The Fondue Toolset

Thomas Baar

3rd International KeY-Workshop
Königswinter near Bonn
June 8, 2004

Outline

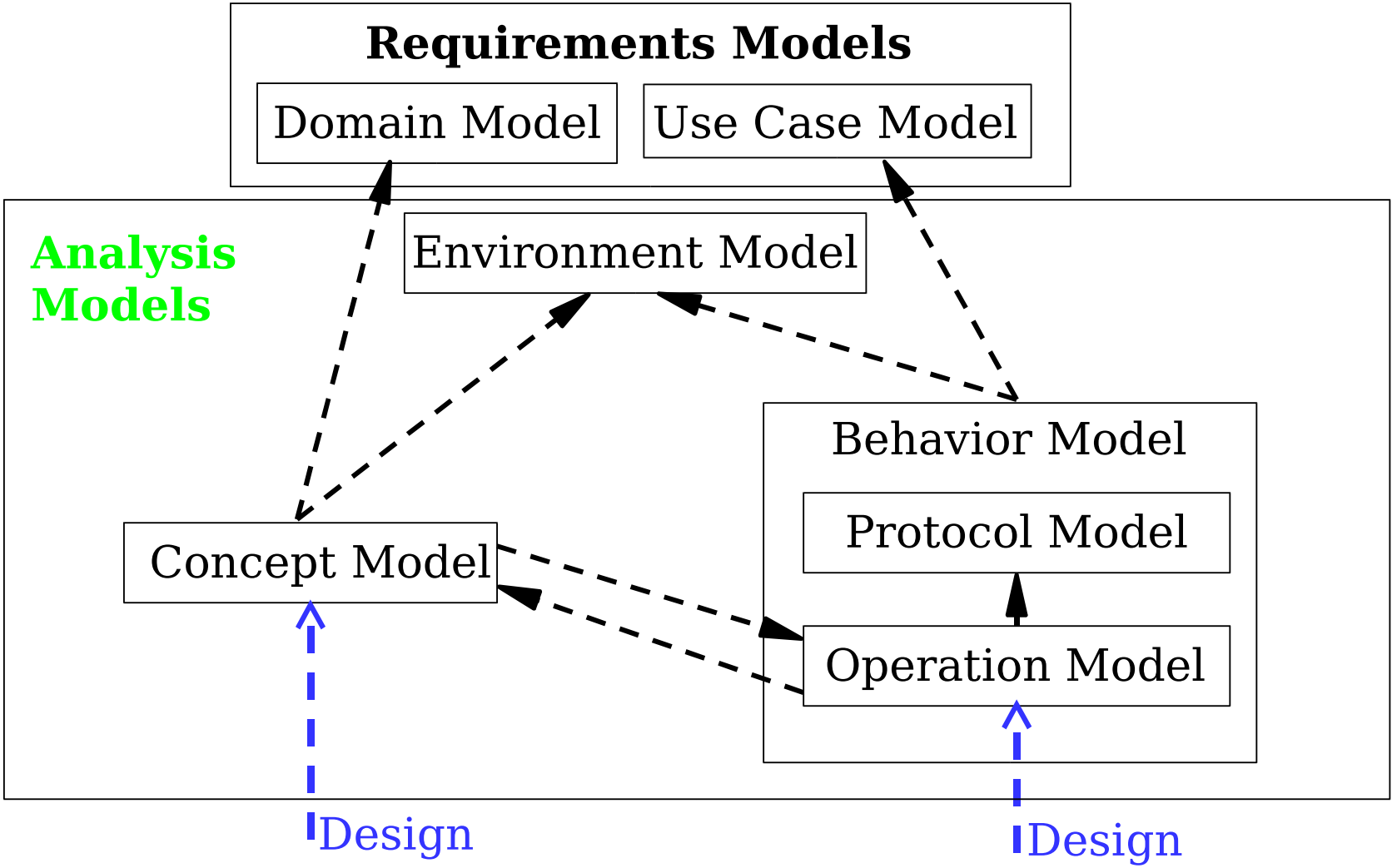
- What is Fondue?
- Toolset
  - Architecture
  - Current State
  - Current Problems

Fondue -- Overview

- Software development methodology inspired by FUSION
- Notation is similar to UML
- Advocates formal specification in Analysis phase
- Applied by students in projects
- Tested by developers of commercial software

