

41. Meta-CASE-Werkzeuge: MetaEdit+ von MetaCase

1

Prof. Dr. Uwe Aßmann
Technische Universität Dresden
Institut für Software- und
Multimediatechnik
<http://st.inf.tu-dresden.de>
Version 13-1.1, 09.01.14

- 1) Meta-CASE-Werkzeuge
- 2) MetaEdit+
- 3) Introduction to Fujaba

Obligatory Reading

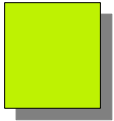
- ▶ MetaCase. Domain-Specific Modeling With Metaedit+: 10 Times Faster Than UML. White paper. http://www.metacase.com/papers/Domain-specific_modeling_10X_faster_than_UML.pdf
- ▶ MetaCase. Abc To Metacase Technology. http://www.metacase.com/papers/ABC_to_metaCASE.pdf
- ▶ A Comparison of ATL and Story-Driven Modeling (Fujaba-style GRS)
 - http://www.es.tu-darmstadt.de/fileadmin/download/publications/spatzina/PP_AGTIVE_2011.pdf



Literatur

- ▶ [Nill] C. Nill. Analysis and Design Modeling Using Metaphorical Modeling Entities. A Modeling Language for the Tools and Materials Approach. Diplomarbeit Technische Universität Dresden, 2006.
- ▶ <http://www.metacase.com/support/45/manuals/index.html>

41.1 Meta-CASE-Werkzeuge



4

Nutzung von Meta-CASE

- ▶ Ein **Meta-CASE-Werkzeug** ist eine metamodelldgesteuerte Entwicklungsumgebung für den Entwurf von SEU und Softwarewerkzeugen
 - Metamodelsteuerung zur Herstellung einer individuell angepassten Werkzeug-Umgebung:
 - Generierung von Repositorien mit Frontend- und Backend-Tools für Austauschformate
 - Generierung von Editoren und Kompositionswerkzeugen für Artefakte
 - Kompositionssysteme zur Komposition von Werkzeugen
 - Modellierung von textuellen und graphischen Sprachen
 - Modellierung von domänenspezifischen Sprachen und ihren Werkzeugen (domain-specific languages, DSL)

Productivity by Meta-CASE

- ▶ Meta-CASE improve the productivity
 - of a software development team
 - of a team of domain engineers
 - Domain-specific methods are 5 to 10 times faster than using (UML-)notation
 - Reference: Domain-Specific Modeling: 10 Times Faster Than UML; Whitepaper MetaCase 2005; <http://www.metacase.com/de/>
- ▶ Meta-CASE are the most productive tools we know for the construction
 - of DSL
 - of tools
 - of composition systems
 - of IDE (SEU)
- ▶ You take part in a course which presents the most productive tools we know!

Weitere Beispiele zu Meta-CASE

- ▶ MetaEdit+ (commercial): Parametrisierbares CASE-Tool mit
 - Editor für rollenorientierte Metamodelle (MetaEdit+ rollenorientierte Metasprache)
 - Generator für die Erstellung der Methodenbeschreibung
 - Gute Anbindung an GUIs with Screen-Flow-Language
- ▶ AdoXX (commercial), BOC Vienna
- ▶ KOGGE, JKOGGE: Generator für grafische Entwurfsumgebungen
 - KOGGE basiert auf einer formalen Meta-Tool-Beschreibung und einem Interpreter (Prof. Ebert, Uni Koblenz)
 - <http://www.uni-koblenz-landau.de/koblenz/fb4/institute/IST/AGEbert/MainResearch>
- ▶ Eclipse Modeling Facility (EMOF):
 - Benutzt eine Teilmenge von MOF
- ▶ OpenArchitectureWare (EMOF): Moved to Eclipse
 - <http://www.openarchitectureware.org/>
- ▶ Netbeans: IDE based on MOF
- ▶ MOFLON: IDE based on MOF, with Storyboards (GRS), Logic (OCL) and TGG (GRS)
- ▶ Fujaba: with Storyboards (GRS)

