



Web Engineering

Development Processes for Web Applications

Prof. Dr. Alexander Knapp

Dr. Nora Koch

SS 07 (8) – 2.07.2007

Ludwig-Maximilians-Universität München

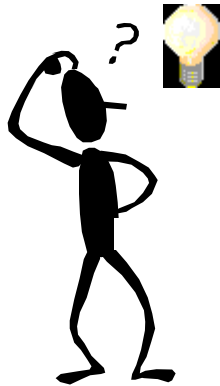
- Web engineering is the application of a **systematic** and **quantifiable approach** to cost-effective requirements analysis, design, implementation, testing, operation, and maintenance of **high-quality Web software**.
- Web engineering is also the **scientific discipline** concerned with the study of these approaches.

*Gerti Kappel, Birgit Pröll, Siegfried Reich,
Werner Retschitzegger: Web Engineering, (2006)*

- Web software is created via a development process, not a manufacturing process. It means that software creation is always **engineering**.

Victor Basili et al. (2006)

From the Idea to the Software Product



```
<html>
<head>
...
</head>
<body>
...
<ul>
<li><a href="blatt-01.pdf">Übungsblatt 1</a></li>
<li><a href="blatt-02.pdf">Übungsblatt 2</a></li>
</ul>
<a href="http://www.pst.informatik.uni-muenchen.
de/projekte/uwe/argouwe.shtml"></a><br />
<ul><li>Lösungsvorschläge zu Blatt 1 (Aufgabe
1a und 1c) und Blatt 4 als <a title="Lösungen
Blatt 1 und 4" href="Web-FilmDB.zip">ArgoUWE
Modell</a></li><li><a title="loesungsvorschlag-02.zip"
href="loesungsvorschlag-02.zip">Lösungsvorschläge
zu Blatt 2</a></li>
...
</bod<>
</html>
```

- The roadmap to building high quality software products is the **software process**
- A software process provides a framework for managing activities that can very easily get out of control
- Project management supports and controls the development

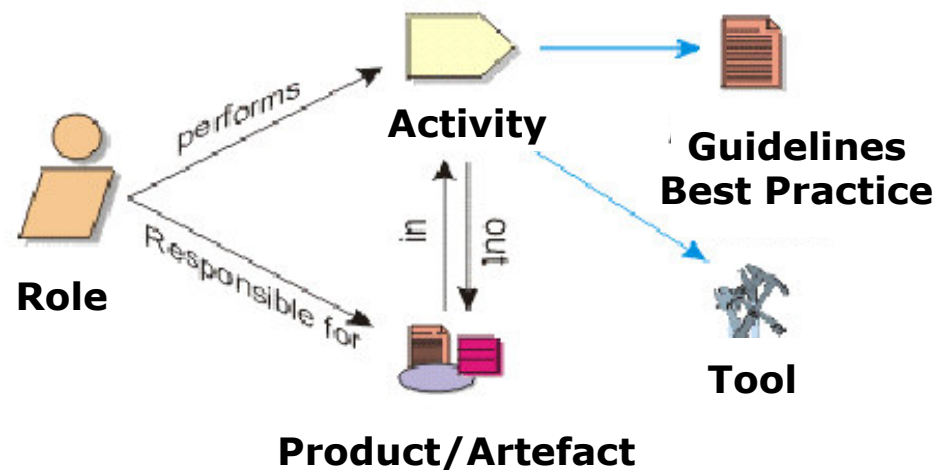
Phases in Development Processes for Web Applications



- Feasibility study
 - analysis of technologies
 - availability of resources
- Requirements analysis
 - elicitation
 - specification
 - validation
- Design
 - modelling
 - Web pages style design
 - selection of Web architecture
- Implementation
 - coding
 - content creation or import
 - testing
 - documentation
- Quality assurance
 - verification
 - performance evaluation
- Deployment
 - distribution
 - configuration
 - training
- Maintenance
 - changes
 - documentation
- Project management
 - planning
 - cost estimation
 - risk management

Development Process Basics

- **Resources** to be used in the project
 - budget, time, team (role, actor), tools, best practice (guidelines)...
- **Activities** to be performed to achieve the project goal
 - tasks
- **Products** to be delivered
 - results, artefacts, deliverables, milestones



Waterfall vs. Iterative Development Processes

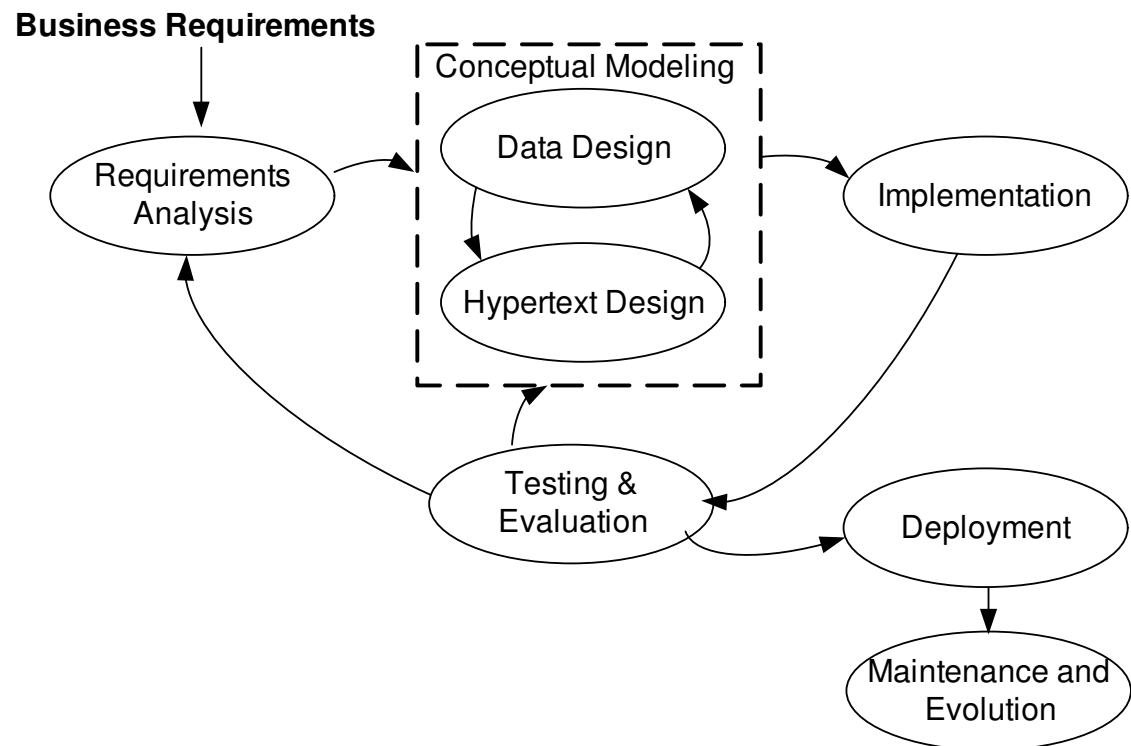
■ Waterfall

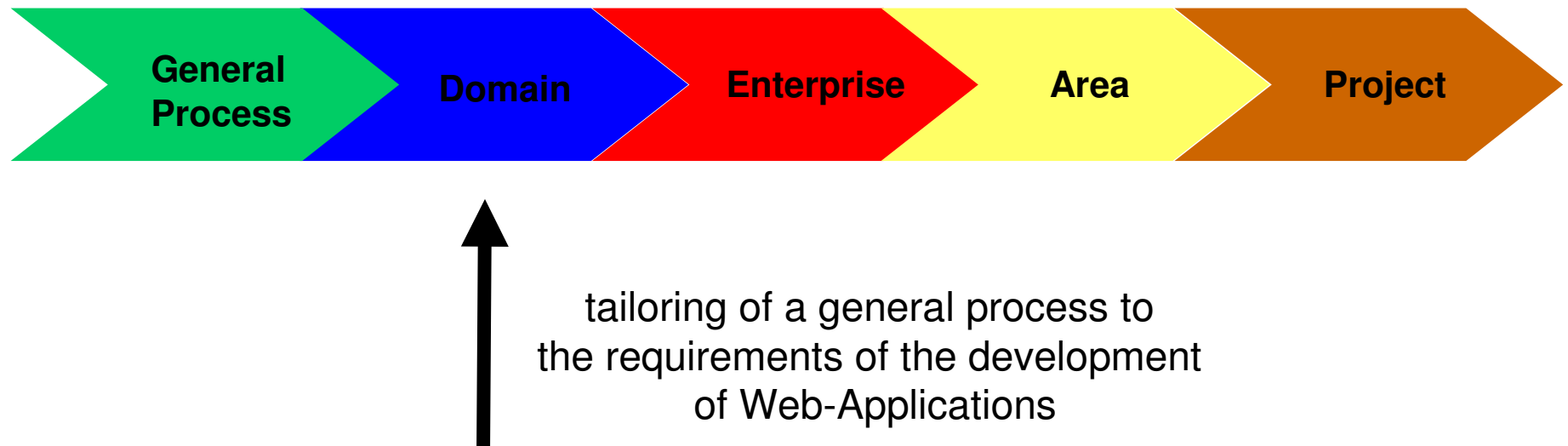
- first step is the capture all requirements followed by estimation of costs and scheduling, then design, implementation and deployment
- not appropriate if requirements change or are unreliable
- examples in the Web Domain
 - RMM: Relationship Management Methodology
 - WSDM: Web-System Design Method

■ Iterative

- approach to build software in several iterations in sequence
- each iteration is a self-contained mini-project (requirements, analysis, design, programming, testing, ...)
- goal: production of an executable iteration release (integration and testing of all software)
- examples in the Web Domain
 - WebML methodology

- WebML is a visual language for specifying the content structure of a Web application and the organization and presentation of such content in a hypertext
- Inspired by Boehm's spiral model (Boehm, 1988)
- iterative and incremental
- prototype or partial version of the application in each iteration
- tested and evaluated and extended
- support WebRatio tool





Requirements for Web Development Processes

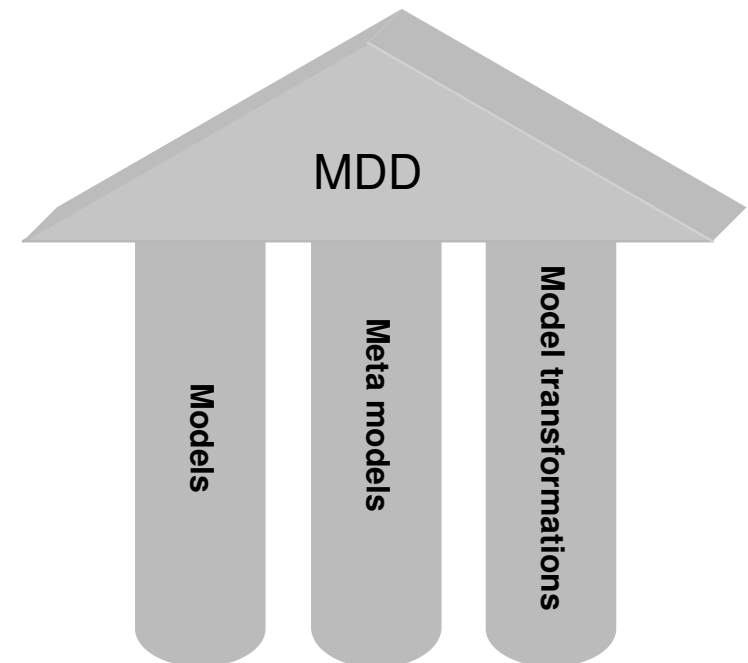
- Appropriate for short software life-cycles
 - average 3 to 6 months
 - time-to-market pressure
 - co-existence of releases
- Separation of concerns
 - content, structure, presentation, ...
- Change management
 - continuous update of content
 - changes in structure
 - changes and improvements in layout and user interface functionality
 - adaptation to new technologies and standards



Model-Driven Development (MDD)
seems to be a promising approach in the Web Domain

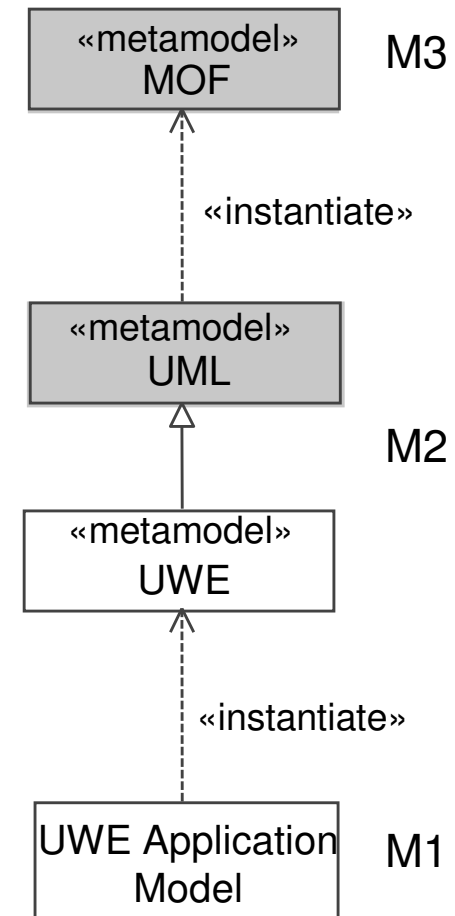
Model-Driven Development Approaches

- MDD approaches based on
 - models and model transformations
- MDD approaches require languages for
 - specification of models
 - description of metamodels
 - definition of model transformations
- MDD in the Web Domain
 - several methods propose **modelling**
 - OOHD, OO-H, UWE, WebML, Hera,...
 - separation of concerns
 - similar Web specific modelling elements
 - different notations
 - some methods define **metamodel** for modelling languages
 - OO-H, UWE, W2000, WebML, ...
 - few approaches address **model transformations**
 - OOWS, UWE, WebSA, ...



