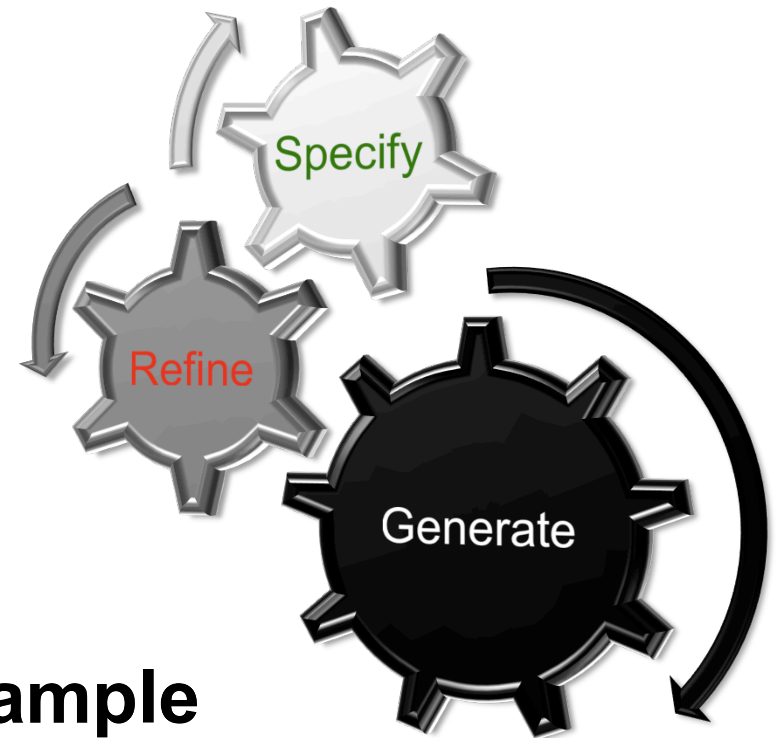
CSSE 490 Model-Based Software Engineering: Play it again Sam… more Software Factories

Shawn Bohner
Office: Moench Room F212
Phone: (812) 877-8685
Email: bohner@rose-hulman.edu

ROSE-HULMAN
INSTITUTE OF TECHNOLOGY

Learning Outcomes: MBE Discipline

Relate Model-Based Engineering as an engineering discipline.



- Demo Final Project
- Finish Software Factories
- Examine Executable UML
- Short Action Language Example
- Recipe Framework for Manual Code Inclusion
- ReviewTerm Paper Assignment

Recall: Software Factories

- 1990's Software Factories emerged as the new automated programming
- Faced an untrained community coupled with limitations in computing capabilities
  - The Virtual Software Factory
  - Software Templates
  - Software Refinery

- Devolved into IDEs configured for efficient development of Domain applications (led by Microsoft these days)

