Using and Extending

... also Introducing Camille & B-Motion Studio

Jens Bendisposto, UDUS

Event-B – Scheduler/scheduler.bum – Rodin Platform – /Users/tech/MyApplications/rodin_1.0/rodin.app/Contents/MacOS/workspace

Event-B Explor

Scheduler
- Processes
- scheduler

scheduler

```
machine scheduler sees Processes

variables active ready waiting

invariants
  @inv1 active ∈ ℙ(PID)
  @inv2 ready ∈ ℙ(PID)
  @inv3 waiting ∈ ℙ(PID)
  @inv4 ready ∩ waiting = ∅
  @inv5 card(active)≤1
  @inv6 active ∩ (ready ∪ waiting) = ∅
  @inv7 active=∅ ⇒ ready=∅

events
  event INITIALISATION
    then
      @act1 active ≔ ∅
      @act2 ready ≔ ∅
      @act3 waiting ≔ ∅
  end

  event new
    any pp
    where
      @grd1 pp ∈ PID
      @grd2 pp∉active
      @grd3 pp∉ready∪waiting
    then
      @act1 waiting ≔ waiting∪{pp}
  end

  event ready_active
    any rr
```

Outline

- scheduler : sees
  - active
  - ready
  - waiting
  - inv1
  - inv2
  - inv3
  - inv4
  - inv5
  - inv6
  - inv7
  - INITIALISATIO
    - act1
    - act2
    - act3
  - new (pp)
    - grd1
    - grd2
    - grd3
    - act1
  - ready_active
    - grd1
    - grd2
    - act1
    - act2
  - ready (rr)
    - grd1
    - grd2
    - act1
    - act2
  - swap_out

Camille Texteditor

Released for Rodin 1.0

Available through ProB Update Site

Rodin Problems | Properties | Tasks

0 errors, 0 warnings, 0 infos

Description	Resource	Path	Location

0 items selected

Event-B – Scheduler/scheduler.bum – Rodin Platform – /Users/tech/MyApplications/rodin_1.0/rodin.app/Contents/MacOS/workspace

Event-B Explor

Scheduler
- Processes
- scheduler

scheduler

```
machine scheduler sees Processes

variables active ready waiting

invariants
  @inv1 active ∈ ℙ(PID)
  @inv2 ready ∈ ℙ(PID)
  @inv3 waiting ∈ ℙ(PID)
  @inv4 ready ∩ waiting = ∅
  @inv5 card(active)≤1
  @inv6 active ∩ (ready ∪ waiting) = ∅
  @inv7 active=∅ ⇒ ready=∅

events
  event INITIALISATION
    then
      @act1 active ≔ ∅
      @act2 ready ≔ ∅
      @act3 waiting ≔ ∅
  end

  event new
    any pp
    where
      @grd1 pp ∈ PID
      @grd2 pp∉active
      @grd3 pp∉ready∪waiting
    then
      @act1 waiting ≔ waiting∪
  end

  event ready_active
    any rr
```

Camille Claudel

1864-1943

Rodin's source of inspiration and lover

Outline

- scheduler : sees
  - active
  - ready
  - waiting
  - inv1
  - inv2
  - inv3
  - inv4
  - inv5
  - inv6
  - inv7
  - INITIALISATIO
    - act1
    - act2
    - act3
  - new (pp)
    - grd1
    - grd2
    - grd3
    - act1
  - ready_active
    - grd1
    - grd2
    - act1
    - act2
  - ady (rr)
    - grd1
    - grd2
    - act1
    - act2
  - wap_out

Rodin Problems | Properties | Tasks

0 errors, 0 warnings, 0 infos

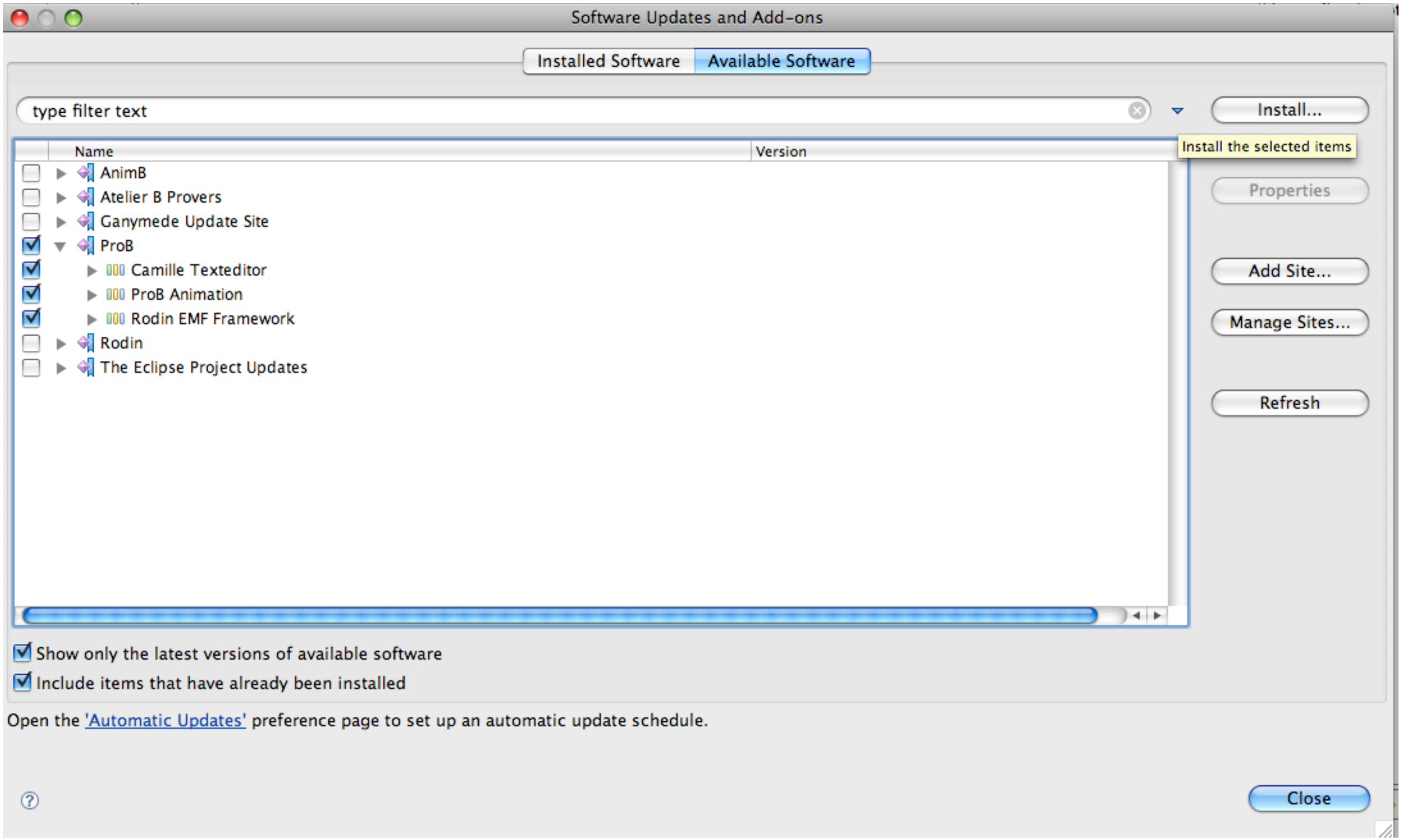
Description	Resource	Path	Location

0 items selected

ProB in a Nutshell

- Animator and Model Checker for B, CSP, Z, Promela and **Event-B**
- Available through the Rodin Update Mechanism
- Tcl/Tk Version available from http://www.stups.uni-duesseldorf.de/ProB