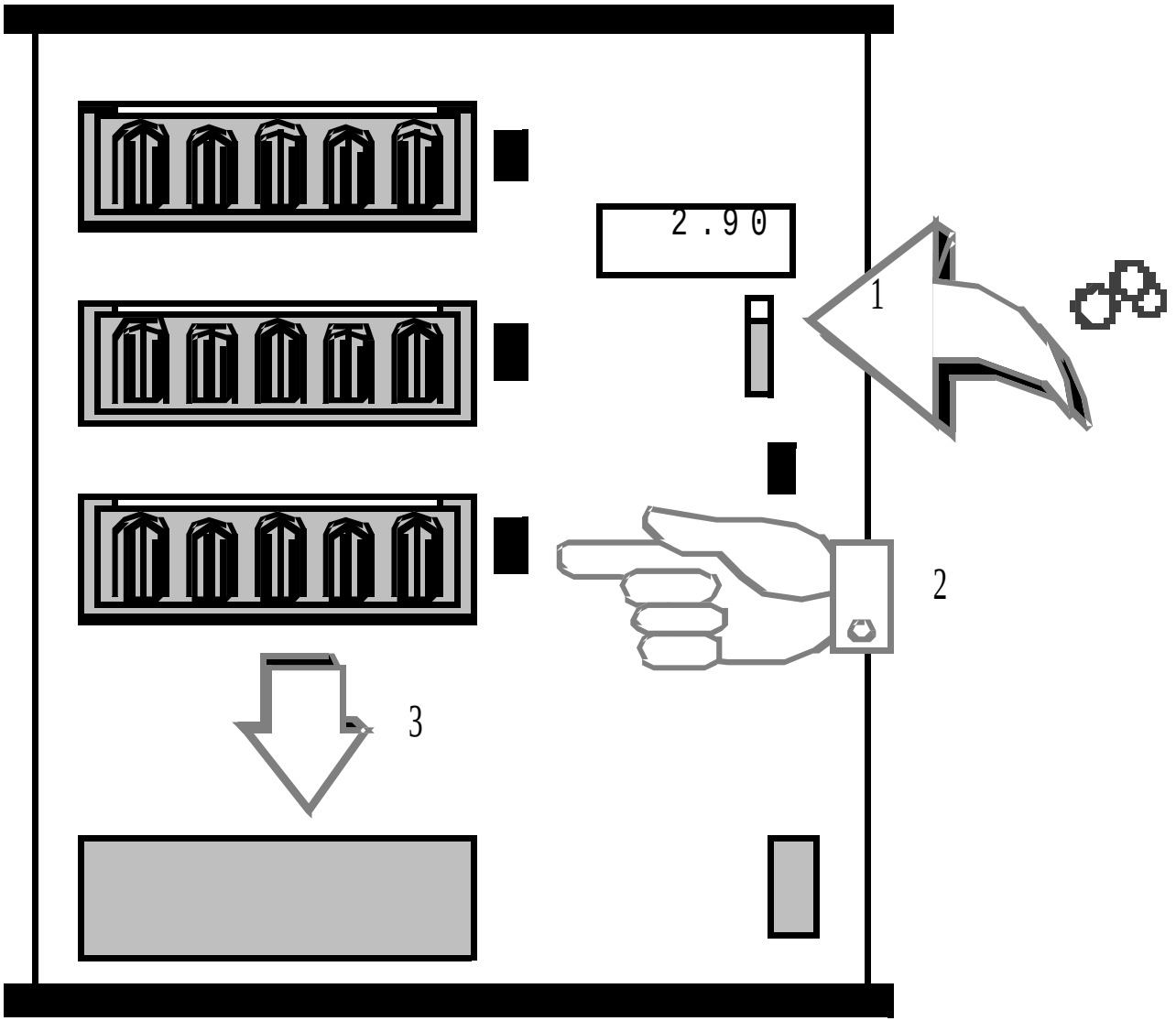
Toolset: Concentrates on Analysis phase

Fondue -- Models

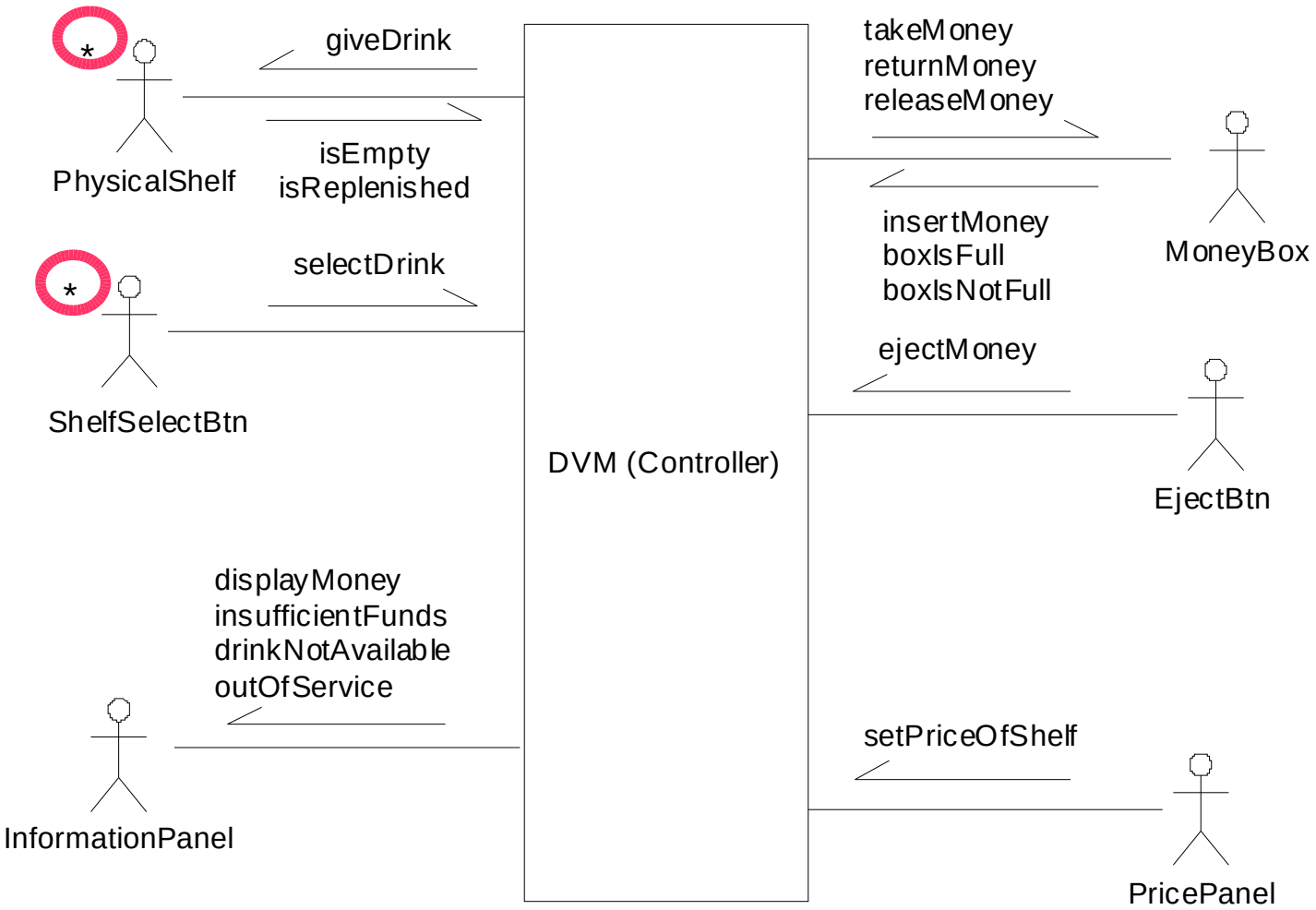


A - - - → B A depends on B: a change in B induces a change in A