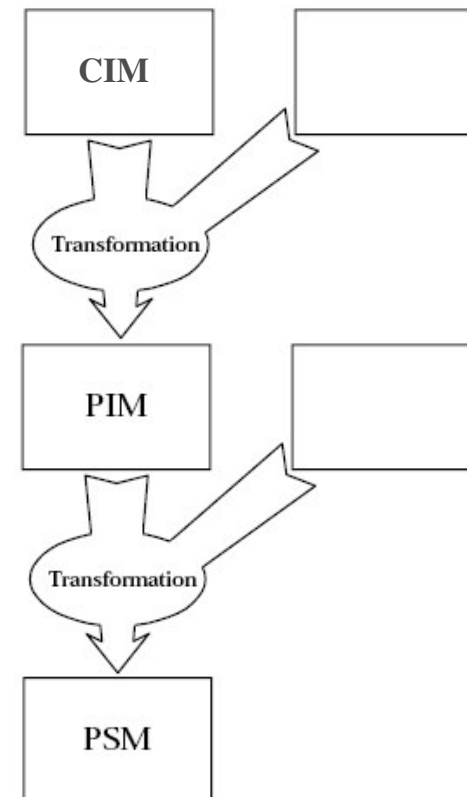
MDD Principles: Models and Metamodels

- Models
 - computational independent model (CIM)
 - platform independent model (PIM)
 - platform specific model (PSM)
- Metamodels
 - definition of **concepts** and **relationships** among concepts
 - compatibility with the **OMG metamodeling architecture**
 - MOF meta-metamodel
 - XMI interchange format → tool compatibility (theoretically)
 - static semantics given by OCL constraints (well-formedness rules)
 - basis for tool support



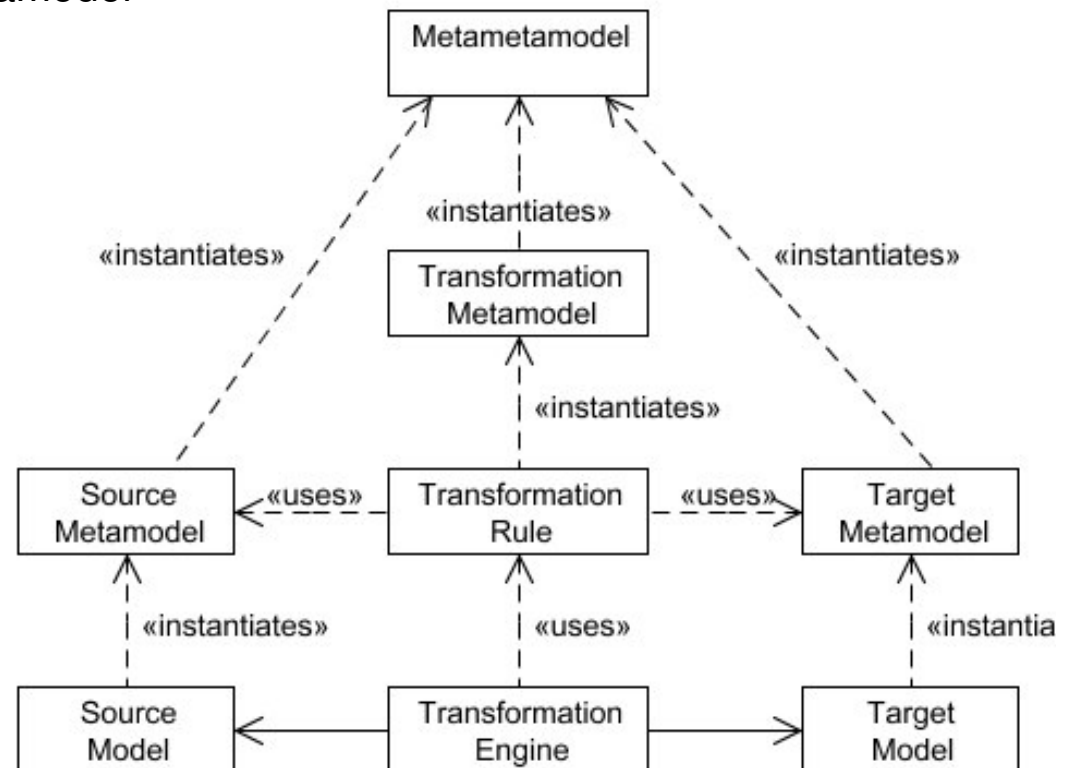
MDD Principles: Model Transformations

- Model transformations
 - CIM→PIM, PIM→PIM, PIM→PSM
- Model transformation languages
 - general programming languages
 - Java
 - graph transformation languages
 - Attribute Graph Grammar (AGG)
 - query/view/transformation languages
 - QVT
 - ATLAS Transformation Language (ATL)
- OMG standards
 - MOF, UML, OCL, XMI, QVT

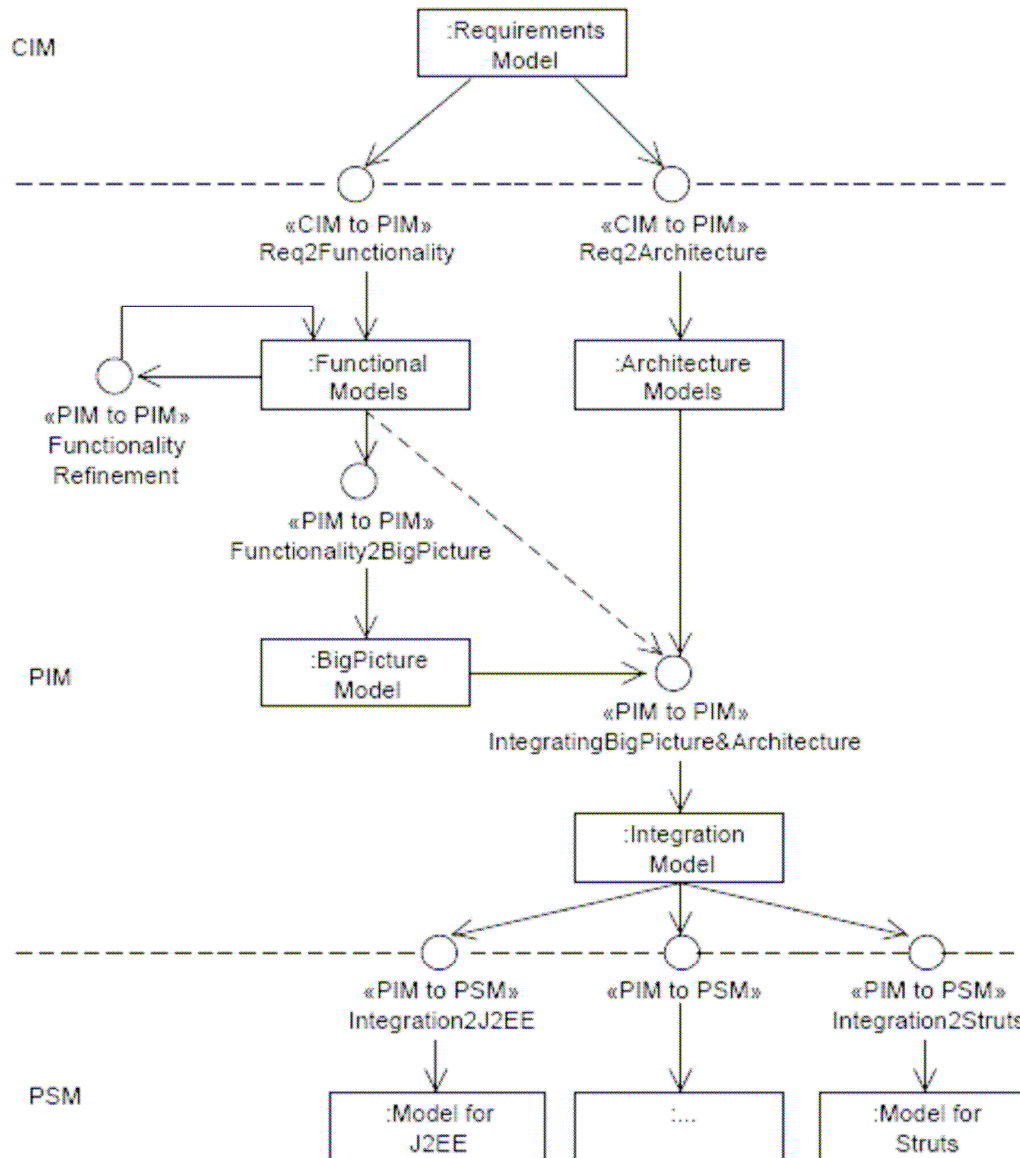


Model Transformations

- Translate between source and target models
 - instances of same or different metamodel
- Translation performed by a transformation engine
- Transformation engine executes rules
- Transformation rules are
 - defined at metamodel level
 - applied at model level
- Set of rules
 - seen as a model
 - with a metamodel
- Metamodels are based on a metamodel



Model transformation pattern (J. Bézivin, 2004)



- Graphical representation of the process

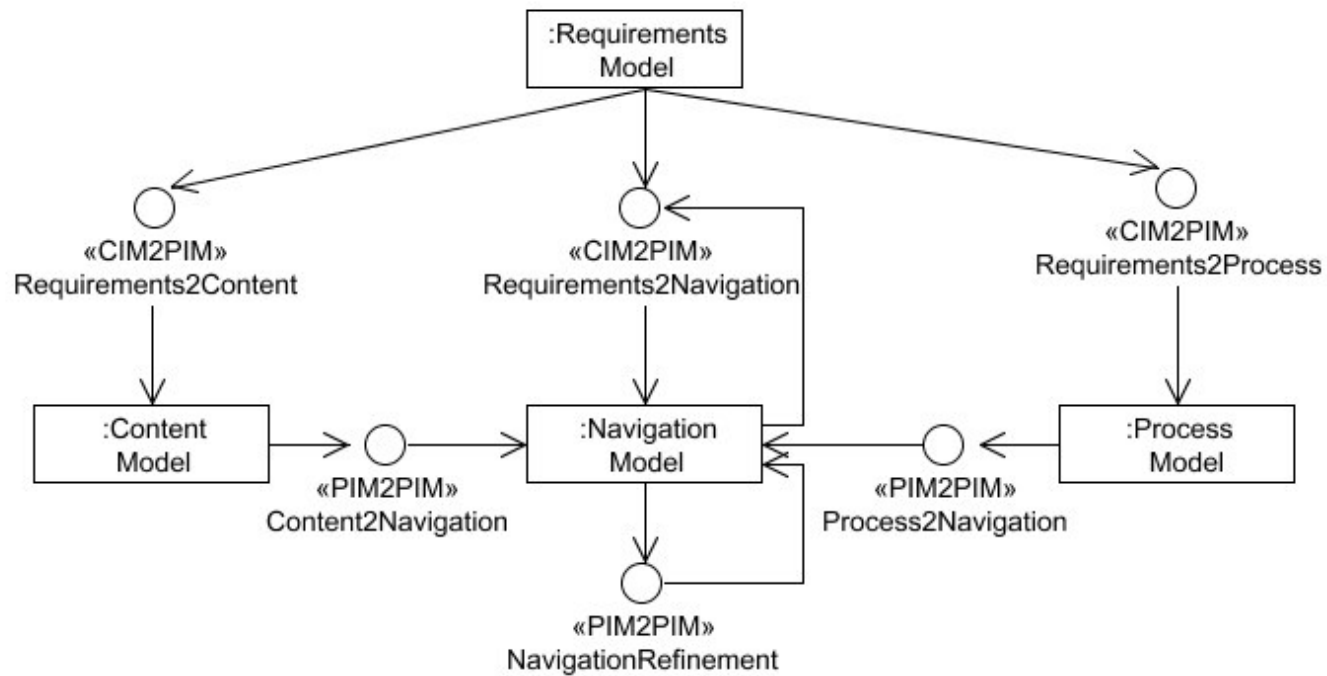
- process as UML activity diagram
- model transformations as stereotyped UML actions
- models as UML object flow states
- implicit initial and final state

- Types of models in UWE

- requirements model (CIM)
- functional models (PIM)
 - content model
 - navigation model
 - ...
- architecture models (PIM)
- integration models (PIM)
- models for J2EE, .Struts (PSM)