41.2 MetaEdit+ von MetaCase

8

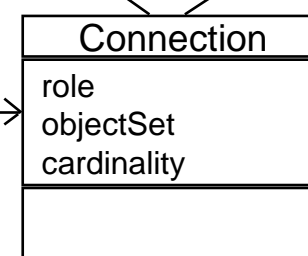
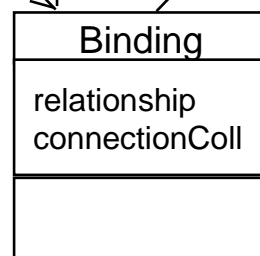
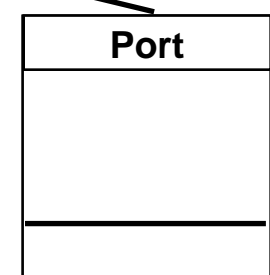
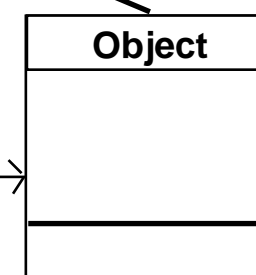
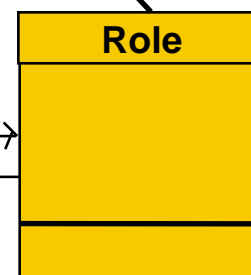
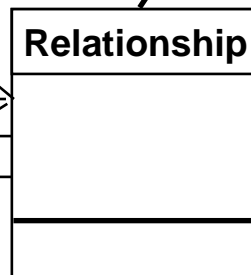
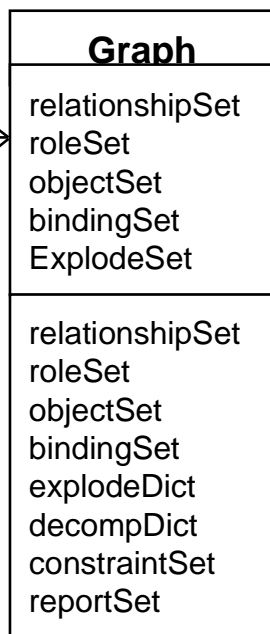
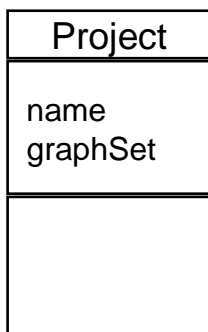
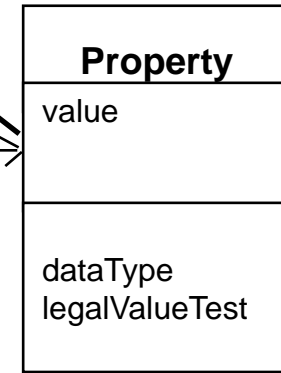
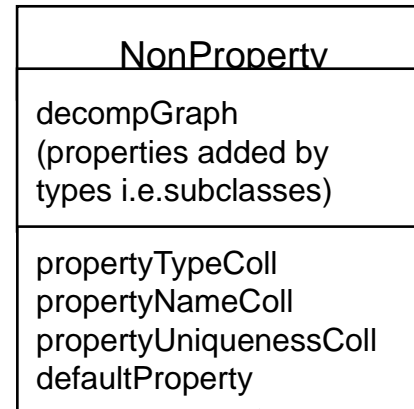
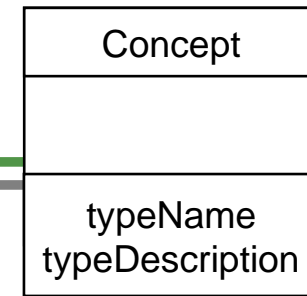
- <http://www.metacase.com/download/> Evaluation version
- http://www.metacase.com/cases/dsm_examples.html Many more DSL examples
- <http://www.metacase.com/resources.html> Articles and handbooks

Wdh: Metasprache von MetaEdit+

Unterstützt Graphen und Rollen mit GOPRR

Metamodell:

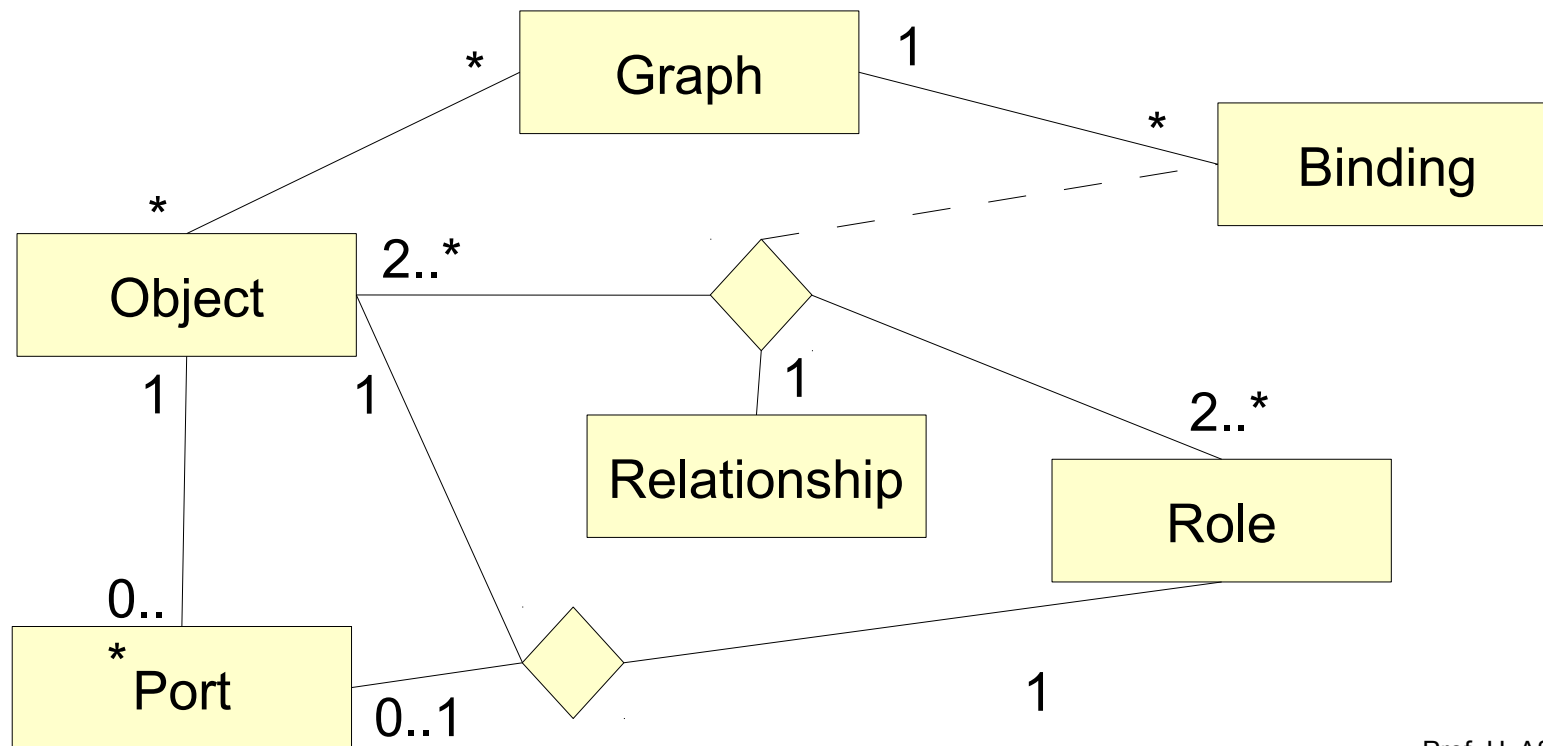
- **G**raph Tool
- **O**bject Tool
- **P**roperty Tool
- **R**elationship Tool
- **R**ole Tool