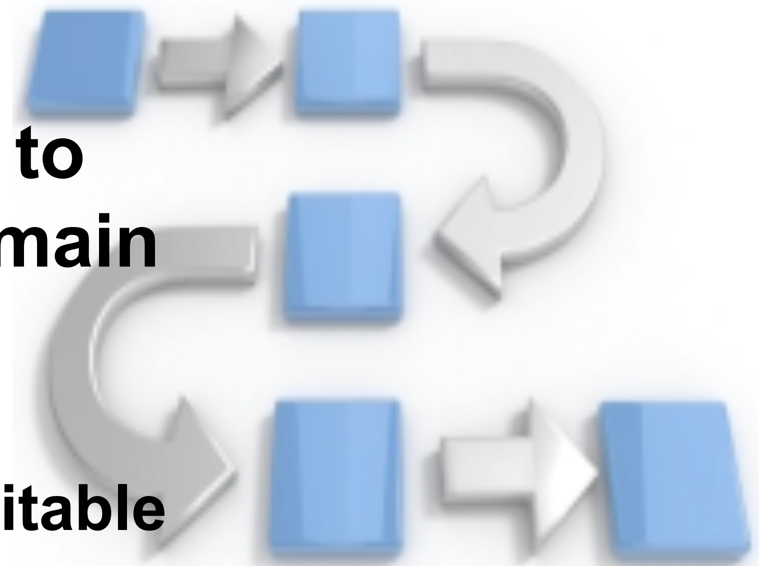
Recall: Software Factories Schema

- Schema defines viewpoints for modeling and building a system (e.g., enterprise system):
  - Presentation, form layout and workflow
  - Component structure and business data model
  - Persistence mapping, Deployment, …
- Schema identifies core artifacts as well as the most efficient way of producing them
  - DSLs, frameworks, patterns, manual programming
- Schema identifies commonalities and differences among applications in the domain