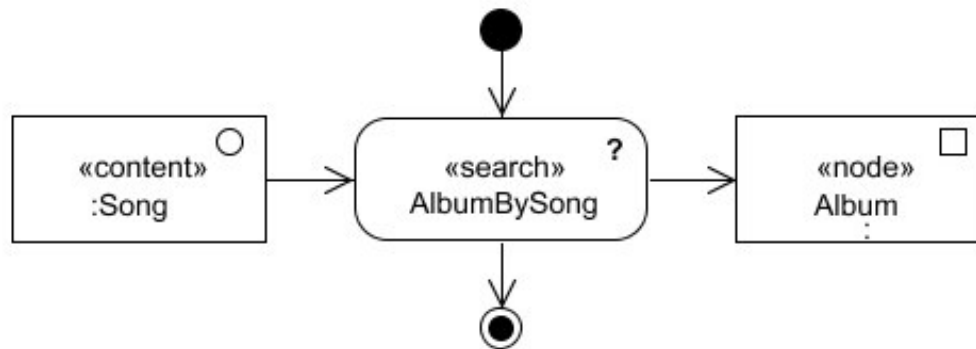
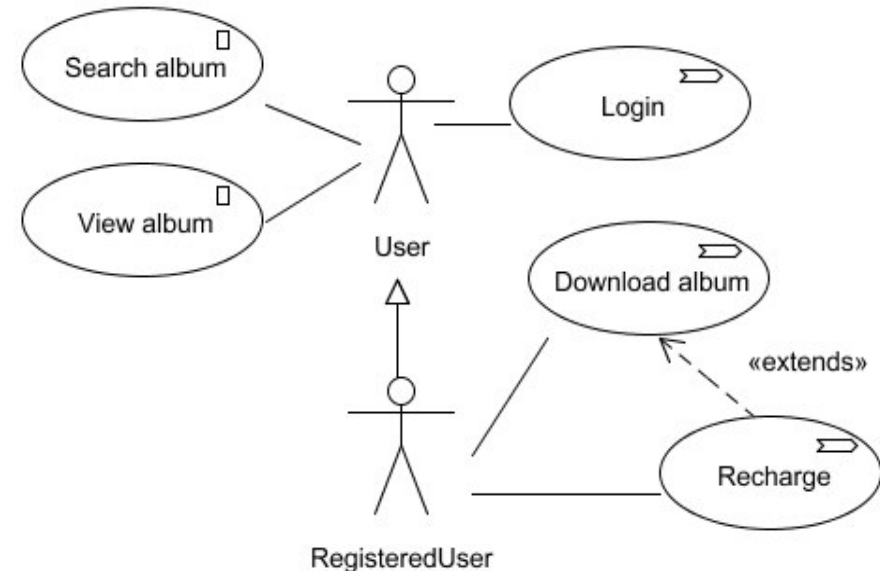
UWE Development Process

Requirements to Functional Models



Music Portal Example (excerpt)

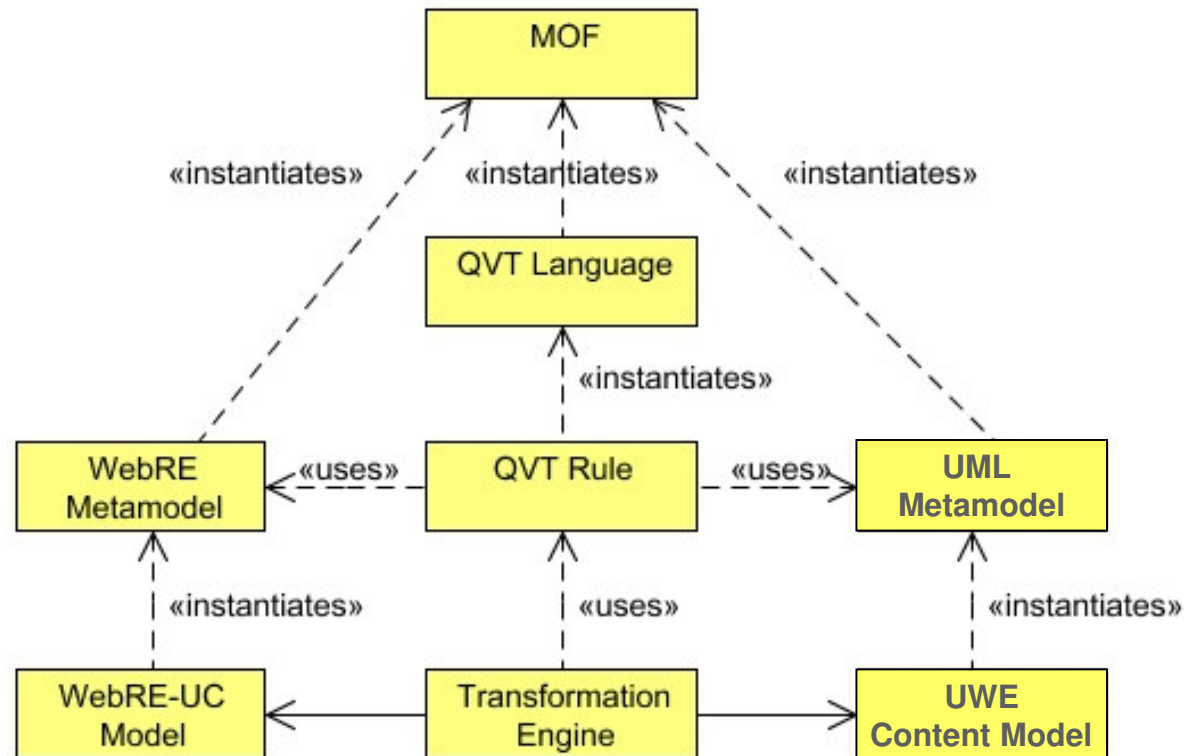
- Requirements model
 - UML use case diagrams
 - UML activity diagrams
- Web requirements engineering profile (WebRE)
 - WebUser,
 - navigation, WebProcess
 - browse, search, ...
 - content, node, ...



Query View Transformation Language (QVT)

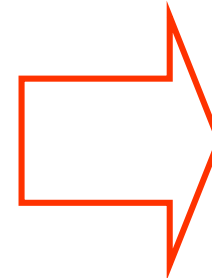
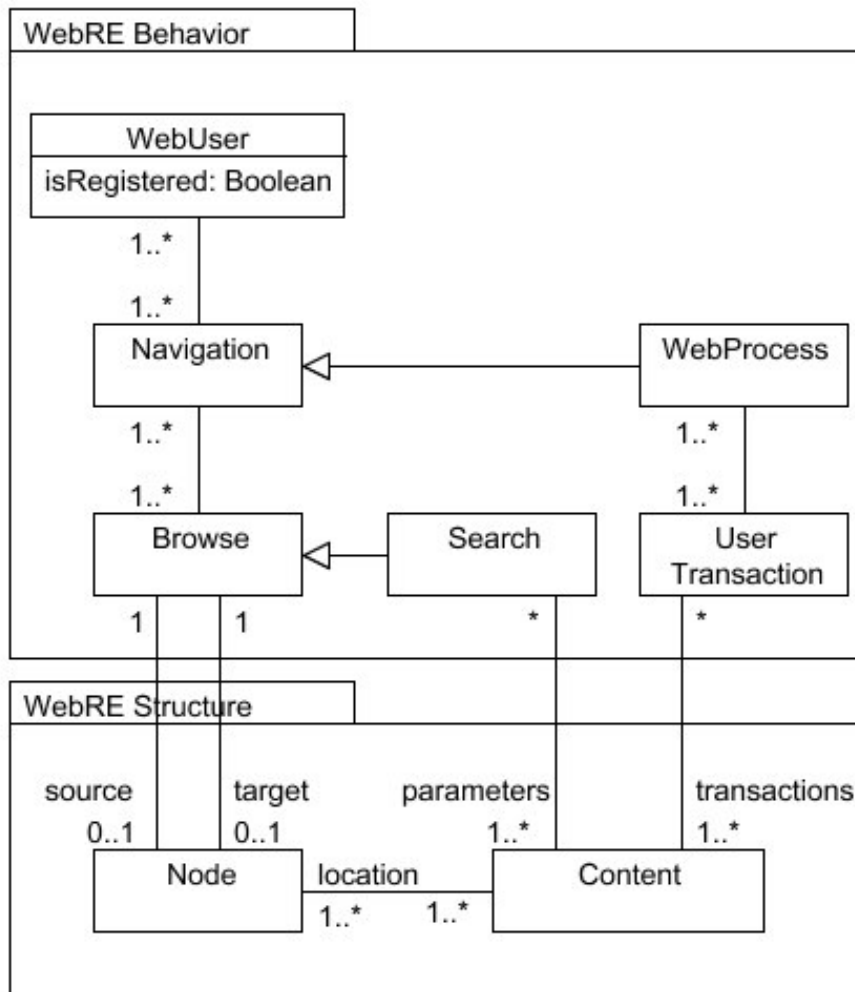
- Request for proposal launched by OMG
 - to define a standard way to transform source models into target models
 - compatible with the MDA recommendation suite: UML, MOF, OCL, etc.
- Ideas in this proposal
 - source and target models may conform to arbitrary MOF metamodels
 - transformation program is considered itself as a model, and as a consequence also conforms to a MOF metamodel
- QVT comprises three languages
 - relations, core and operational mappings
- QVT relations and core are declarative languages
 - relations language has a graphical concrete syntax
- QVT operational mapping language is an imperative language
 - that extends both QVT relations and QVT core
 - syntax of the QVT operational mappings language provides constructs commonly found in imperative languages (loops, conditions, etc.)
- Partial QVT implementations
 - SmartQVT (op), OptimalJ (core), ATL (QVT-like)

Model Transformations Pattern

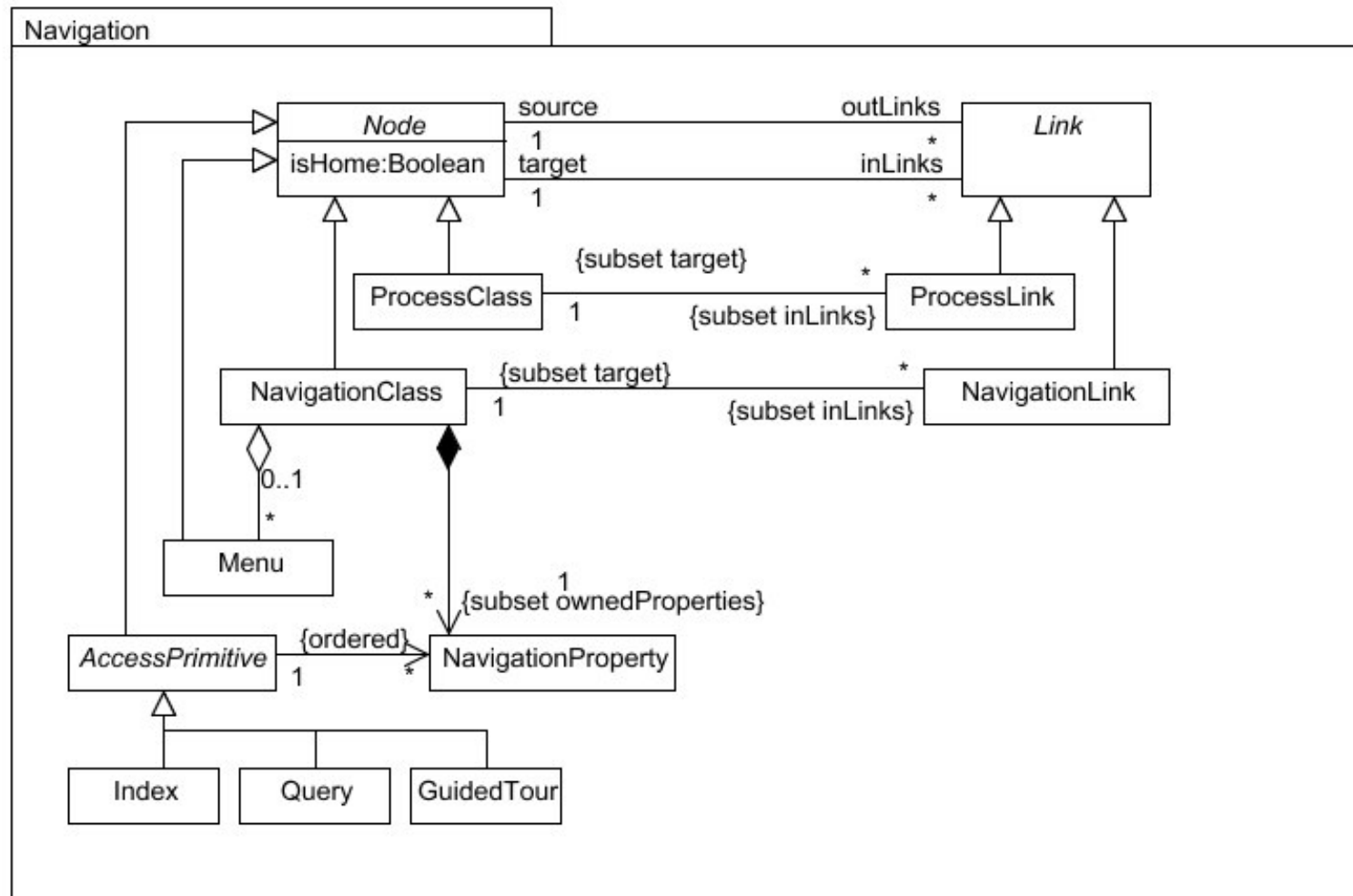
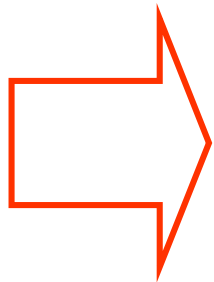