Wdh: Graph Types in MetaEdit+

► A graph type (diagram) defines:

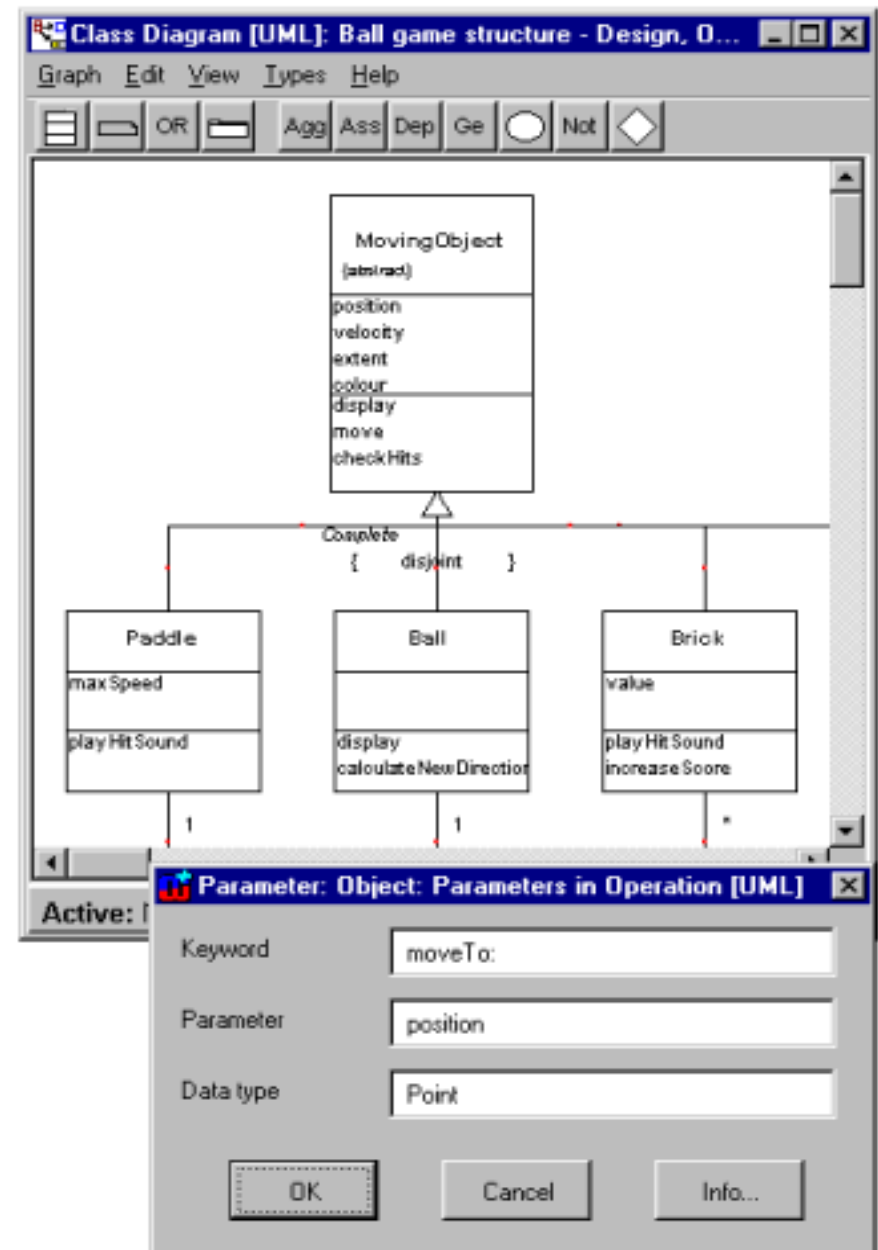
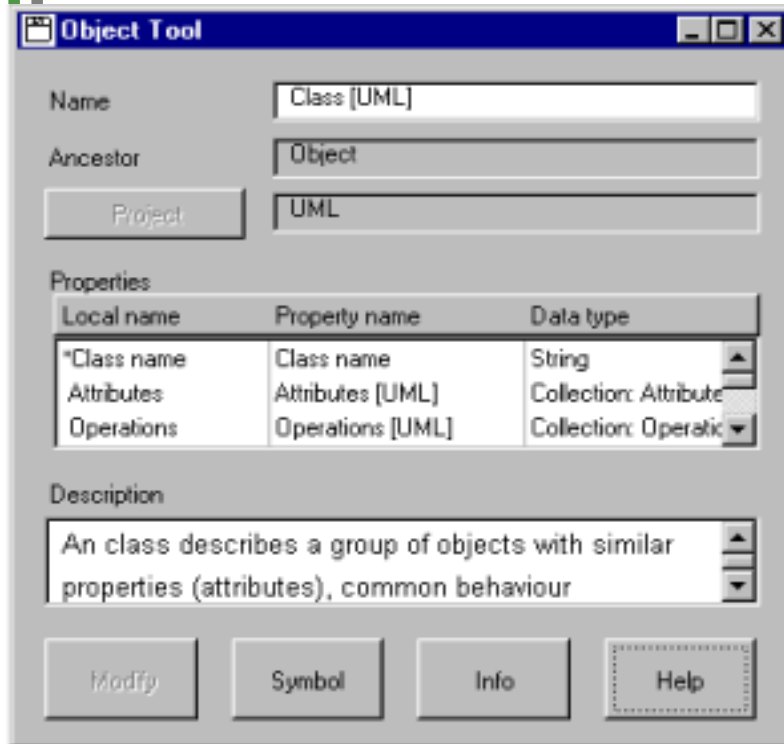
- Objects
- Roles
- Relationships
- Allowed Bindings between all entities:
 - a binding consists of a relationship with roles and playing objects