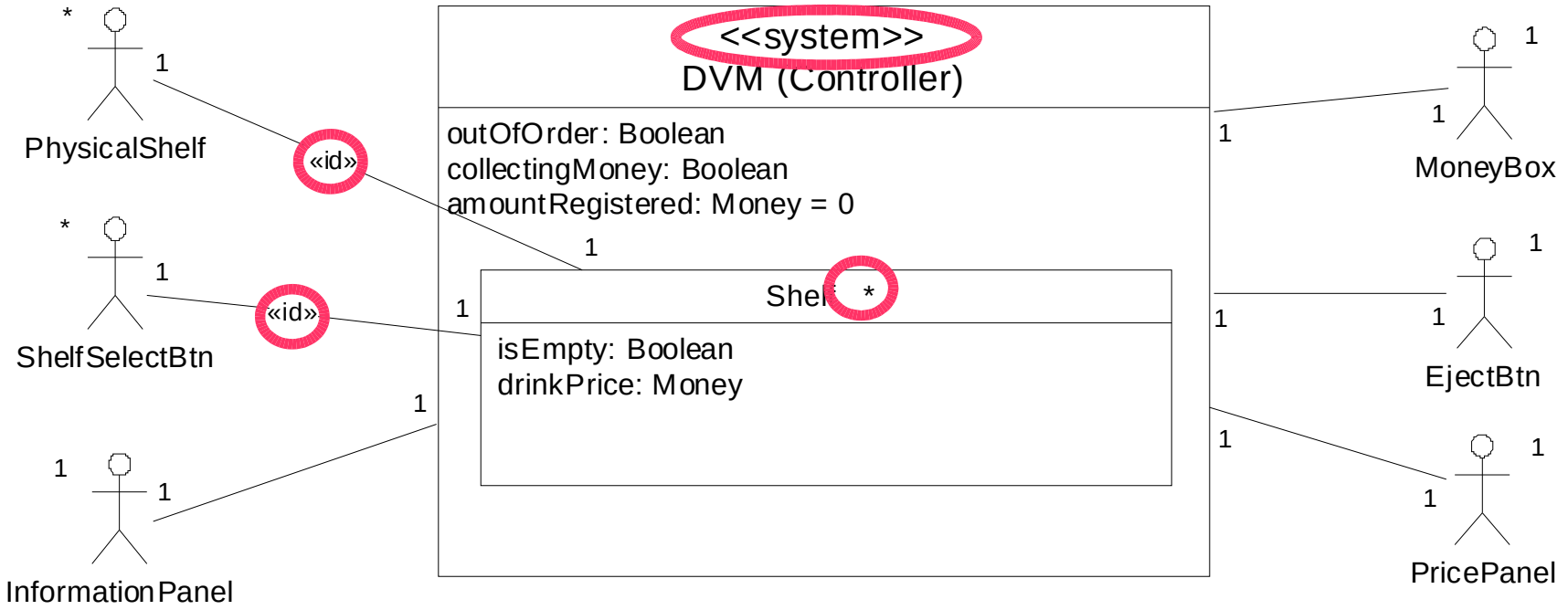
Example - DrinkVendingMachine



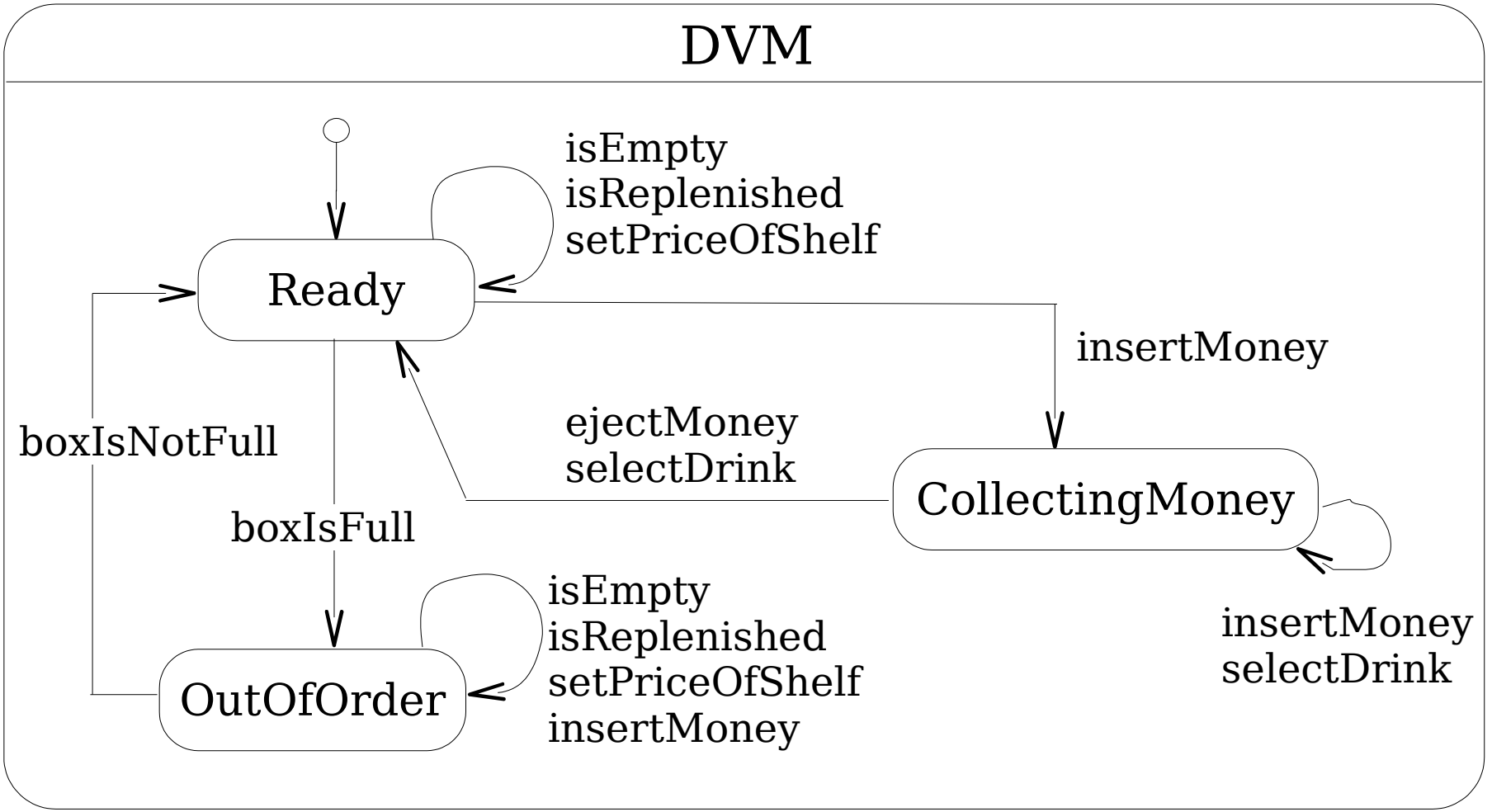
DVM – Environment Model



DVM – Concept Model



DVM -- ProtocolModel



DVM -- OperationModel

