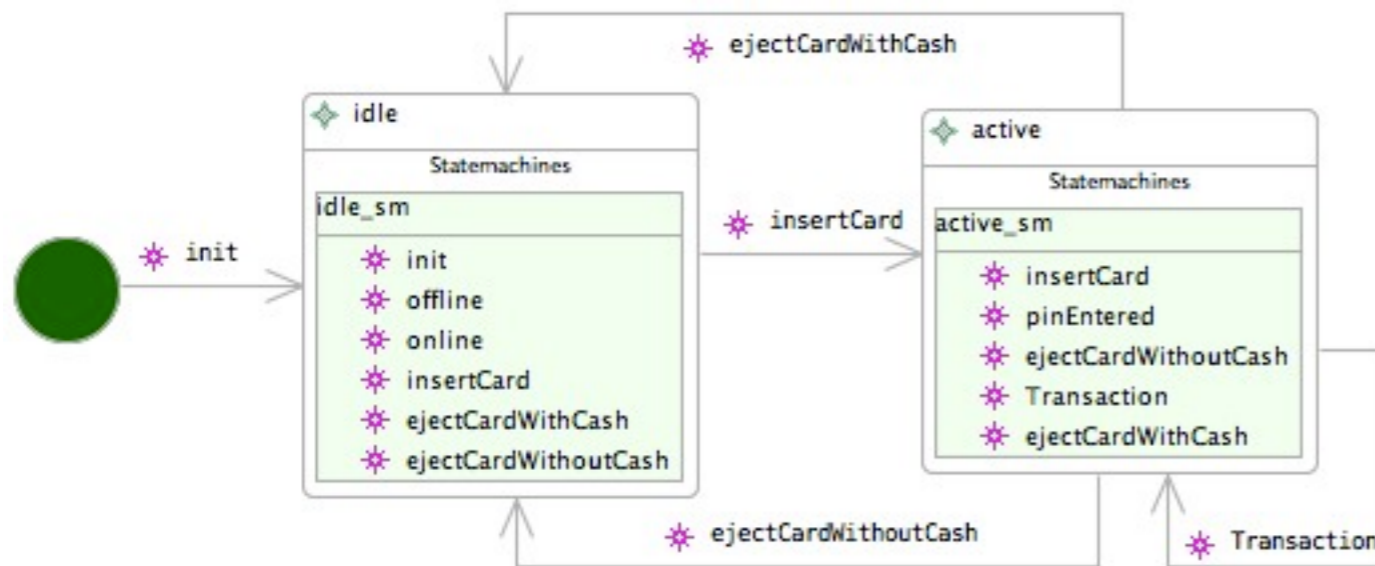


UML-B State-machine Animation



UML-B Modelling Environment

UML-B:



Event-B:

```

EVENTS
INITIALISATION ≙
STATUS
ordinary
BEGIN
C.init : C = ∅
atm_sm.init : atm_sm = ∅
idle_sm.init : idle_sm = ∅
active_sm.init : active_sm = ∅
END

insertCard ≙
STATUS
ordinary
REFINES
insertCard
ANY
self // contextual instance of class C
WHERE
self.type : self ∈ C
idle_sm_isin_available_wd : self ∈ dom(idle_sm)
idle_sm_isin_available : idle_sm(self) = available
THEN
  
```



ProB Animator and Model Checker

The screenshot displays the ProB Animator and Model Checker interface, which is divided into several panels:

- Events Panel:** Shows a list of events with their parameters. The event 'ejectCardWithCash' with parameter 'C_SET1' is currently selected.
- State Panel:** Displays the current state of the system. It includes variables for 'ATM_A' and 'ATM_R', their values, and previous values. Below the variables, it shows the status of various formulas, including invariants, all of which are currently true (T).
- Event-B Panel:** Shows the current event being executed, 'Transaction', and the model being simulated, 'Rodin Pro'.
- History Panel:** Shows a sequence of actions performed by the model checker, including 'Transaction', 'pinEntered', 'insertCard', and 'INITIALISATION'.