Erstellen eines eigenen CASE-Tools mit MetaEdit+

Entwurf der eigenen Methode

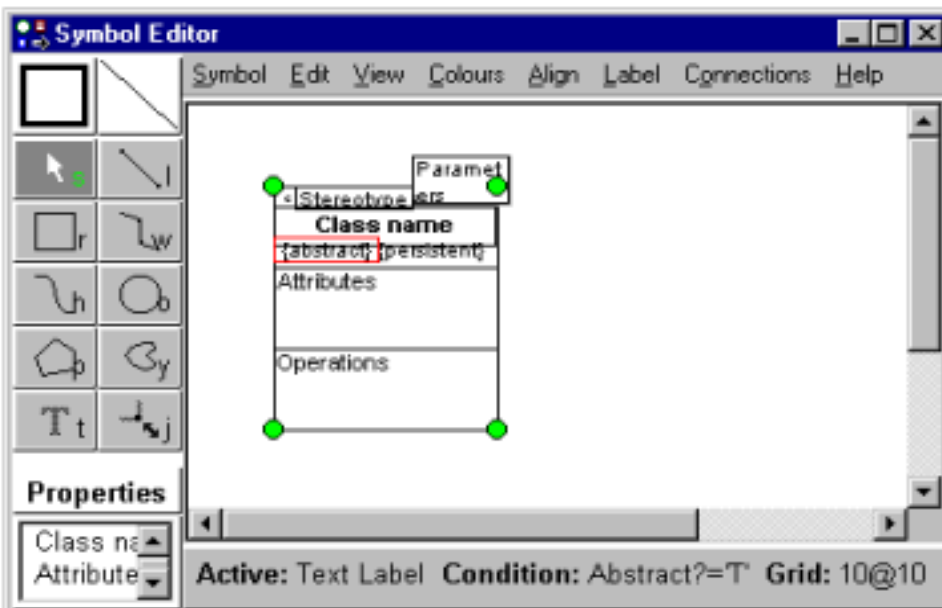
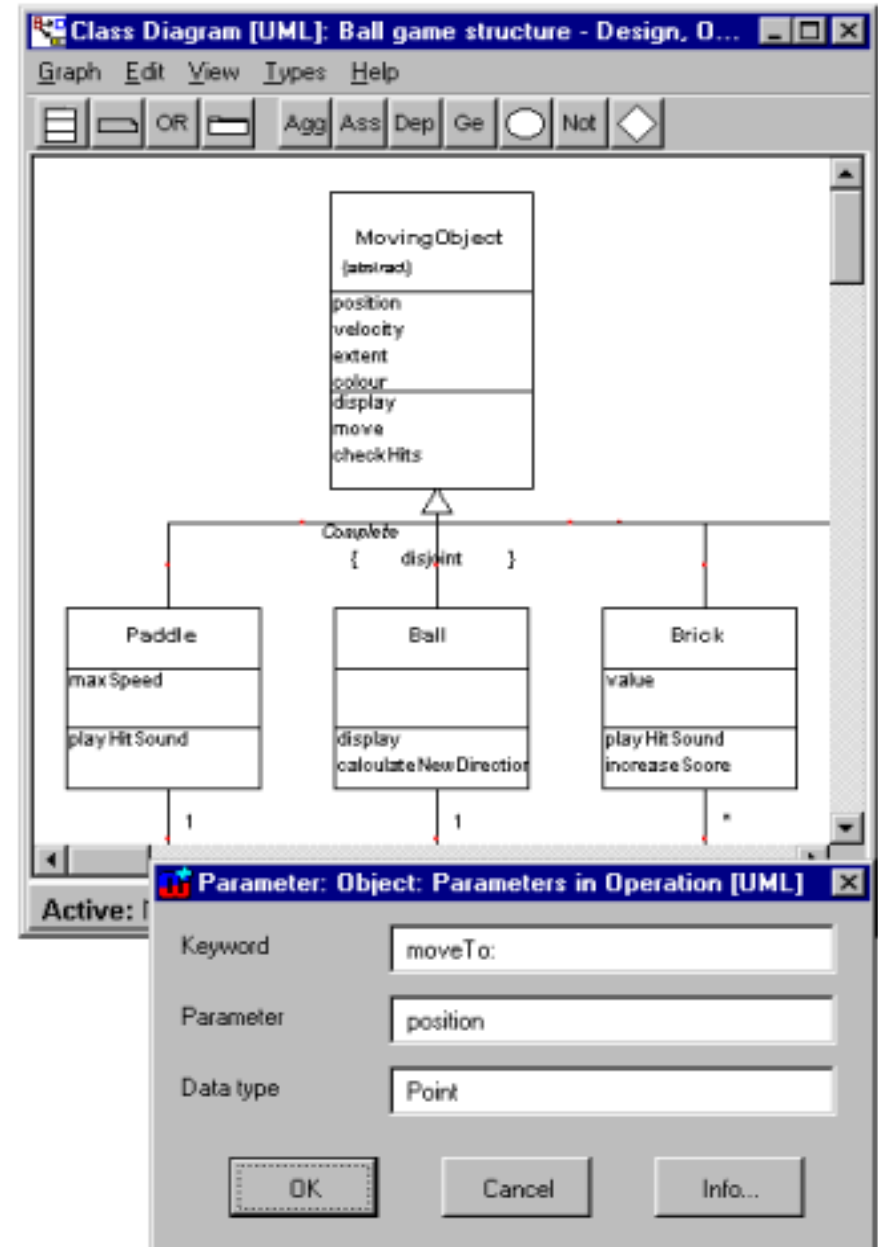
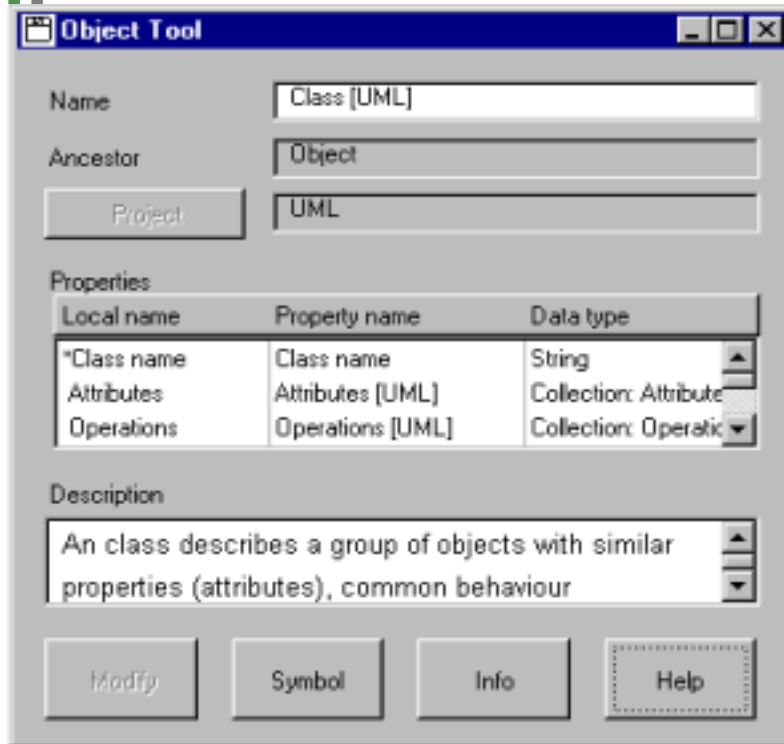
Benutzen der eigenen Methode