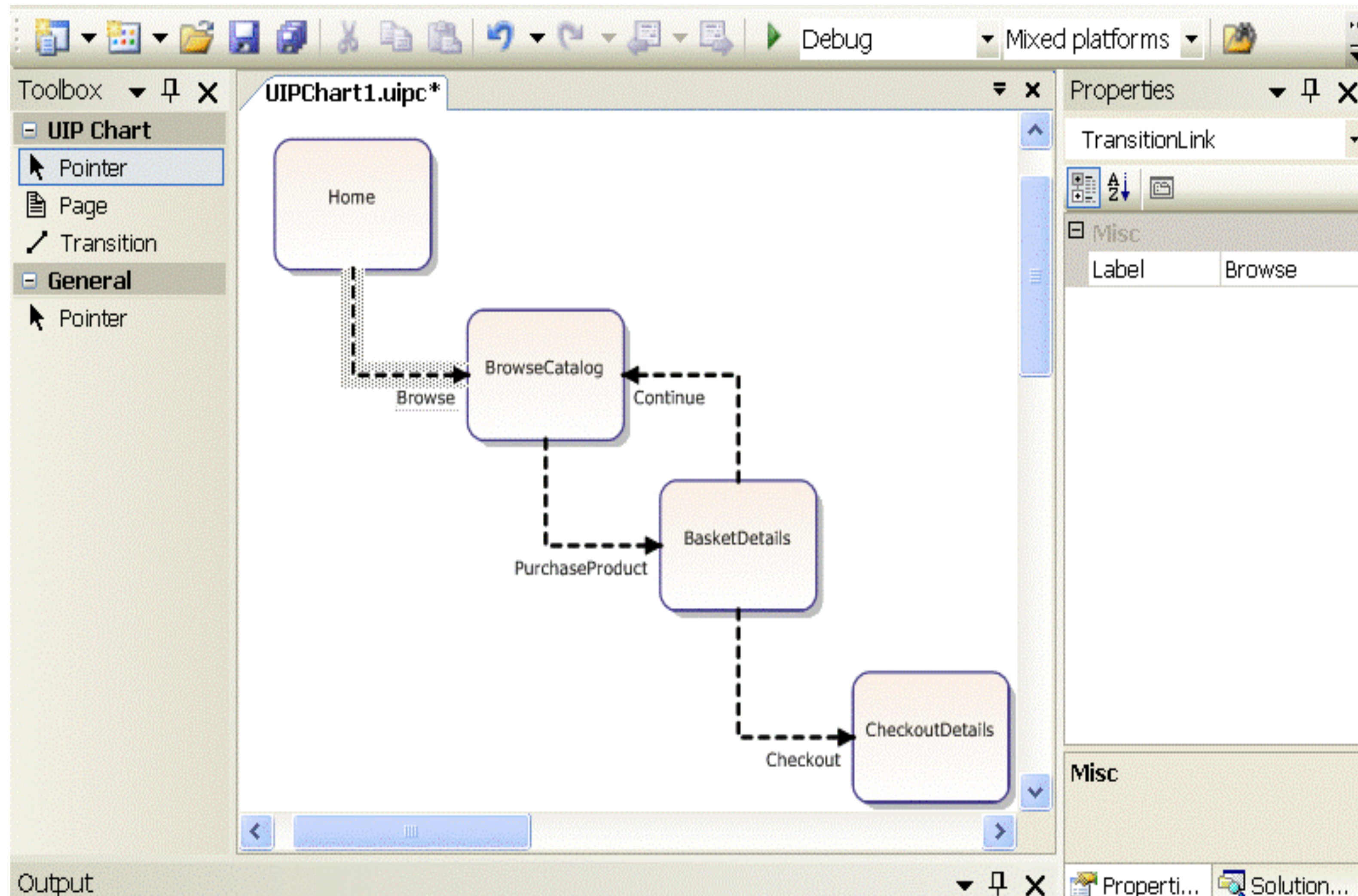
Software Factories Templates

- Makes the Schema usable

- Load SF Template into IDE to configure it for specific domain
  - Provides the necessary frameworks or libraries
  - Contributes project types suitable for the factory
  - Delivers build scripts
  - Extends IDE with DSL editors and transformations



MS DSL Tools Example



Source: http://galilee.microsoft.fr/(ncqibzbvkp2ezr45aevbcqjk)/a17fdcfb90f14a7592045f1c0fc5e97f/

