



Web Engineering

Modelling of Web Applications

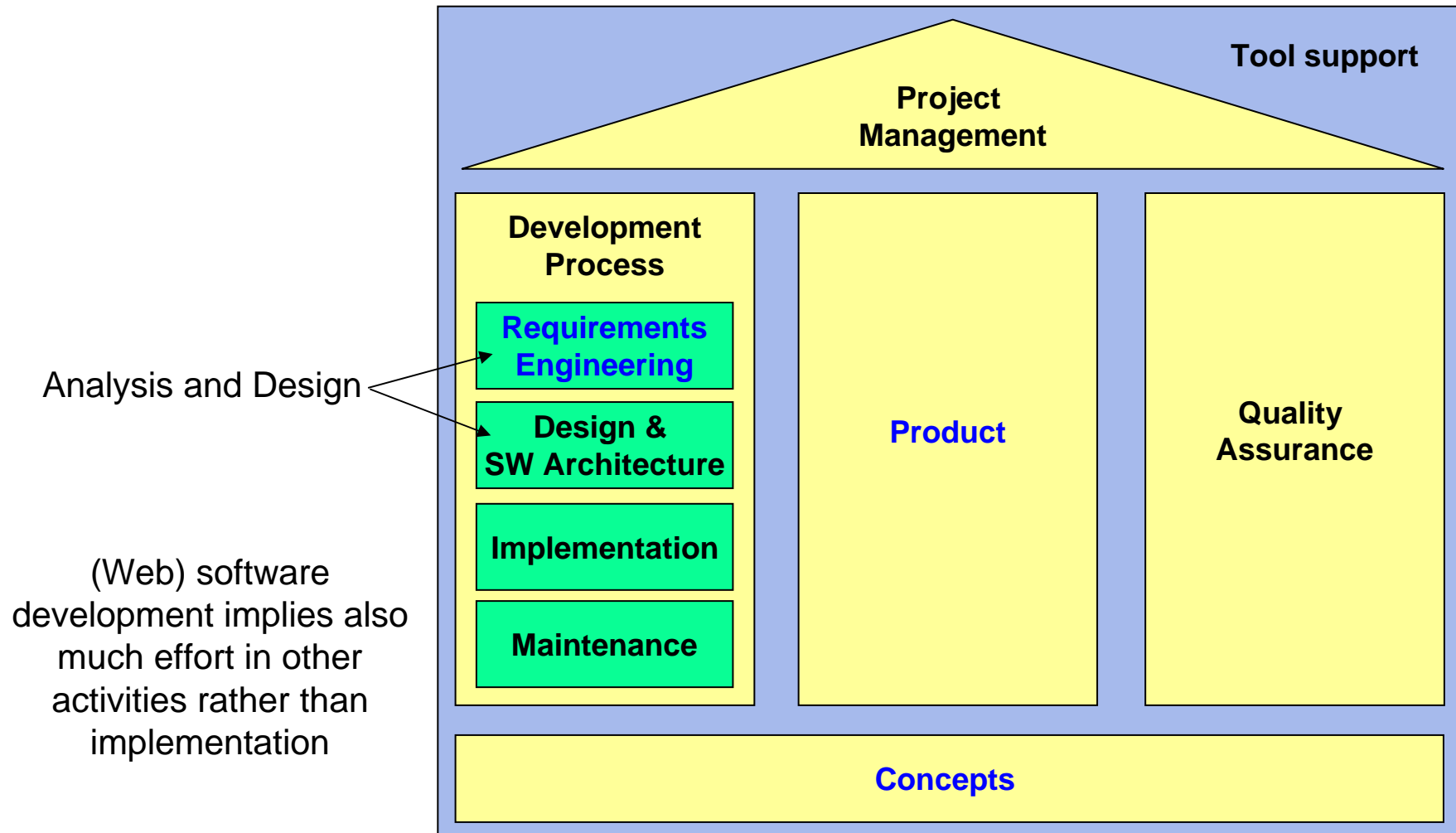
Prof. Dr. Alexander Knapp

Dr. Nora Koch

SS 07 (4) – 21.05.07

Ludwig-Maximilians-Universität München

Software/Web Engineering Areas



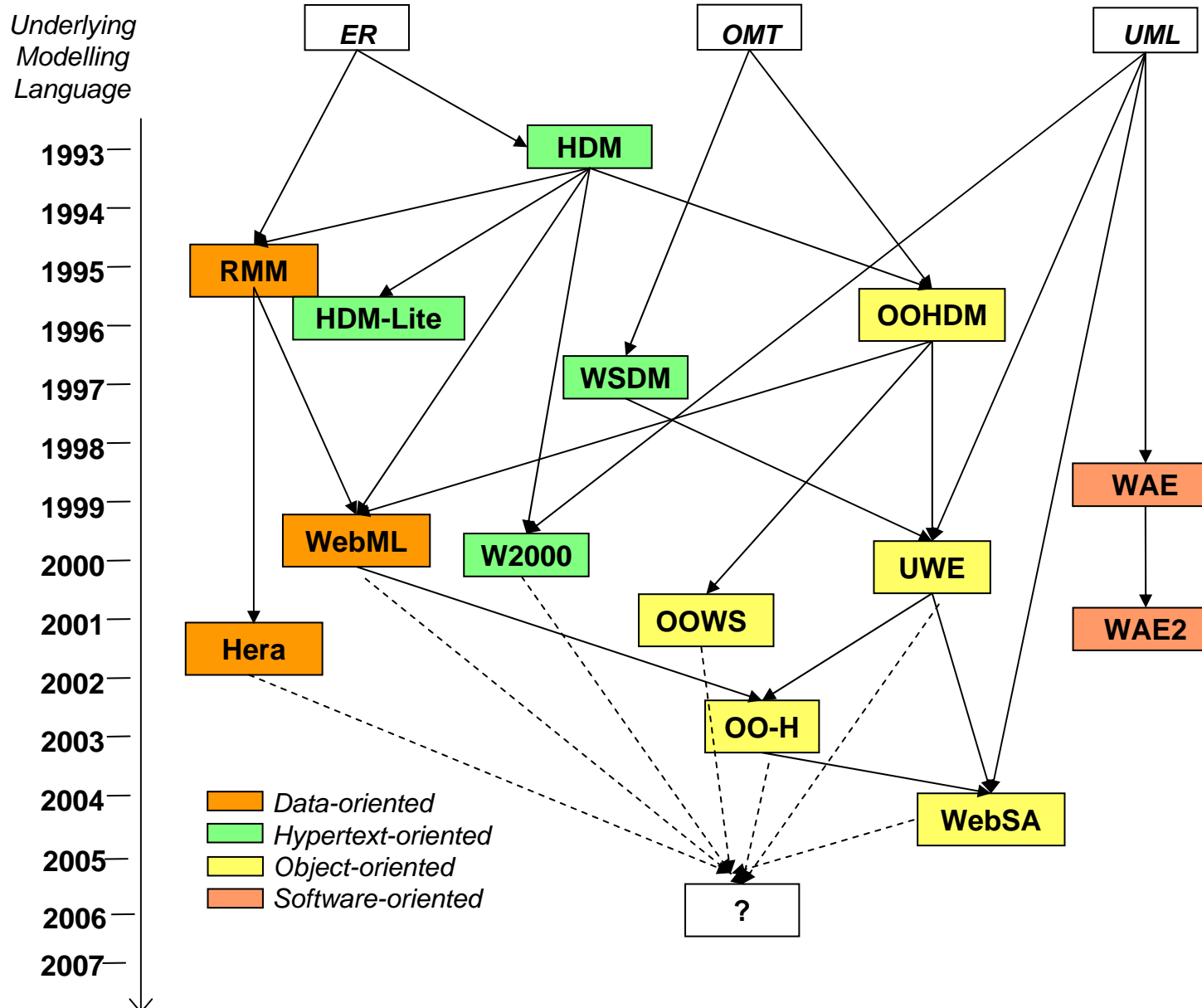
Contents

- Web Engineering Methods
- Modelling of Web Applications
- UWE Approach
 - Modelling Requirements
 - Modelling Content
 - Modelling Navigation
 - Modelling Process
 - Modelling Presentation
 - Modelling Adaptivity
- Metamodel and Profile for UWE
- WebML Approach (additional slides)

Methods for Development of Web Applications

- Describe the development process
 - steps
 - expected results
- Provide modelling techniques
 - construction of models
 - model transformations
- Provide tool support
 - building models
 - generation of applications

Methods for Web Engineering



Names of Methods

HDM / HDM-lite	Hypertext Design Model
Hera	
NDT	Navigational Development Technique
OO-H	Object-Oriented Hypermedia Method
OOHDM	Object-Oriented Hypermedia Design Method
OOWS	Object-Oriented Web Solution
RMM	Relationship Management Methodology
UWE	UML-based Web Engineering
W2000	
WAE / WAE2	Web Application Extension
WebML	Web Modeling Language
WebSA	Web Software Architecture
WSDM	Web Site Design Method

ER	Entity Relationship Model
OMT	Object Modeling Technique
UML	Unified Modeling Language

Problems of Current Methods

- Life cycle is only partially covered
 - focus on modelling
 - weak requirements analysis
 - poor support of (semi-)automatic generation of Web applications
- Mainly appropriate for development from scratch
 - weak support of reengineering & reuse
- Different notations and concepts
 - similar approaches with slight differences
 - “method war”
- Lack of use of standards
 - metamodeling (MOF, EMF)
 - modelling language (BPMN, UML, proprietary notation)