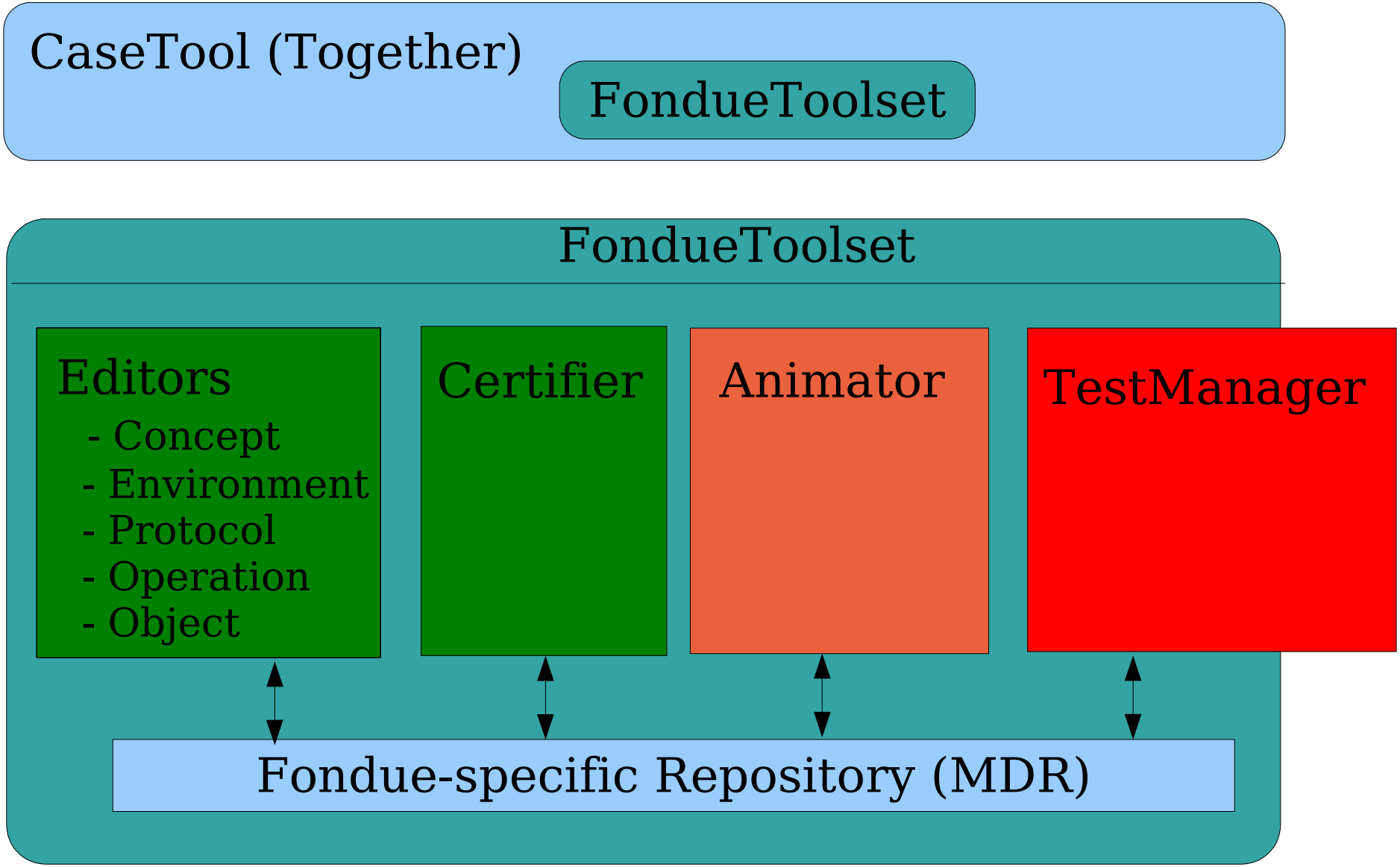
Operation: DVM::ejectMoney()
UseCases: buy drink
Messages: InformationPanel::{DisplayMoney, InsufficientFunds, DrinkNotAvailable}
MoneyBox::{ReleaseMoney}
Pre:
Post:

```
self.amountRegistered = 0 and
self.moneyBox^releaseMoney() and
self.informationPanel^displayMoney(0) and
self.informationPanel^insufficientFunds(false) and
self.informationPanel^drinkNotAvailable(false)
```

