# An EMF Framework for Event-B

Colin Snook - Soton,

Fabian Fritz - HHU,

Alexei Illiasov - Newcastle

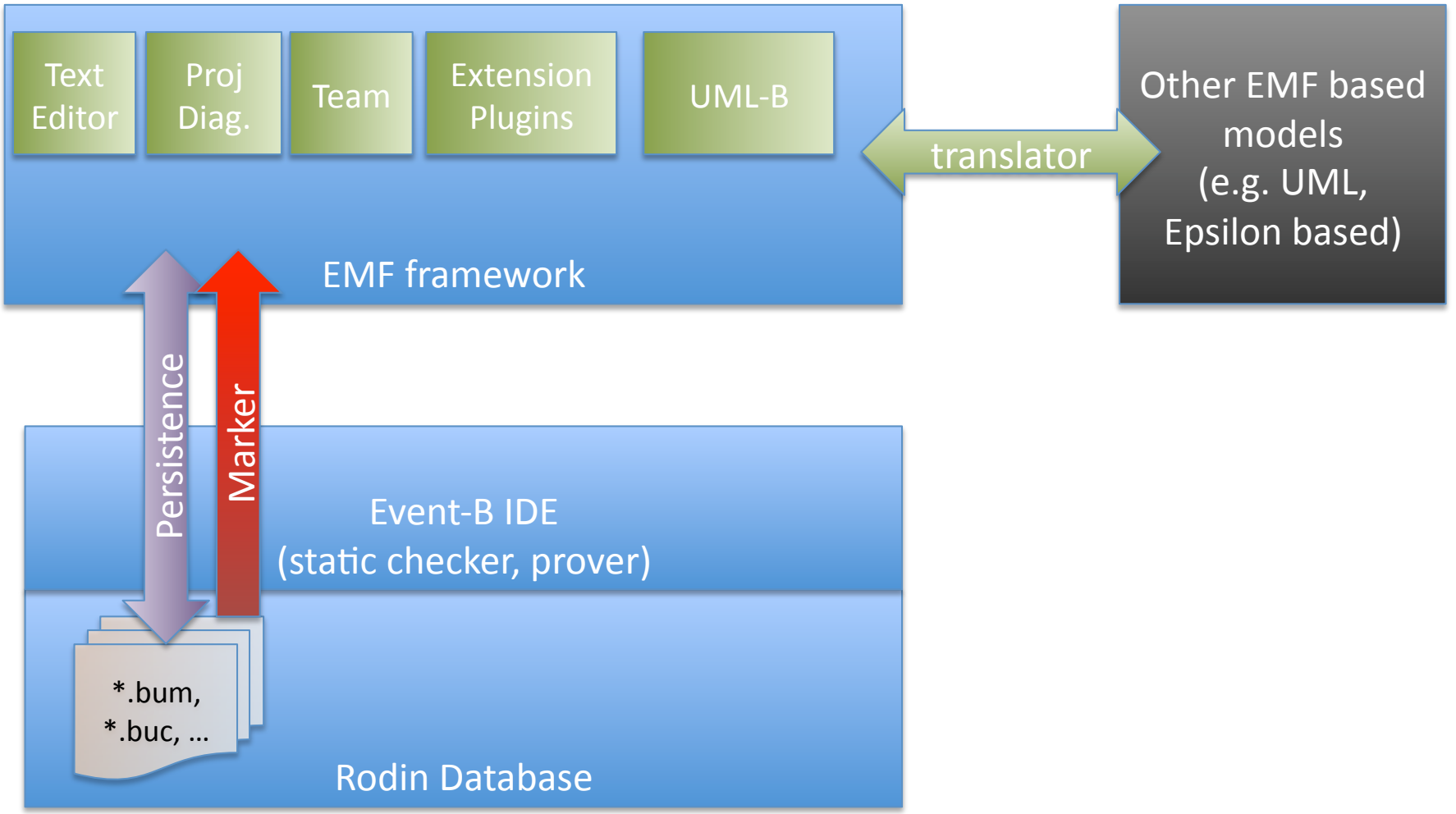
# EMF

- Eclipse Modelling Framework
- Meta-modelling notation (abstract syntax)
- Code Generator
  - Model repository (database)
  - Editor facilities
- Support for developers:
  - Command framework
  - Persistence API
  - Dynamic EMF
- Lots of other tool support
  - Editors – text, graphical, mixed
  - Translators – model transformation

# Background Work

- UML-B
  - EMF based model repository
  - GMF to generate diagram code
  - Translator to generate Event-B projects
- Alexei's Text Editor
  - Used EMF with customised persistence
  - Used TEF to generate prototype text editor