Installing ProB



Event-B – Scheduler/scheduler.bum – Rodin Platform – /Users/tech/MyApplications/rodin_1.0/rodin.app/Contents/MacOS/workspace

Event-B Explor

Scheduler
- Processes
- scheduler

Open
Open With
Properties
Rename
Refine
Validate with ProB
Delete
Retry Auto Provers
Recalculate Auto Status
Simplify Proof(s)
Purge Proofs...
Prove Interactively

Animate/Model Check
ProB Open with ProB classic
ProB Export for use in ProB classic

scheduler

```
machine scheduler sees Processes

variables active ready waiting

invariants
  @inv1 active ∈ ℙ(PID)
        eady ∈ ℙ(PID)
        aiting ∈ ℙ(PID)
        eady ∩ waiting = ∅
        ard(active)≤1
        ctive ∩ (ready ∪ waiting) = ∅
        ctive=∅ ⇒ ready=∅

        t2 ready := ∅
        t3 waiting := ∅

  event new
    any pp
    where
      @grd1 pp ∈ PID
      @grd2 pp∉active
      @grd3 pp∉readyuwaiting
    then
      @act1 waiting := waitingu{pp}
  end

  event ready_active
    any rr
```

Outline

- scheduler : sees
  - active
  - ready
  - waiting
  - inv1
  - inv2
  - inv3
  - inv4
  - inv5
  - inv6
  - inv7
  - INITIALISATIO
    - act1
    - act2
    - act3
  - new (pp)
    - grd1
    - grd2
    - grd3
    - act1
  - ready_active
    - grd1
    - grd2
    - act1
    - act2
  - ready (rr)
    - grd1
    - grd2
    - act1
    - act2
  - swap_out

Rodin Problems | Properties | Tasks

0 errors, 0 warnings, 0 infos

Description	Resource	Path	Location

1 items selected

ProB – Scheduler/scheduler.bum – Rodin Platform – /Users/tech/MyApplications/rodin_1.0/rodin.app/Contents/MacOS/workspace

Event-B | Rodin Pr

Scheduler
- Processes
- scheduler

scheduler

```
machine scheduler sees Processes

variables active ready waiting

invariants
  @inv1 active ∈ ℙ(PID)
  @inv2 ready ∈ ℙ(PID)
  @inv3 waiting ∈ ℙ(PID)
  @inv4 ready ∩ waiting = ∅
  @inv5 card(active)≤1
  @inv6 active ∩ (ready ∪ waiting) = ∅
  @inv7 active=∅ ⇒ ready=∅

events
  event INITIALISATION
    then
      @act1 active ≔ ∅
      @act2 ready ≔ ∅
      @act3 waiting ≔ ∅
  end

  event new
    any pp
    where
      @grd1 pp ∈ PID
      @grd2 pp∉active
      @grd3 pp∉ready∪waiting
    then
      @act1 waiting ≔ waiting∪{pp}
end
```

State

Name	Value
Variables	
active	
ready	
waiting	

History

Operations	Loops
(root)	

Operations

Operation	Parameter(s)
ready_active	
swap_out	
ready	
swap	
new	
initialise_machine	{}, {}, {}

Evaluate Expression

[Rodin Label] Functor	Value	Predicate / Expression
AXIOMS		
INVARIANT		
THEOREMS		

Predicate / Expression | Value

1 items selected

ProB – Scheduler/scheduler.bum – Rodin Platform – /Users/tech/MyApplications/rodin_1.0/rodin.app/Contents/MacOS/workspace

Event-B | Rodin Pr

Scheduler
- Processes
- scheduler

State

Name	Value	Previous value
Variables		
active	{}	{PID2}
ready	{}	
waiting	{PID1,PID2}	{PID1}

invariant ok

History

Operations	Loops
swap_out()	
swap(PID2)	
ready(PID2)	
ready_active(PID1)	
new(PID2)	
new(PID1)	
initialise_machine({},{},{})	
(root)	

Operations

Operation	Parameter(s)
ready_active	PID1
swap_out	
ready	
swap	
new	

Evaluate Expression

[Rodin Label] Functor	Value	Predicate / Expression
AXIOMS		
INVARIANT		
THEOREMS		

Predicate / Expression | Value

ProB – Simple/count.bum – Rodin Platform – /Users/tech/MyApplications/rodin_1.0/rodin.app/Contents/MacOS/...

Operations

Operation	Parameter(s)
inc	0

0
1
2
3

count

```
machine count

variables x

invariants
  @type x∈ℕ

events
  event INITIALISATION
    then
      @init x := 0
  end

  event inc
    any y
    where
      @grd y∈ℕ
    then
      @inc x := x +y
  end
end
```

State

Name	Value	Previou
Variables		
x	2	

invariant ok

Preferences

type filter text

- General
- Event-B
- Help
- Install/Update
- ProB
  - ProB Classic
- Run/Debug
- Team

ProB

MaxInt, used for expressions such as xx::NAT (2147483647 for 4 byte ints) 3

MinInt, used for expressions such as xx::INT (-2147483648 for 4 byte ints) -1

Size of unspecified deferred sets in SETS section 2

Max Number of Initialisations Computed 4

Max Number of Enablings per Operation Computed 10

Show top-level ANY variable values of Event B Events as operation parameters ☐

Animate operations which are skip or PRE C THEN skip ☑

Convert lazy form back into explicit form for Variables, Constants, Operation Arguments ☑

Lazy expansion of lambdas set comprehensions ☐

Lazy expansion of *Recursive* set Comprehensions and lambdas ☐

Symmetry Mode: off,flood,canon,nauty,hash off

Time out for computing enabled transitions (in ms) 2500

Animate multiple levels of refinement in Event-B (experimental) ☐

Restrict invariant checking to affected clauses. Also remove clauses that are proven (EventB) ☑

Restore Defaults Apply

Cancel OK

ProB – Simple/count.bum – Rodin Platform – /Users/tech/MyApplications/rodin_1.0/rodin.app/Contents/MacOS/...