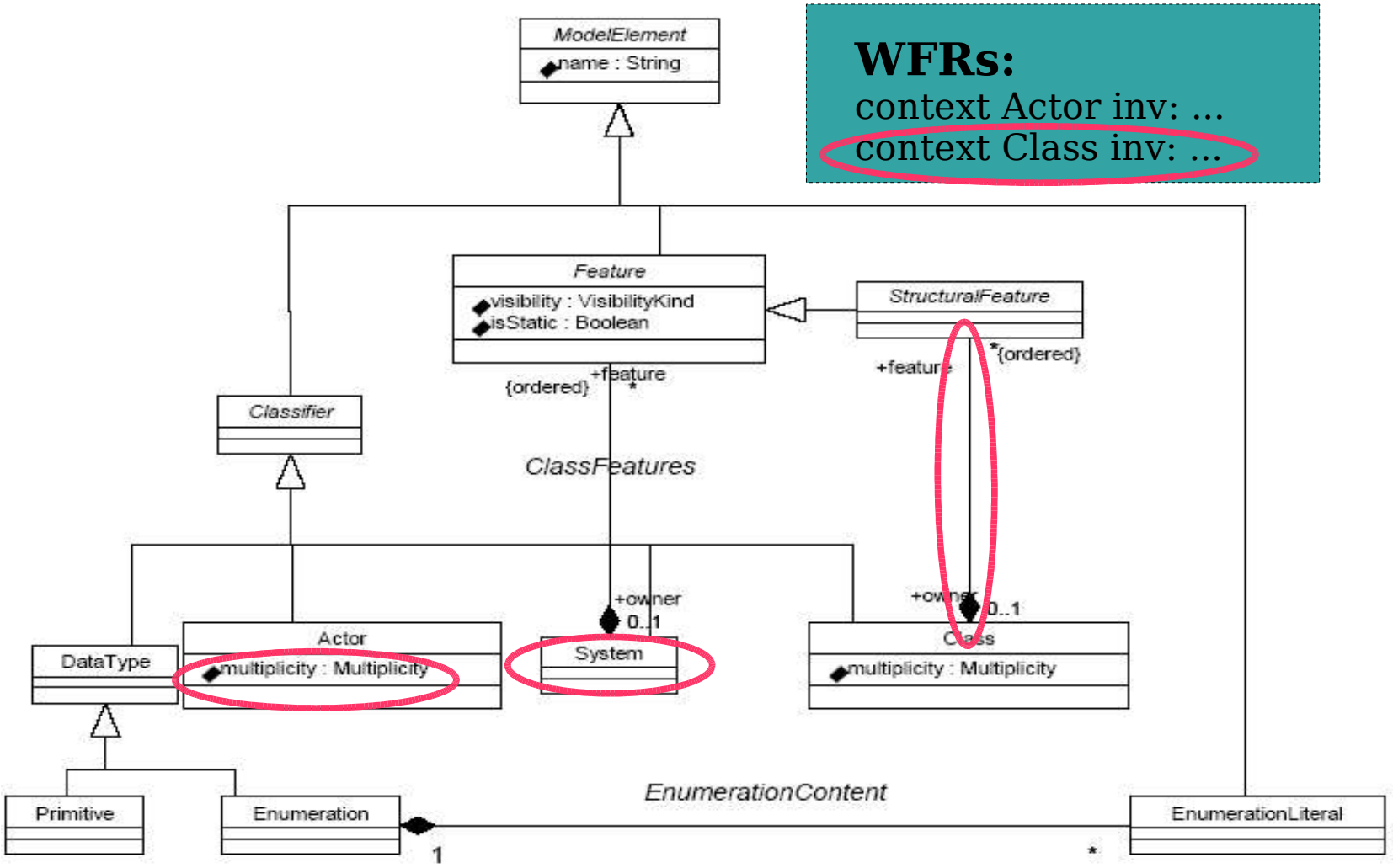
Fondue Toolset -- Overview

- Support for Fondue-Notation
  - Editors for Concept-, Environment-, Protocol-Model (ensure strict compliance to Fondue-MM)
  - Cross-check for syntactical correctness (Certifier)
  - Editor for Operation-Model (OCL)
- Animation of Specification
  - Display of system state (object diagram)
- (Test Case Generation)