Erstellen eines eigenen CASE-Tools mit MetaEdit+

Entwurf der eigenen Methode

Benutzen der eigenen Methode



MetaEdit+ Workbench für ein State Diagram (STD)

The image displays the MetaEdit+ Workbench interface for designing a State Diagram (STD) for a stopwatch application. The interface is divided into several windows:

- Symbol Editor:** Shows a state symbol with the text "State name" and a green box containing "Bli" and "DisplayFn".
- Object Tool:** A panel with buttons for "Open...", "New...", "Close", "State (Watch)", "Object", and "Watch". It also includes a "Properties" table and a "Description" field.
- State Diagram Graph:** A graph showing the state transitions for the stopwatch application. It includes states like "Stopped" and "A", and transitions labeled with "Down" and "stopTime".
- State [Watch]: Object:** A dialog box for editing the state properties. It includes fields for "State name" (Stopped), "DisplayFn", "Blinking", and "Documentation".

The "Object Tool" window contains the following properties table:

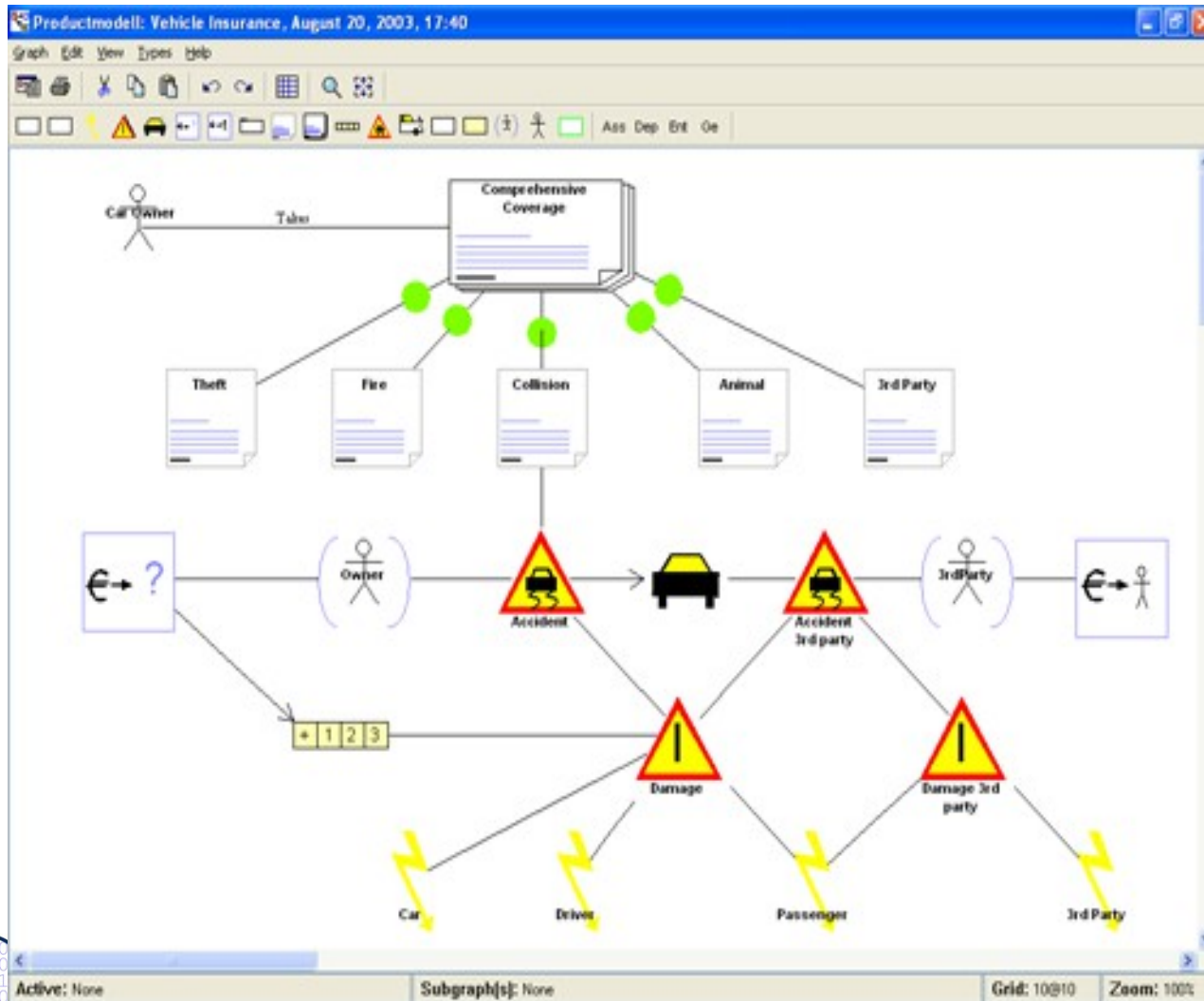
Local name	Property name	Data type
*State name	Name [Watch]	String
DisplayFn	DisplayFnField	DisplayFn
Blinking	Time unit	String (Overridable)
Documentation	Documentation [Watch]	Text