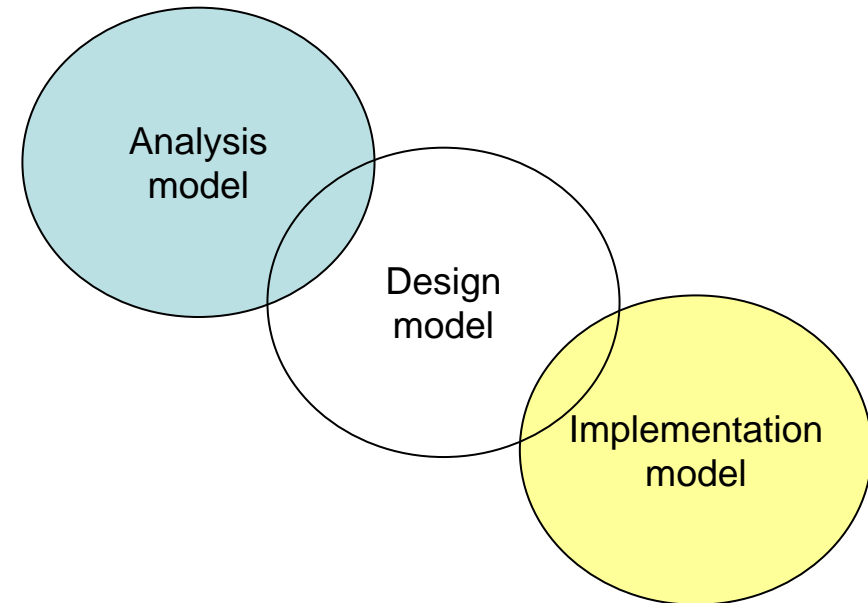


Field for research and development

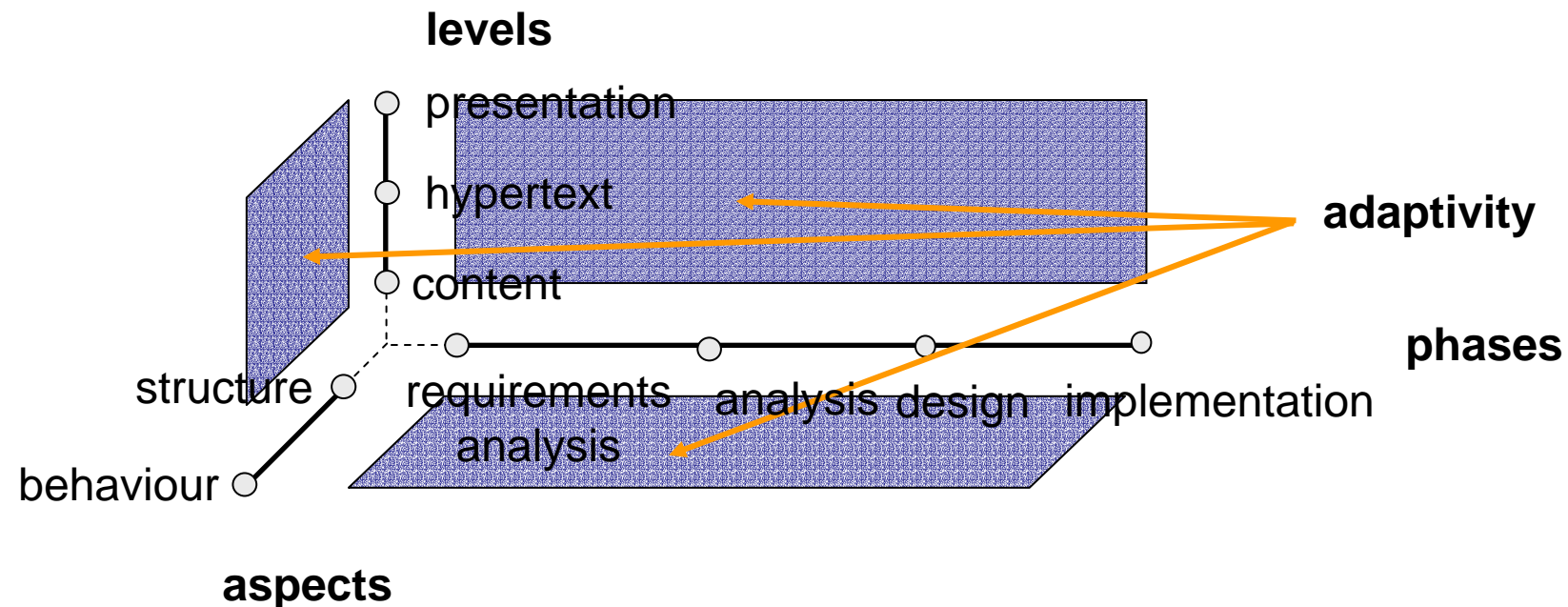
Analysis and Design of Web Applications

- Analysis Phase
 - specification of requirements
- Design Phase
 - representation of the software to be built
 - technology/platform independent
 - technology/platform dependent
- Implementation Phase
 - executable models
 - executable code

A **model** is a pattern, plan, representation, or description designed to show the structure or workings of an object, system, or concept.



- Modelling
 - is a technique used in analysis and design
 - uses
 - notation (syntax)
 - graphical representation
 - language
 - process for building models
 - rules
 - guidelines
 - graphical editors



■ Modelling process

- information-driven ("content first")
- presentation-driven ("layout first")
- functionality-driven ("test first") → agile process

Source: Kappel et al. *Web Engineering, d-punkt (2003)*


Advantages of Modelling Approaches

- A rigorous modelling approach
 - can reduce development efforts (cost and time)
 - allows a more structured development process
 - produces more usable and coherent final results
 - design models are self-documenting
 - immediate prototyping can be achieved

Requirements for Web Modelling

- Expressiveness
 - real-life cases should be expressible
 - frequently used design patterns should be captured
- Ease of use
 - intuitive notation
 - clear semantics
 - consistency checks
- Implementability
 - efficient mapping to physical data structures
 - effective generation of Web pages
 - flexible code generation from behavioural specifications

UWE Approach

- UWE is a an engineering approach for the development of Web applications
- Main characteristic is the use of UML for all models
 - “pure” UML whenever possible
 - UML extension for Web specific features  UML profile
- Focuses on **systematisation** and **automatic generation**
- UWE comprises
 - a **notation** for the graphical representation of the model elements
 - a **method** (technique) supporting semi-automatic generation
 - a **metamodel** for UWE modelling elements
 - a **process** supporting the development life-cycle of Web applications
 - a case-tool supporting the development of Web applications
 - **ArgoUWE** editor for the design
 - set of transformations for model-to-model and model-to-code transformations

Why UML?

- UML is a graphical language for specifying, constructing and documenting software artifacts
- UML is a **de facto industry standard** and an **OMG standard**
- UML includes
 - notation
 - diagram types
 - metamodel
 - Object Constraints Language (OCL) for invariants, pre- and post-condition
 - well-formedness rules
- UML does not provide a development process

How expressive is UML for the development of Web applications?

- UML does not include specific Web model elements
- UML defines extension mechanisms → **UML profiles**

UML Profiles

- An UML extension is called a **UML profile** if it only uses the extension mechanisms provided by the UML
 - stereotypes
 - tagged values
 - OCL constraints
- **UML profile = light weight extension**



eases
tool support

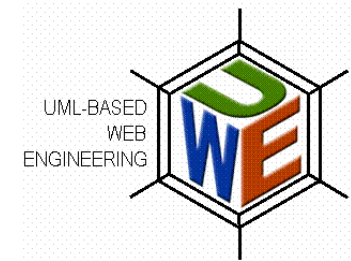


- **heavy weight extension**
 - uses other diagram types not defined in the UML
 - uses different notation

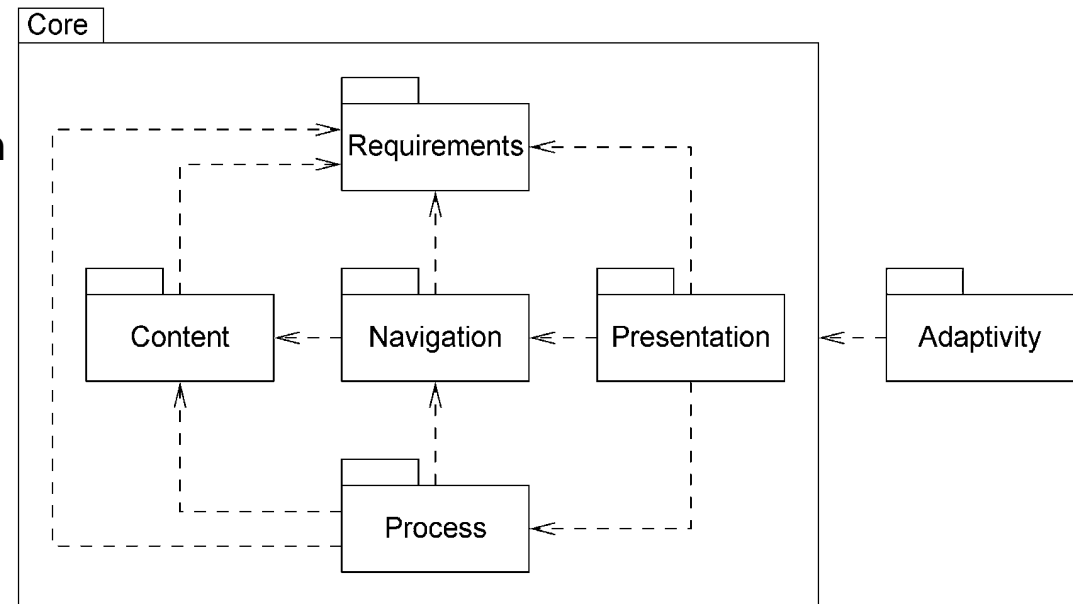
Modelling with UWE

- Use of UML and a UML Profile
 - UML diagram types and model elements
 - additional Web specific constructs (extension for Web)
- Based on guidelines for the construction of the models (development process)
 - set of steps for each model
 - relationship between models
- Model construction is
 - systematic
 - semi-automatic
 - tool-supported