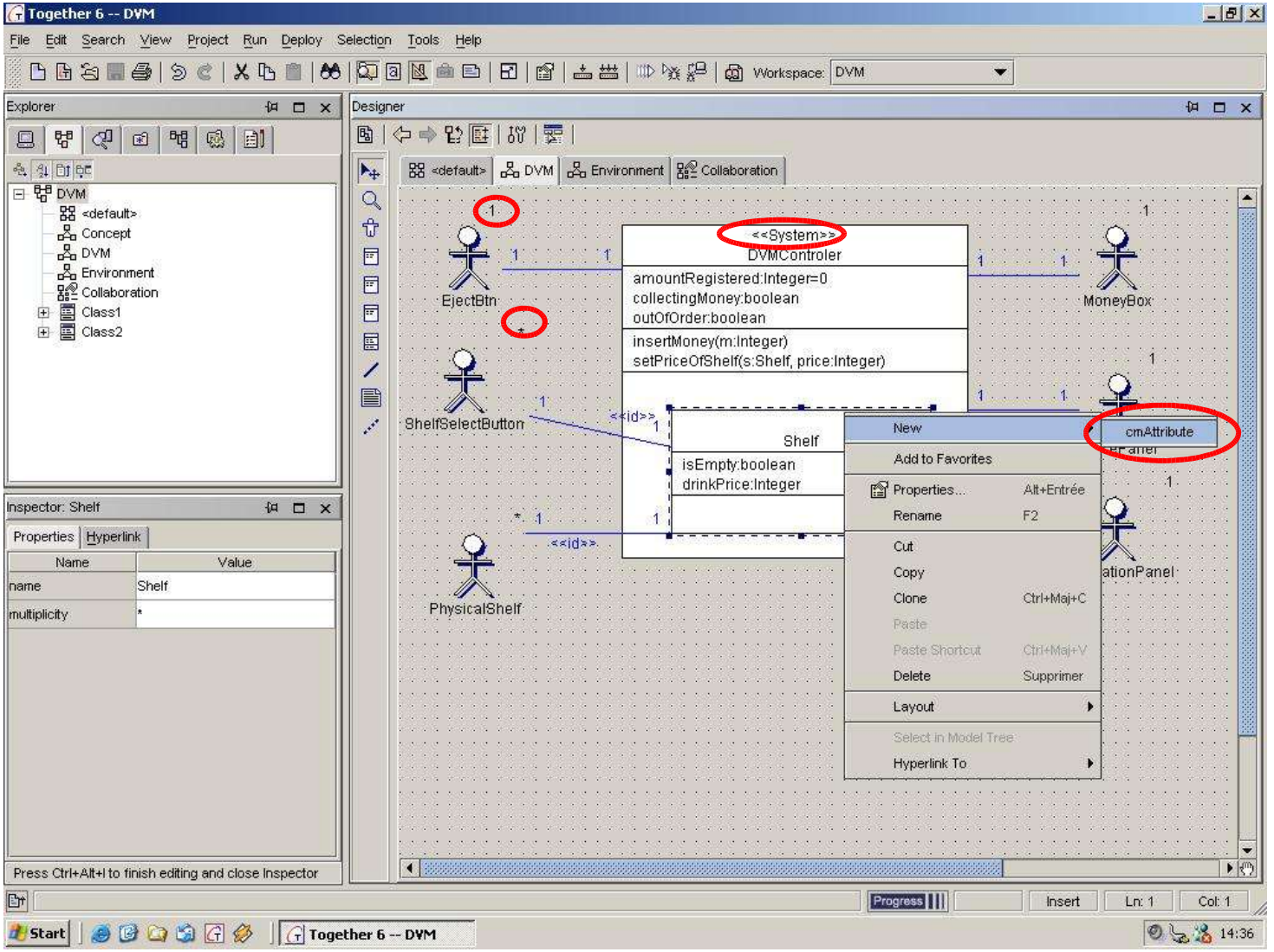
Fondue Toolset -- Architecture



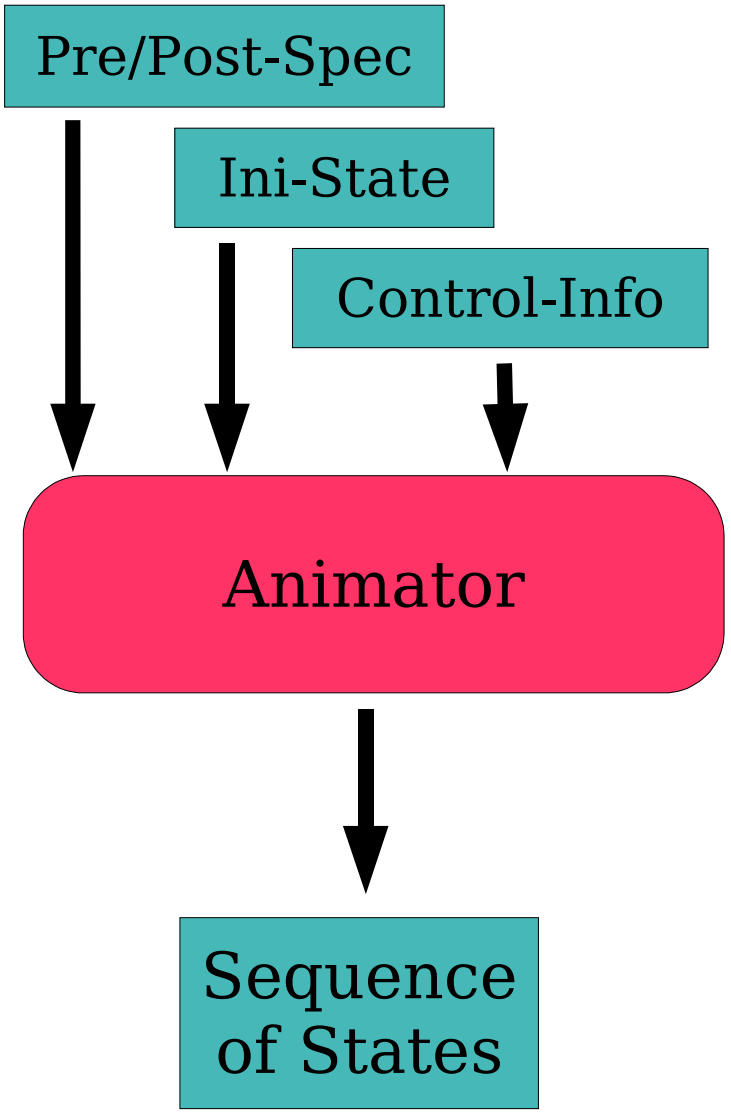
Notation – Defined by Metamodel



Notation – Implemented by Editor



Animator

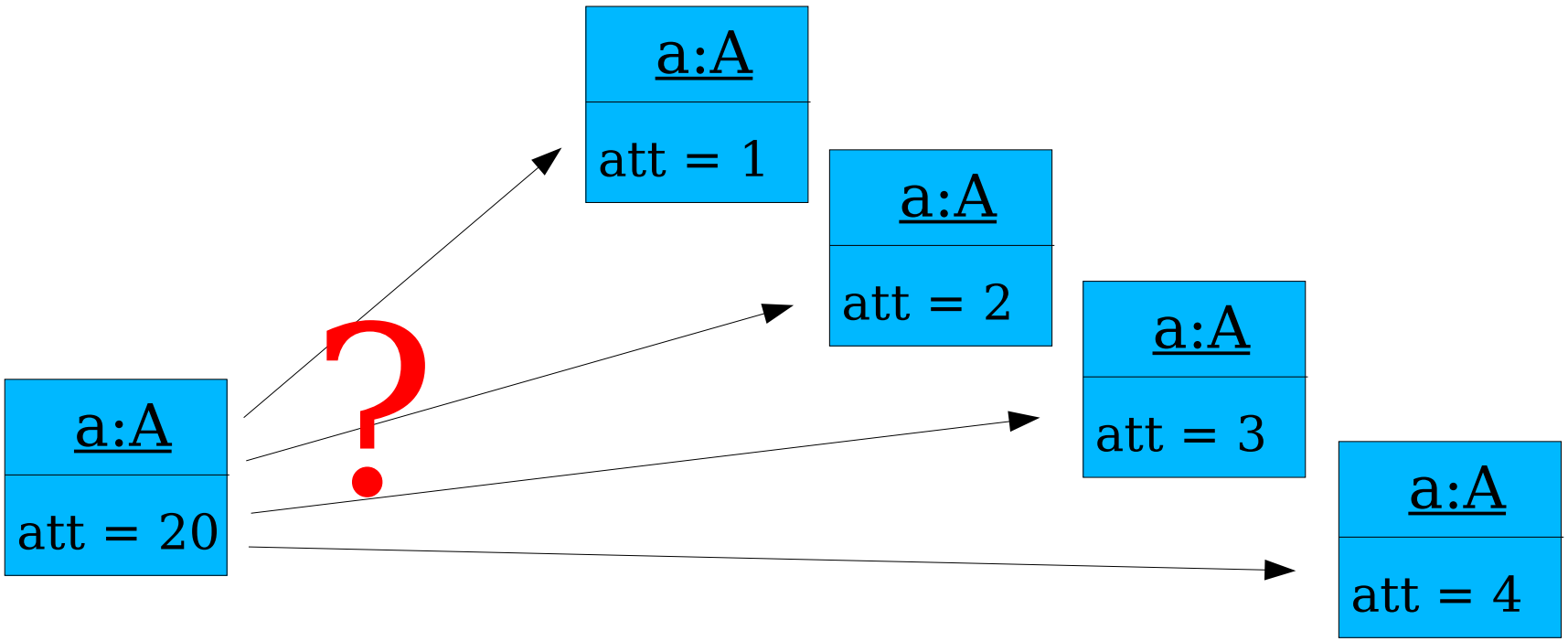


Purpose:

- Satisfiability-Check
- Reality-Check
- Implementation of UML as High-level Programming Language (Executable UML)

Animator – Non-Determinism

```
context A:foo()
    pre:  self.att > 0
    post: self.att > 0 and self.att < 5
```



Non-Deterministic Animation-- Solution I

Forbid non-deterministic specifications

- Post-state specifications must have only one solution for given pre-state
- Specification style is (mainly) adopted in B (mainly: non-deterministic specification are made explicit by usage of non-deterministic constructs)
- OCL-dialect OCLScript attempts something similar

```
context A:foo()
   pre:  self.att > 0
   post: self.att > 0 and self.att < 5
```

```