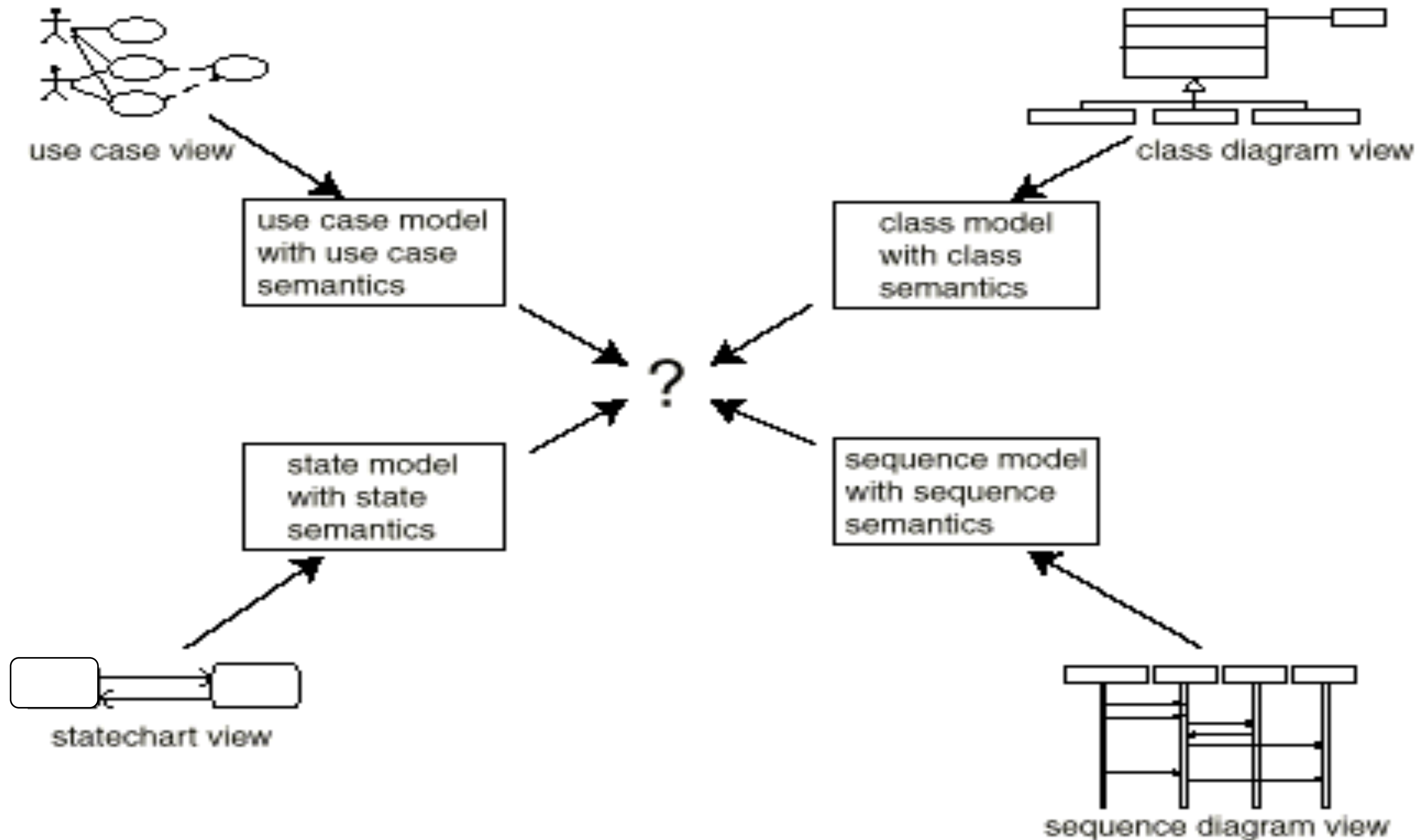
Defining a Metamodel

Source: http://galilee.microsoft.fr/(ncqibzbvkp2ezr45aevbcqjk)/a17fdcfb90f14a7592045f1c0fc5e97f/

Software Factory and MBSE's

- SFs use model-based concepts without major changes
  - DSLs are used to build models, Languages often graphical
  - Some provide tooling to define the metamodels as well as concrete syntax and editors
- SFs seldom use OMG standards for their infrastructure
  - DSLs are not UML based
  - Metamodels are not based on the MOF, and not QVT
- Application developer's perspective
  - Models are first class artifacts in development projects
  - Editors and transformations integrate seamlessly with the IDE
- Infrastructure developer's perspective
  - Metamodels, editor definitions and transformations are first class artifacts
  - Tools to build them are seamlessly integrated into the IDE