Operations

Operation	Parameter(s)
inc	0

4
5
6
7

count

```
machine count

variables x

invariants
  @type x∈ℕ

events
  event INITIALISATION
    then
      @init x ≔ 0
  end

  event inc
    any y
    where
      @grd y∈ℕ
    then
      @inc x ≔ x +y
  end
end
```

State

Name	Value	Previou
Variables		
x	0	

invariant ok

State

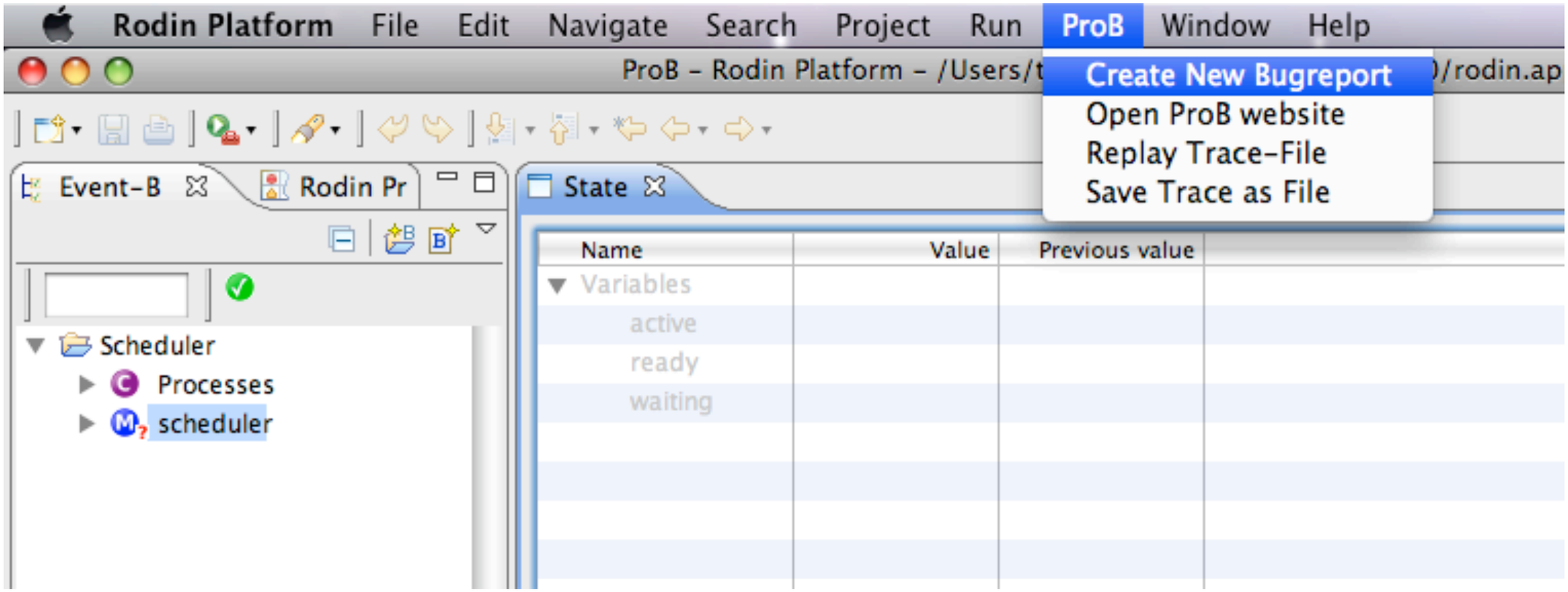
Name	Value	Previous value
ctx0		
discharged	{(e1|->i1),(e2|->i2)}	
initial	{s1}	
label	{((s1|->s2)|->e1),((s1|->s3)|->e2),((s2|->s3)|->e1),((s3|->s4)|->e1)}	
preserve	{s1,s3}	
proofobligations	{(e1|->i1),(e1|->i2),(e2|->i1),(e2|->i2),(e3|->i1),(e3|->i2)}	
trans	{(s1|->s2),(s1|->s3),(s2|->s3),(s3|->s4)}	
truth	{(s1|->i1),(s1|->i2),(s2|->i1),(s3|->i1),(s3|->i2),(s4|->i1)}	
violate	{s2,s4}	
mc2		
inv_broken	{s2}	{}
inv_ok	{}	
open	{s1,s3,s4}	TYPES:STATES
mc1		
inv_broken	{s2}	{}
inv_ok	{}	
open	{s1,s3,s4}	TYPES:STATES
mc0		
inv_broken	{s2}	{}
inv_ok	{}	
open	{s1,s3,s4}	TYPES:STATES

invariant ok

Something is wrong?

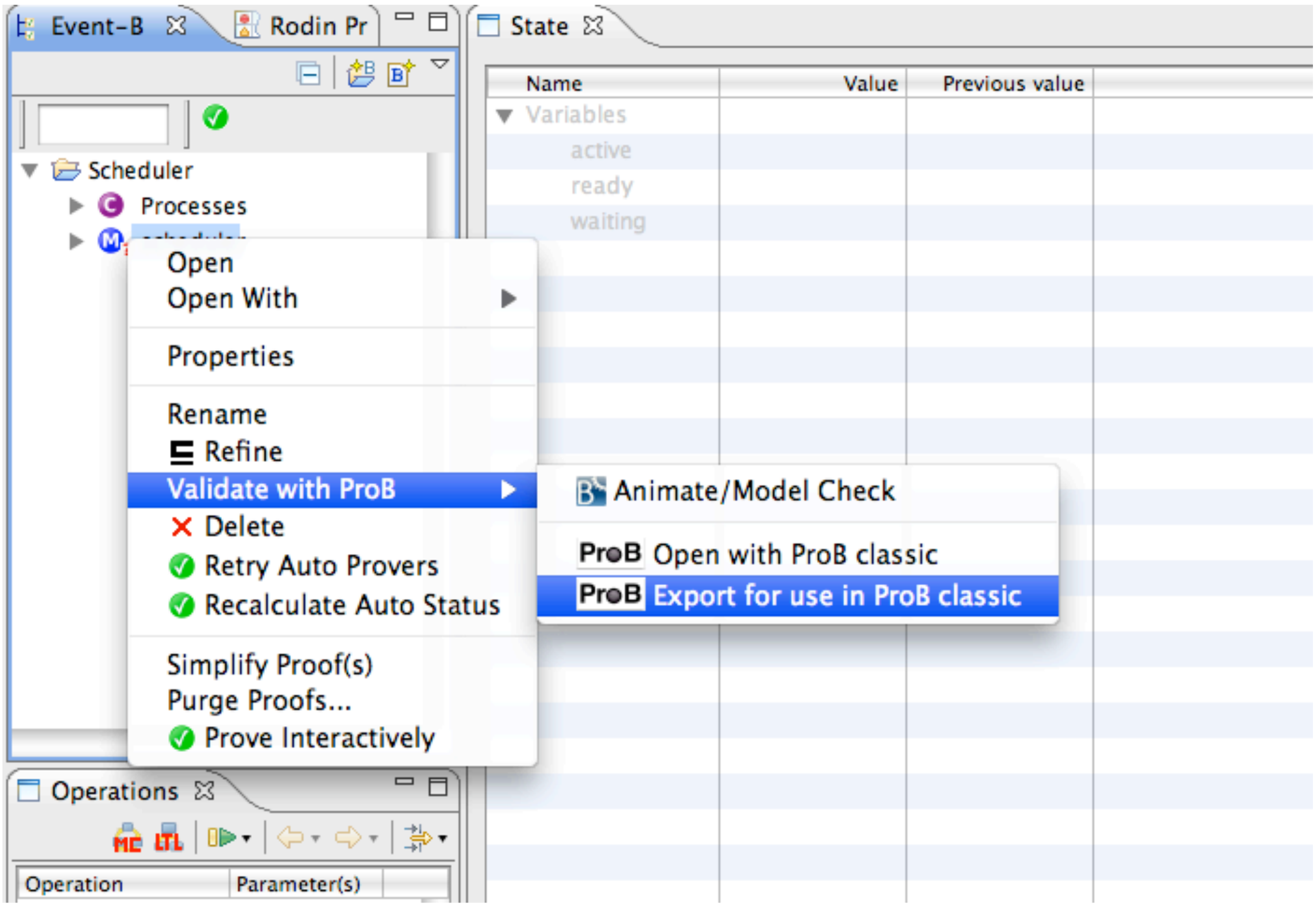
Something is missing?