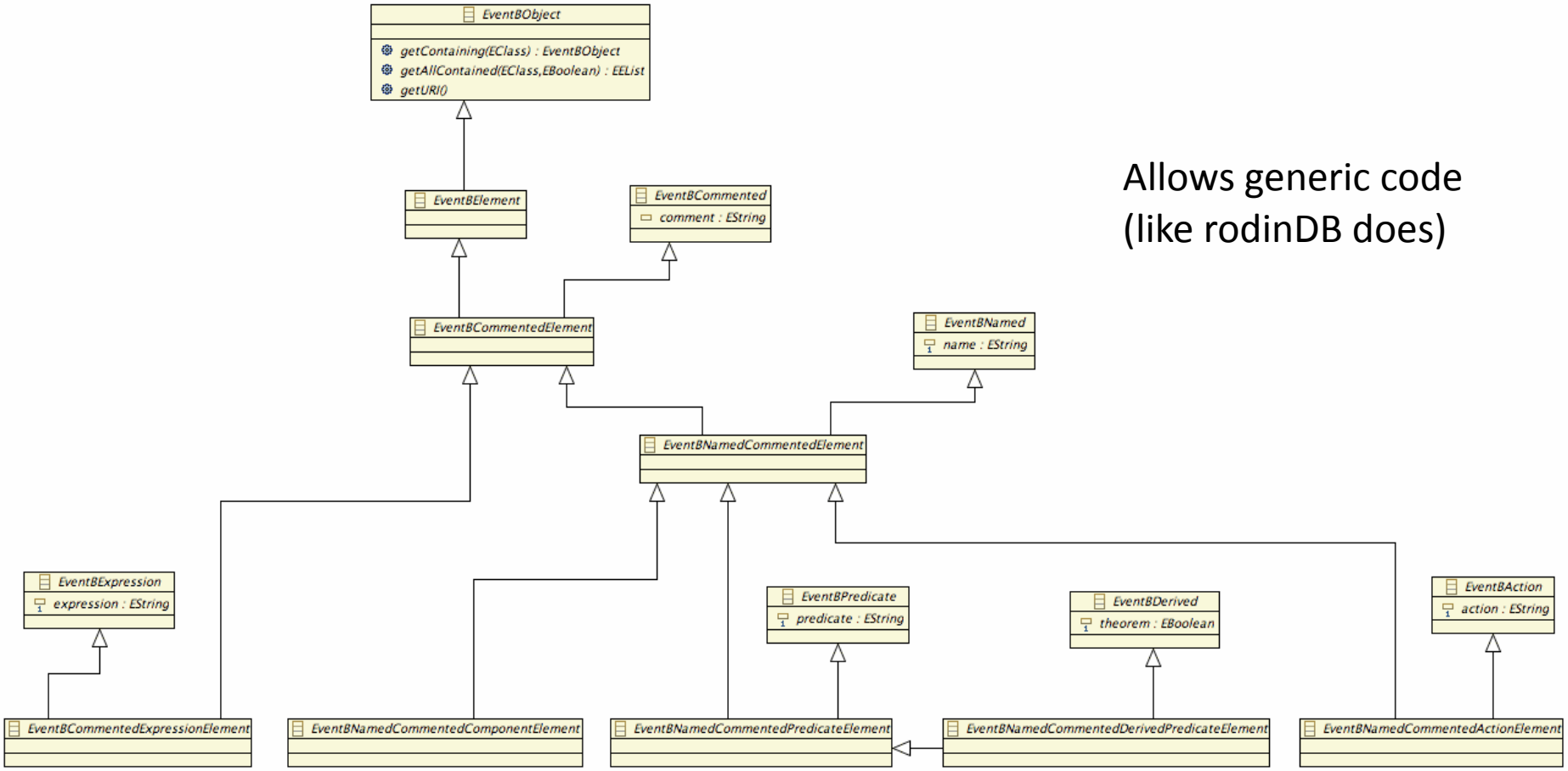
# Front-End Approach



# Event-B Metamodel

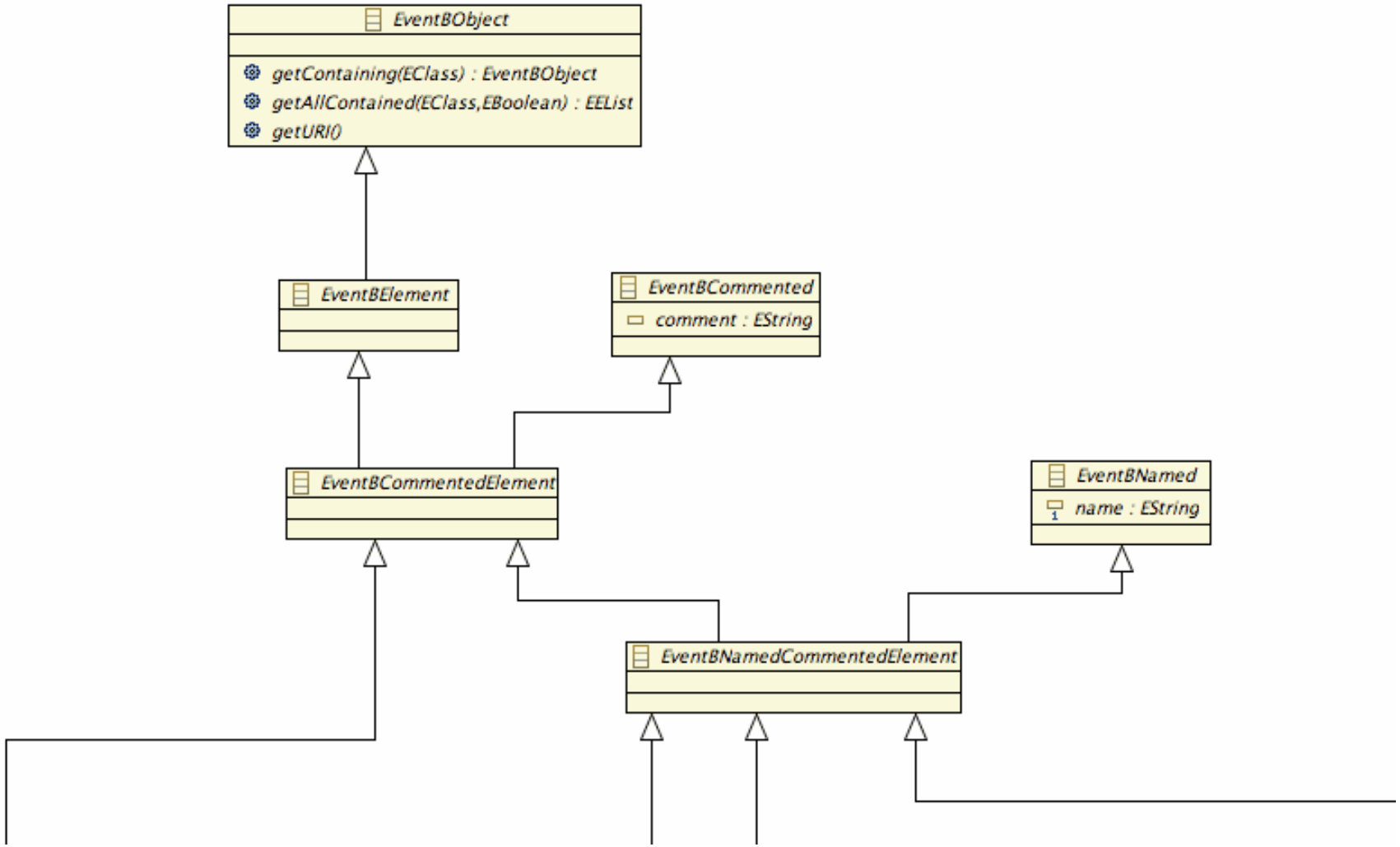
- Core package
  - Abstract basis
  - Extension mechanism
  - Project
- Machine package
- Context package

# Abstract Core

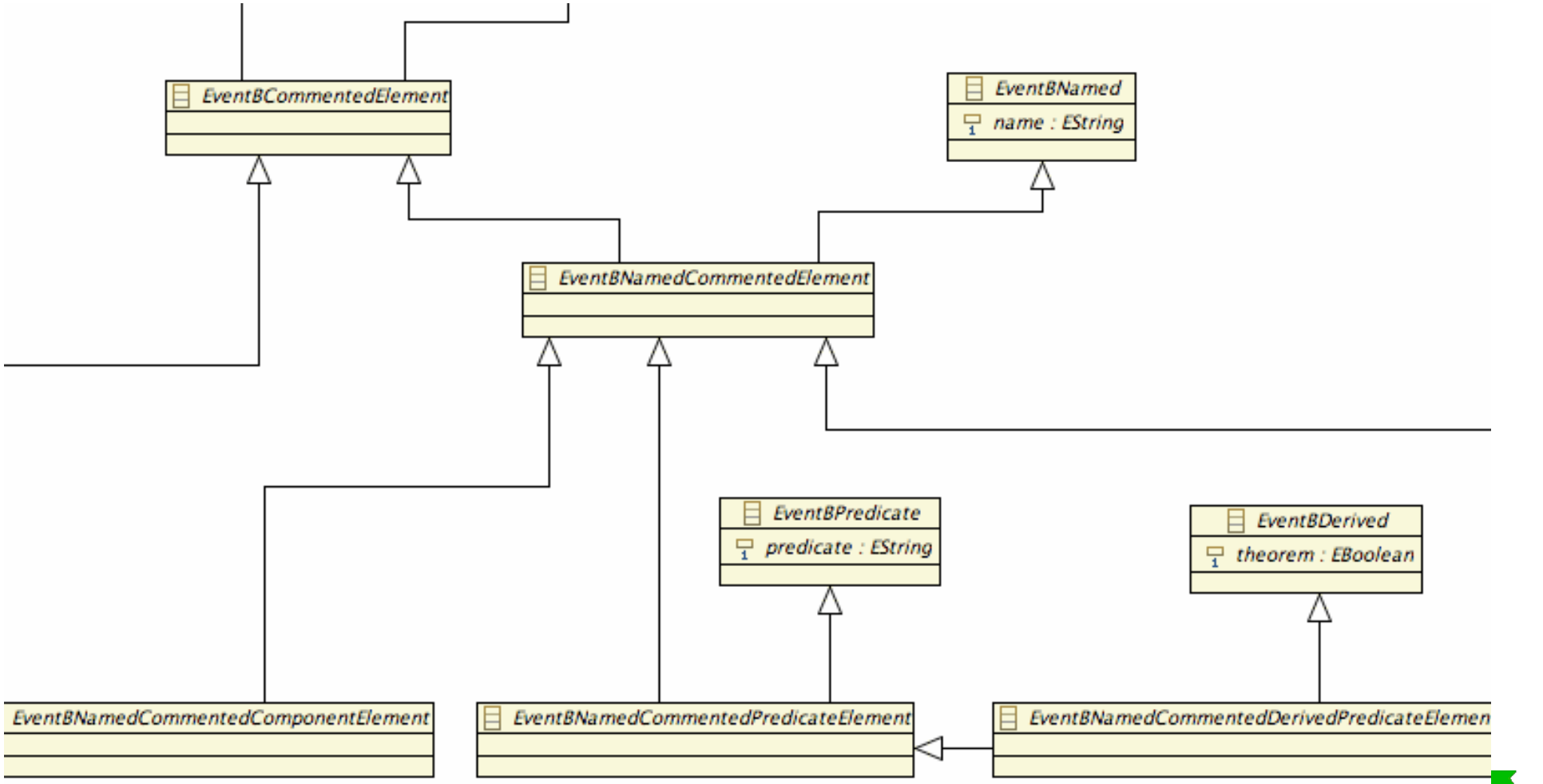



Allows generic code
(like rodinDB does)