Same Semantics for Different Views



Graphical Model Editor

Menubar

Toolbar

Modebar

Model Editing Windows

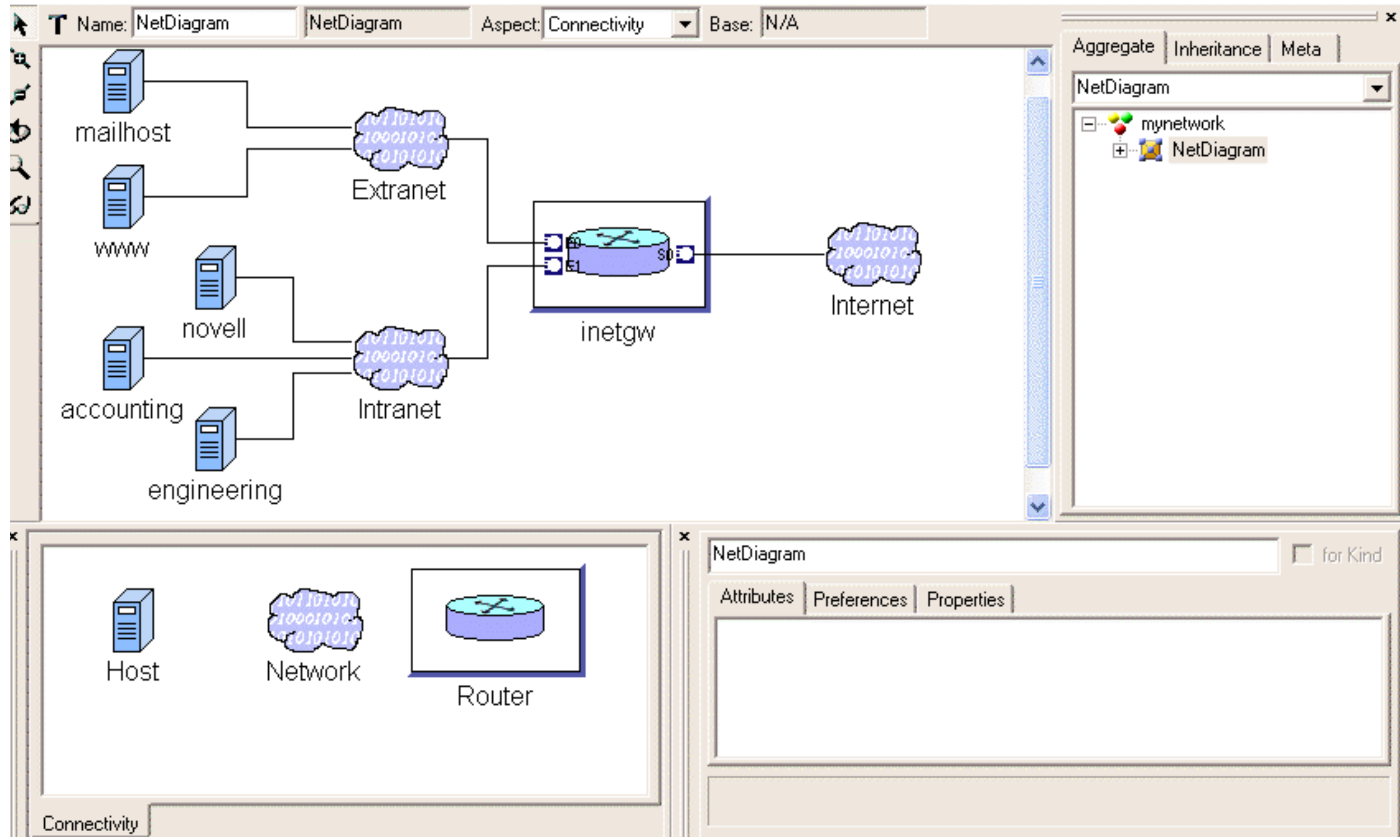
Partbrowser

Statusbar

Model Browser

Attribute Browser

GME: Modeling based on previously defined Metamodel



GME: OCL Constraint Validation

Name: SF2000 | ParadigmSheet | Aspect: Constraints | Base: N/A

Folder
<<Folder>>

Processing
<<Model>>

Primitive
<<Model>>
Firing : enum
Script : field
Priority : field

Compound
<<Model>>

AtLeastOnePart

Signal
<<Atom>>

InputSignal
<<Atom>>

OutputSignal
<<Atom>>

DataflowConn
<<Connection>>

ParameterConn
<<Connection>>

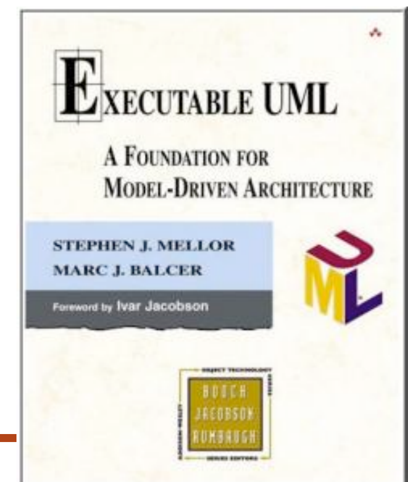
AtLeastOnePart | for Kind

Attributes | Preferences | Properties

Description:	Compounds must have parts
Default parameters:	
Equation:	self.parts()->size > 0
Priority (1=High):	2
Depth:	1
On close model	False
On create	False
On delete	False
On new child	False
On lost child	False
On move	False
On derive	False
On connect	False
On disconnect	False
On change attribute	False
On change property	False
On change assoc.	False
On refer	False
On unrefer	False
On include in set	False
On exclude from set	False