Analysis and Design Models in UWE



- Analysis models of a Web application
 - functional requirements are specified by
 - use case model
 - workflows
 - data (content) requirements are specified by
 - domain model
- Design models of a Web application
 - information aspects
 - content model
 - hypertext structure and navigation functionality
 - navigation model
 - layout schema
 - presentation model
 - functionality
 - process model
 - adaptivity model



Example: Simple Music Portal

- Inspired by `www.mp3.com`
 - offers albums for downloading
 - contains information about albums, songs, singer, composer, and publisher
 - this information is available for free
 - registered users can search albums and download them
 - for downloading they need to have enough credit on their prepaid account
 - accounts are rechargeable

Modelling Requirements with UWE

- Use cases are the most appropriate/used technique for modelling requirements
- Graphical visualization by UML use case diagram
 - to model required functionality
 - distinguishes between navigation and process use cases

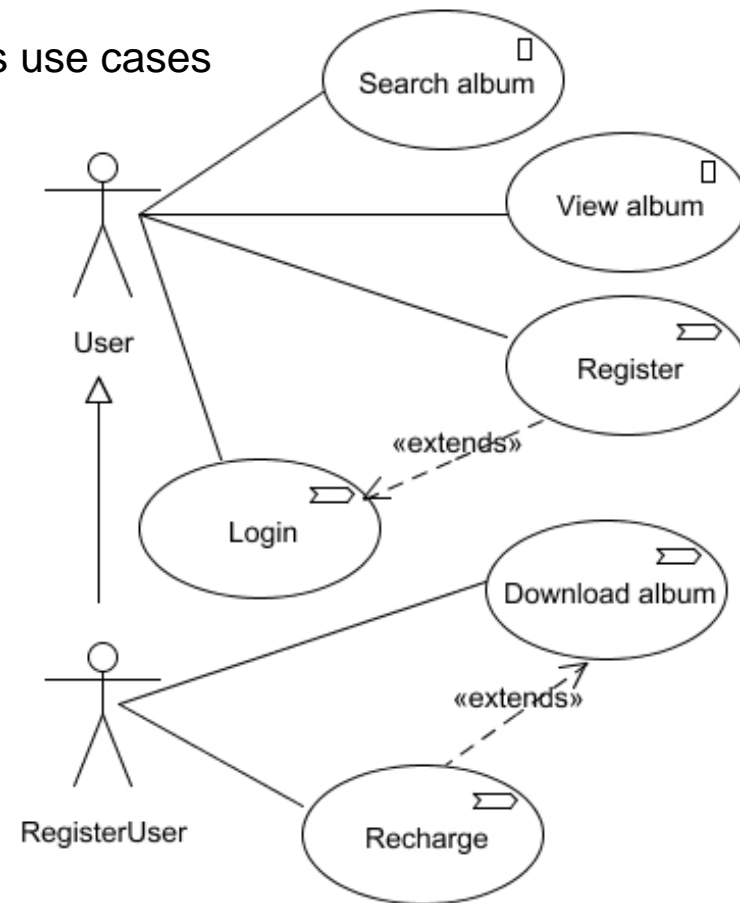
- Modeling constructs

- «navigation»



- «web process»

(in ArgoUWE normal use case)



CASE Tool ArgoUWE

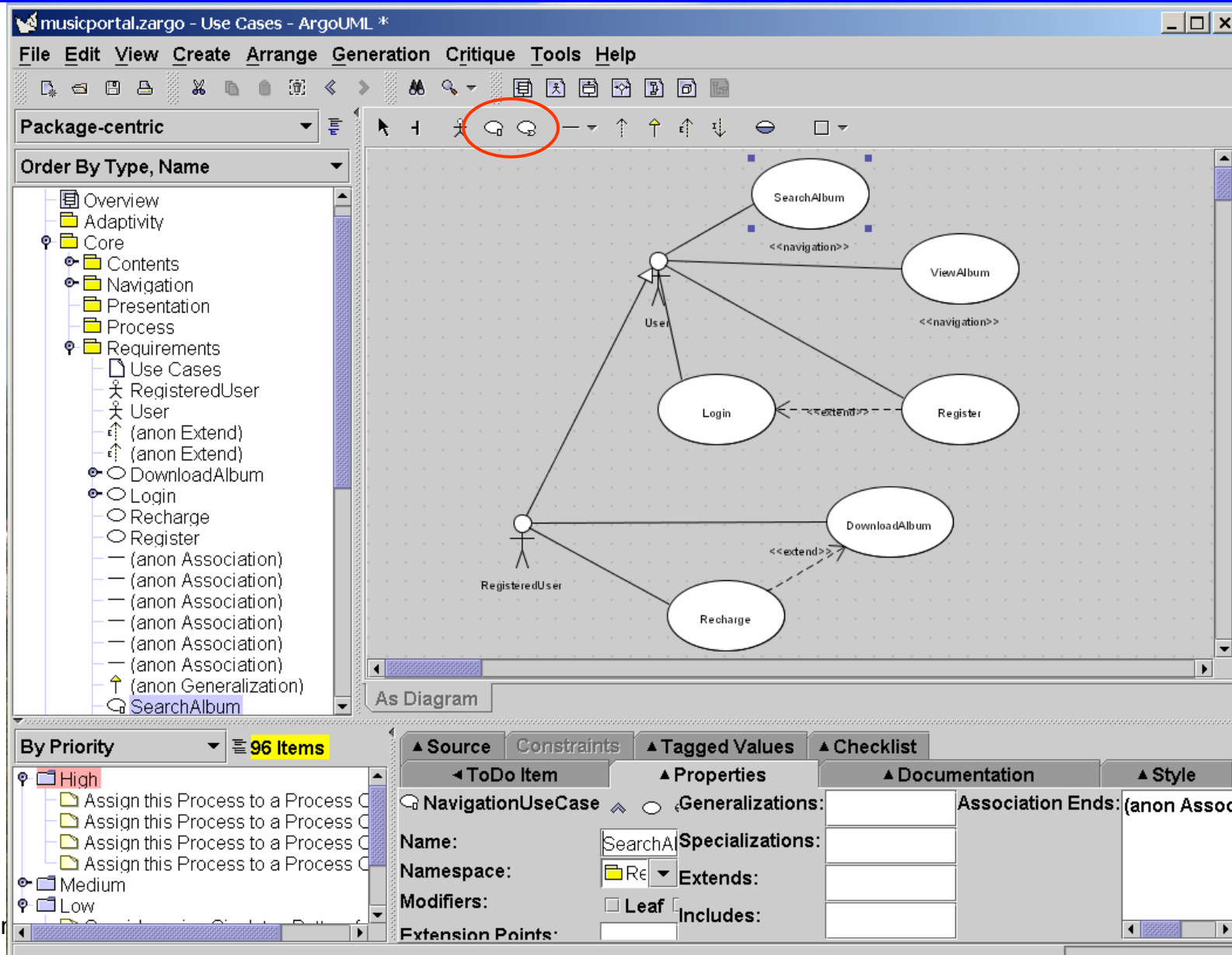
- Defined as plugin for ArgoUML
 - advantages
 - open source UML CASE tool
 - design critics for checking model consistency
 - disadvantages
 - still based on UML 1.x
 - usability problems
- Implemented so far
 - provides UWE notation
 - supports systematic construction of Web applications
 - allows for separation of concerns: content (conceptual), navigation, presentation and business processes
 - supports model validation checking models consistency
- Currently we are developing plugins for other CASE tools

ArgoUWE: Consistency Check & Design Critics

- ArgoUML cognitive design critics
 - [background thread](#)
 - warning mechanism for model inconsistencies
 - wizard for design improvement
 - uses critics pane for the critiques
 - displays wizards comments in detail pane

- ArgoUWE design critics features
 - inherits from ArgoUML (e.g. name collision)
 - is extended by all [wellformedness constraints of UWE](#) (e.g. relationship between use case and process model)