The "State [Watch]: Object" dialog box contains the following fields:

- State name: Stopped
- DisplayFn: [Empty]
- Blinking: [Dropdown]
- Documentation: In this state the Stopwatch is stopped and current stop time is shown on the display.

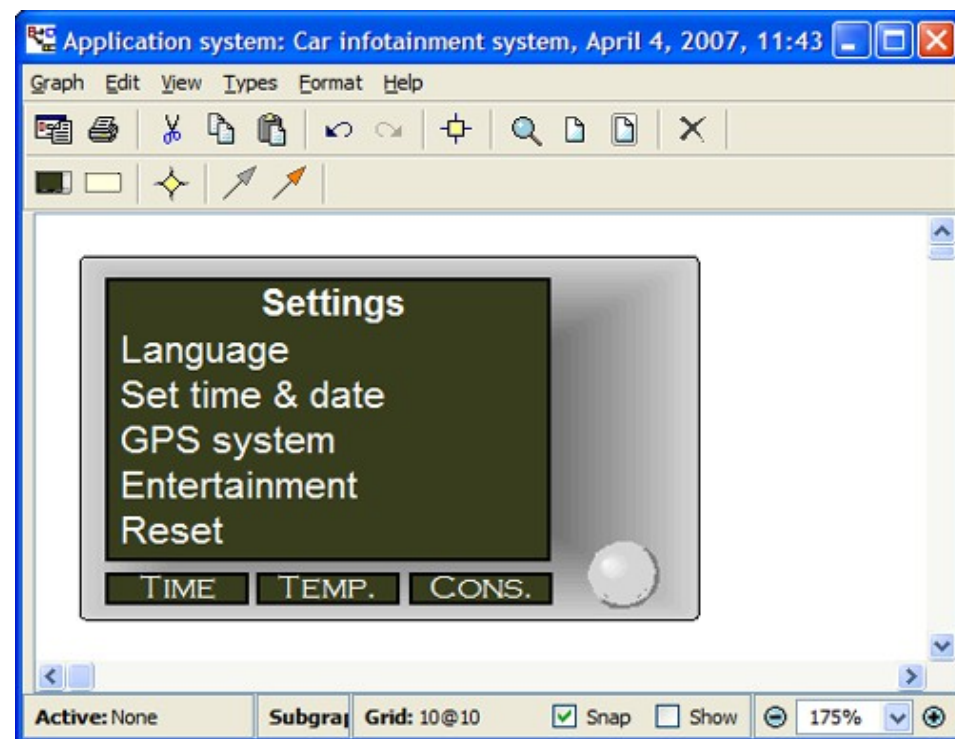
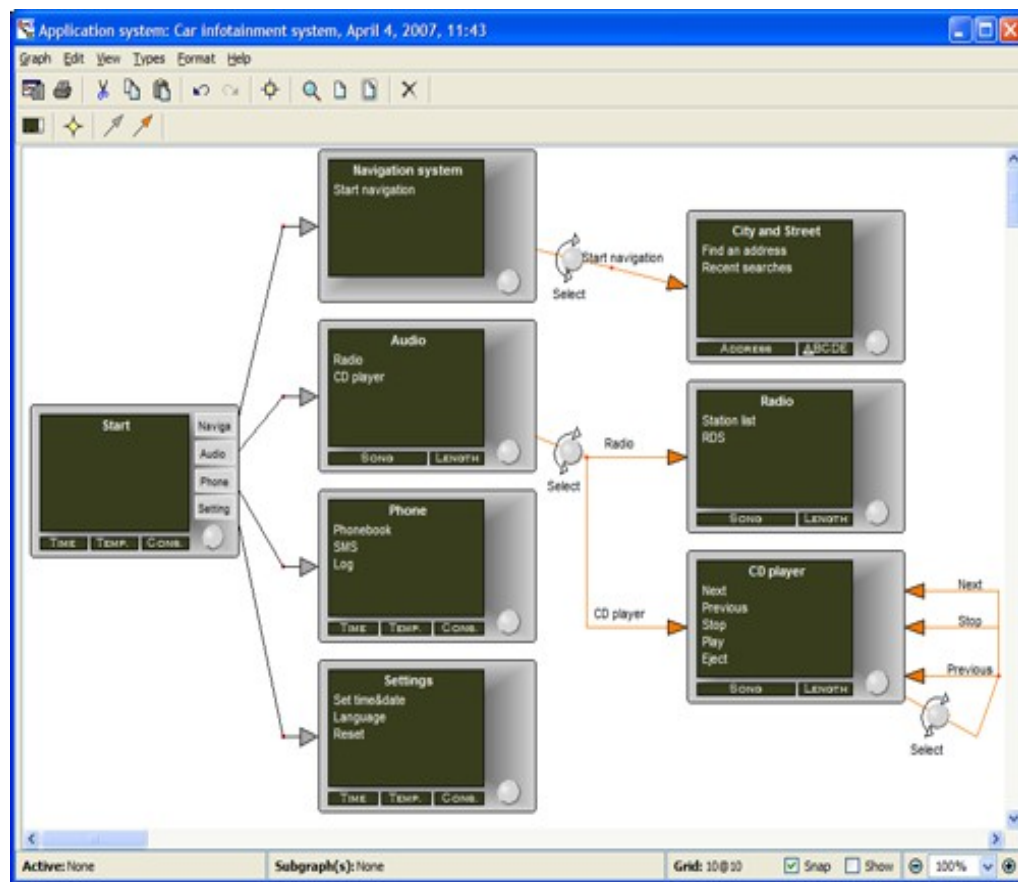
Insurance DSL