Important: Let us know!



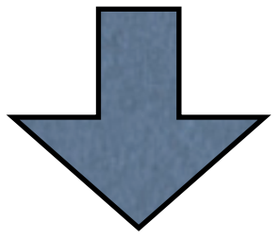
If possible: Attach Trace

Meanwhile ...

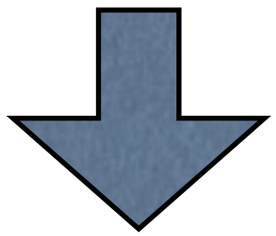

Event-B
Rodin Pr
State
Name
Value
Previous value
Variables
active
ready
waiting
Scheduler
Processes
Open
Open With
Properties
Rename
Refine
Validate with ProB
Delete
Retry Auto Provers
Recalculate Auto Status
Simplify Proof(s)
Purge Proofs...
Prove Interactively
Animate/Model Check
Open with ProB classic
Export for use in ProB classic
Operations
Operation
Parameter(s)

Extending ProB

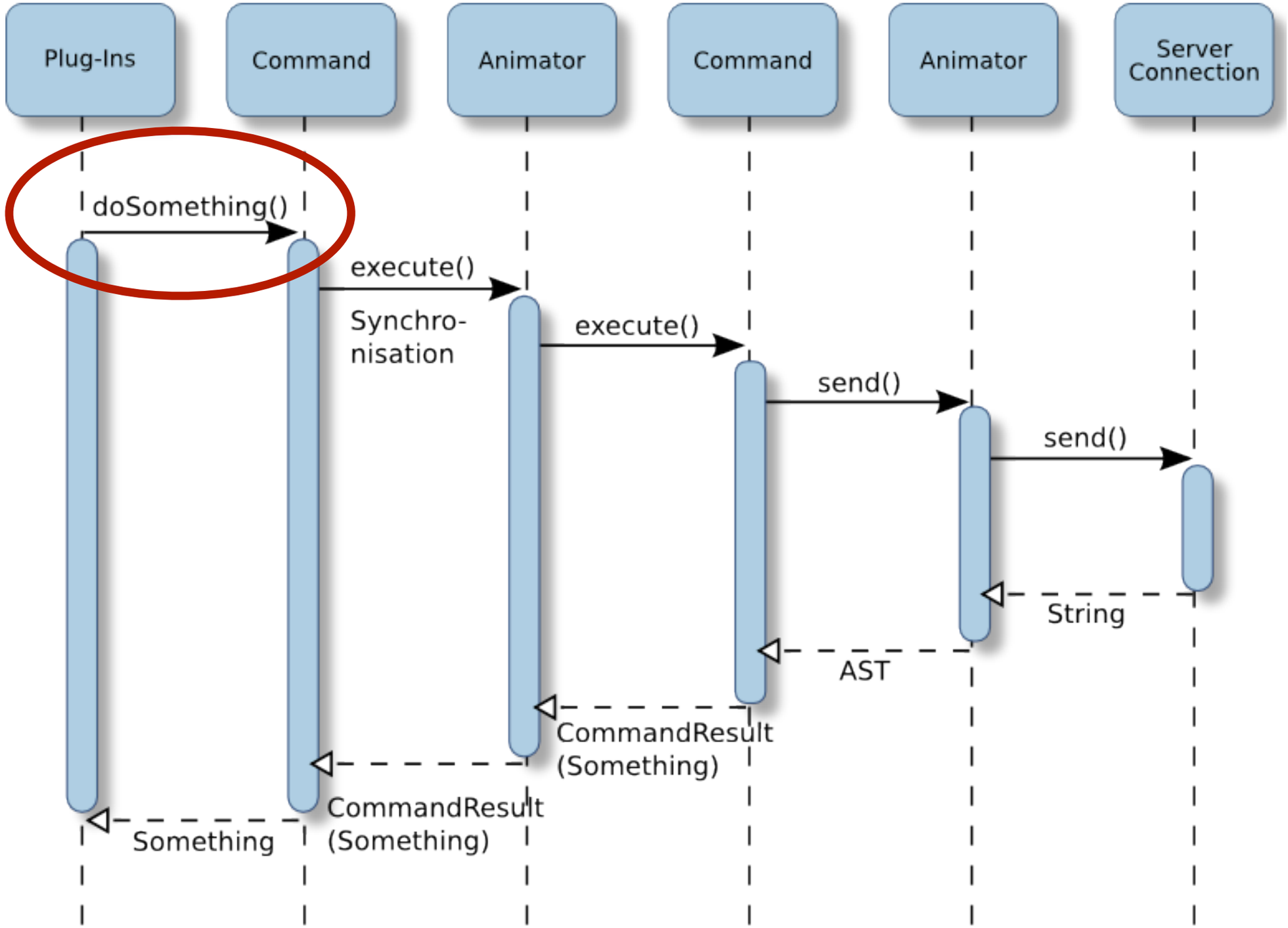
Command

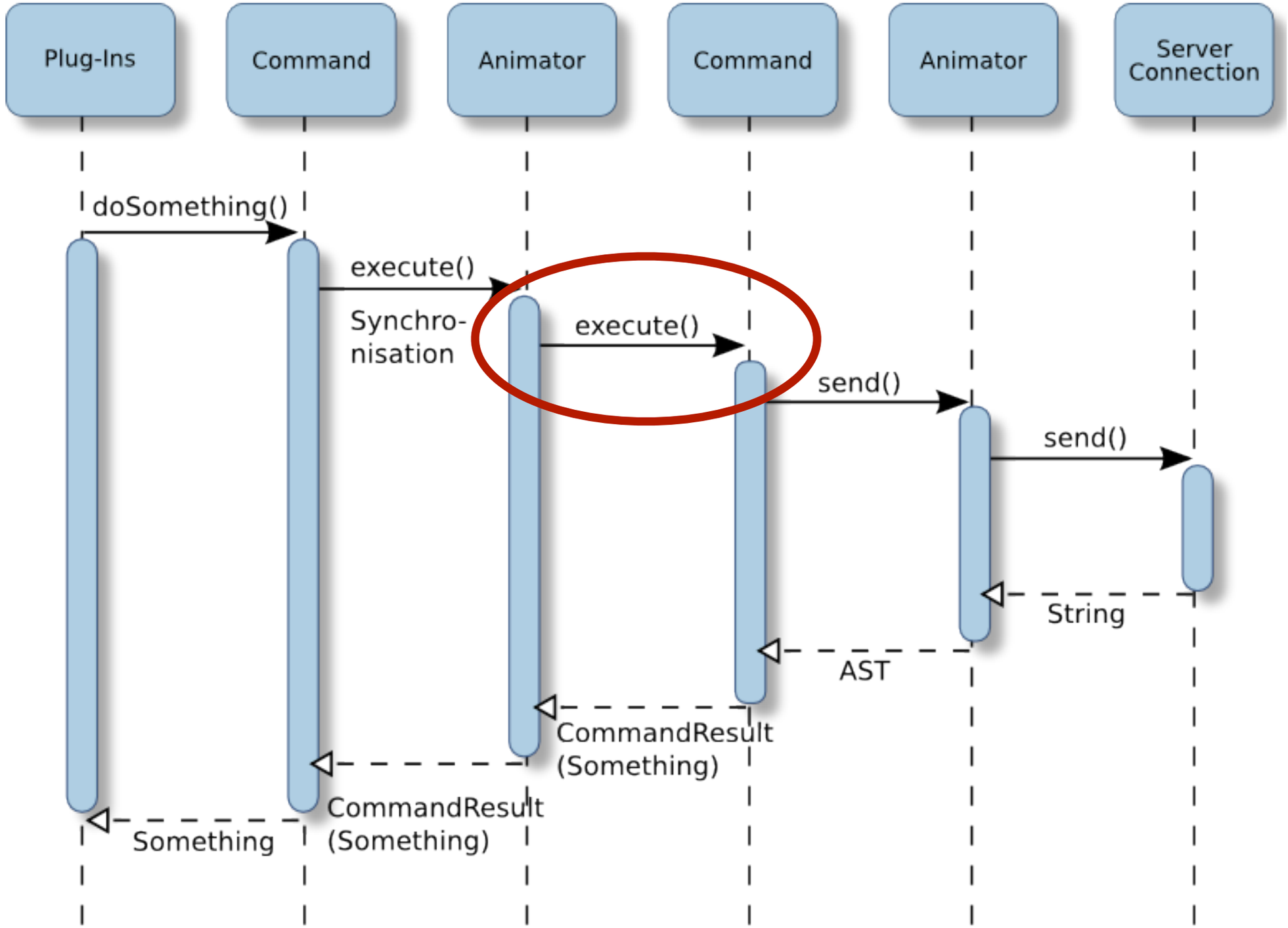


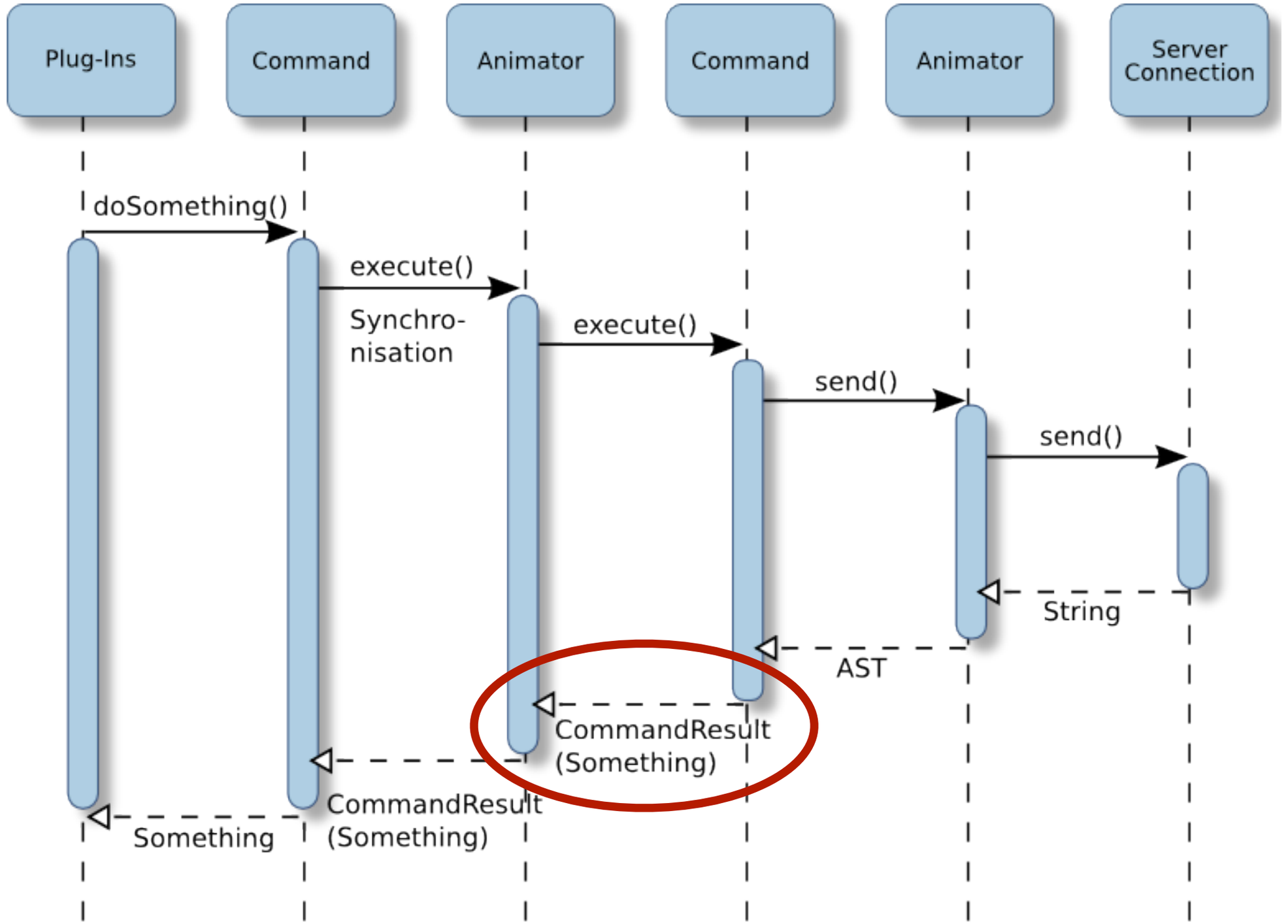
Animator

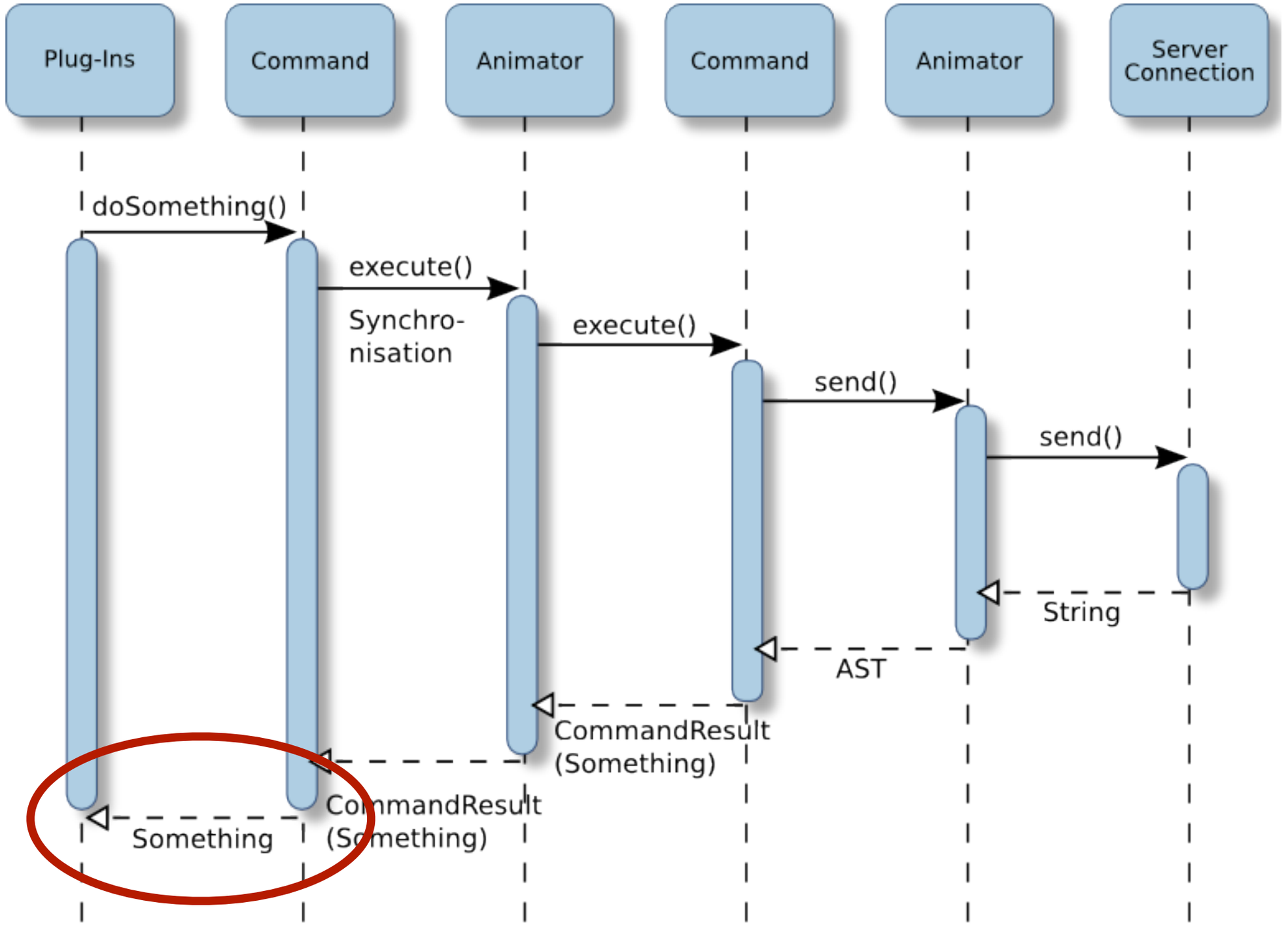


Result

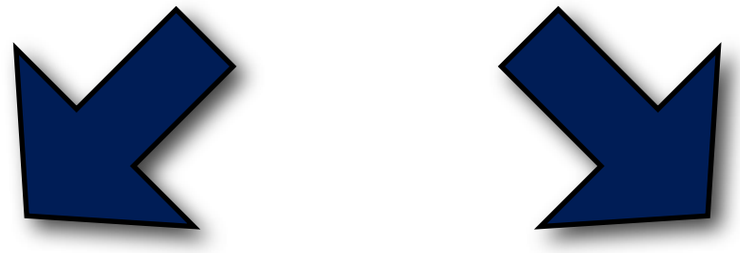

Plug-Ins
Command
Animator