Model transformation pattern (J. Bézivin, 2004)

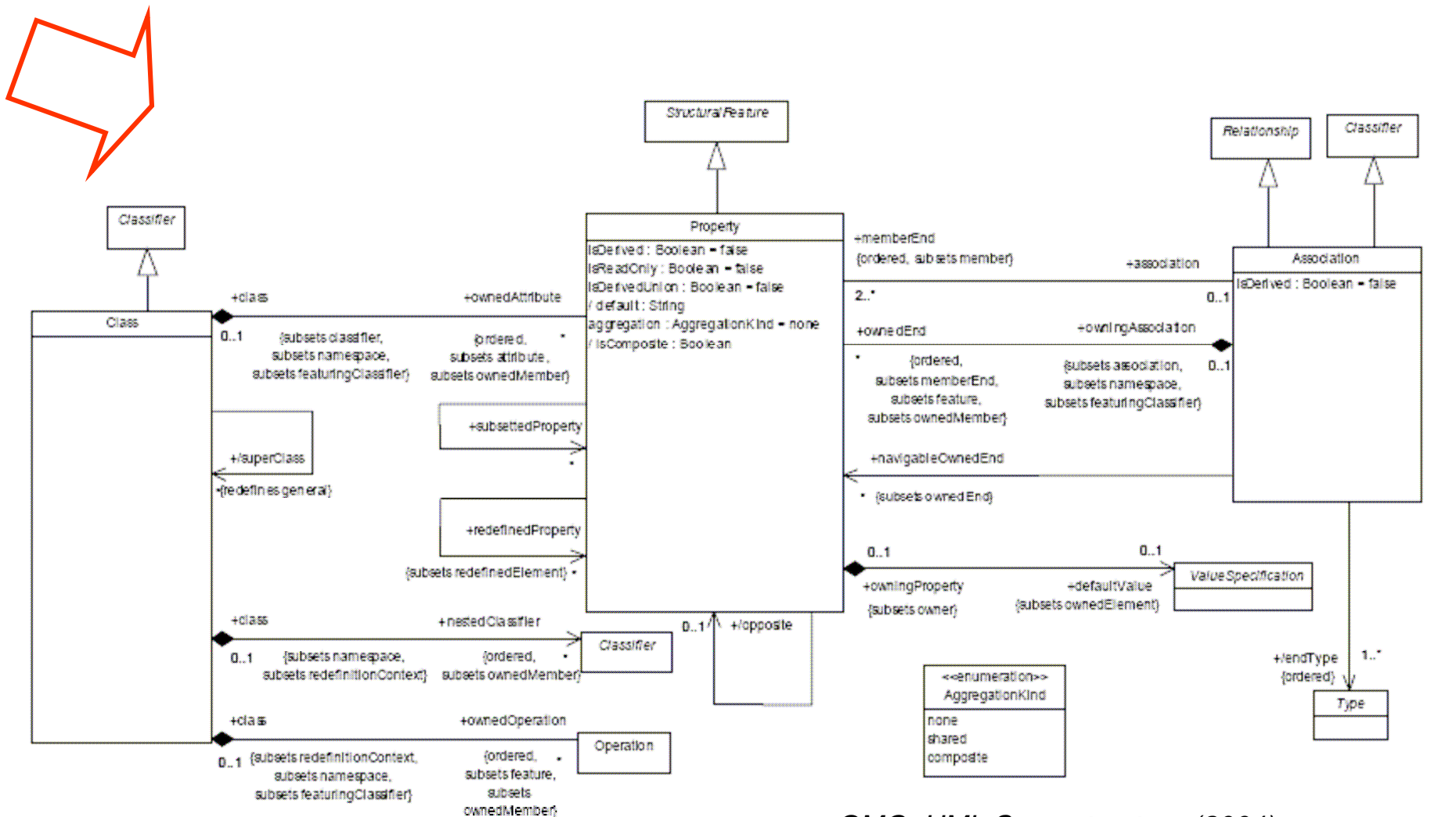
WebRE Metamodel



UWE Metamodel: Navigation Package



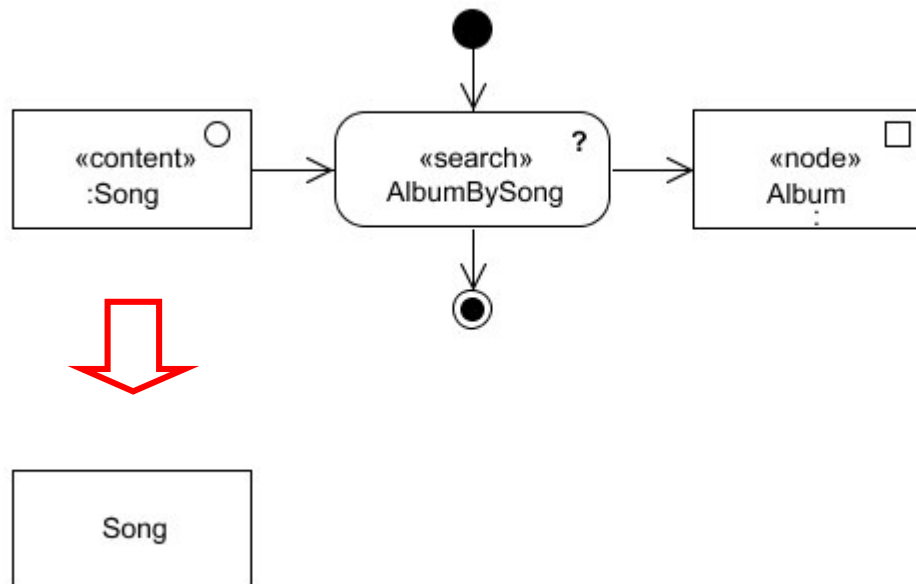
UML Metamodel



OMG, UML Superstructure (2004)

Transforming Requirements to Content (1)

- Source: requirements model (UML activity diagram)
 - objects input for Web actions
 - objects result of Web actions
- Target: content model
 - classes of the content model
- Profile-based transformation



Transforming Requirements to Content (2)



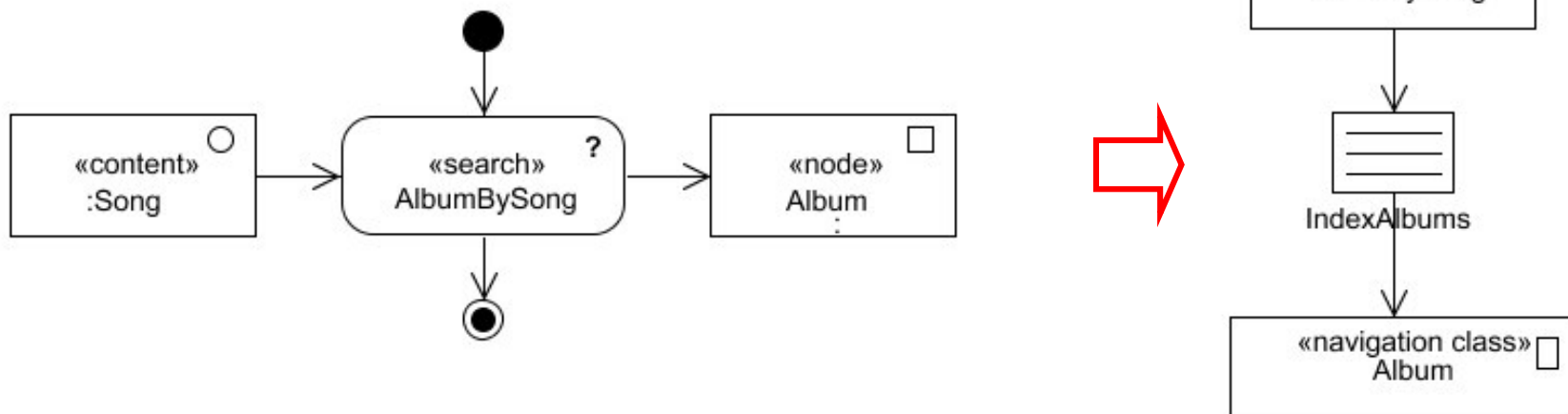
- Model transformation language
 - Query View Transformation (QVT)
 - textual notation
- Transformation rule

```
transformation ReqContent2ContentClass (webre:WebRE, uwe:UWE) {
  top relation R1 {
    checkonly domain webre  c: Content {name = n};
    enforce    domain uwe    cc: Class {name = n}; }
  top relation R2 {cn: String;
    checkonly domain webre  p: Property {namespace=c:
                                     Content{}, name = cn};
    enforce domain uwe  p1: Property {namespace = cc: Class{};
                                     name = cn }

    when {R1 (c, cc);}
  }
}
```

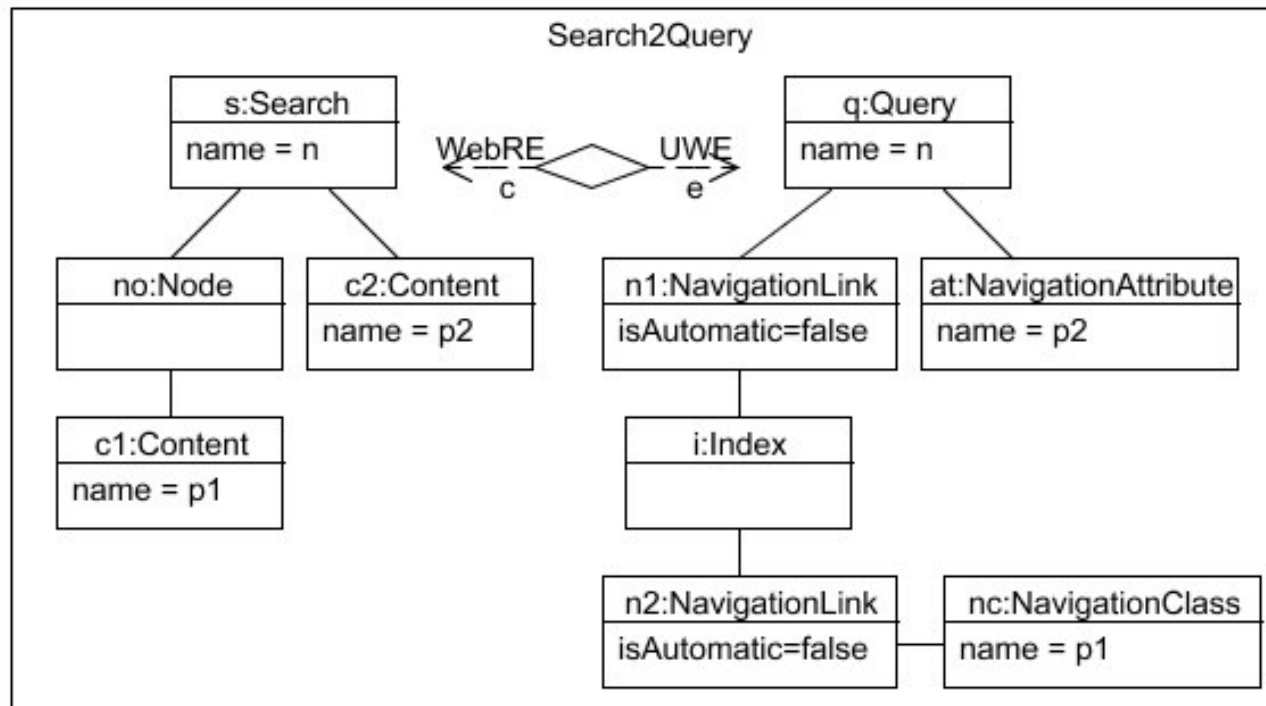
Transforming Requirements to Navigation Elements (1)

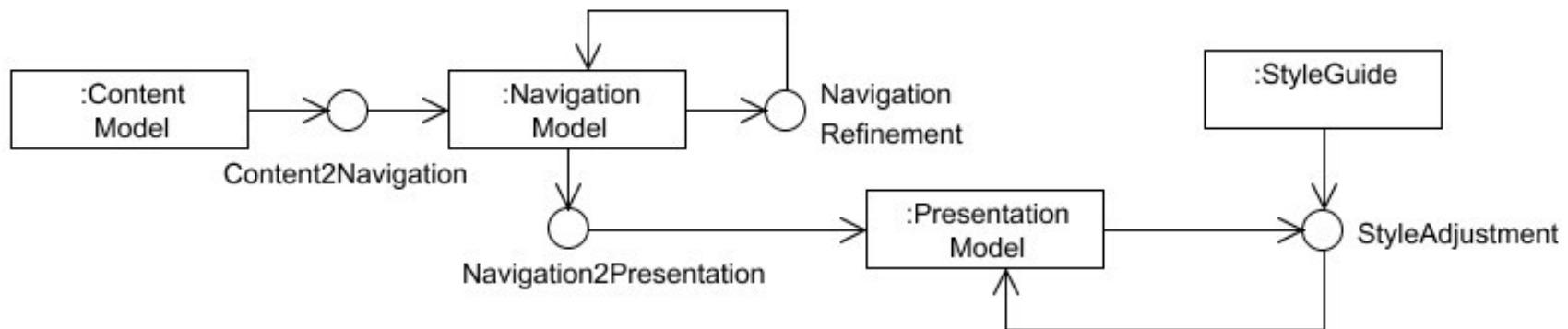
- Source: requirements model (UML activity diagram)
 - search action
 - content and node object flow states
- Target: navigation model
 - query and index class
 - navigation class
 - navigation link
- Pattern-based transformation