Executable UML (xUML) Concepts

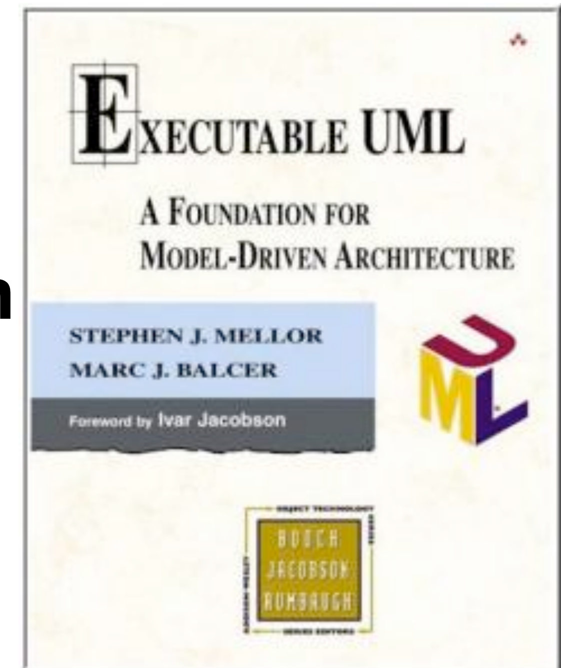
- Executable UML is not a formal standard, but a goal for a UML-based programming language
- Must eliminate redundancy and ambiguities, to increase executability of UML
- Action language needed to define complete implementations of software systems
- Not a DSL, but rather a universal, UML-based programming language



Executable UML → Action Semantics

- Hard to model a complete system today via UML or even MOF-based languages

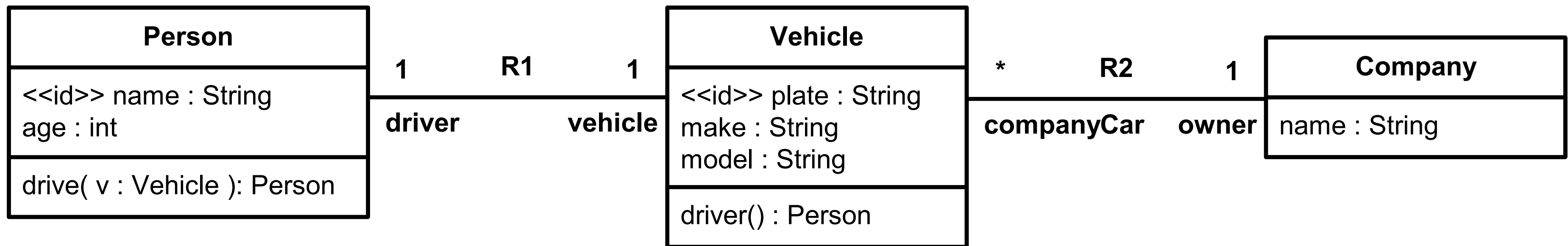
- Action semantics do not contain structural constructs (classes, attributes & relationships)
  - Already defined in the structural part of the model
  - Merely define behavioral building blocks



Action Semantics in UML 2.0

- Models procedural behavior via abstract syntax
- Variables for assigning/reading (sets, bags…)
- Arithmetic and logical operations
- Control flow (if-then-else, case, block…)
- Class extents that be queried (SQL-like)
- Creation, deletion, and navigation of associations
- Generation of signals and timers
- Definition of functions

Action Languages Example 1/3




```
myJeep = create Vehicle with plate = "IYQ2"
myJeep.make = "Chrysler Jeep"
myJeep.model = "Liberty CRD"
```

Action Languages Example 2/3

```
shawn = create Person with name = "Shawn"
```

We can now call the operation drive() to let the driver drive the vehicle.

```
[actualDriver] = drive[aVehicle] on shawn
```