Command
Animator
Server Connection
doSomething()
execute()
Synchro-
nisation
execute()
send()
send()
String
AST
CommandResult
(Something)
CommandResult
(Something)
Something


Plug-Ins
Command
Animator
Command
Animator
Server Connection
doSomething()
execute()
Synchro-
nisation
execute()
send()
send()
String
AST
CommandResult
(Something)
CommandResult
(Something)
Something


Plug-Ins
Command
Animator
Command
Animator
Server Connection
doSomething()
execute()
Synchro-
nisation
execute()
send()
send()
String
AST
CommandResult
(Something)
CommandResult
(Something)
Something


Plug-Ins
Command
Animator
Command
Animator
Server Connection
doSomething()
execute()
Synchro-
nisation
execute()
send()
send()
String
AST
CommandResult
(Something)
CommandResult
(Something)
Something

Two (basic) Flavours



Basic Commands

- Call Prolog Code
- May need support from UDUS

Composed Commands

- Call other Commands
- May not need support from UDUS

No real difference

- Prolog calls and Command calls can be mixed

Example: Exploring a new State

```
List<Operation> enabledOperations =
GetEnabledOperationsCommand.getOperations(animator, id);

StateValuesResult stateValues =
GetStateValuesCommand.getStateValues(animator, id);

boolean initialised =
CheckInitialisationStatusCommand.isInitialized(animator, id);

boolean invariantKo =
CheckInvariantStatusCommand.isInvariantViolated(animator, id);

boolean maxOperationReached = CheckMaxOperationReachedStatusCommand
            .maxOperationReached(animator, id);

boolean timeoutOccured = CheckTimeoutStatusCommand.isTimeout(
             animator, id);
```