At the bottom of the state panel, a green bar indicates 'invariant ok' and 'no event errors detected'.



Power of Two: UML-B and ProB

The screenshot displays the Rodin Pro IDE interface with several panels:

- Events Panel:** Lists events and their parameters.

Event	Parameter(s)
▶ insertCard	C_SET2
⊖ init	
▶ ejectCardWithCash	C_SET1
⊖ Transaction	
⊖ ejectCardWithoutCash	
▶ offline	C_SET2
⊖ online	
⊖ pinEntered	
- State Diagram Panel:** Shows a state machine with states `idle` and `active`. Transitions are labeled with events like `insertCard`, `ejectCardWithCash`, and `Transaction`. Each state contains instances of `C_SET2` and `C_SET1`.
- State Panel:** A table showing the current state of the system.

Name	Value	Previous value
★ ATM_A		
★ C	{C_SET1,C_SET2}	{C_SET1}
★ atm_sm	active),(C_SET2→idle)}	{{(C_SET1→active)}
★ ATM_R		
active_sm	{{(C_SET1→dispensing)}	{{(C_SET1→dispensing)}
★ idle_sm	{{(C_SET2→available)}	∅
★ C	{C_SET1,C_SET2}	{C_SET1}
★ atm_sm	active),(C_SET2→idle)}	{{(C_SET1→active)}
- History Panel:** Shows a sequence of events for `ATM_R` and `ATM_A`.

ATM_R	ATM_A
init	init
Transaction	Transaction
pinEntered	
insertCard	insertCard

At the bottom, two green status bars indicate: **invariant ok** and **no event errors detected**.