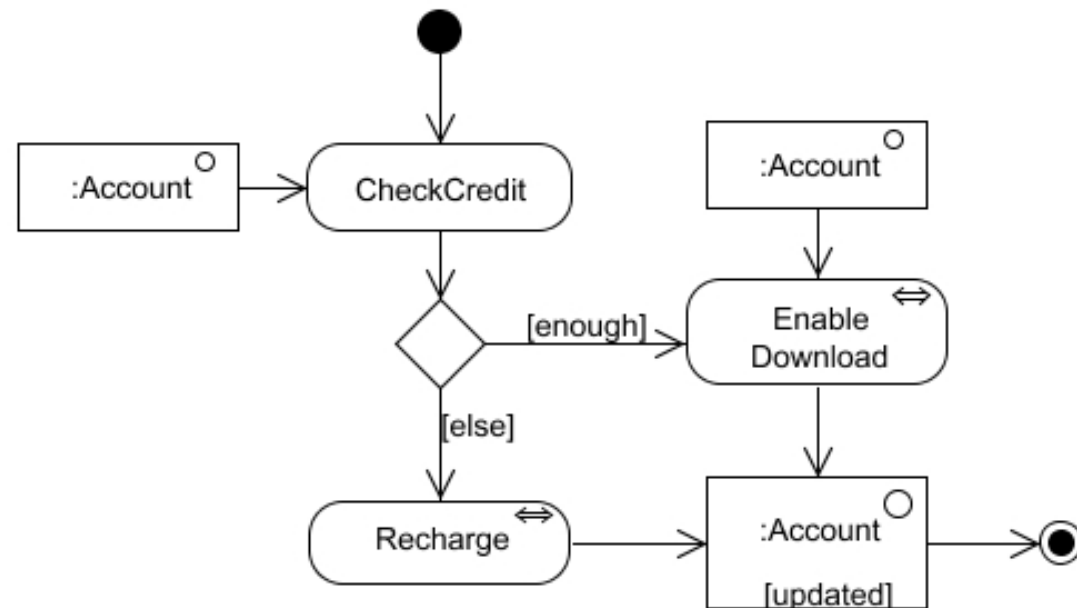
ArgoUWE: Using UWE Stereotypes for Use Cases



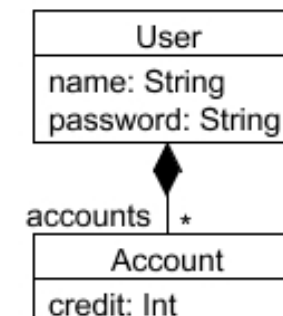
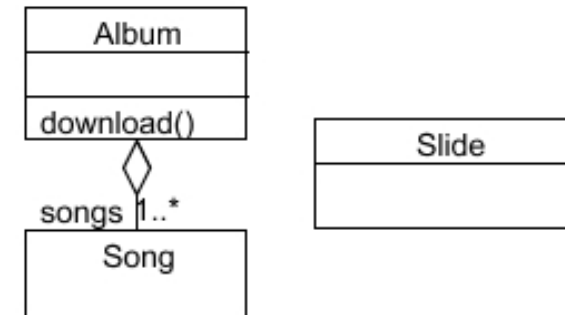
Modelling Business Processes

- Business Process can be modelled as workflows
- Workflows are visualized by UML activity diagram
 - set of actions
 - control flow elements as decisions, merges, forks, joins, etc.
 - object flows including relevant objects (instance of classes)
- Example: Download Album

RE Element	UML Metaclass	Icon
Browse	Action	⇒
Search	Action	?
User Transaction	Action	↔
Content	Classifier	○
Node	Classifier	□



- Representation with “pure” UML is sufficient
 - stereotypes are not needed
 - UML class diagram
- UML class diagrams used at analysis-level and design-level
 - elements to be modelled represented by classes
 - units of information
 - users of the system (user model)
 - associations represent relationships between classes
 - aggregation used to show an whole/part relationship
 - composition is a strong form of aggregation that requires a part instance be included in at most one composite at a time. If a composite is deleted, all of its parts are normally deleted with it.
 - inheritance used to show hierarchies between classes

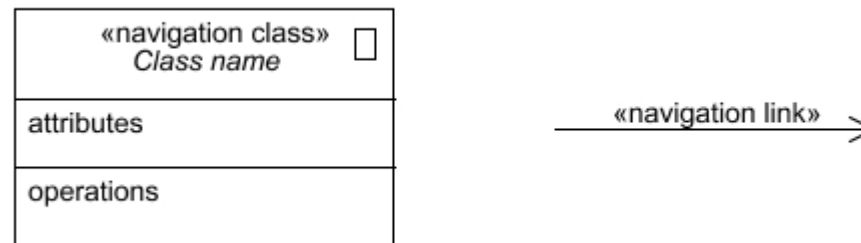


Navigation Modelling

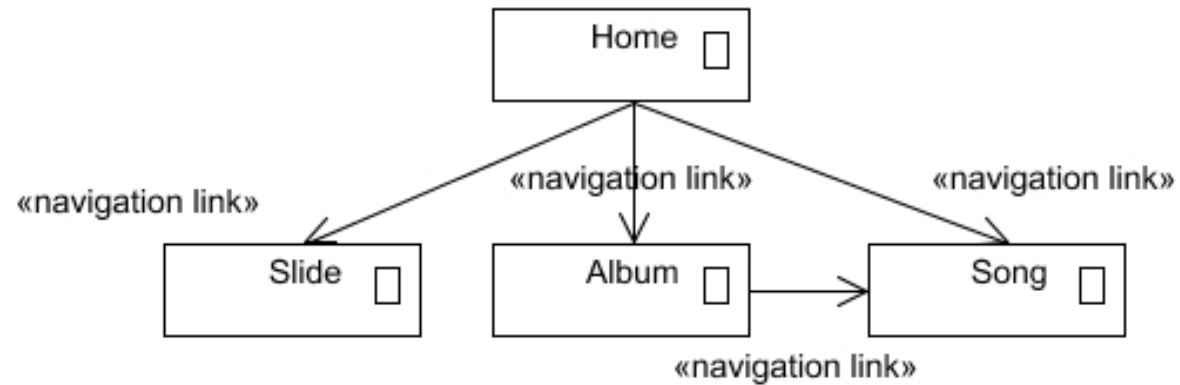
- Goals
 - to represent nodes and links of the hypertext structure
 - to design navigation paths
 - to avoid disorientation and cognitive overload
- Result: [navigation model](#)
 - represented by a UML class diagram
 - uses specific modelling elements for Web concepts
- Modelling elements
 - basic elements
 - access primitives
- Other methodologies
 - own notation for hypertext elements
 - different diagram types: UML state diagram or own diagram type

Basic Navigation Elements

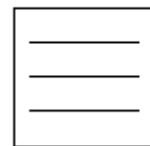
- Navigation elements used to model the core hypertext structure
 - **navigation class** specifies the hypertext nodes is visited by a user through browsing (a navigation class will be given the same name of the domain class which it maps)
 - **navigation link** specifies that the target navigation object is accessed by navigation from the source navigation object
- Stereotypes for basic navigation elements
 - «navigation class»
 - «navigation link»



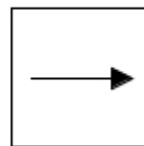
Navigation Model: Hypertext Structure



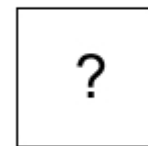
- Additional elements for the selection of one target navigation element from a set of instances of a navigation element
 - `index`
 - `guidedTour`
 - `query`
 - `menu`
- Shortcuts for more complex constructs (if represented in UML without extension)



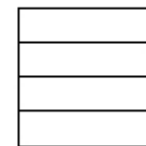
Index



GuidedTour



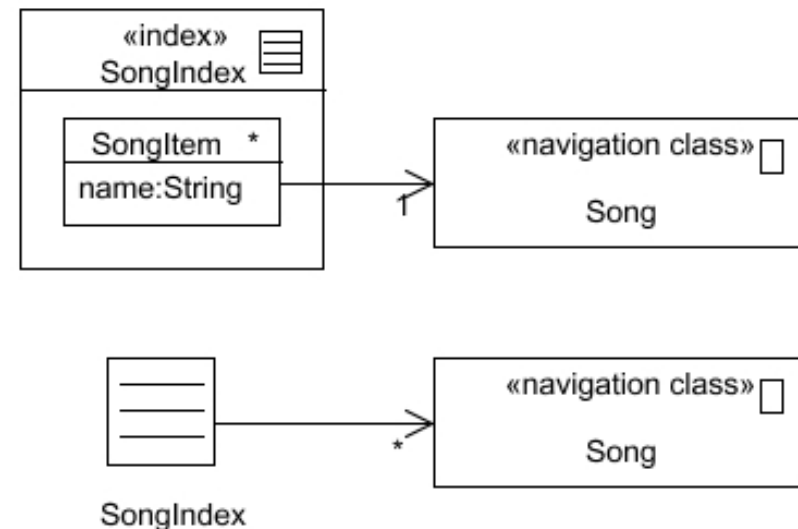
Query



Menu

Access Primitive: Index

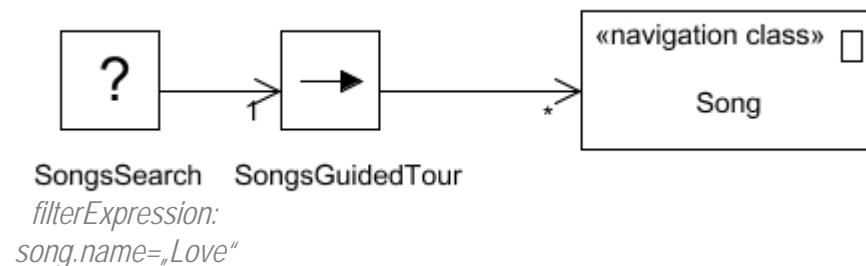
- **Index** specifies the direct access to all instances of the target by providing a list of all elements from which one can be selected to continue navigating in the Web application
 - contains an arbitrary number of index items
 - each index item is an object which has a name and owns a link to an instance of a navigation class
- UML stereotype: «index»