Example: Getting the current State ID

```
final static String COMMAND ="getCurrentStateID(ID).";

public CommandResult<String> execute(final Animator a) throws
ProBException {
      Map<String, PrologTerm> bindings = BindingGenerator
            .createBindingMustNotFail(COMMAND,
                  animator.sendCommand(COMMAND));

      String id = bindings.get("ID").toString();
      return new CommandResultAdapter<String>(id);
   }

public static String getID(final Animator a) throws ProBException {
      GetCurrentStateIdCommand getID =
                        new GetCurrentStateIdCommand();
      return (String) a.execute(getID).unwrap();
}
```

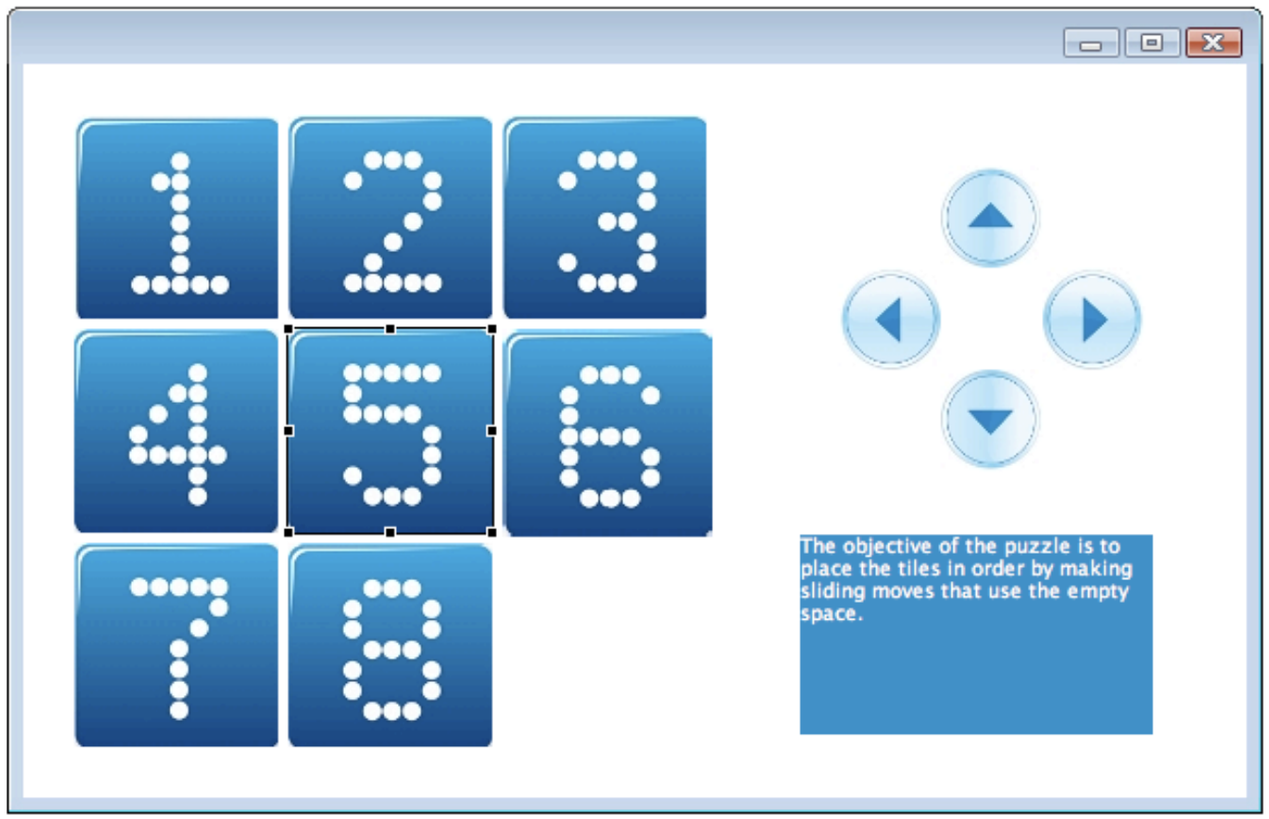
Coming soon

B-Motion Studio - Puzzle8/Visualization.bmso - Rodin Platform - /Users/tech/MyApplications/rodin_1.0/rodin.app/Contents/MacOS/workspace

100%

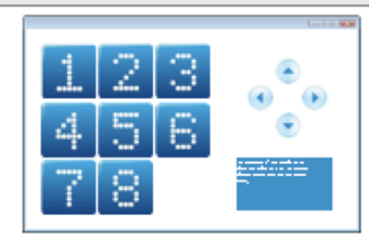
B-Motion Studio Editor

Industry
B-Canister
Main
B-Image
B-Button
B-Composite
B-Text



Outline

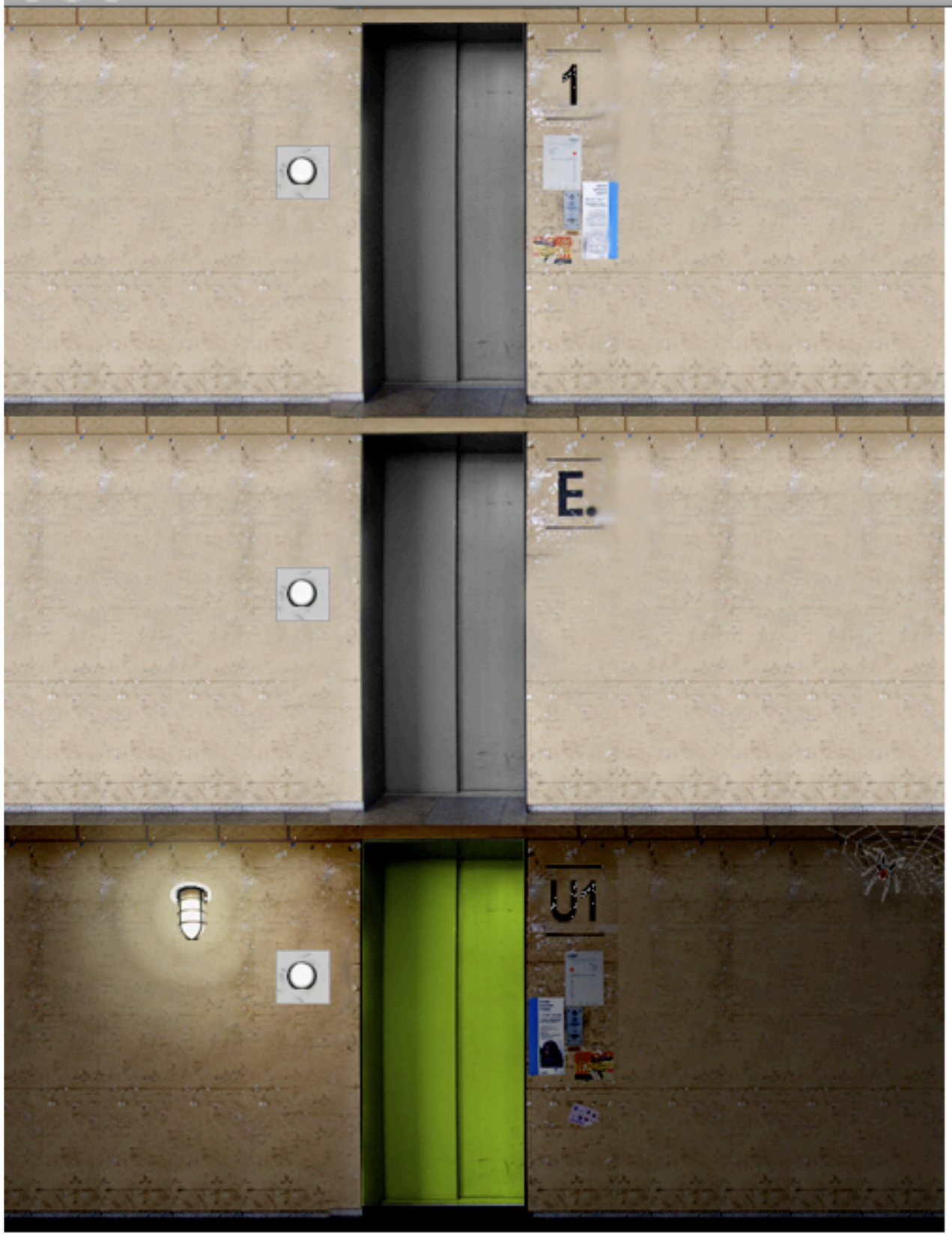
puzzle_4
puzzle_6
puzzle_7
puzzle_8
puzzle_5
bt_down
bt_right
bt_left
bt_up
instruction