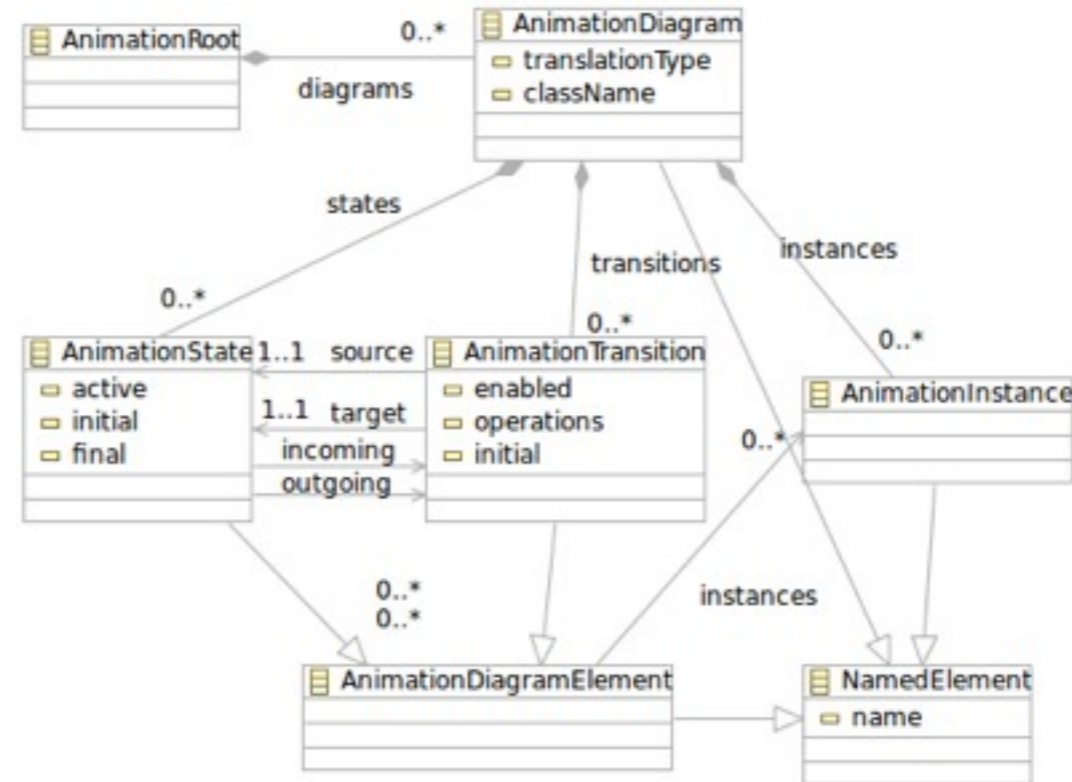


Implementation

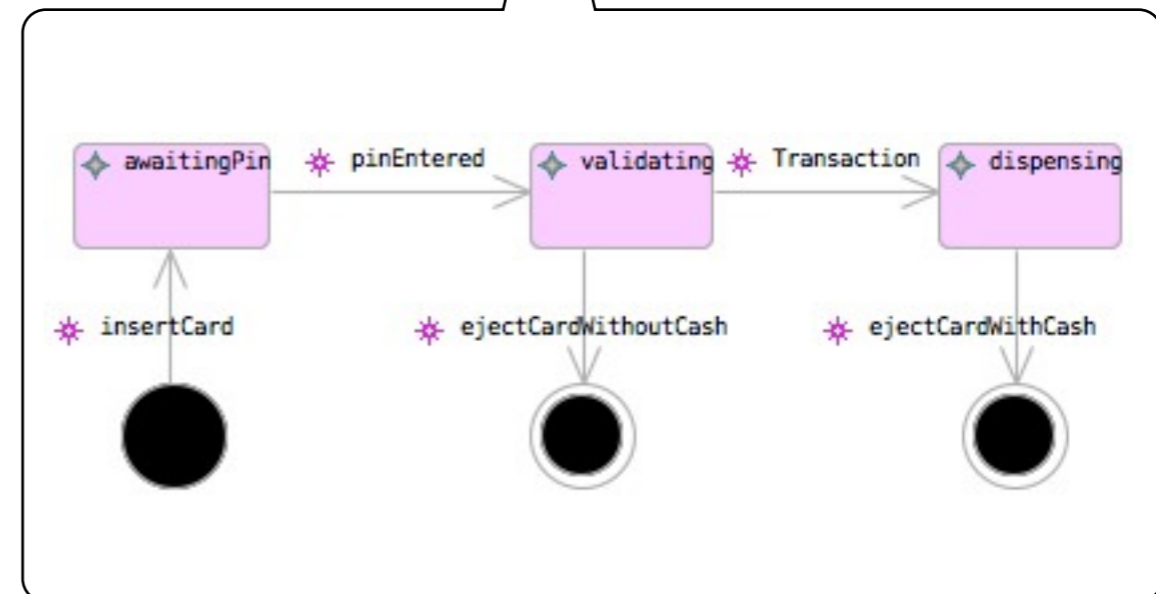
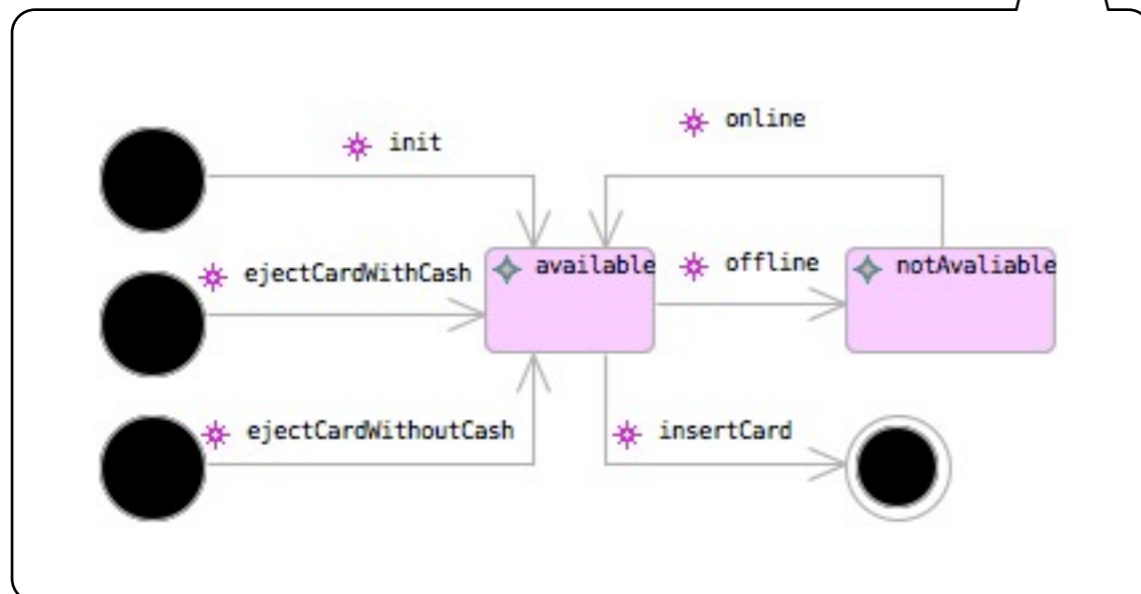
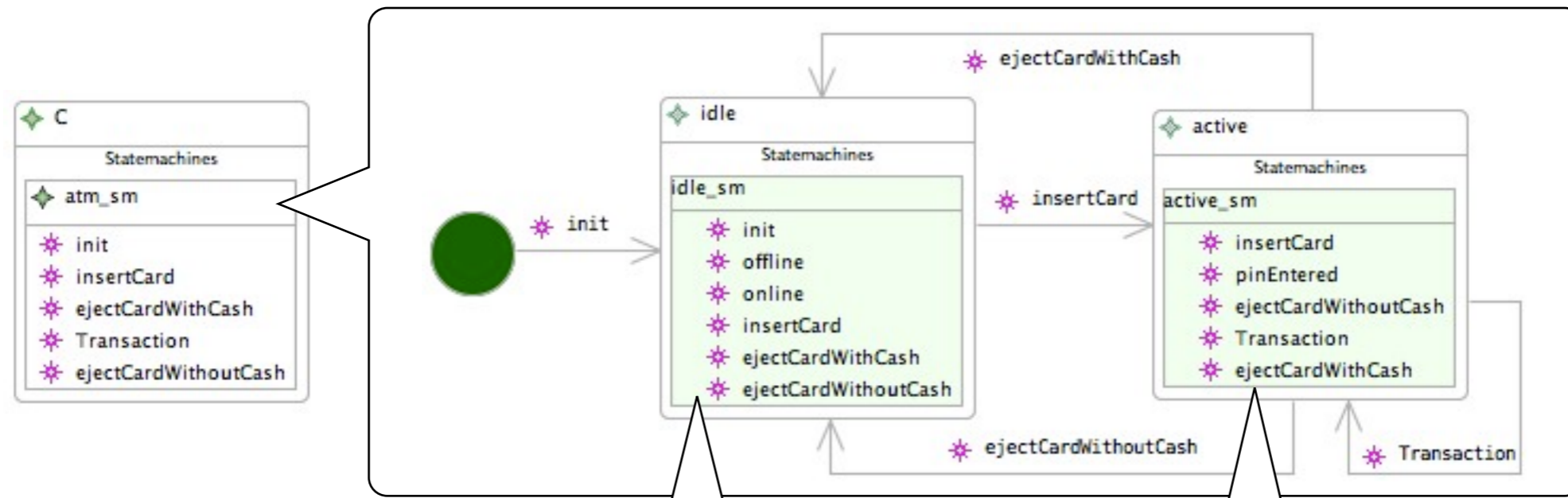
- Animation Metamodel (EMF)
- Customised Graphical Metamodel (GMF)
- org.eclipse.ui extension
- de.prob.core.animation: <listener> extension

dependencies:

- UML-B Metamodel
- org.rodinp.core API
- de.prob.core API
- de.prob.ui.perspective



Animation Example: ATM



Animation Example: ATM (start)

The screenshot displays the Rodin Pro IDE interface for an ATM state machine. The main window shows a state machine diagram with two states: 'idle' and 'active'. Transitions are labeled with events: 'init' (to idle), 'insertCard' (to active), 'Transaction' (to active), 'ejectCardWithoutCash' (back to idle), and 'ejectCardWithCash' (back to idle). The 'idle' state is currently selected.