Further Access Primitives



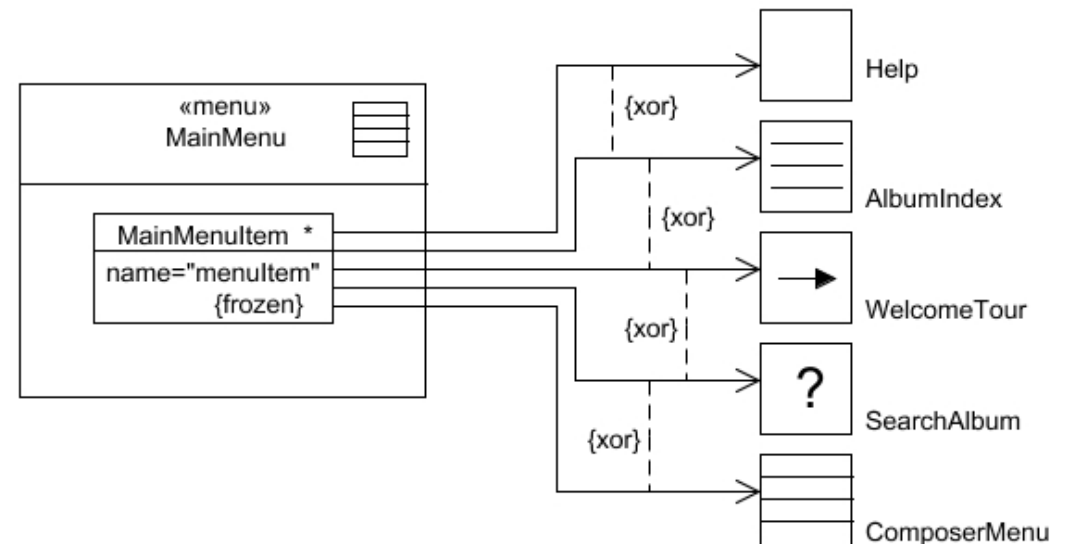
- **Query** represents the possibility to search for instances of the target node where instances are selected based on the property *FilterExpression*
 - UML stereotype: «query»
- **GuidedTour** provides sequential access to instances of the target node
 - order in the sequential access is given by a property *SortExpression*
 - different options may be specified
 - circular access to the instances
 - access is triggered by a user-interaction
 - UML stereotype: «guidedTour»



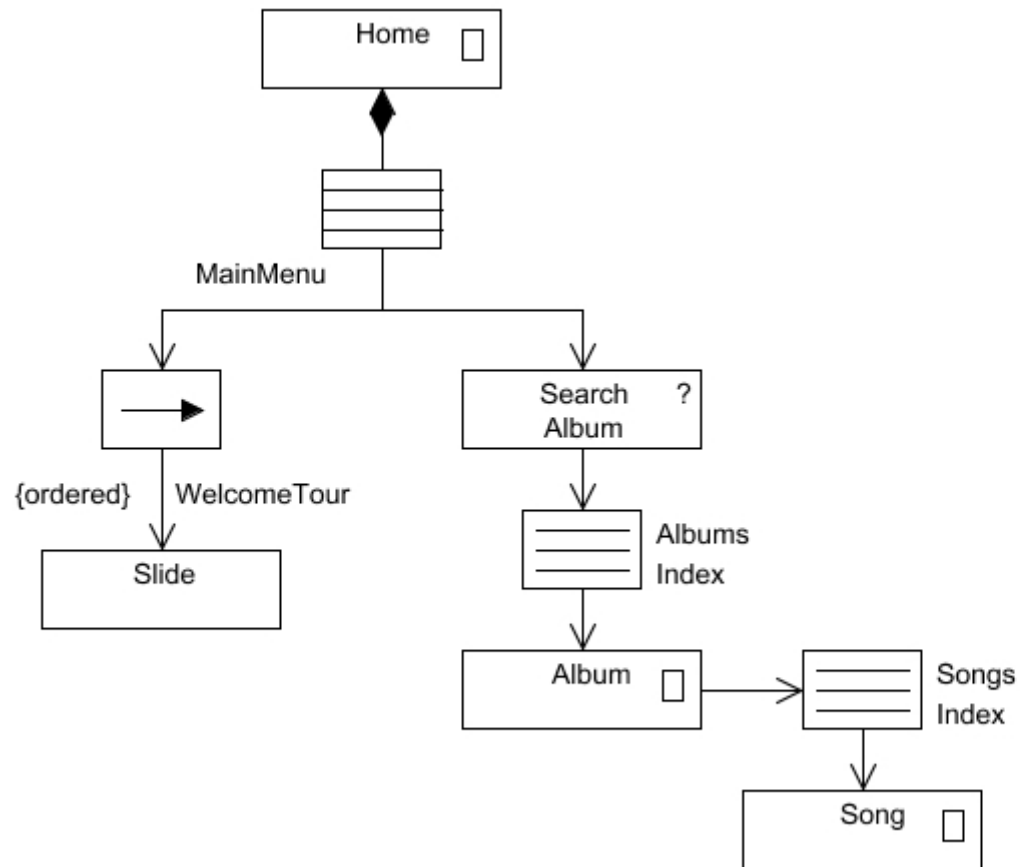
Navigation Model Elements: Menu and External Node

- **Menus** are used to structure the outgoing links from a node
 - associated to a navigation class by composition
 - consists of a set of links to heterogeneous elements, such as indexes, guided tours, queries, instances instances of navigation classes or other menus
 - UML stereotype: «menu»

- **External Links** are links pointing to nodes that do not belong to the Web application
 - UML stereotype: «external link»



Navigation Model: Hypertext Structure



Navigation

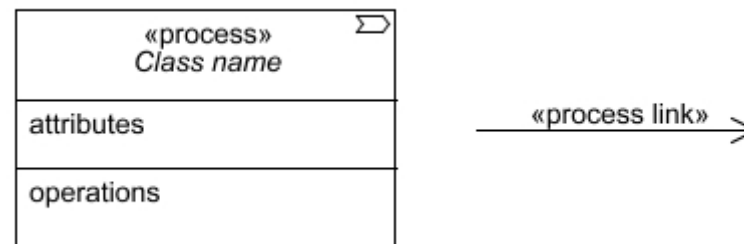
- Enhancement of the navigation structure model by
 - access primitives: [index](#), [guided tour](#) and [queries](#)
 - [menus](#)
 - [processes](#)
 - tagged value [home](#) to indicate starting point of the application (node without ingoing links)
 - tagged value [landmark](#) to indicate that a node is reachable from everywhere (all other nodes include link to landmark node)
- Construction of views of the navigation space
 - partial views if the number of nodes is large
 - different views for different users
 - different views for different environments

Modelling Processes in UWE

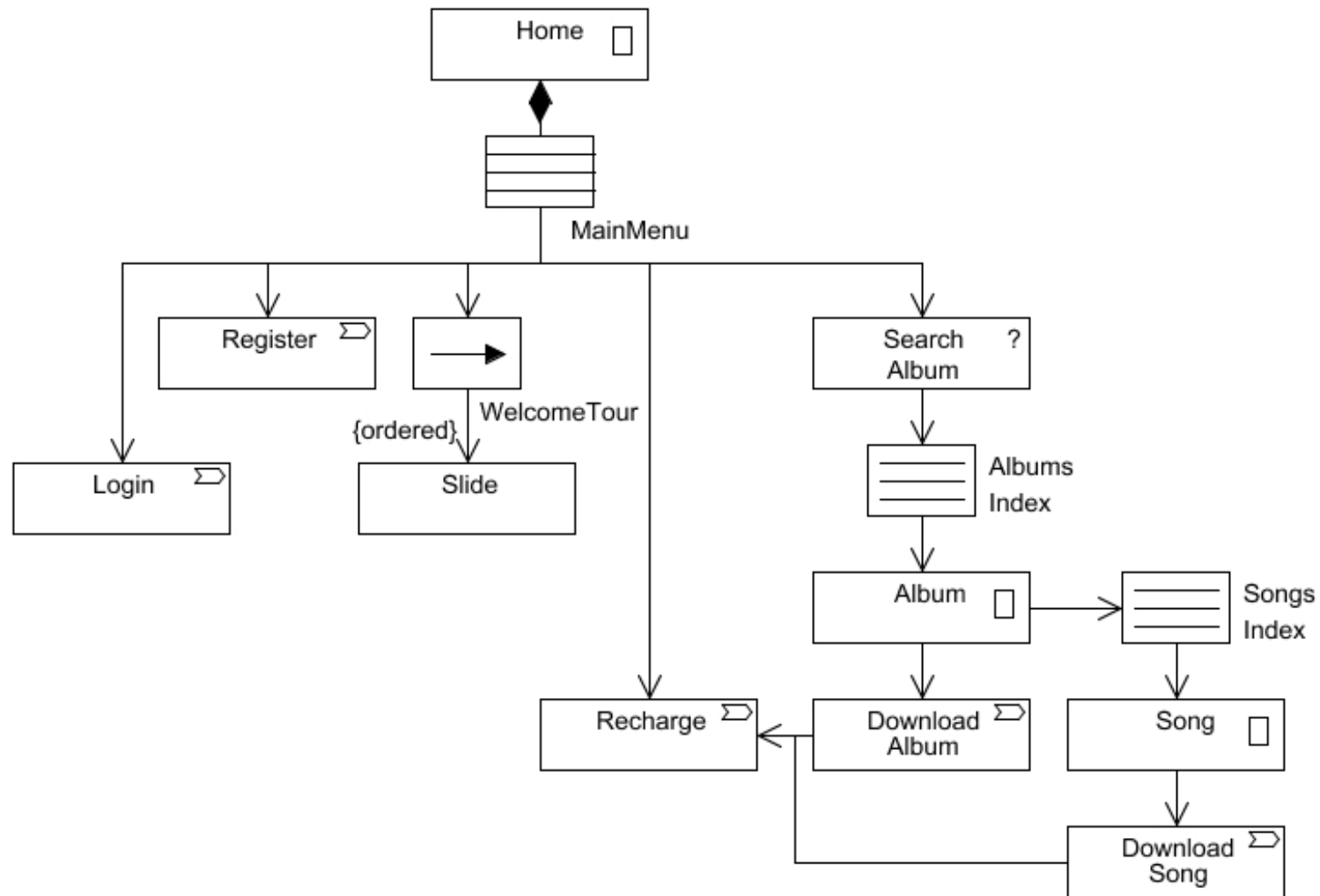
- Navigation model of a Web application
 - represents the static information structure accessible to a user of the system
 - specifies browsing (navigation) functionality
- Process model
 - represent the dynamic aspects of a Web application
 - specifies functionality, such as transactions and complex workflows of activities
- Process modelling consists of
 - definition of **process classes** (for **non-navigation use cases**)
 - **integration** of these process classes in the **navigation model**
 - description of the behaviour through a **process flow**
 - represented as UML activity diagram