Transforming Requirements to Navigation Elements (2)

- Model transformation language
 - QVT
 - graphical notation
- Transformation rule

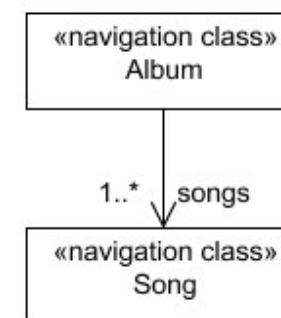
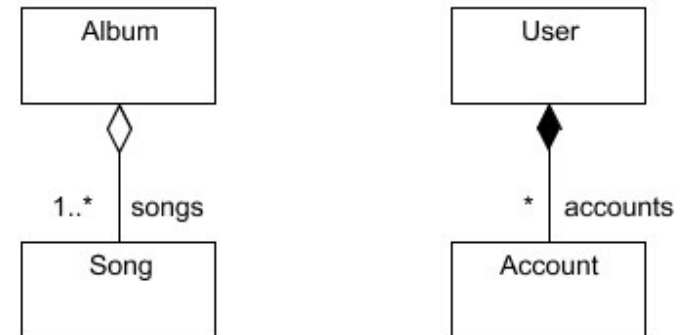




- UWE metamodel and UWE profile
 - navigation elements: *navigation class, navigation link, index, ...*
 - presentation elements: *presentation class, anchor, image, ...*
- Case tool ArgoUWE
 - extension of ArgoUML
 - provides stereotypes
 - supports (semi-)automatic execution of transformations

Transforming Content to Navigation (1)

- Content2Navigation
 - generates navigation classes from content classes
 - adds a navigation links based on associations of the content model
- Marking elements
 - identification of classes of the content model that are relevant for the navigation view
 - task performed by designer
- Semi-automatic transformation



ATLAS Transformation Language (ATL)

- ATL is a model transformation language
 - developed at INRIA to answer the QVT Request for Proposal
- ATL can be used for syntactic or semantic translation.
 - It is built on top of a model transformation Virtual Machine
- ATL development toolkit
 - plugin available in open source from the GMT Eclipse Modeling Project (EMP)
 - implements the MOF Query/View/Transformation language QVT

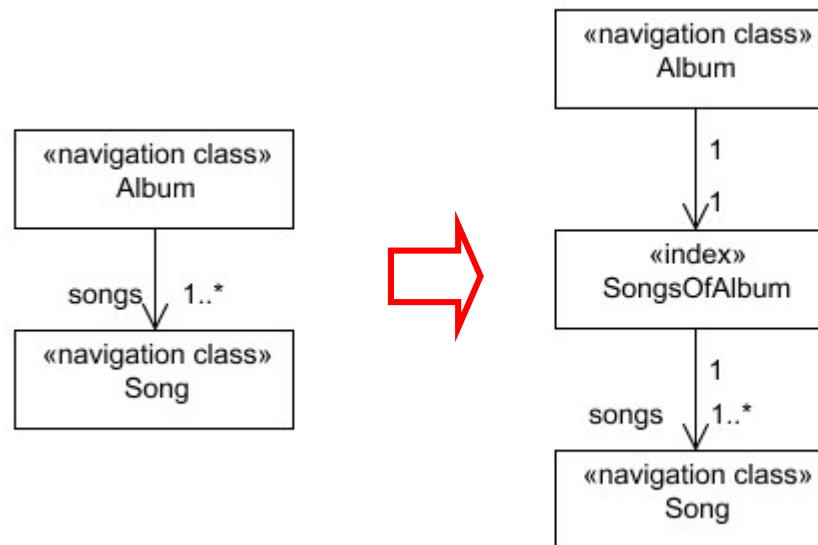
Transforming Content to Navigation (2)

- Implementation
 - Java within CASE tool ArgoUWE
 - ATL (ATLAS Transformation Language)
- ATL transformation rule

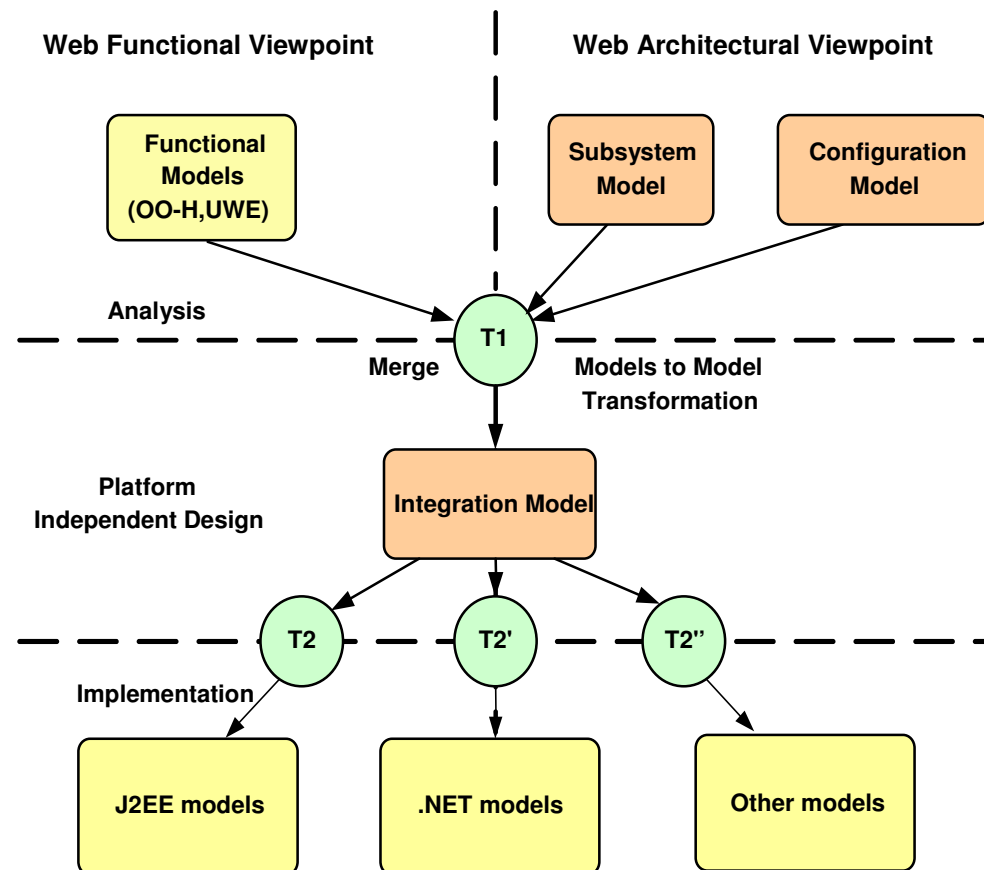
```
rule Class2NavigationClass {  
  from c : UML!Class ( c.ocIsTypeOf( UML!Class ) )  
  to nc : UWE!NavigationClass (  
    name <- c.name,  
    ownedAttribute <- c.ownedAttribute->select( p |  
      p.association.ocIsUndefined() ) )  
}
```

Refinement of Navigation Model

- Improvement based on patterns
 - index for associations with multiplicity greater than one at the directed association end
 - menu for navigation classes with multiple outgoing associations
- Implementation
 - Java in ArgoUWE
 - ATL

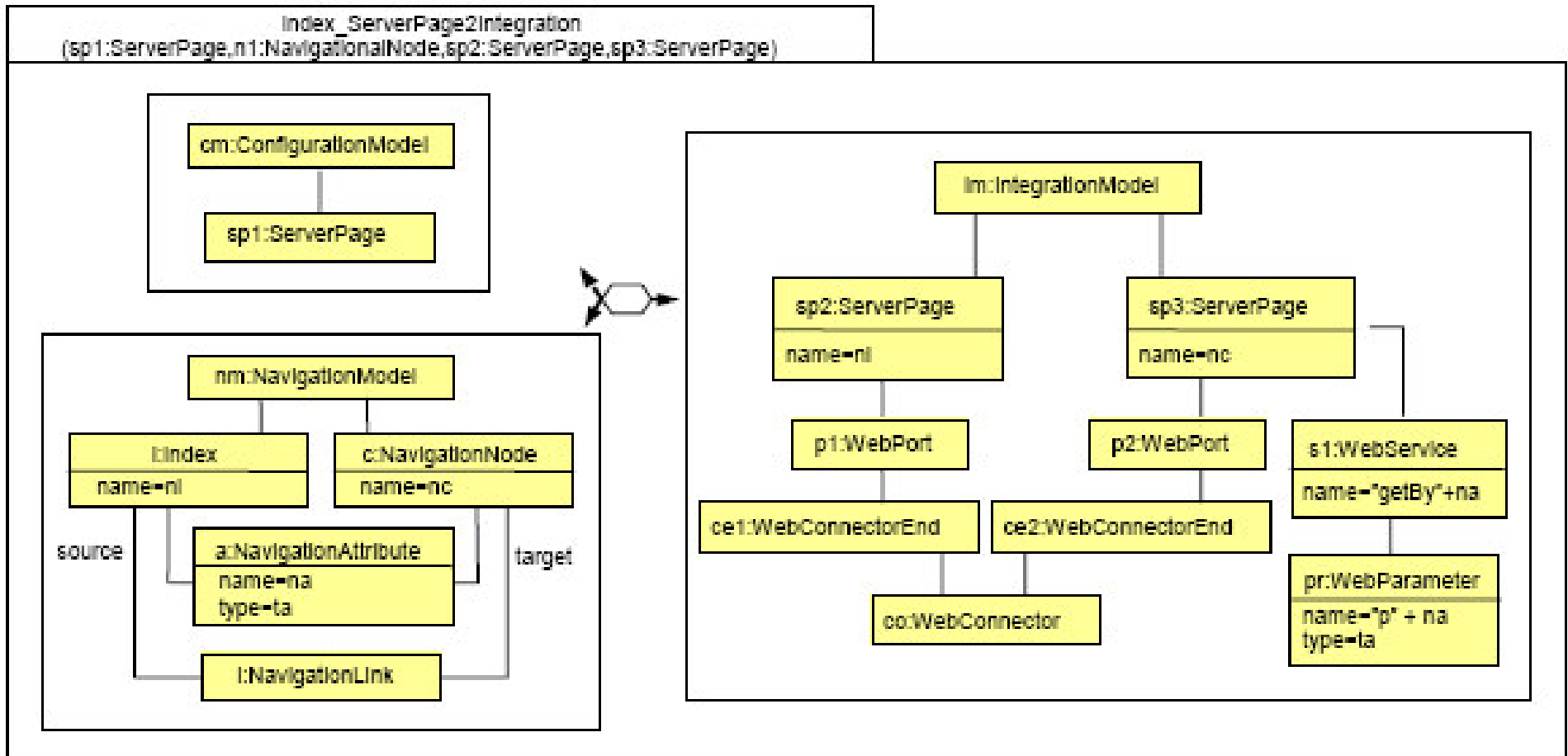


- Web Software Architecture (WebSA) approach*
 - domain specific language for modelling architectural views of Web applications
 - subsystem model
 - configuration model
 - integration model
 - UML profile of architectural modelling elements
 - Web component
 - Web port
 - Web connector
 - server page,
 - etc.
 - QVT-like transformations



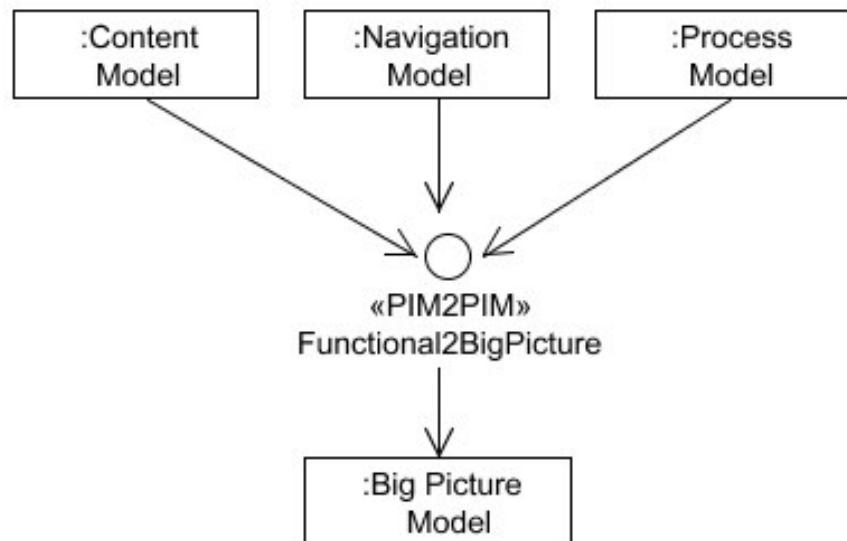
* Santiago Melía, University of Alicante, PhD Thesis (2007)