Below the diagram is the 'State' table, which tracks the state of the system across different components:

Name	Value	Previous value
ATM_A		
C		
atm_sm		
ATM_R		
active_sm		
idle_sm		
C		
atm_sm		

Below the state table, a green bar indicates 'no event errors detected'. At the bottom, the 'History' table shows the current state of the system:

ATM_R	ATM_A
(uninitialised state)	

On the left side, the 'Events' panel lists various events with their parameters:

Event	Parameter(s)
insertCard	
init	
ejectCardWithCash	
Transaction	
ejectCardWithoutCash	
offline	
online	
pinEntered	
INITIALISATION	∅,∅,∅,∅



Animation Example: ATM (initialisation)

State

Name	Value	Previous value
★ ATM_A		
★ C	∅	
★ atm_sm	∅	
★ ATM_R		
★ active_sm	∅	
★ idle_sm	∅	
★ C	∅	
★ atm_sm	∅	

Events

Event	Parameter(s)
insertCard	
init (x2)	C_SET1
ejectCardWithCash	
Transaction	
ejectCardWithoutCash	
offline	
online	
pinEntered	

History

ATM_R	ATM_A
INITIALISATION (uninitialised state)	INITIALISATION

invariant ok no event errors detected



Animation Example: ATM (animation I)

The screenshot shows the Rodin IDE interface for animating a state machine. The main window displays a state machine diagram with states 'idle' and 'active'. A red arrow points from the 'idle' state to the 'State' table below. The 'State' table shows the current state of objects ATM_A, ATM_R, and C. The 'Events' table shows a sequence of events like 'insertCard' and 'init'. The 'History' table shows the sequence of states and events.

Name	Value	Previous value
ATM_A		
C	{C_SET1}	∅
atm_sm	{{(C_SET1→idle)}}	∅
ATM_R		
active_sm	∅	∅
idle_sm	{{(C_SET1→available)}}	∅
C	{C_SET1}	∅
atm_sm	{{(C_SET1→idle)}}	∅

Event	Parameter(s)
▶ insertCard	C_SET1
▶ init	C_SET2
◻ ejectCardWithCash	
◻ Transaction	
◻ ejectCardWithoutCash	
▶ offline	C_SET1
◻ online	
◻ pinEntered	

ATM_R	ATM_A
init	init
INITIALISATION	INITIALISATION
(uninitialised state)	



Animation Example: ATM (animation II)

State

Name	Value	Previous value
★ ATM_A		
★ C	{C_SET1,C_SET2}	{C_SET1}
★ atm_sm	active),(C_SET2→idle)	{{(C_SET1→active)}
★ ATM_R		
active_sm	{{(C_SET1→awaitingPin)}	{{(C_SET1→awaitingPin)}
★ idle_sm	{{(C_SET2→available)}	∅
★ C	{C_SET1,C_SET2}	{C_SET1}
★ atm_sm	active),(C_SET2→idle)	{{(C_SET1→active)}

Events

Event	Parameter(s)
▶ insertCard	C_SET2
◻ init	
◻ ejectCardWithCash	
◻ Transaction	
◻ ejectCardWithoutCash	
▶ offline	C_SET2
◻ online	
▶ pinEntered	C_SET1

History

ATM_R	ATM_A
init	init
insertCard	insertCard
init	init

invariant ok no event errors detected



Animation Example: ATM (animation III)

The screenshot shows the Rodin Pro IDE with an ATM state machine animation. The main window displays a state transition diagram with states `awaitingPin`, `validating`, and `dispensing`. The left sidebar shows a project tree with ATM-related files. The bottom panels show the State and History views.