Process Elements

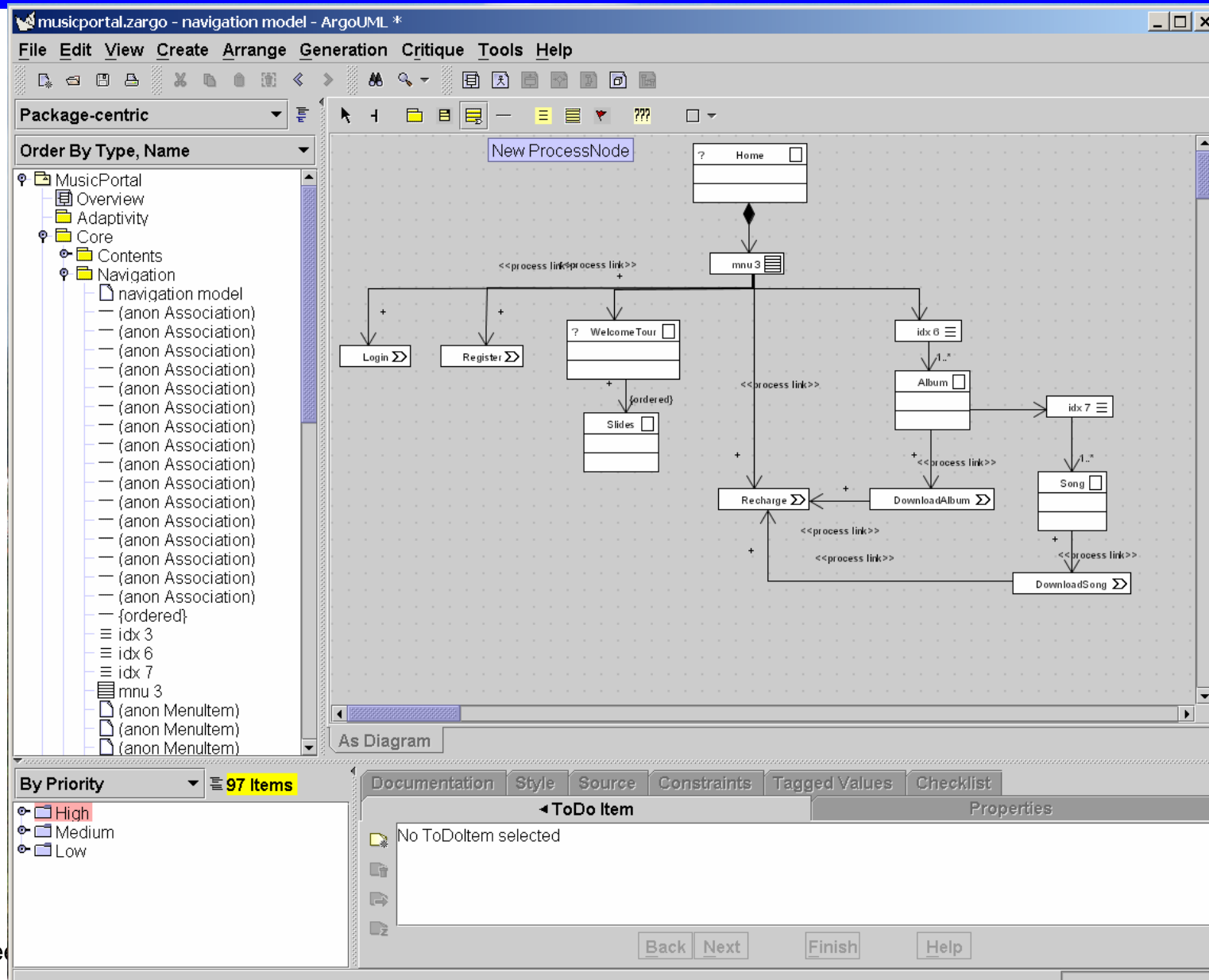
- **Process class** represents the process through which the user will be guided in the Web application
 - for complex process that require more than a single class, an additional process model is built
 - UML stereotype: «process class»
- **Process link** is used to model the association between a «navigation class» and a «process class»
 - indicates **entry points** and **exit points** of processes within the navigation structure
 - UML stereotype: «process link»