context A:foo()
   pre:  self.att > 0
   post: self.att = 2
```

Non-Deterministic Animation-- Solution II

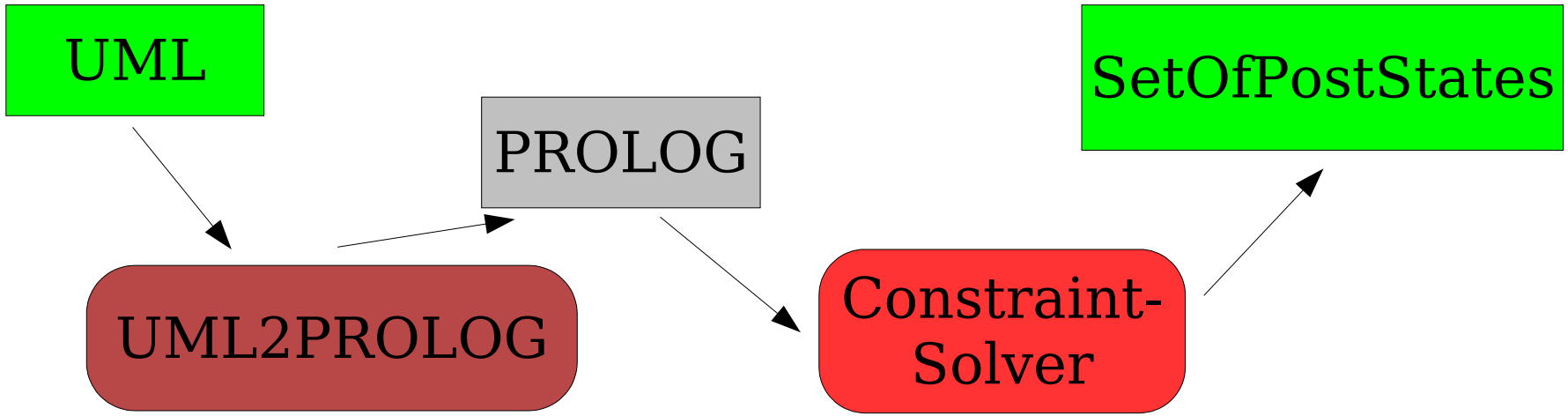
Choose post-state among current possibilities

1) Compute all solutions for post-state spec

2) Next state is chosen (by user or automatically)

   i. Choose-operation is backtrackable

Approach of B.Legeard in BZTT:



Summary

Results

- Concept-,Environment-Editor compliant to MM
- Fondue-specific version of OCL-Editor
- Fondue-specific repository
- Check of well-formedness rules with certifier
- Stand-alone object editor

Summary

Future work

- Integration (ObjectEditor, import, OCLE, ...)
- Refined Metamodel (including layout information)
- Parsing of OCL (requires Fondue2UML preprocessing)
- Animation
  - Translation into format for constraint solver
  - Adaptation of standard-solver ???
  - Synchronization with Protocol model
  - Front-end for refactoring application