Properties

Property	Value
height	100
id	puzzle_5
image	images/5.jpg
width	100
x	130
y	130

Visualization for machine mch.bum



Reverse UP

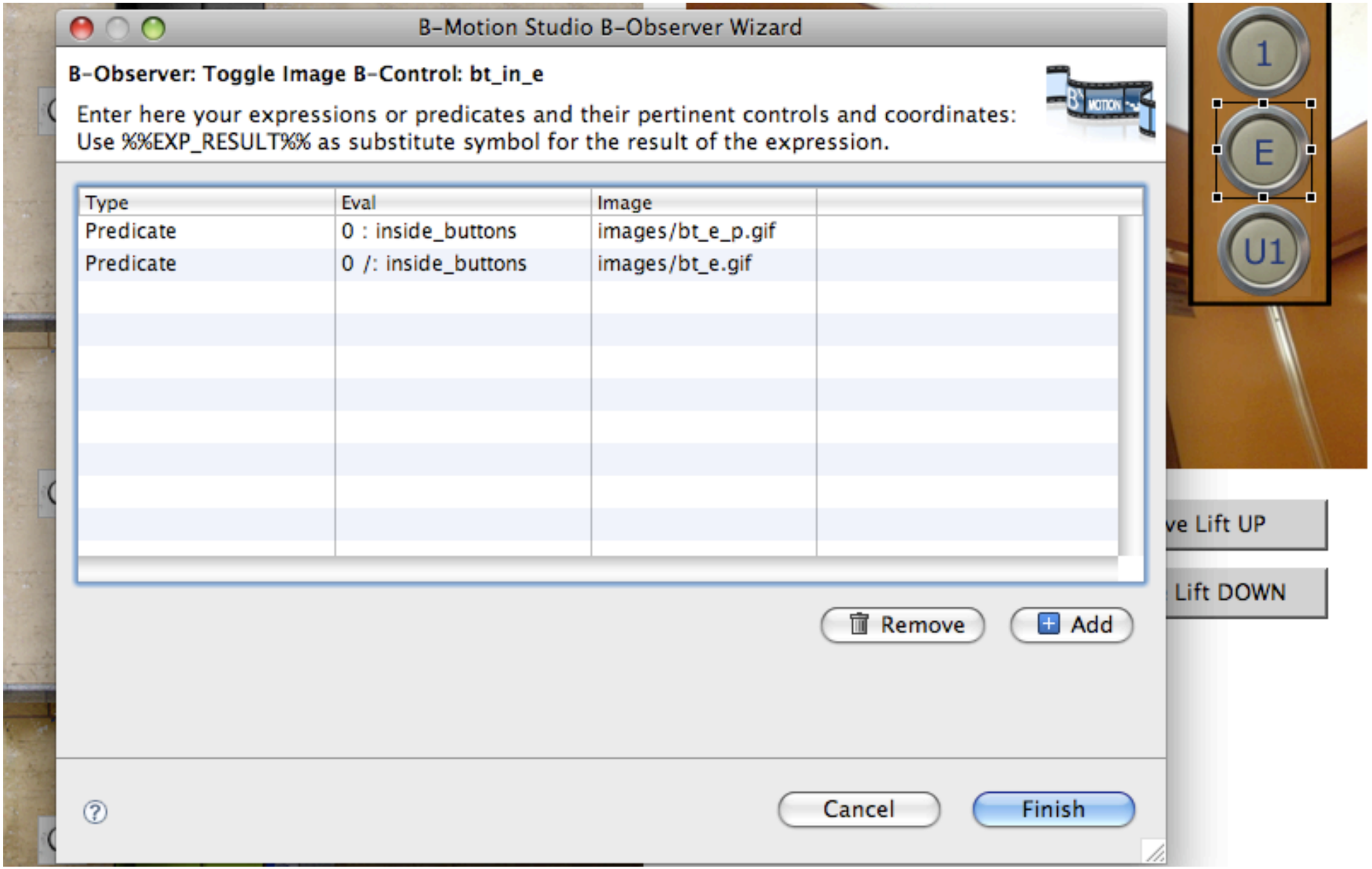
Move Lift UP

Reverse Down

Move Lift DOWN

Open Door

Close Door


B-Motion Studio B-Observer Wizard
B-Observer: Toggle Image B-Control: bt_in_e
Enter here your expressions or predicates and their pertinent controls and coordinates:
Use %%EXP_RESULT%% as substitute symbol for the result of the expression.
Type
Eval
Image
Predicate
0 : inside_buttons
images/bt_e_p.gif
Predicate
0 /: inside_buttons
images/bt_e.gif
Remove
Add
Cancel
Finish
1
E
U1
ve Lift UP
Lift DOWN

Acknowledgements

- Camille Texteditor and EMF Framework: Fabian Fritz, Colin Snook, Alexei Iliasov
- B-Motion Studio: Lukas Ladenberger