Example WebSA



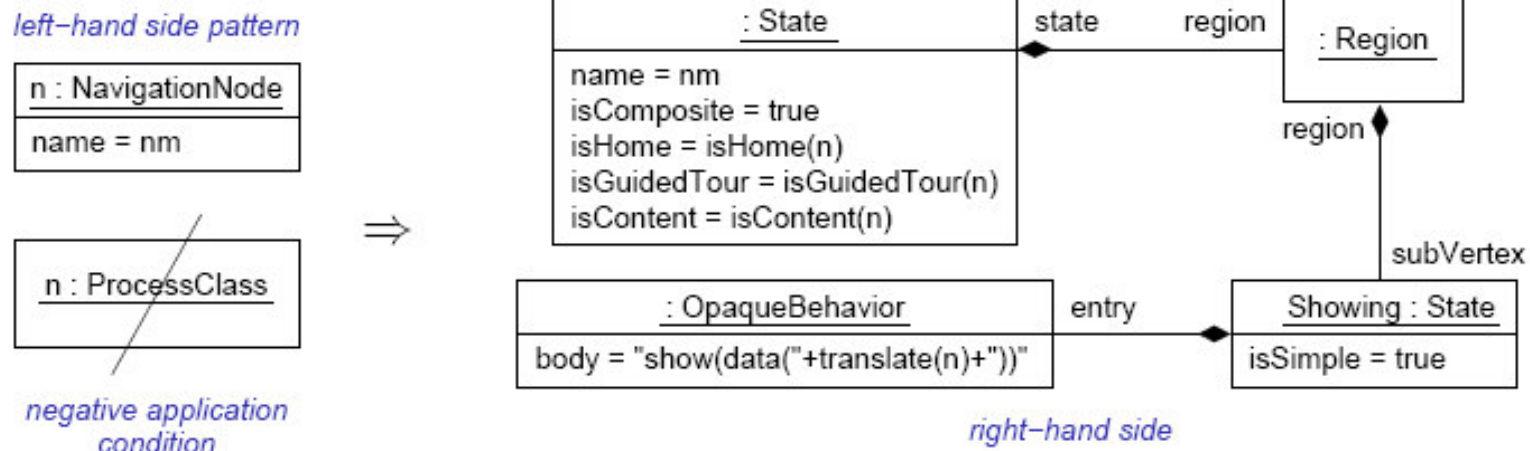
Generating “Big Picture” Model

- Generation of an integrated functional model (“big picture”)
 - transformation target UML state machine for integration of content, navigation and process models
 - graph transformation language
 - tool: Attributed Graph Grammar System (AGG)
 - validation of correctness by model checking

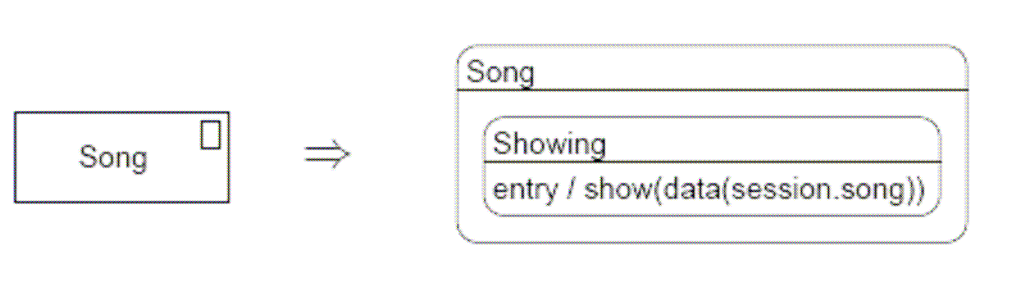


Big Picture: Transformation of Navigation Model

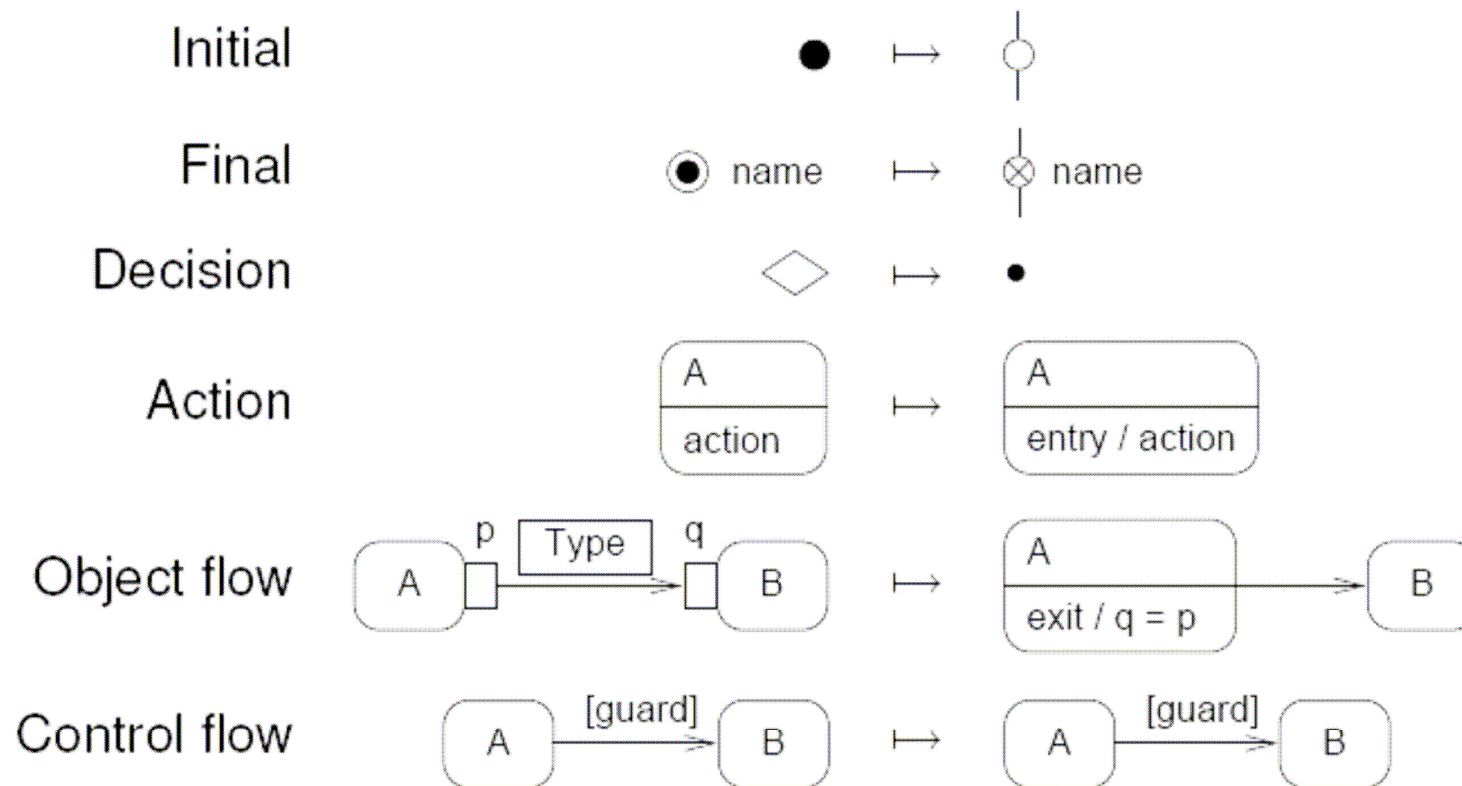
- capture navigation nodes as states (with parameters for data)



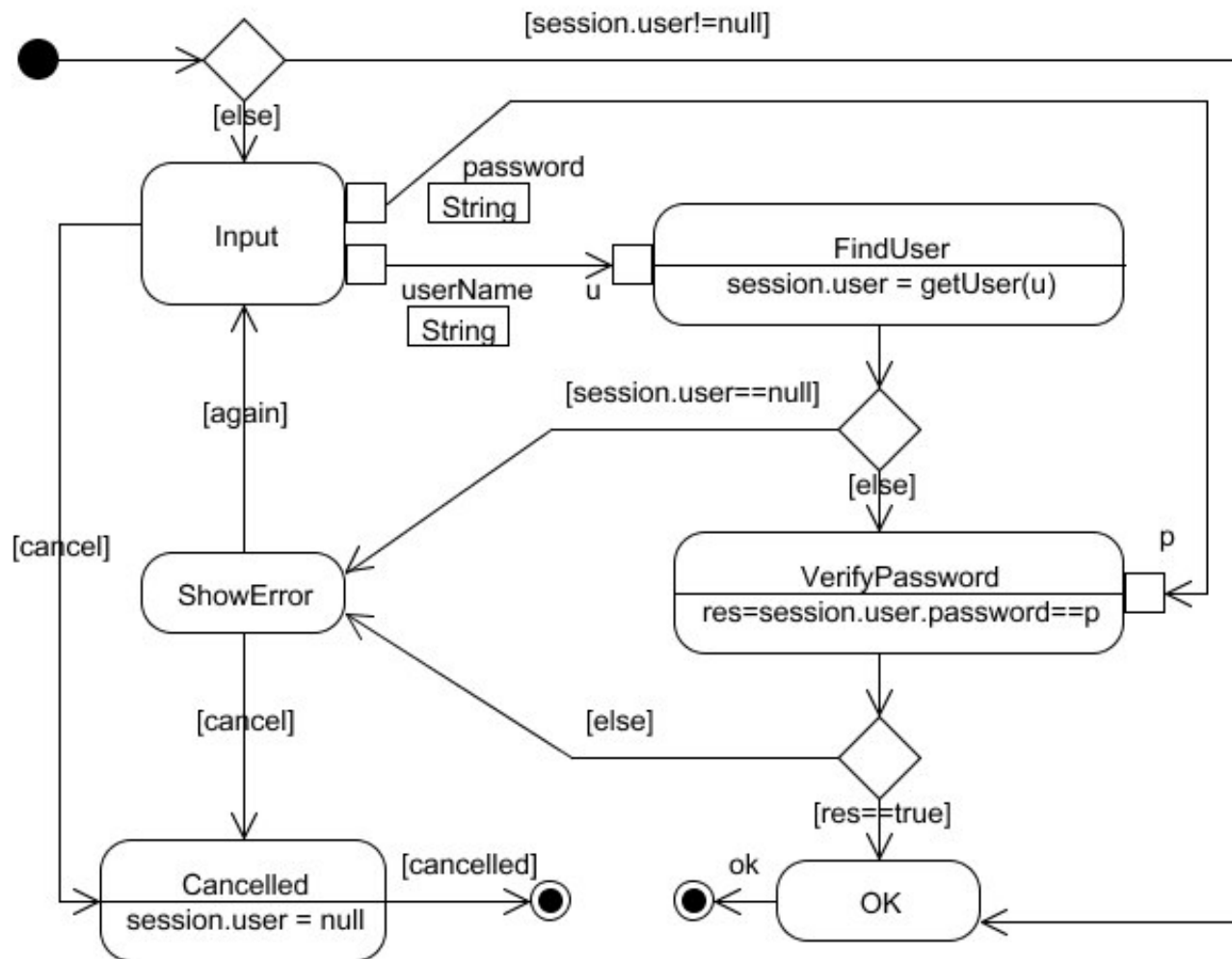
- Example: music portal: transformation for navigation node “song”



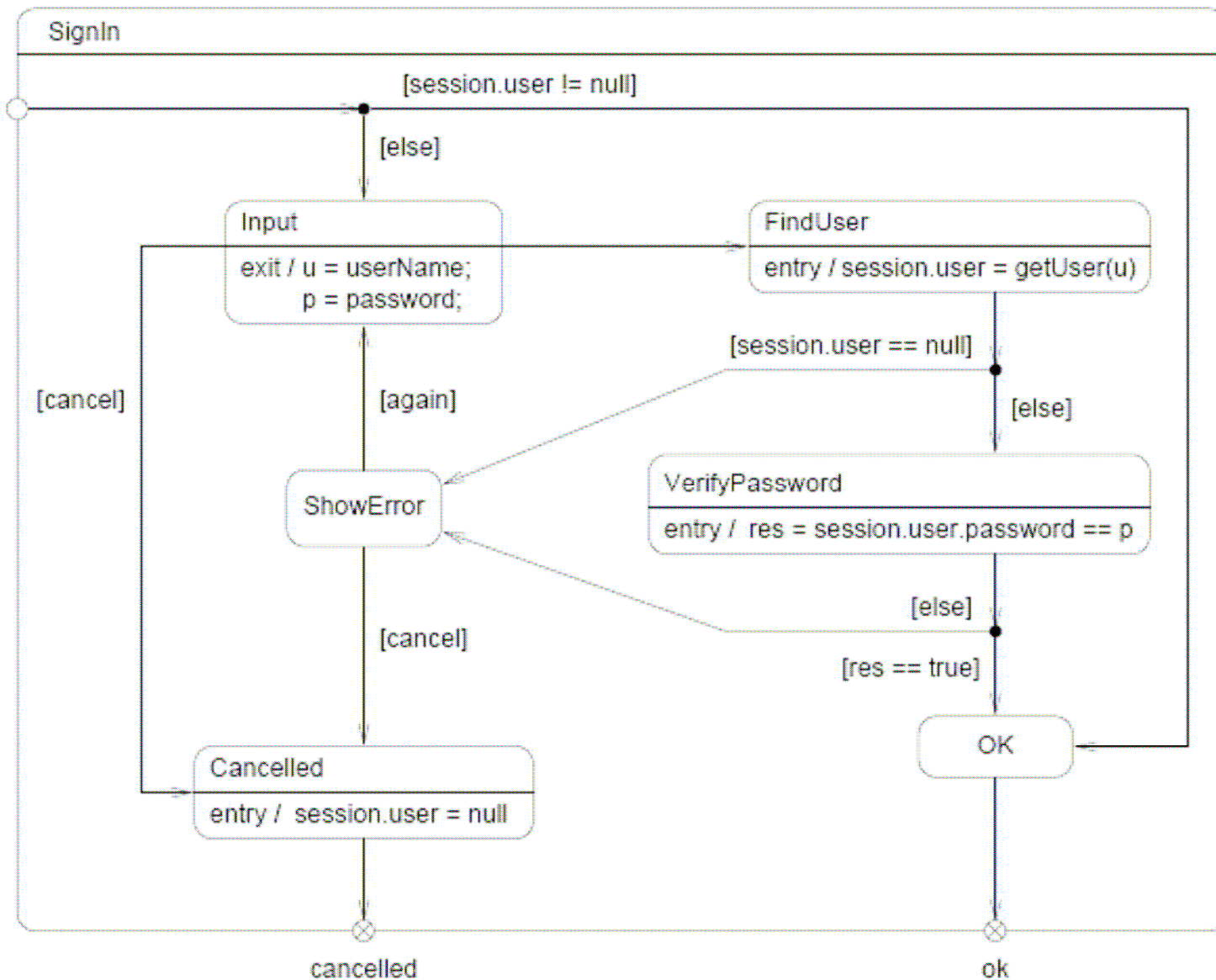
Big Picture: Transformation of Business Process