State View:

Name	Value	Previous value
★ ATM_A		
★ C	{C_SET1,C_SET2}	{C_SET1}
★ atm_sm	active),(C_SET2→idle}}	{{(C_SET1→active}}
★ ATM_R		
active_sm	{{(C_SET1→awaitingPin}}	{{(C_SET1→awaitingPin}}
★ idle_sm	{{(C_SET2→available}}	∅
★ C	{C_SET1,C_SET2}	{C_SET1}
★ atm_sm	active),(C_SET2→idle}}	{{(C_SET1→active}}

History View:

ATM_R	ATM_A
init	init
insertCard	insertCard
init	init



Animation Example: ATM (animation IV)

State

Name	Value	Previous value
▼ ATM_A		
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
atm_sm	.→active),(C_SET2→idle)	.→active),(C_SET2→idle)
▼ ★ ATM_R		
★ active_sm	{{(C_SET1→validating)}} C_SET1→awaitingPin}}	
idle_sm	{{(C_SET2→available)}}	{{(C_SET2→available)}}
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
atm_sm	.→active),(C_SET2→idle)	.→active),(C_SET2→idle)

History

ATM_R	ATM_A
pinEntered	
init	init
insertCard	insertCard



Animation Example: ATM (animation V)

State

Name	Value	Previous value
▼ ATM_A		
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
atm_sm	.→active),(C_SET2→idle)	.→active),(C_SET2→idle)
▼ ★ ATM_R		
★ active_sm	{{(C_SET1→validating)}} C_SET1→awaitingPin}}	
idle_sm	{{(C_SET2→available)}} {{(C_SET2→available}}	
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
atm_sm	.→active),(C_SET2→idle)	.→active),(C_SET2→idle)

Events

Event	Parameter(s)
▶ insertCard	C_SET2
● init	
● ejectCardWithCash	
▶ Transaction	C_SET1
▶ ejectCardWithoutCash	C_SET1
▶ offline	C_SET2
● online	
● pinEntered	

History

ATM_R	ATM_A
pinEntered	
init	init
insertCard	insertCard



Animation Example: ATM (animation VI)

State

Name	Value	Previous value
▼ ATM_A		
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
atm_sm	→active),(C_SET2→idle)	→active),(C_SET2→idle)
▼ ★ ATM_R		
★ active_sm	{{(C_SET1→validating)}} C_SET1→awaitingPin}}	
idle_sm	{{(C_SET2→available)}} {{(C_SET2→available}}	
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
atm_sm	→active),(C_SET2→idle)	→active),(C_SET2→idle)

History

ATM_R	ATM_A
pinEntered	
init	init
insertCard	insertCard



Animation Example: ATM (animation VII)

The screenshot shows the Rodin Pro IDE interface for animating a state machine. The main window displays a state transition diagram for the ATM state machine. The states are `awaitingPin`, `validating`, and `dispensing`. Transitions are labeled with events: `pinEntered` (from `awaitingPin` to `validating`), `Transaction` (from `validating` to `dispensing`), `insertCard` (to `awaitingPin`), `ejectCardWithoutCash` (from `validating`), and `ejectCardWithCash` (from `dispensing`). The `dispensing` state has an instance `C_SET1`.

The left sidebar shows the project tree for 'ATM', including state machines and class diagrams for `ATM_A` and `ATM_R`.

The bottom panels show the 'State' and 'History' views.