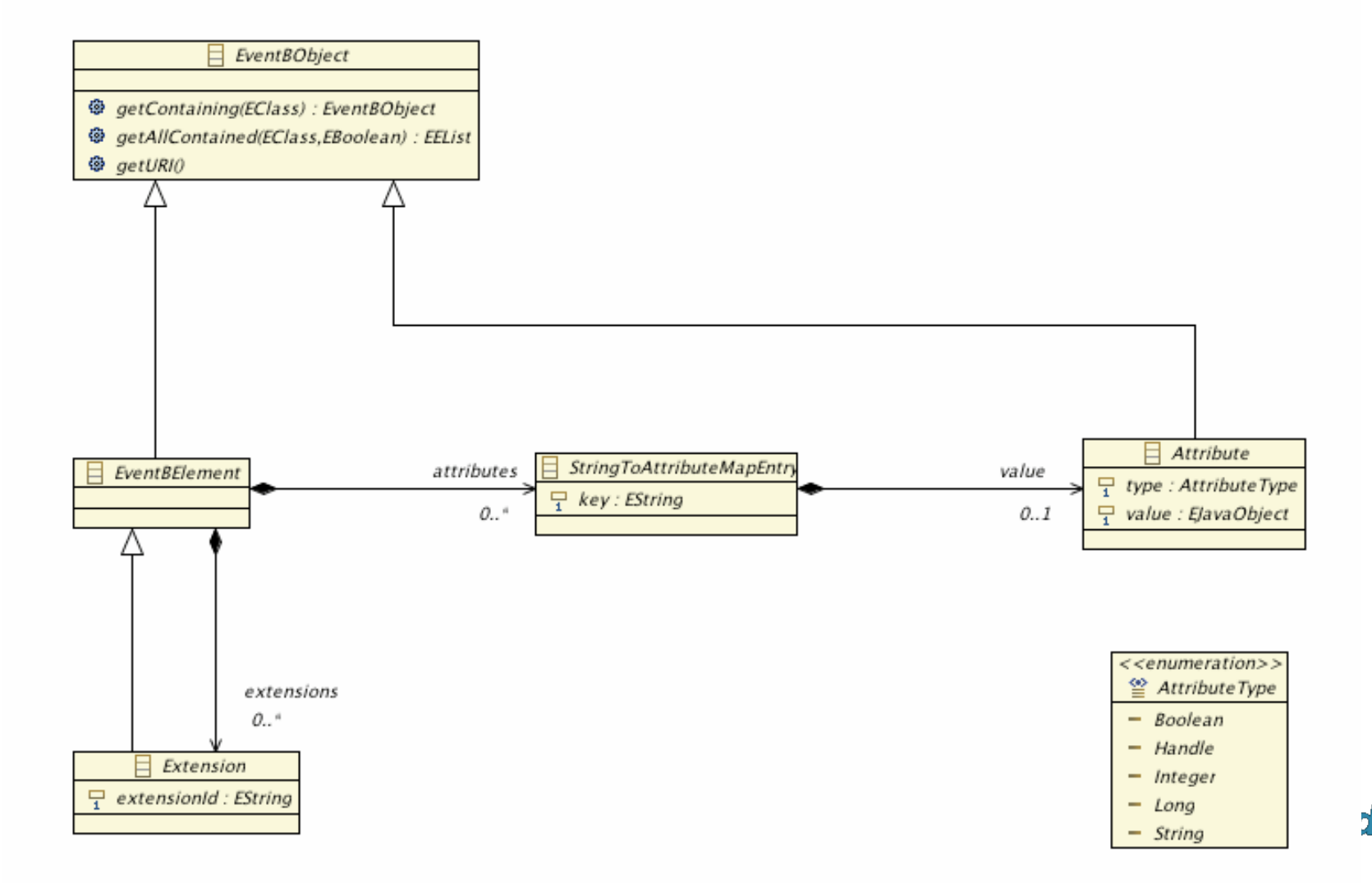
# Abstract Core



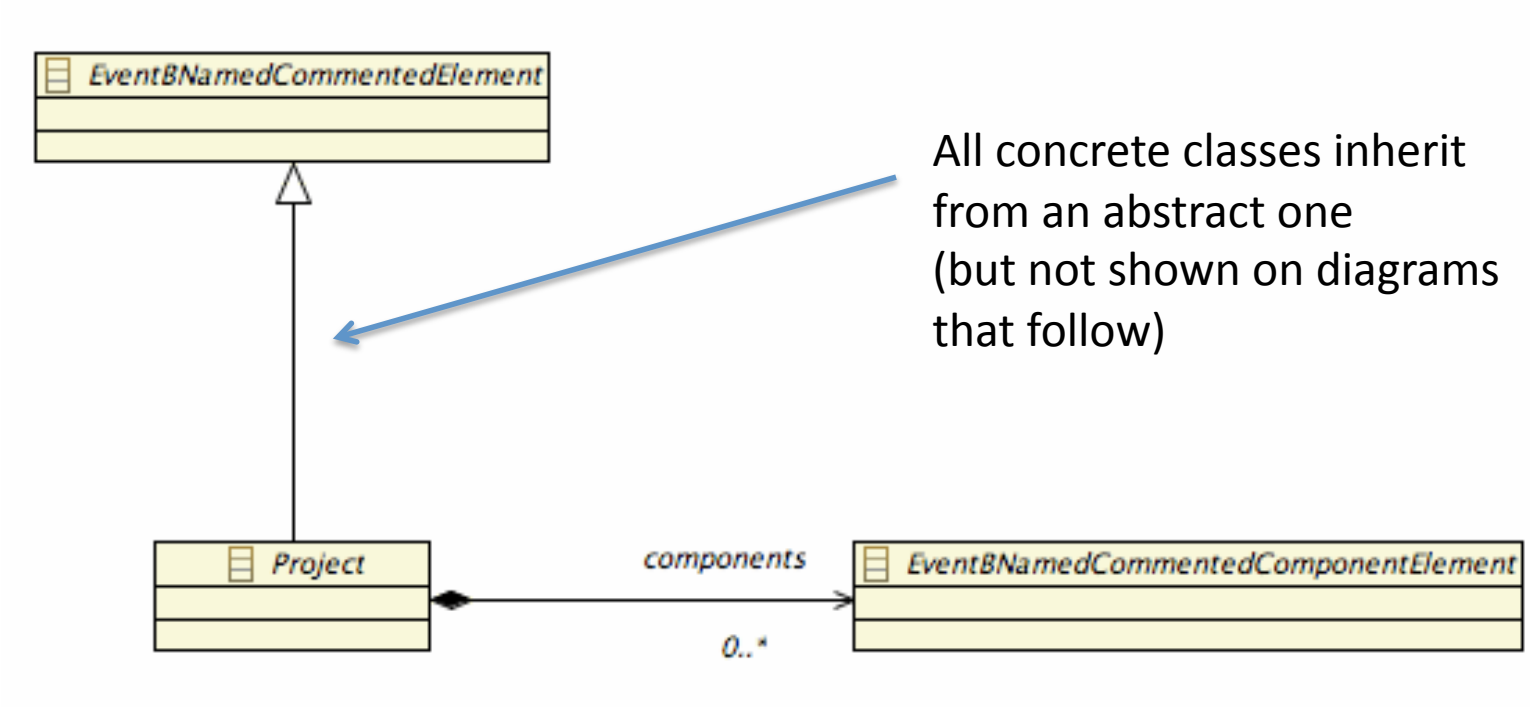
# Abstract Core



# Extension Mechanism

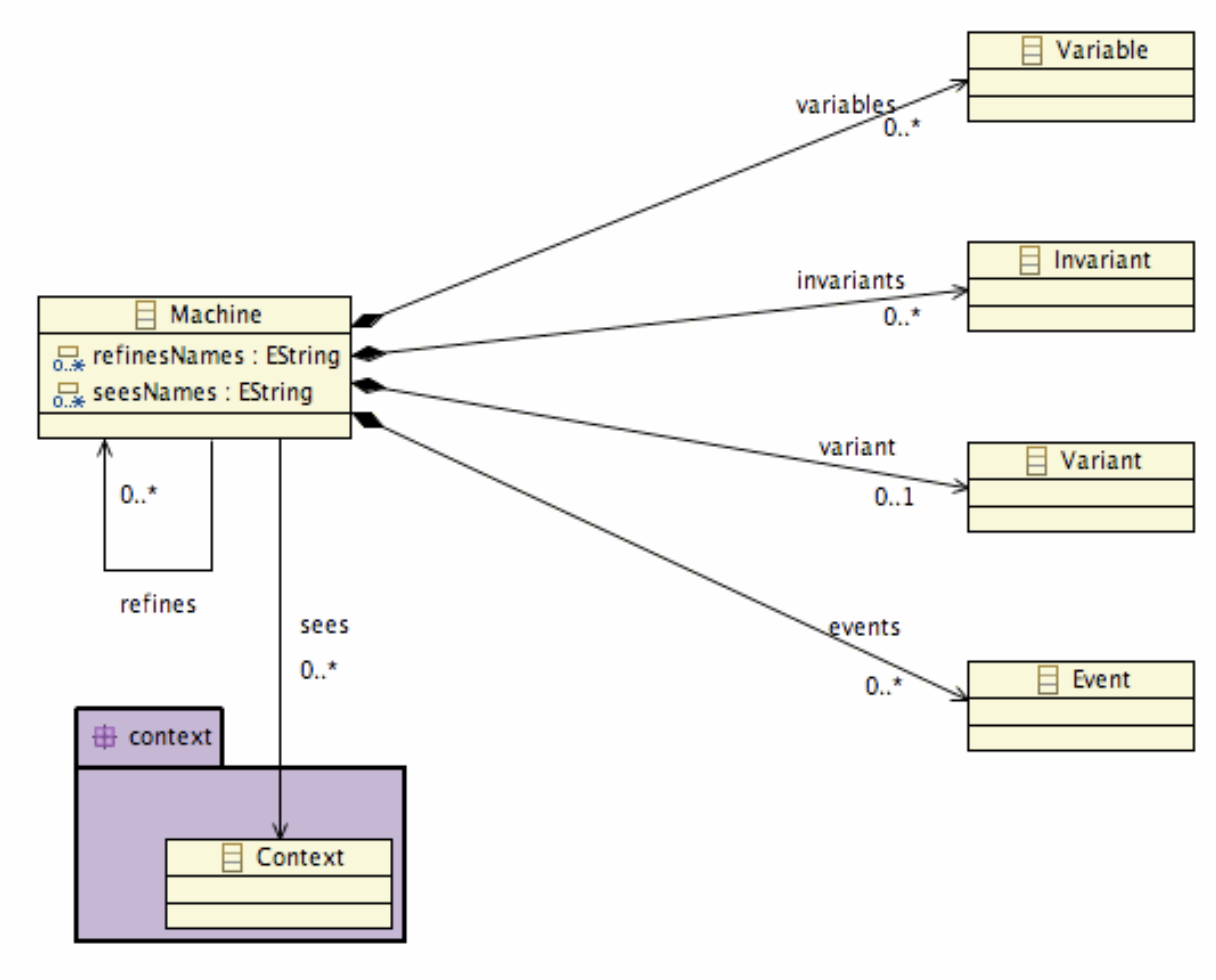