- ▶ For modeling of insurance products
- ▶ Generators produce the required insurance data and code for a J2EE website



Automotive Entertainment DSL

- ▶ Domain: car infotainment system and user interface elements
- ▶ Design of the logic and flow via connecting the modeling concepts between GUI and application concept metamodel editor



Werkzeuge in MetaEdit+

- ▶ Report Generator:
 - Skriptgesteuert, zur Erzeugung von Texten und Code
- ▶ API (API-Server):
 - MetaEdit+ ist in Smalltalk implementiert
 - Zugreifbar über Web Server (SOAP mit WSDL)

```
Report 'ExportToolUIModel'  
'<?xml version="1.0" encoding="UTF-8"?>'newline;  
'<model>'newline;  
foreach .Graph {  
  do :Graph {  
    if type; = 'Tools UIs Model' then  
      subreport; 'ToolUI_XML' run;  
    else  
      subreport; 'structureXML' run;  
    endif  
  }  
}  
'</model>'newline;  
endreport
```

[Null]

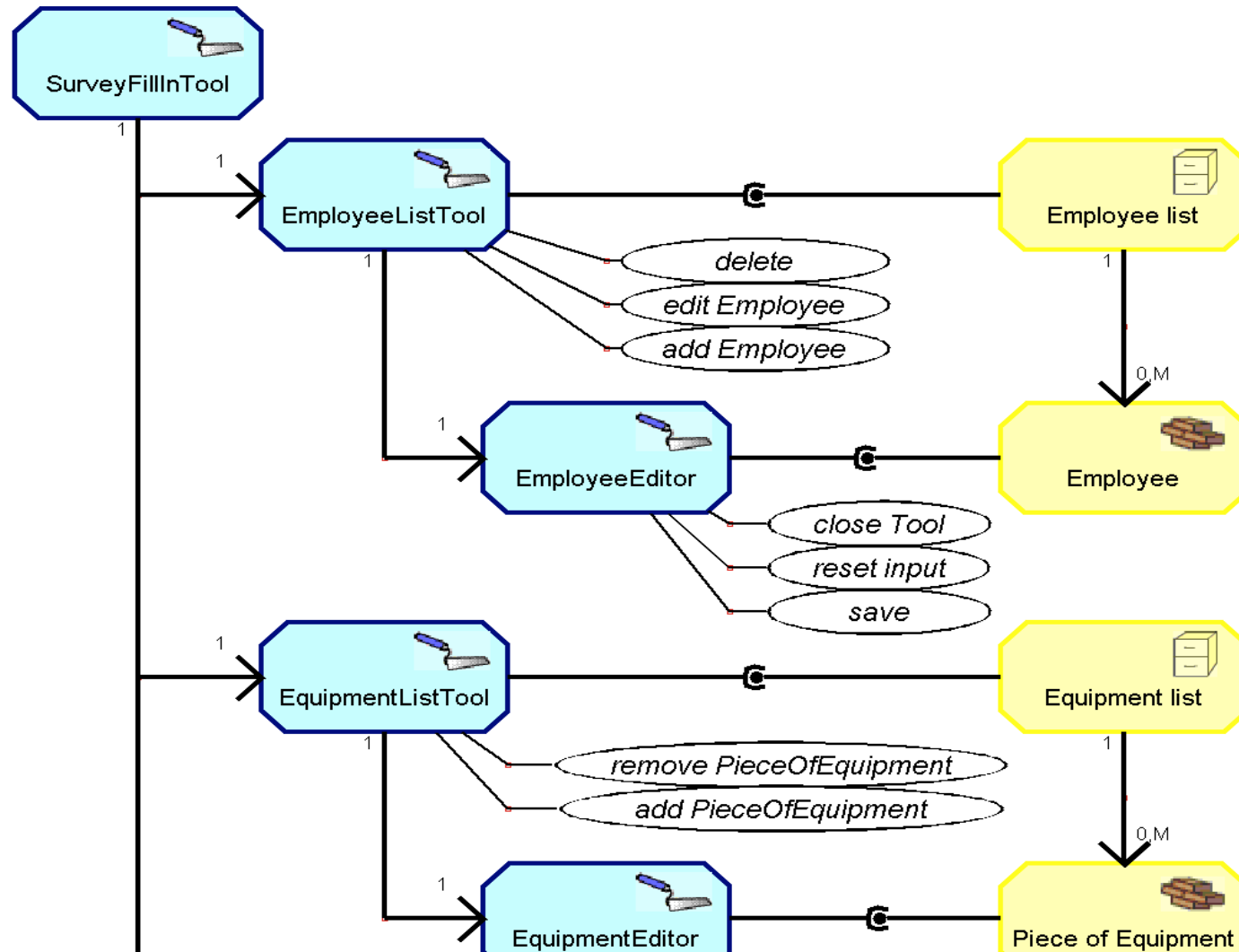
Editor for Scripts for Code Generation

```
Report 'C state machine'  
subreport; '_C_Enums'; run;  
'int state = Start;'; newline;  
'int button = None; /* pseudo-button for following  
buttonless transitions */'; newline; |  
subreport; '_C_RunWatch'; run;  
'void handleEvent()'; newline;  
'('; newline;  
' int oldState = state;'; newline;  
' switch (state)'; newline; ' ('; newline;  
foreach .(State [Watch] | Start [Watch]);  
{ ' case ';  
  if type = 'Start [Watch]' then 'Start'; else id;  
endif;  
'.'; newline;
```