What is still missing, of course, is the implementation of the operation drive(). The least it must do is to instantiate the association R1 (that is, to create a link between the two concerned objects).

```
link this R1 aVehicle
```

Action Languages Example 3/3

```
theCurrentDriver = this.R1.“driver“
```

Let’s assume we want to find all people in the system:

```
{allPersons} = find-all Person
```

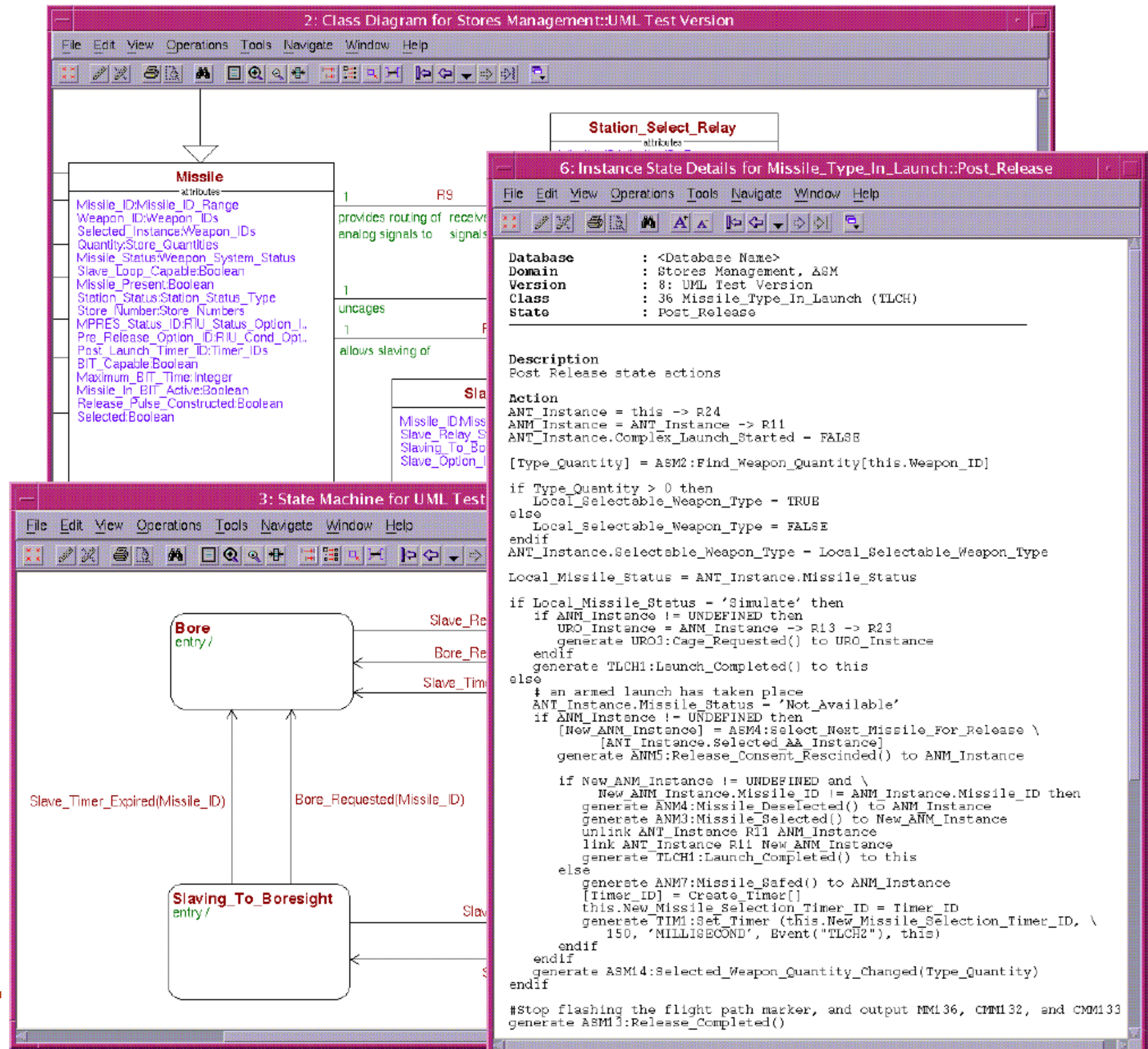
The braces state that allPersons is a set of objects instead of just one.