Process Model: UML Activity Diagram: Login

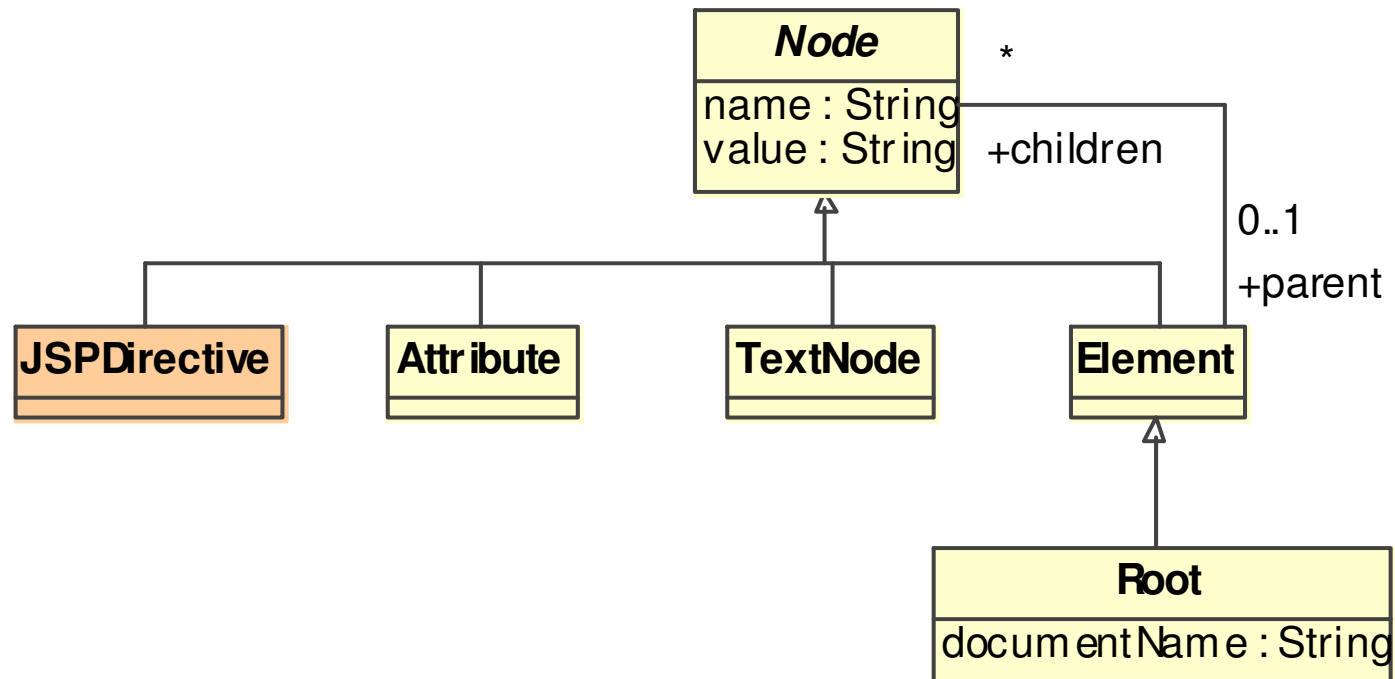


Big Picture: Example Transformation of Business Process



Generation of Web Applications in UWE

- UWE uses a transformational approach
 - to generate data model and presentation layer
 - based on content, navigation structure and presentation models
 - transformation rules from UWE content model to Java beans
 - transformation rules from UWE presentation model to Java Server Pages (JSPs)
- UWE uses an interpretational approach
 - using a virtual machine
 - to interpret the process model (activity diagrams)
 - configuration data for the virtual machine is generated from process and navigation model
- Implemented so far
 - using the Spring framework
 - transformations defined in ATLAS Transformation Language (ATL)



Example: Code Generation in UWE



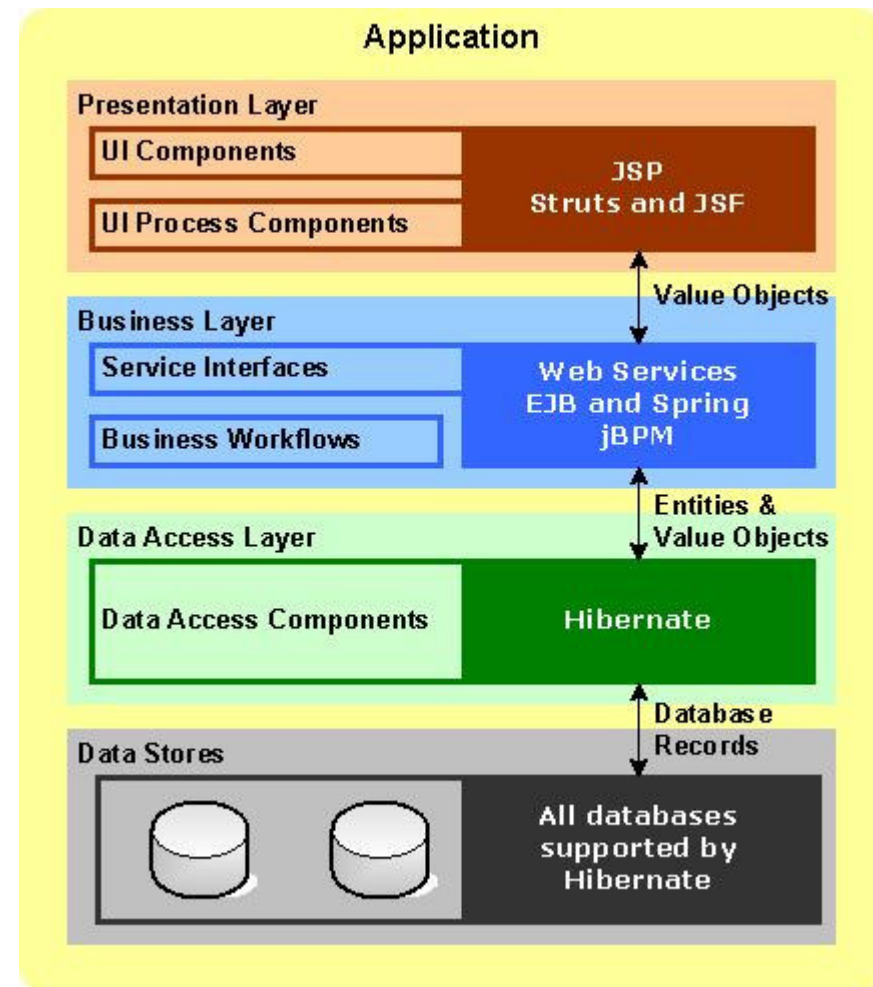
- Generation of an JSP element for UWE presentation element
 - ATL Transformation

```
rule PresentationClass2JSP {
  from
    pc : UWE!PresentationClass
  to
    jsp : JSP!Root(documentName <- pc.name + '.jsp',
      children <- Sequence{ htmlNode })),
    htmlNode : JSP!Element(name <- 'html',
      children <- Sequence{ headNode, bodyNode })),
    headNode : JSP!Element(name <- 'head',
      children <- Sequence{ titleNode })),
    titleNode : JSP!Element(name <- 'title',
      children <- Sequence{ titleTextNode })),
    titleTextNode : JSP!TextNode(value <- pc.name),
    bodyNode : JSP!Element(name <- 'body',
      children <- Sequence{ pc.ownedAttribute->collect(p |
p.type) })
}
```

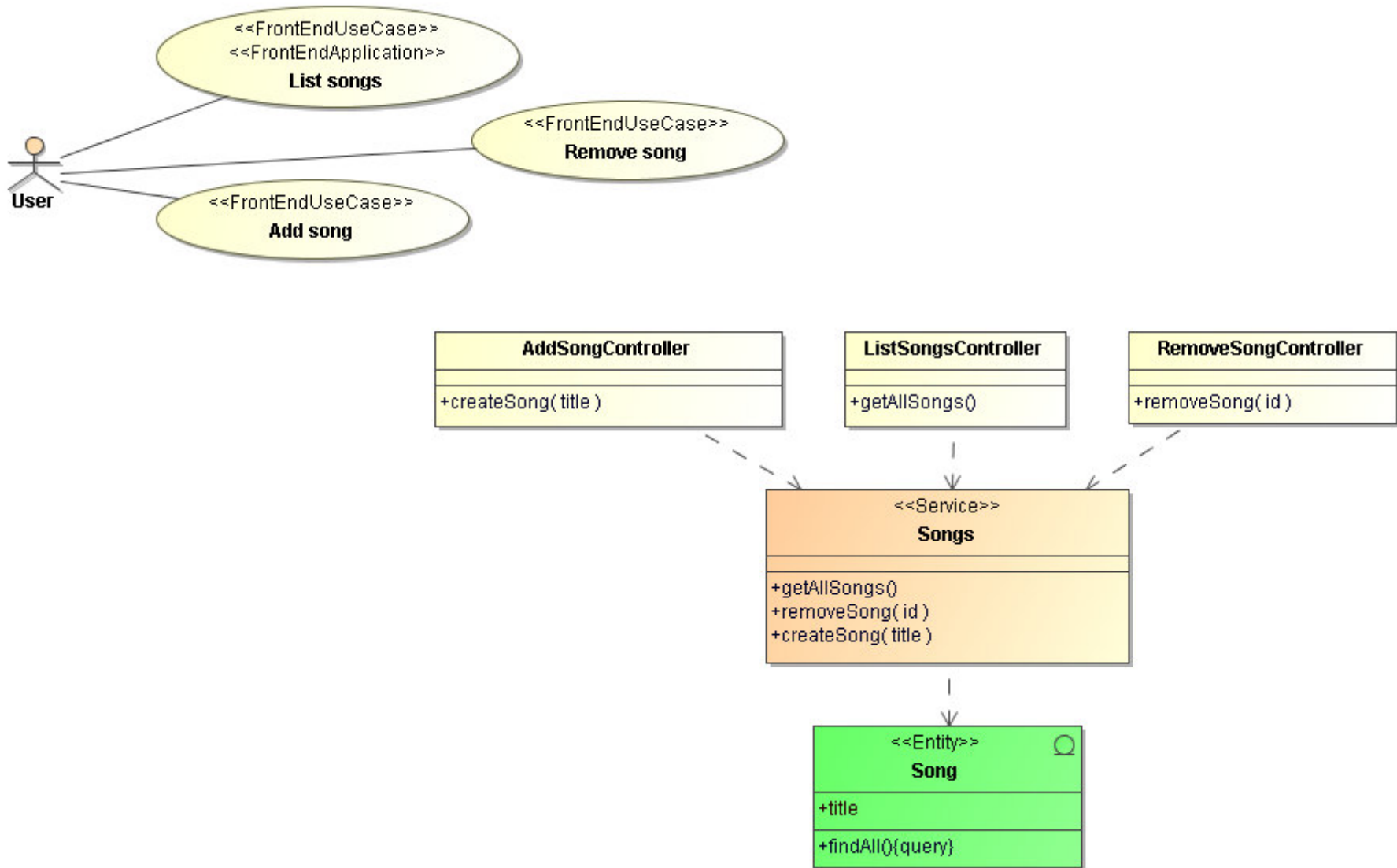
Generation of Web Applications in WebRatio