# Project

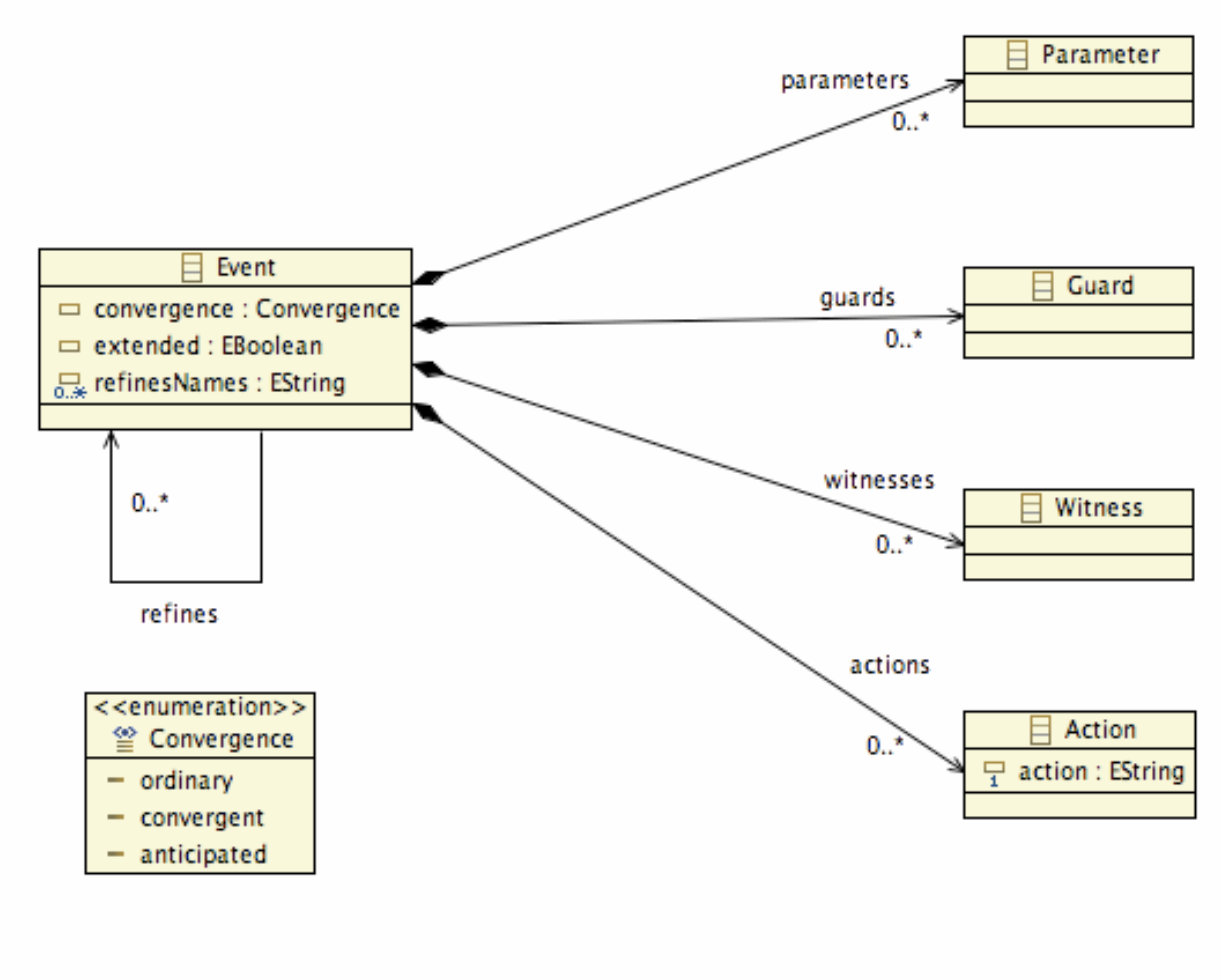



All concrete classes inherit from an abstract one (but not shown on diagrams that follow)