Navigation Model: Including Web Processes



Modelling with ArgoUWE

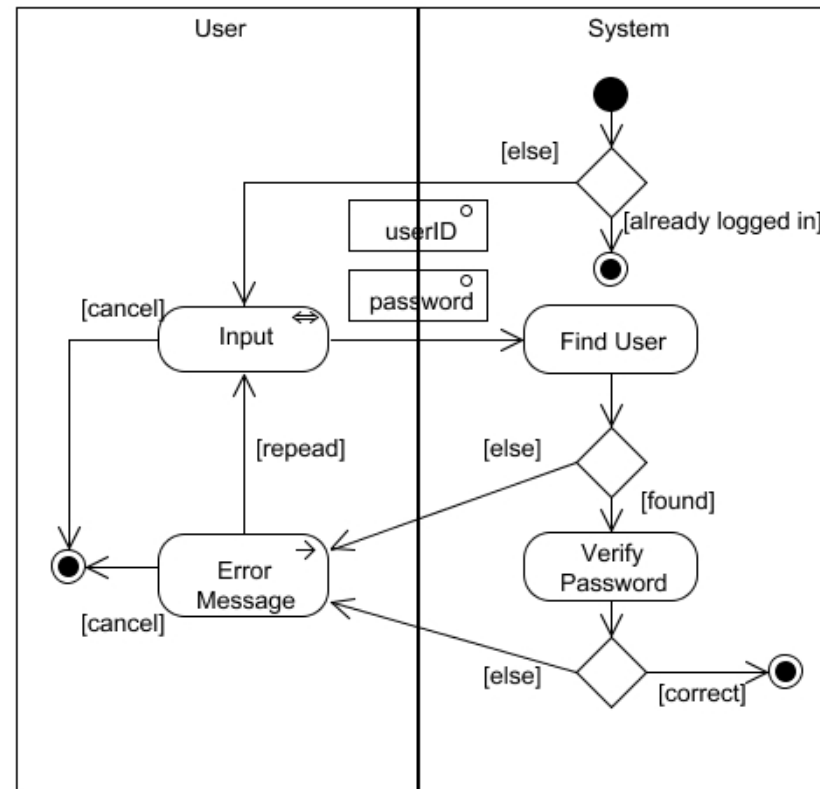


Modelling the Process Flow

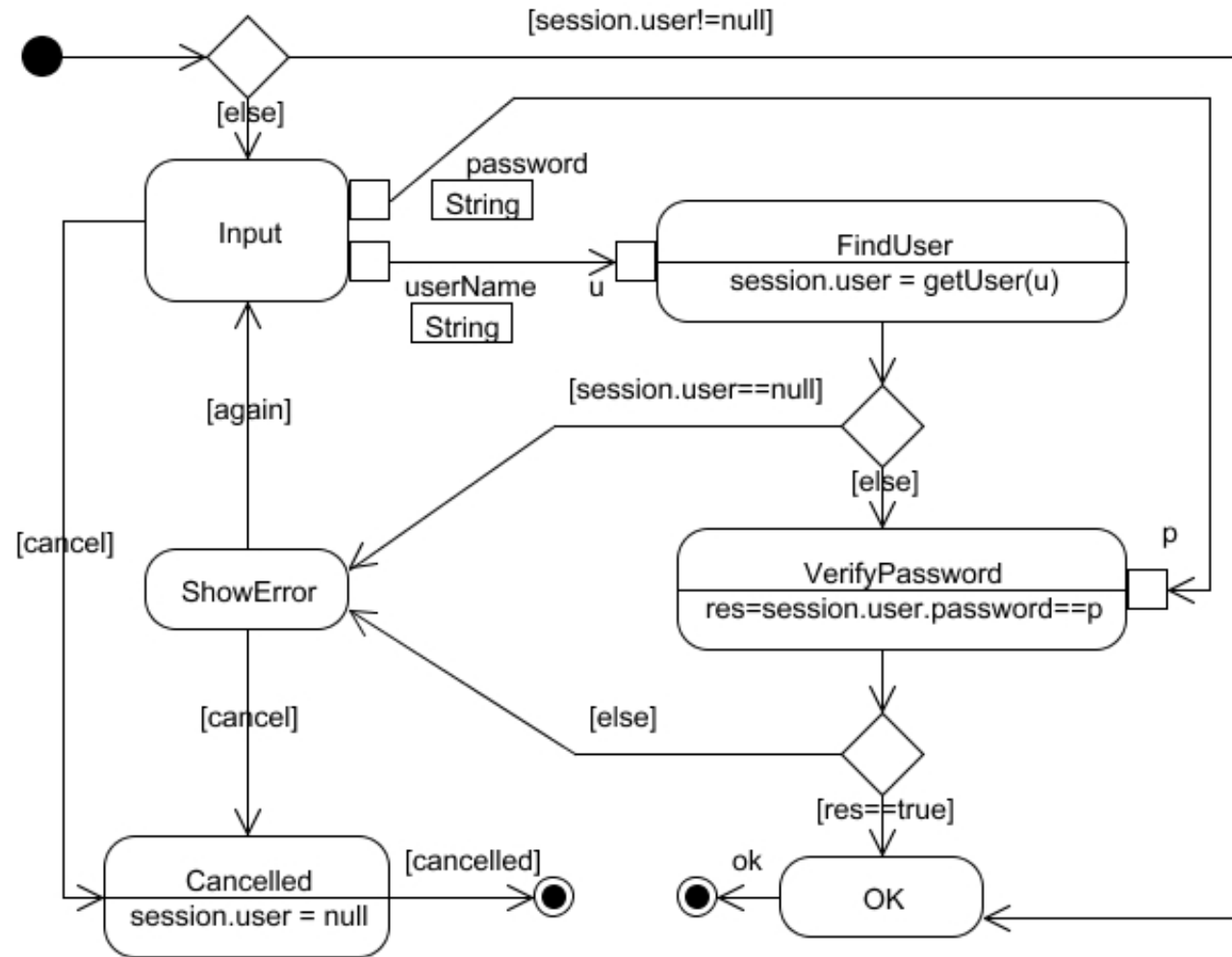
- The behavior of a Web process is defined by the process flow model
 - represented by an UML activity diagram
 - result of the refinement of the activity diagram drawn for requirements specification
 - no need of use of stereotypes

- Process flow consists of
 - flow of execution represented by activity nodes connected by activity edges
 - control nodes that provide flow-of-control constructs, such as decisions and synchronization
 - object nodes that represent data flowing along object flow edges or pins associated to the actions
 - semantic of activities is based on control and data token flows, similar to Petri nets

Requirements: UML Activity Diagrams for Login



Process Model: UML Activity Diagram: Login



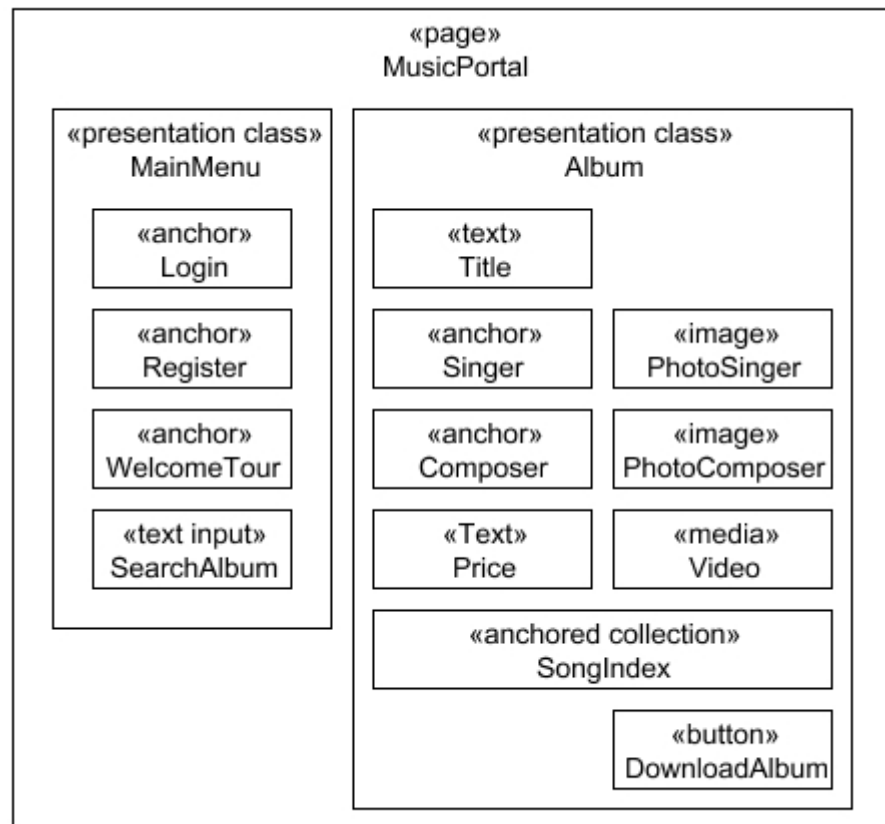
Modelling Presentation

- Representation of layout for the underlying navigation and process models
 - is an abstract presentation
 - concrete presentation requires specification of additional physical properties of the layout
 - colour
 - position
- Presentation classes represent Web pages and are composed of user interface element and other presentation classes
 - definition of pages as a hierarchical composition of presentation elements
- UML class diagram for the [structure of the presentation](#)
 - using in UML container notation
- UML interaction diagram (sequence diagrams) can be used for modelling behaviour

- Alternative: development of a prototype

Presentation Model Elements

- «page» is a presentation group that contains all elements that will be presented together to the user as response to one request
- «presentation class» groups a set of user interface elements representing a logic unit of presentation
- user interface elements
 - «anchor»
 - «text»
 - «image»
 - «button»
 - «form»
 - ...



Modelling Adaptivity

UWE uses a technique called Aspect-Oriented Modelling (AOM) for the construction of models to specify adaptive Web applications

- What are adaptive Web applications?
- What is AOM?

Adaptive Web Applications

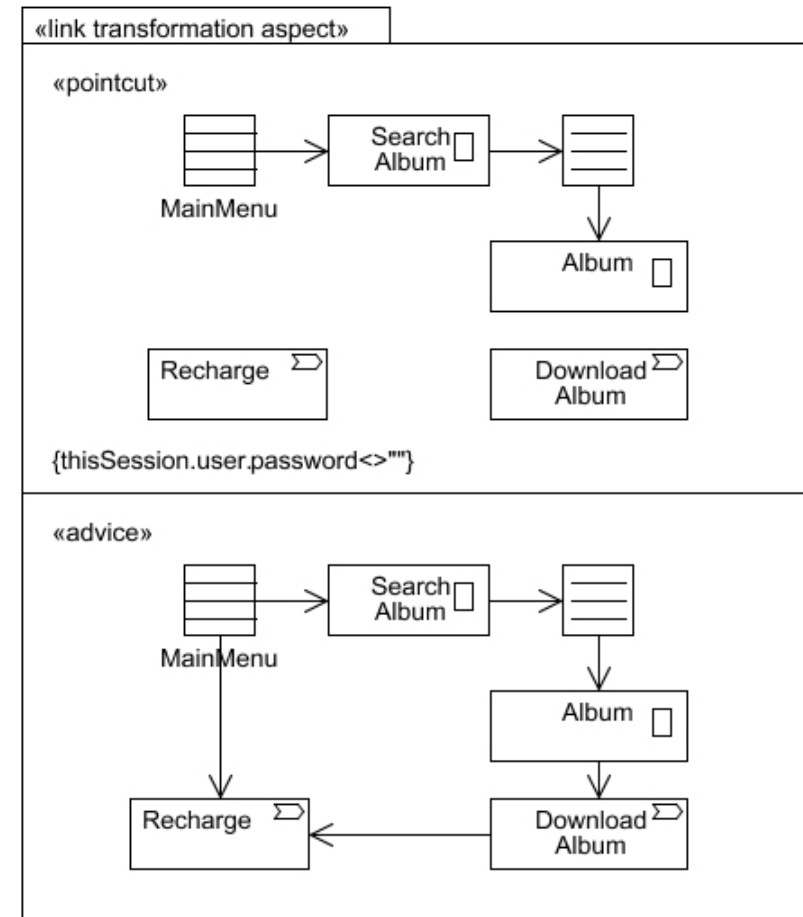
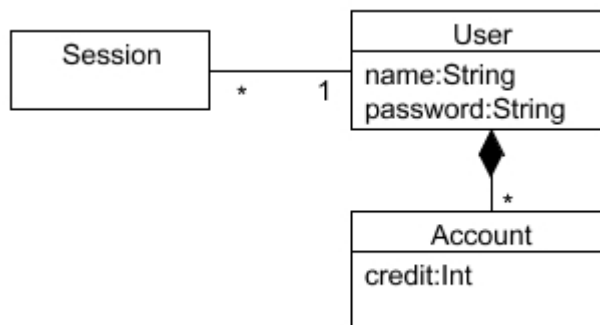
- Adaptation for
 - user properties: **knowledge**, **tasks**, **preferences**, **interests**
 - context properties: **location** (place and time) & **platform** (hw, sw, network)
- Update of a user model / context model
 - **observation** of the **user behaviour** or **environment** by the system
- Techniques for adaptation
 - **content** adaptation
 - inserting and removing text/multimedia features
 - content variants
 - **navigation** adaptation
 - link ordering
 - link annotation
 - link hiding
 - link generation
 - **presentation** adaptation
 - modality adaptation (audio or text)
 - language selection
 - layout variants (resizing of fonts, images, changing colours)