State View:

Name	Value	Previous value
ATM_A		
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
atm_sm	.→active),(C_SET2→idle)	.→active),(C_SET2→idle)
ATM_R		
* active_sm	{(C_SET1→dispensing)}	{(C_SET1→validating)}
idle_sm	{{(C_SET2→available)}	{{(C_SET2→available)}
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
atm_sm	.→active),(C_SET2→idle)	.→active),(C_SET2→idle)

History View:

ATM_R	ATM_A
Transaction	Transaction
pinEntered	
init	init

Events View:

Event	Parameter(s)
▶ insertCard	C_SET2
● init	
▶ ejectCardWithCash	C_SET1
● Transaction	
● ejectCardWithoutCash	
▶ offline	C_SET2
● online	
● pinEntered	

Summary: invariant ok, no event errors detected



Animation Example: ATM (animation VIII)

Events

Event	Parameter(s)
▶ insertCard (x2)	C_SET1
◻ init	
◻ ejectCardWithCash	
◻ Transaction	
◻ ejectCardWithoutCash	
▶ offline (x2)	C_SET1
◻ online	
◻ pinEntered	

State

Name	Value	Previous value
★ ATM_A		
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
★ atm_sm	1→idle),(C_SET2→idle)	active),(C_SET2→idle)
★ ATM_R		
★ active_sm	∅ (C_SET1→dispensing))	
★ idle_sm	e),(C_SET2→available))	{{(C_SET2→available)}
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
★ atm_sm	1→idle),(C_SET2→idle)	active),(C_SET2→idle)

History

ATM_R	ATM_A
ejectCardWithCash	ejectCard
Transaction	Transaction
pinEntered	

Summary: invariant ok, no event errors detected



Animation Example: ATM (animation IX)

The screenshot shows the Rodin Pro IDE interface for animating a state machine. The main window displays a state transition diagram for 'ATM.ATM_R.C.atm_sm.i'. The diagram includes states like 'available', 'notAvailable', and 'offline', with transitions triggered by events such as 'init', 'online', 'offline', 'insertCard', 'ejectCardWithCash', and 'ejectCardWithoutCash'. The 'available' state contains instances 'C_SET2' and 'C_SET1'. The left sidebar shows a project tree for 'ATM'. The bottom panels show 'Events' and 'History' logs.

Events Log:

Event	Parameter(s)
▶ insertCard (x2)	C_SET1
◻ init	
◻ ejectCardWithCash	
◻ Transaction	
◻ ejectCardWithoutCash	
▶ offline (x2)	C_SET1
◻ online	
◻ pinEntered	

History Log:

ATM_R	ATM_A
ejectCardWithCash	ejectCard
Transaction	Transaction
pinEntered	

State Log:

Name	Value	Previous value
★ ATM_A		
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
★ atm_sm	1→idle),(C_SET2→idle)) active),(C_SET2→idle))	
★ ATM_R		
★ active_sm	∅ {C_SET1→dispensing))	
★ idle_sm	e),(C_SET2→available)) {(C_SET2→available))	
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
★ atm_sm	1→idle),(C_SET2→idle)) active),(C_SET2→idle))	

Status: invariant ok | no event errors detected



Animation Example: ATM (animation X)

Events

Event	Parameter(s)
▶ insertCard (x2)	C_SET1
◻ init	
◻ ejectCardWithCash	
◻ Transaction	
◻ ejectCardWithoutCash	
▶ offline (x2)	C_SET1
◻ online	
◻ pinEntered	

State

Name	Value	Previous value
★ ATM_A		
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
★ atm_sm	1→idle),(C_SET2→idle)) active),(C_SET2→idle))	
★ ATM_R		
★ active_sm	∅ {C_SET1→dispensing))	
★ idle_sm	e),(C_SET2→available)) {(C_SET2→available))	
C	{C_SET1,C_SET2}	{C_SET1,C_SET2}
★ atm_sm	1→idle),(C_SET2→idle)) active),(C_SET2→idle))	

History

ATM_R	ATM_A
ejectCardWithCash	ejectCard
Transaction	Transaction
pinEntered	

Summary: invariant ok, no event errors detected



Summary

- Developing a useful tool that extends the functionality of other Rodin tools is relatively easy
- State-machine Animation elaborates the visualisation of UML-B diagrams with the aid of ProB Animator
- Animation and verification of Event-B models becomes more obvious

Future Plans

- Maintain Animation tool after the realisation of UML-B refactoring plans
- Consolidate Class diagram animation ideas and integrate them into existing Animation tool



Thank You!