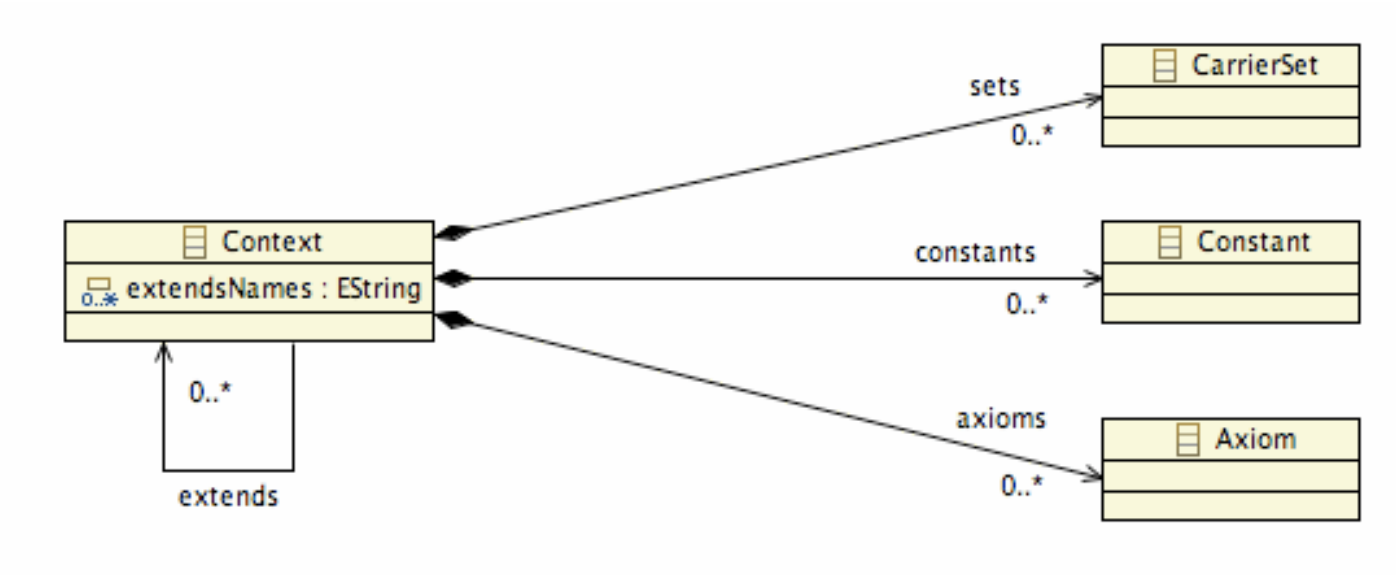
# Machine


Variable
variables
0..*
Invariant
invariants
0..*
Machine
refinesNames : EString
seesNames : EString
0..*
refines
sees
0..*
Variant
variant
0..1
events
0..*
Event
context
Context

# Machine (Events)



# Context


CarrierSet
sets
0..*
Context
extendsNames : EString
0..*
constants
Constant
0..*
0..*
extends
axioms
Axiom
0..*

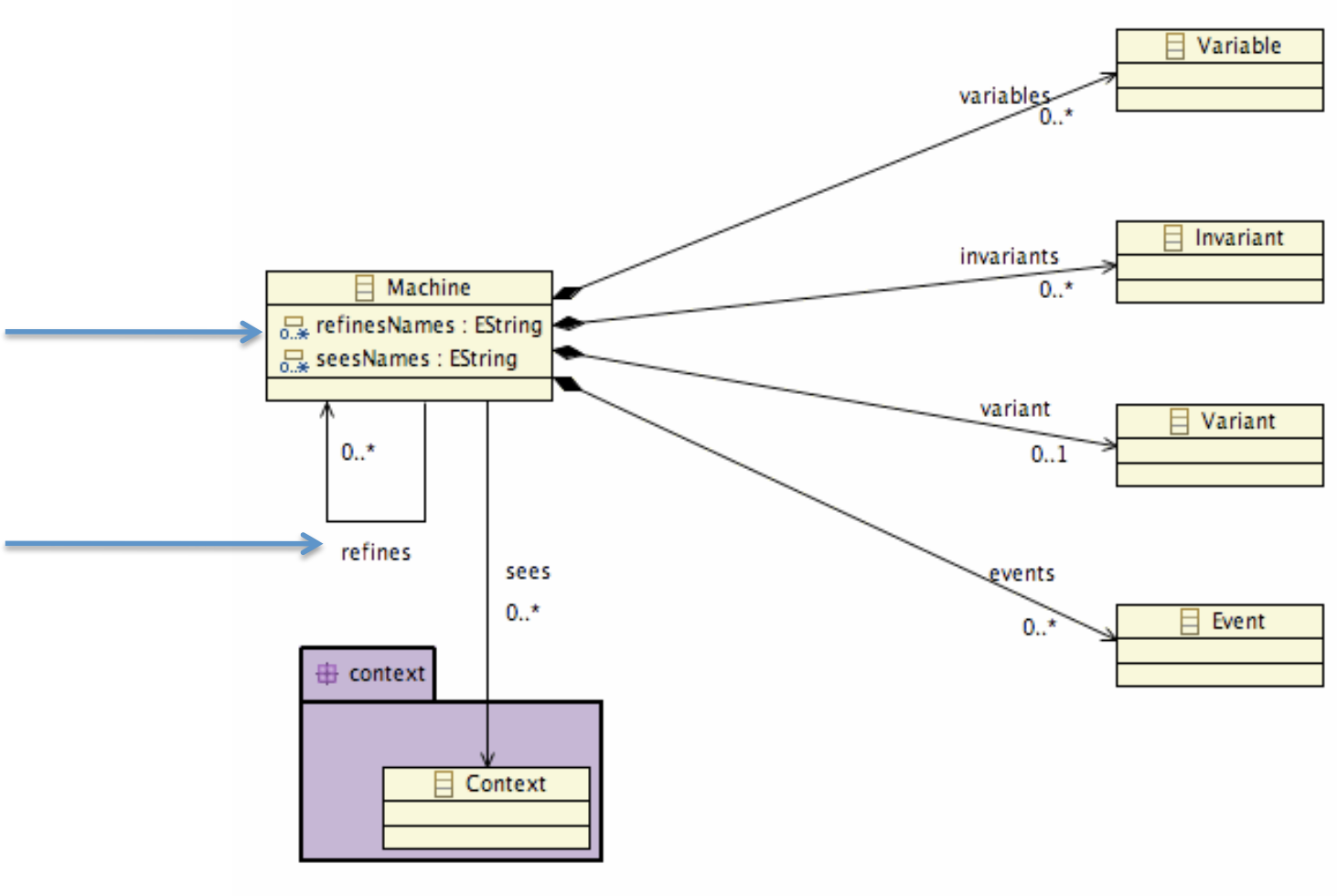
# Inter-Resource References

- Refines Machine, Refines Event, Sees Context
- Some tools work on multiple resources
  - Need direct model references
  - EMF proxy facilities for resolving/loading when needed
- Some tools work on a single resource
  - Don't want to load referenced resources
  - Leave references to other resources unresolved
  - Often they  are un-resolvable (i.e. do not exist)