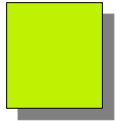
```
typedef enum { Start, Running, Stopped, Stop } States;  
typedef enum { None, Down, Mode, Up } Buttons;  
int state = Start;  
int button = None; /* pseudo-button for following buttonless transitions */  
  
void runWatch()  
{  
  while (state != Stop)  
  {  
    handleEvent();  
    button = getButton(); /* waits for and returns next button press */  
  }  
}  
  
void handleEvent()  
{  
  int oldState = state;  
  switch (state)  
  {  
    case Start:  
      switch (button)
```

Tool/Material DSL, Modeled in MetaEdit+

- ▶ [Nill] presented a DSL for Tools and Materials (TAM-DSL), modelled in in GOPRR with MetaEdit+
- ▶ Editor represents Tools and Materials graphically



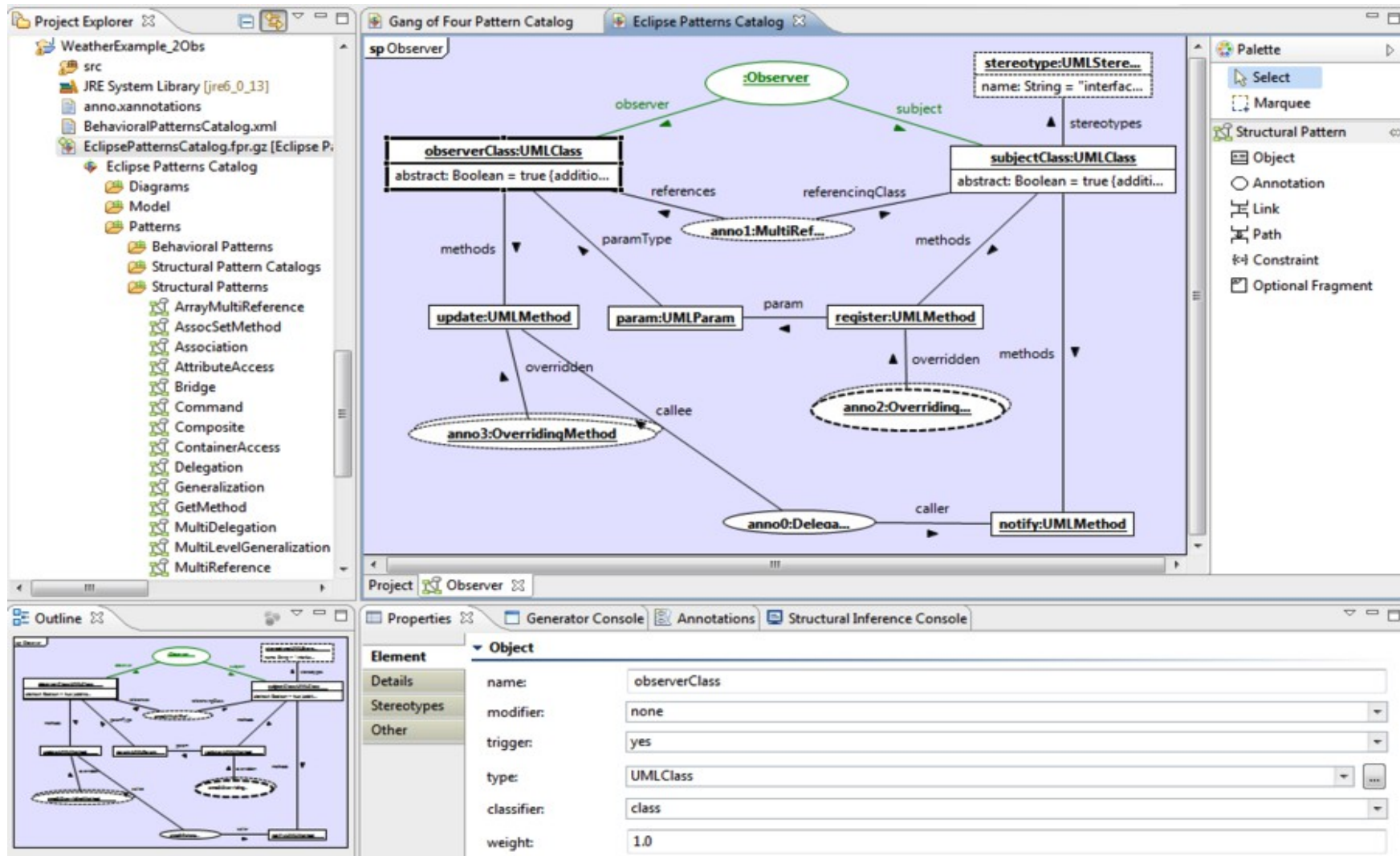
41.2 Introduction to Fujaba



19

www.fujaba.de

- ▶ Fujaba is a MetaCASE-tool based on GRS
- ▶ Basic technology: graph pattern matching and rewriting





The End