- WebRATIO graphical editor
 - produces an internal representation of the WebML models in XML
- WebRATIO integrated EasyStyler presentation designer
 - allows the user to define XSL style sheets from XHTML mockups
- WebRATIO code generator
 - translates XML specifications into application code
 - produces dynamic page templates (e.g. JSP file) to express content and mark-up of pages in HTML
 - produces unit descriptors containing dependency with data layer (name of database and SQL queries)
- WebRATIO runtime layer
 - is built on top of J2EE platform
 - executes the application code

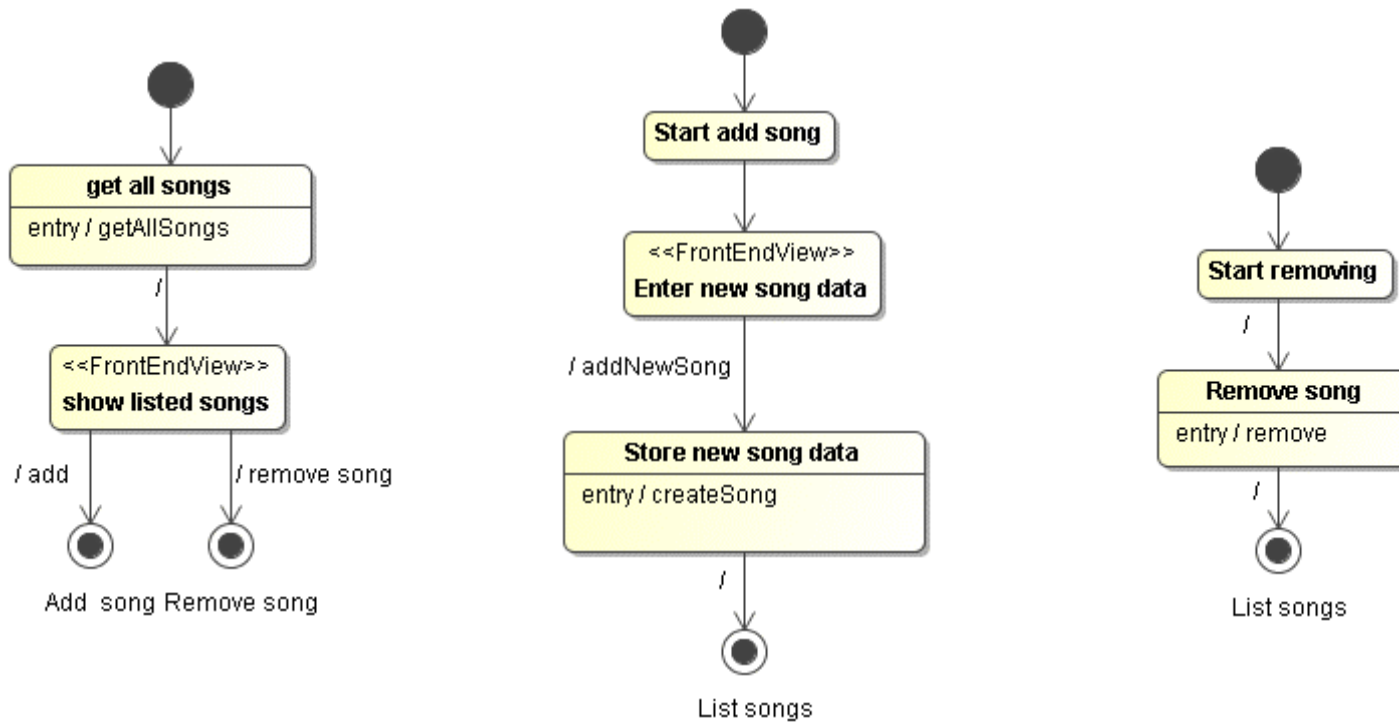
- Generator framework
 - transformation of UML (1.4) models
 - input validation using OCL
 - further meta-model planned for AndroMDA 4.0
- Transformations
 - model-to-model currently in Java
 - QVT and ATL planned for AndroMDA 4.0
 - model-to-code currently template based (Velocity)
 - MOFScript planned for AndroMDA 4.0
 - bundled as cartridges
 - EJB, Spring, Hibernate, Struts, JSF, ...



AndroMDA: UML Models (1)



AndroMDA: Models (2)



AndroMDA: Manual Coding



```
public class SongsImpl extends SongsBase {
    protected java.util.Collection handleGetAllSongs()
        throws java.lang.Exception{
        return this.getCarDao().findAll();
    }
    protected void handleRemoveSong(String id) throws Exception {
        this.getSongDao().remove(Long.valueOf(id));
    }
    protected void handleCreateSong(String title) throws Exception {
        this.getSongDao().create(title);
    }
}
```

Show listed

Show listed

Add

Add

Remove

Id

Remove

4 items found, displaying all items.

1

Id Songs

1 A

2 B

Export options: [CSV](#) | [Excel](#) | [XML](#) | [PDF](#)

Help

Latest News

AndroMDA 3.1

- Completely new engine core
- New cartridges, lots of new features
- Major performance improvements...

more »

Other links

List

Remove

Add

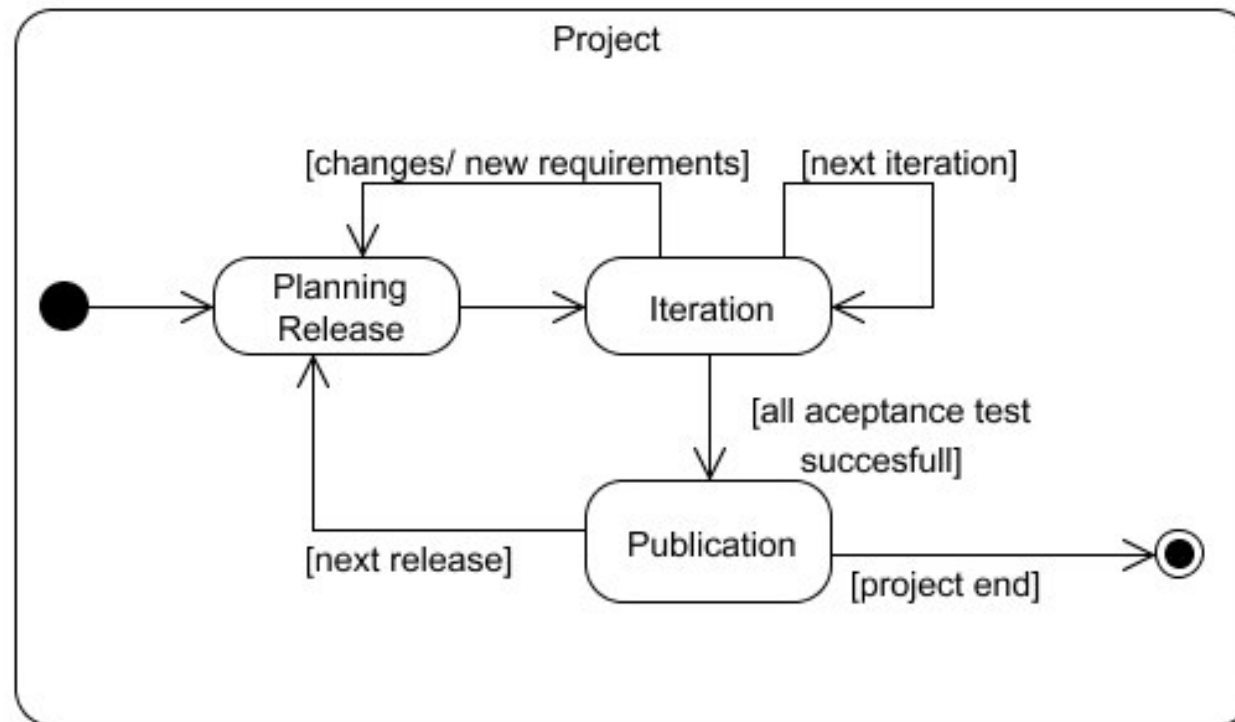
Help

This web application has been generated using AndroMDA's Bpm4Struts cartridge, check the Docs for more information.

The AndroMDA Team
© 2005

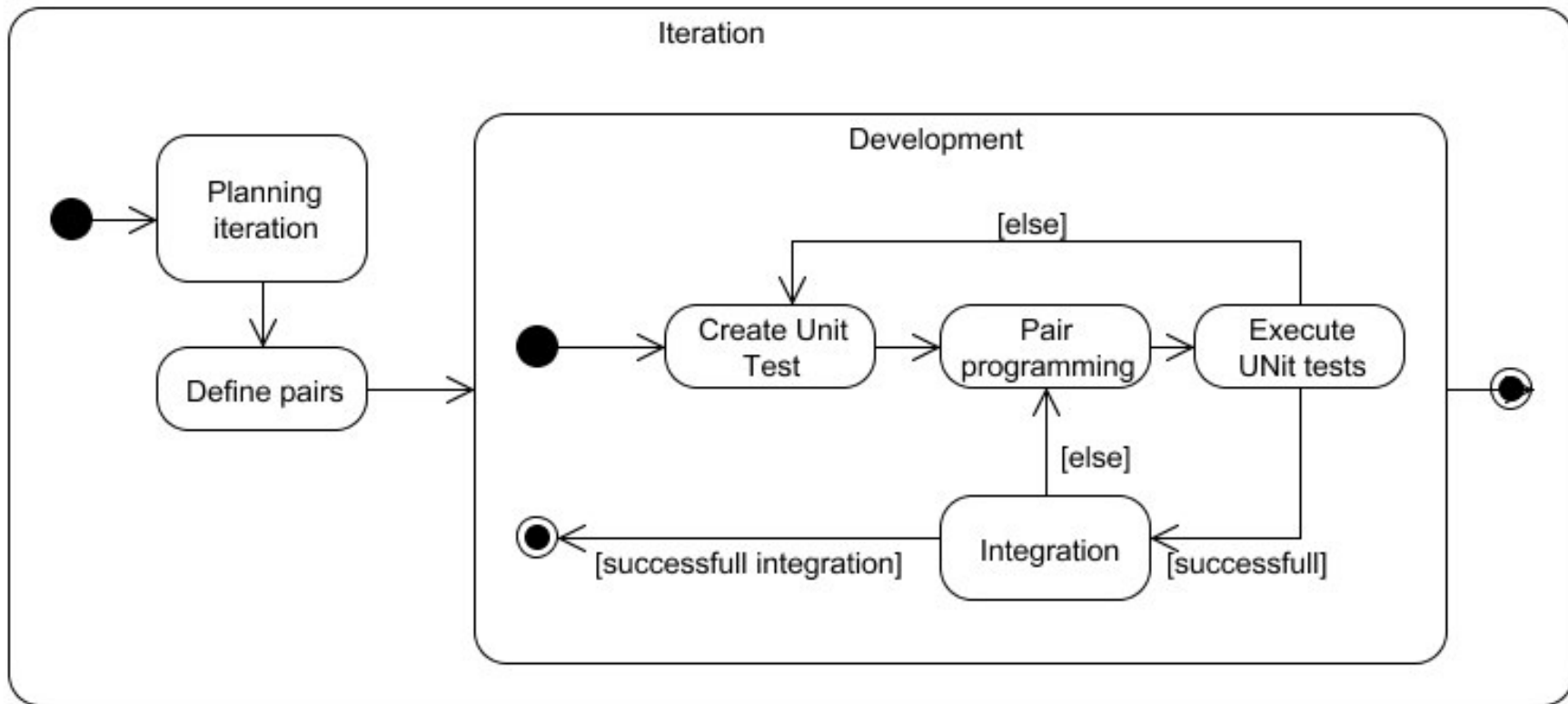
Agile Methods

- Agile methods are a family of development processes
 - XP- Extreme Programming
- Agile methods focus on adapting quickly to changing realities
- Iterative and light weighted processes (documentation)
- Agile Manifesto (2001)
 - individuals and interactions over processes and tools
 - working software over comprehensive documentation
 - customer collaboration over contract negotiation
 - responding to change over following a plan .



Engels, Lohmann & Wagner (2003)

XP Iteration



Engels, Lohmann & Wagner (2003)

- Web application development requires flexible and adaptable development processes
- Trend is model-driven development (MDD)
- Model transformation languages play an important role in MDD (QVT, ATL)

Literature

- **Transformation Techniques in the Model-Driven Development Process of UWE**
Nora Koch, ICWE'06 Workshops, 2006,
http://www.lcc.uma.es/~av/mdwe2006/camera_ready_papers/koch-mdwe-2006-final.pdf
- **Model-Driven Generation of Web Applications in UWE**
Andreas Kraus, Alexander Knapp and Nora Koch
3rd International Workshop on Model-Driven Web Engineering (MDWE 2007), Como, Italy, to appear
- **Query/View/Transformation (QVT) Language**
OMG. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification Final Adopted Specification.
<http://www.omg.org/docs/ptc/05-11-01.pdf>
- **ATLAS Transformation Language and Tool**
<http://www.eclipse.org/m2m/atl/doc/>
- **Attribute Graph Grammar System**
<http://www.eclipse.org/m2m/atl/doc/>
- **W3C. XSL Transformations (XSLT) Version 1.0**
www.w3.org/TR/xslt
- **ArgoUWE:** <http://www.pst.informatik.uni-muenchen.de/projekte/uwe/argouwe.shtml>
- **WebRATIO:** <http://www.webratio.com/>
- **AndroMDA:** <http://www.andromda.org/>