# Solution – Dual Representation

- List of References (EMF proxies)
  - Use Lazy proxy construction
  - URI fragment = reference name (persisted)
  - Automatically construct rest of URI …
    - Project/resource from container component
  - … but not until resolve attempt
- List of Names
  - Transient (no storage)
  - Derived from proxy fragments (by getter)
  - Can be edited .. Notifies parent …
  - .. Proxy fragments kept in step (even if not resolvable)

# Machine


Variable
variables
0..*
Invariant
invariants
0..*
Machine
refinesNames : EString
seesNames : EString
0..*
refines
sees
0..*
variant
0..1
Variant
events
0..*
Event
context
Context

# Persistence

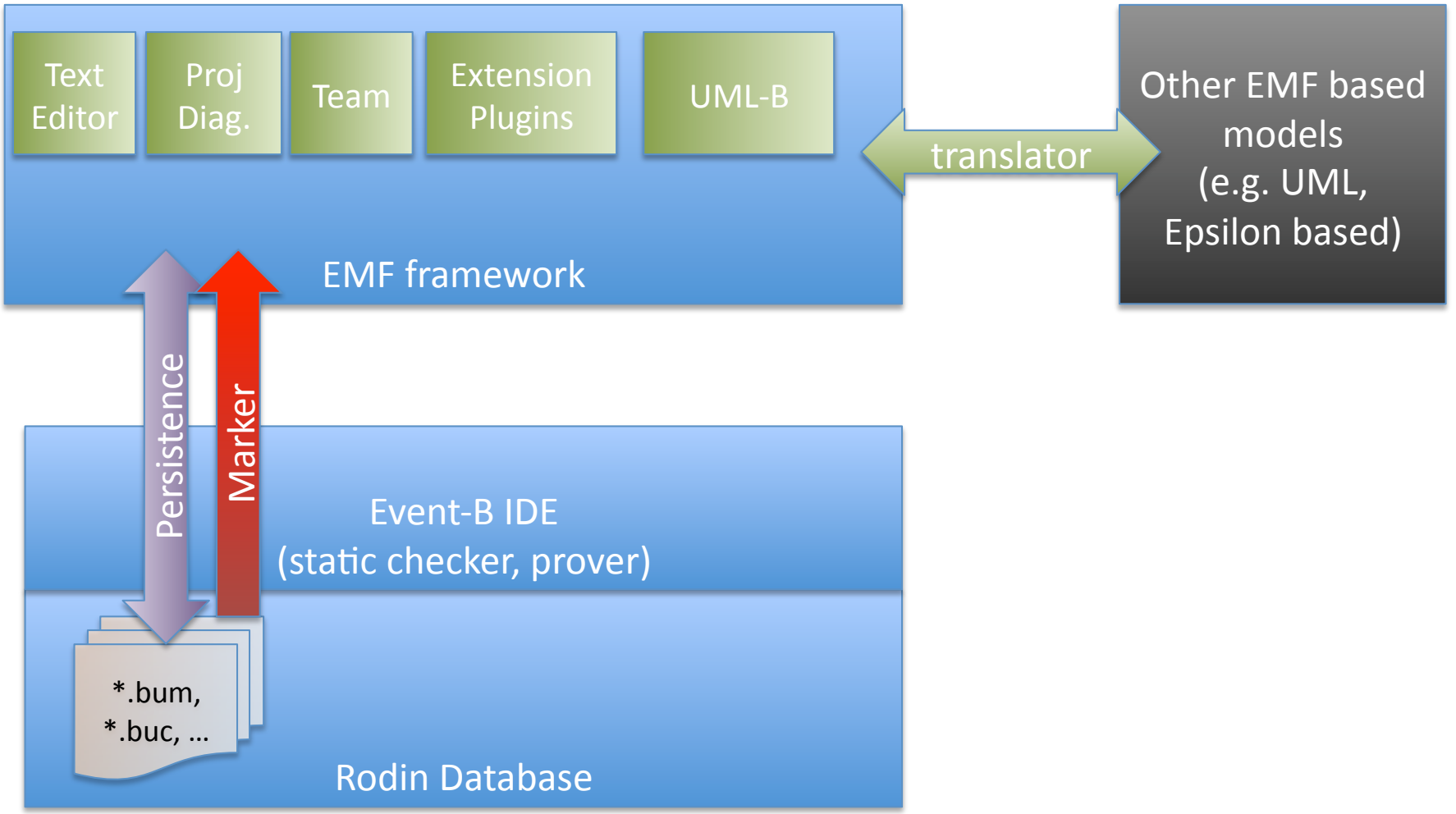
- Overrides EMF default XMI persistence
- Load and Save into Rodin DB via API
- Synchronisers for each element type
  - Registered via extension point
  - Allows for new elements to be defined by plugins
  - Volatile extensions (no synchroniser)
- Attributes
  - Can be Dealt with explicitly in Synchroniser… or
  - Left to Generic Attribute handler

# Interfaces for Synchronisers

*ISynchroniser.java

```
package org.eventb.emf.persistence;

import org.eclipse.core.runtime.IProgressMonitor;

public interface ISynchroniser {

    public <T extends EventBElement> EventBElement load
            (IInternalElement rodinElement, EventBElement emfParent, final IProgressMonitor monitor)
            throws RodinDBException;

    public IInternalElement save
            (EventBElement emfElement, IRodinElement rodinParent, final IProgressMonitor monitor)
            throws RodinDBException;

}
```