One can also limit such a search. For example, all vehicles of the brand Jeep can be looked for.

```
{Jeeps} = find Vehicle where make = “Jeep”
```

Example: Kennedy Carter's iUML

- Model Diagrams
- Code
- Integration



ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Recipe Framework for Integrating Manually Developed Code

Package Explorer

>scmGenerator [cvs.sourceforge.net]
>scmHelloWorld [cvs.sourceforge.net]
>src
>example
>BC1.impl
CalculatorImplementatio
BC2.impl
tests
>src-gen
>example
>BC1
>impl
>CannotAdd.java (AS
>IC1.java (ASCII -kkv
>ICalculator.java (AS
>BC2
>exsys.client
model
build.properties 1.4 (ASCII -kkv)
build.xml 1.3 (ASCII
>dump.dump (Binary
>helloWorld.recipes (Binary)

Ant

oAW - antlrInstantiator - build
oAW - antSupport - build
oAW - core - build
oAW - metamodelGenerator - build
oAW - recipe.ant - build
oAW - recipe.core - build
oAW - recipe.plugin - update
oAW - recipe.simpleChecks - build
oAW - umlMetamodel - 1) generate

Outline

An outline is not available.

oc | Declaration | Search | Console | Progress | Error Log | Properties | Recipes

e:BC1:C1: manual implementation of component
have to provide an implementation class
mplementation class has to extend t
e:BC2:C2: manual implementation of component
e:BC2:Decider: manual implementation of component
e:BC1:Calculator: manual implementation of component

Evaluate

Reevaluate All
Reload Recipe File

Name	Value
_type	org.open
_type	org.open
className	example.BC1.impl.C1Implementation
element	:example:BC1:C1
projectName	scmHelloWorld
srcPath	src

the generator generates a base class forcomponents, in this case

Problems | Javadoc | Declaration | Search | Console | Progress | Error Log | Properties | Recipes

:example:BC1:C1: manual implementation of component
you have to provide an implementation class.
the implementation class has to extend the generated componen
:example:BC2:C2: manual implementation of component
:example:BC2:Decider: manual implementation of component
:example:BC1:Calculator: manual implementation of component

Name	Value
_type	org.openarchitectureware.recipe.ecl...
_type	org.openarchitectureware.recipe.ch...
className	example.BC1.impl.C1Implementation
element	:example:BC1:C1
projectName	scmHelloWorld
srcPath	src

the generator generates a base class forcomponents, in this case example.BC1.impl.C1ImplementationBase. From this base classyou have to extend your own class that has to be called example.BC1.impl.C1Implementation

resource not found: scmHelloWorld/src/example/BC1/impl/C1Implementation.java

helloWorld.recipes - scmHelloWorld

1 2 3 4 5 6 7

Write and Present a Term Paper

- Use IEEE/ACM format for the paper (template provided on Angel)
- Include abstract, introduction, background/related work, analysis, and conclusion (along with references)
- Target 5-7 pages
  - If you are not a strong writer, use a lot of tables and figures to organize your work
  - Use your own words - copied elements without reference are considered plagiarism
- Paper due 11:55pm Tuesday, May 17th, 2011 (Angel dropbox)
- Presentation due by class time on May 19th, 2011 (pres. Delivered to Angel dropbox)

Homework and Milestone Reminders

- Term Paper
  - Paper Due by 11:55pm, Tuesday, May 17th, 2011.
- Term Paper and Presentation
  - Presentation Due by 1:35pm, Thursday, May 19th, 2011.