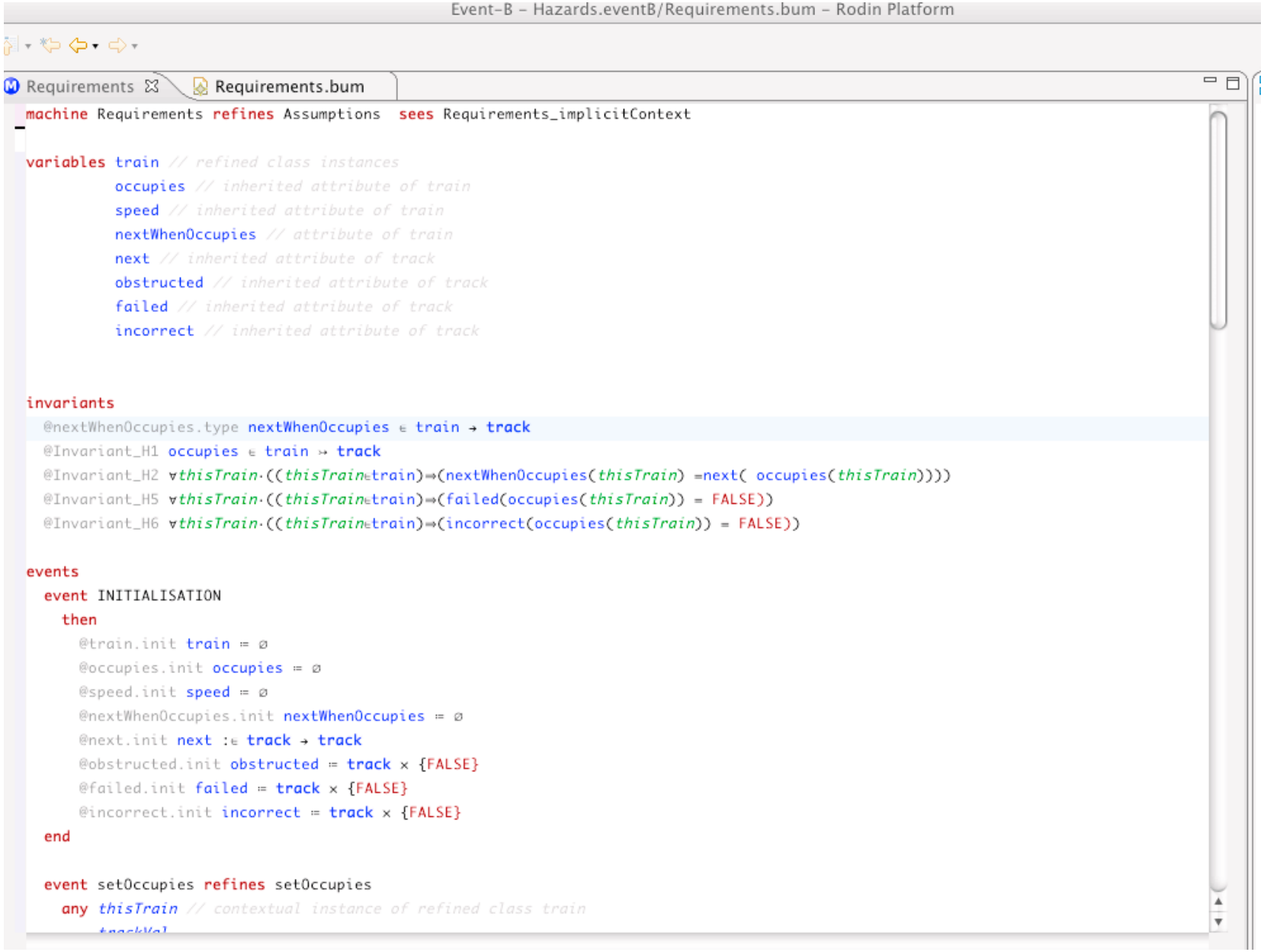
# Hence - Front-End Approach



# Current Users

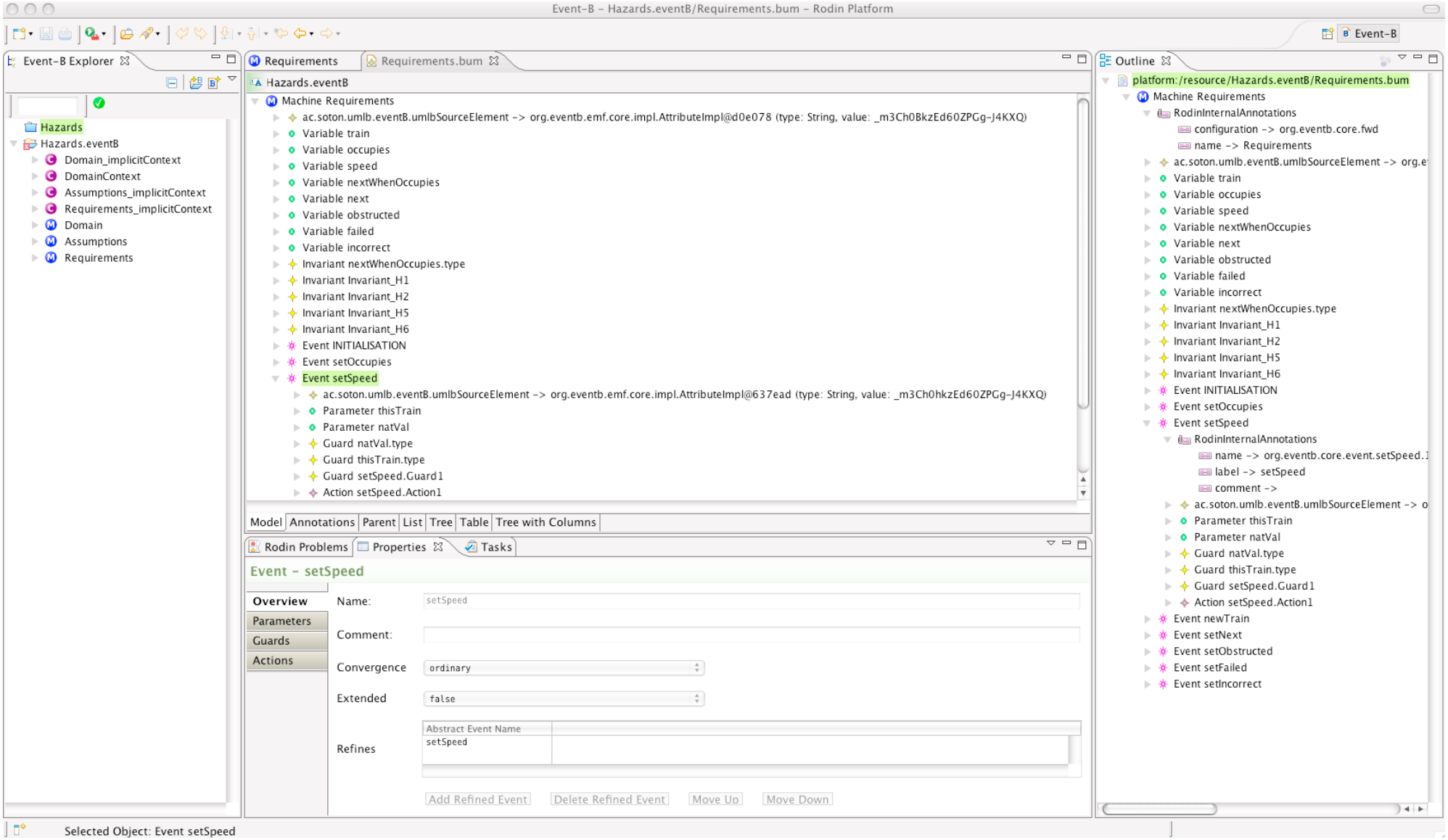
- Text Editor
  - Released by Fabian Fritz at HHU
  - Uses EMF model with text tools extensions
  - BTW - Uses EMF's compare/merge framework programmatically for local synchronisation of changes (before persistence)

# Text Editor

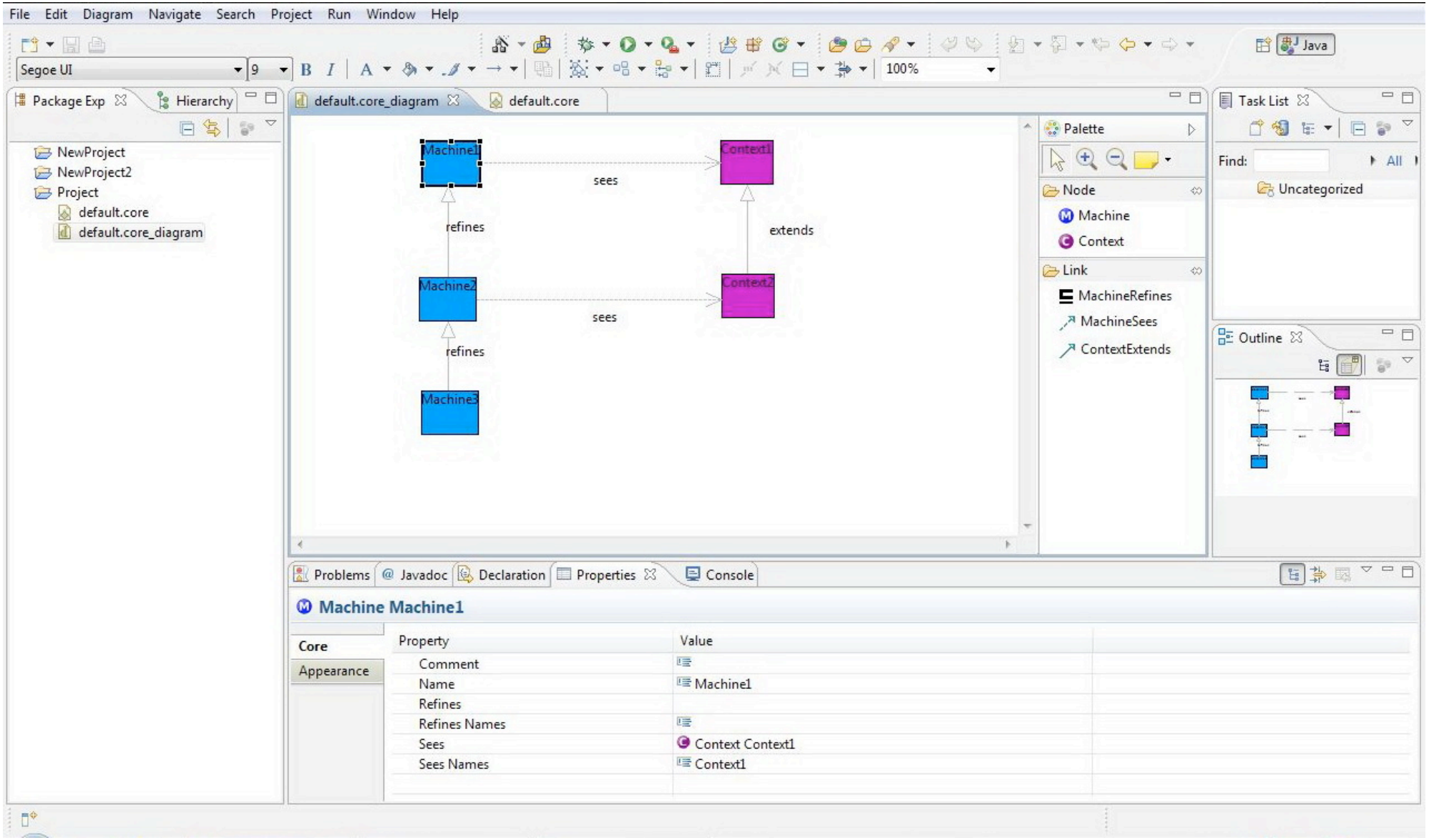

Event-B – Hazards.eventB/Requirements.bum – Rodin Platform
Requirements
Requirements.bum
machine Requirements refines Assumptions sees Requirements_implicitContext
variables train // refined class instances
occupies // inherited attribute of train
speed // inherited attribute of train
nextWhenOccupies // attribute of train
next // inherited attribute of track
obstructed // inherited attribute of track
failed // inherited attribute of track
incorrect // inherited attribute of track
invariants
@nextWhenOccupies.type nextWhenOccupies ∈ train → track
@Invariant_H1 occupies ∈ train ⇸ track
@Invariant_H2 ∀thisTrain·((thisTrain∈train)⇒(nextWhenOccupies(thisTrain) =next( occupies(thisTrain))))
@Invariant_H5 ∀thisTrain·((thisTrain∈train)⇒(failed(occupies(thisTrain)) = FALSE))
@Invariant_H6 ∀thisTrain·((thisTrain∈train)⇒(incorrect(occupies(thisTrain)) = FALSE))
events
event INITIALISATION
then
@train.init train ≔ ∅
@occupies.init occupies ≔ ∅
@speed.init speed ≔ ∅
@nextWhenOccupies.init nextWhenOccupies ≔ ∅
@next.init next :∈ track → track
@obstructed.init obstructed ≔ track × {FALSE}
@failed.init failed ≔ track × {FALSE}
@incorrect.init incorrect ≔ track × {FALSE}
end
event setOccupies refines setOccupies
any thisTrain // contextual instance of refined class train

# Other Users:
# Prototypes and Future Plans

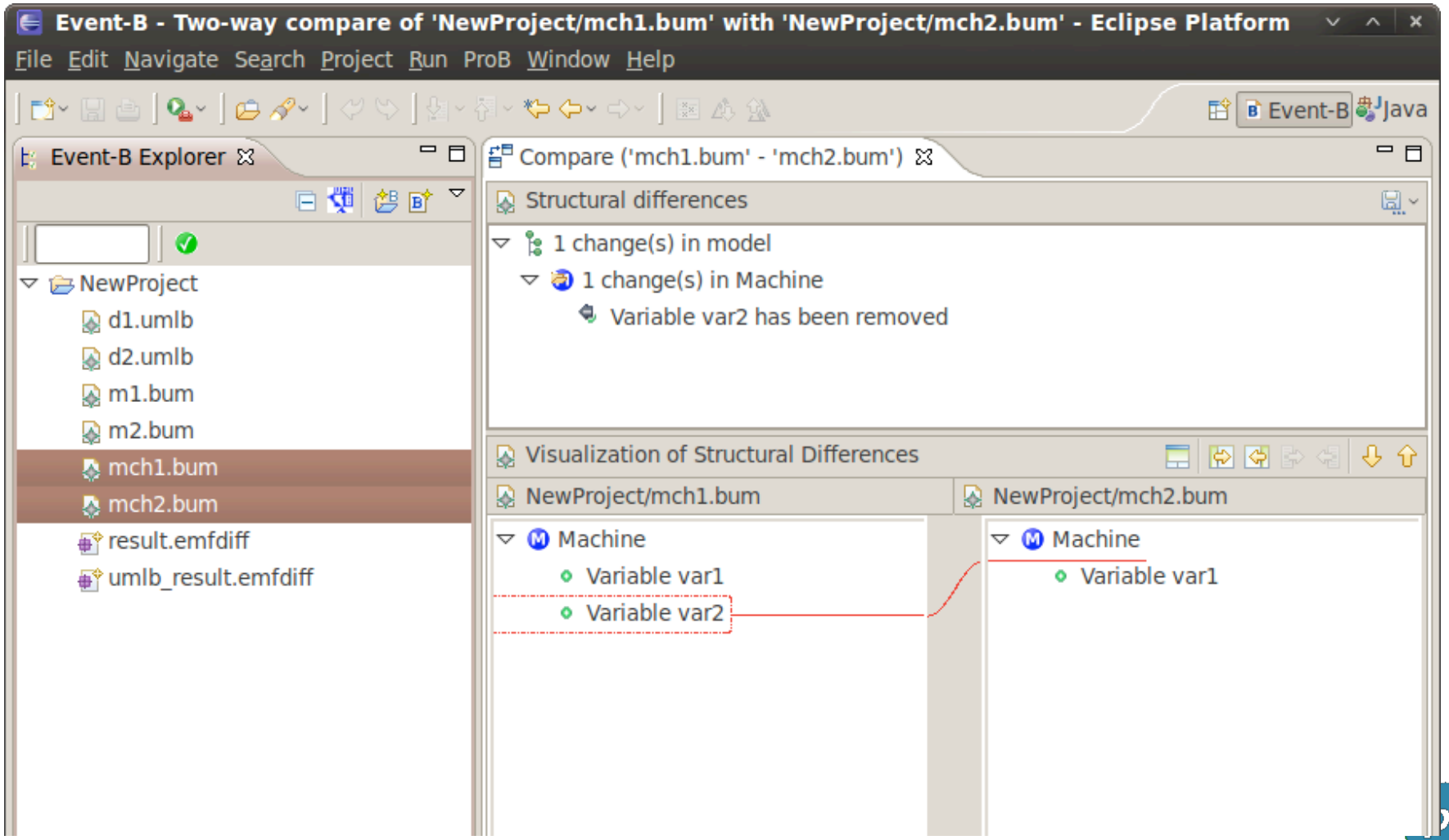
# Structured Editor



# Project Diagram



# Compare/Merge Editor



# Under Investigation

- NEW
  - Records Extension
- Already in Rodin
  - Theories Extension
  - Shared Event Composition Tool
- Already in EMF
  - UML-B
  - Feature Composition Tool
  - Refinement Pattern Editor

# M2M transformations

- UML (to UML-B)

- Kaos
- Problem Frames
- CSP

# Questions