Aspect-Oriented Modelling (AOM) in a Nutshell

- AOM identifies and defines
 - **join point**: well-defined place in a model / program where additional features can be attached
 - **pointcut**: set of join points
 - **advices**: feature to add/execute at a join point
- AOM specifies a **weaving** process (composition) of
 - **core** functional modules (elements of a pointcut)
 - **aspects** (defined by an advice)

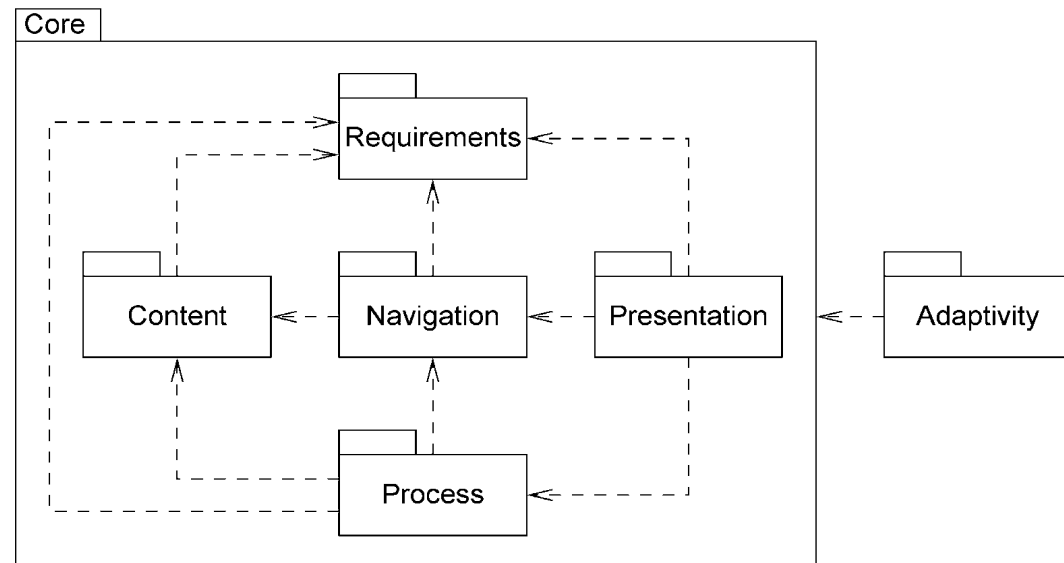
Modelling Adaptivity

- Specification of
 - pointcuts (including conditions) and
 - advices
- Weaving the result into the web application based on
 - current state of the user model
 - information provided by link traversal
- Example: links only visible for registered users to
 - [download album](#)
 - [recharge](#)



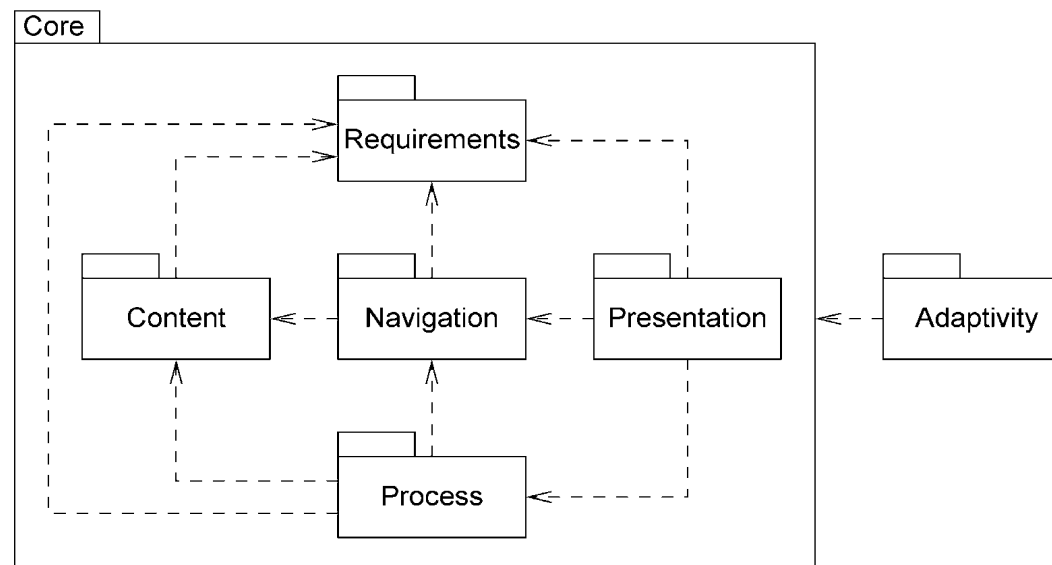
UWE Metamodel

- UWE Metamodel is defined as a conservative extension of UML 2.0
 - model elements of the UML metamodel are not modified
 - all new elements are related by inheritance to at least one model element of the UML
 - use of OCL to constraints to specify additional semantics of the new elements
- UWE extension consists in adding 2 top level packages
 - Core
 - Adaptivity



UWE Metamodel

- UWE metamodel structure reflects
 - reflects separation of concerns in the structure of Core
 - cross-cutting of adaptation by the dependency of Adaptivity on Core
- UWE metamodel is profileable
 - mapping to a UML profile is possible
- UWE metamodel is MOF compatible
 - uses XML metadata interchange format (XMI)

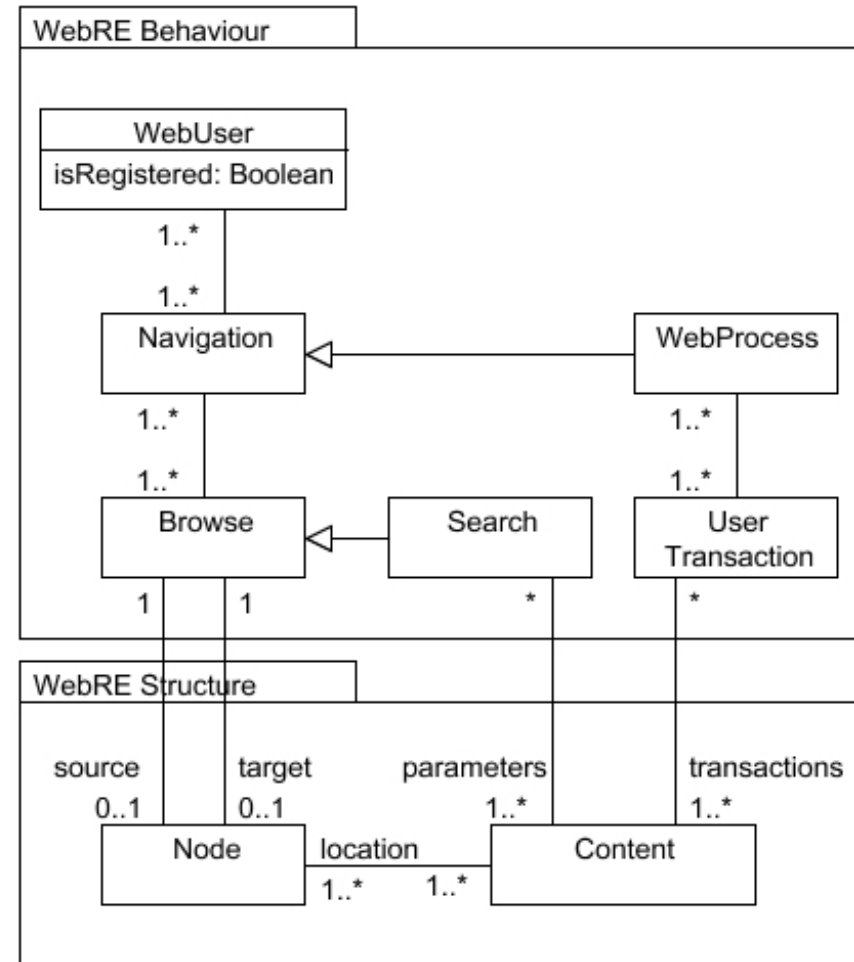


UWE Metamodel: Requirements Model

- Model elements grouped in packages
 - WebRE Behaviour
 - WebRE Structure
- Defines relationships among elements
 - inheritance (e.g. search defined as an extended browse)
 - associations (e.g. a browse requires a source and a target node)
- Defines invariants (OCL constraints)

```

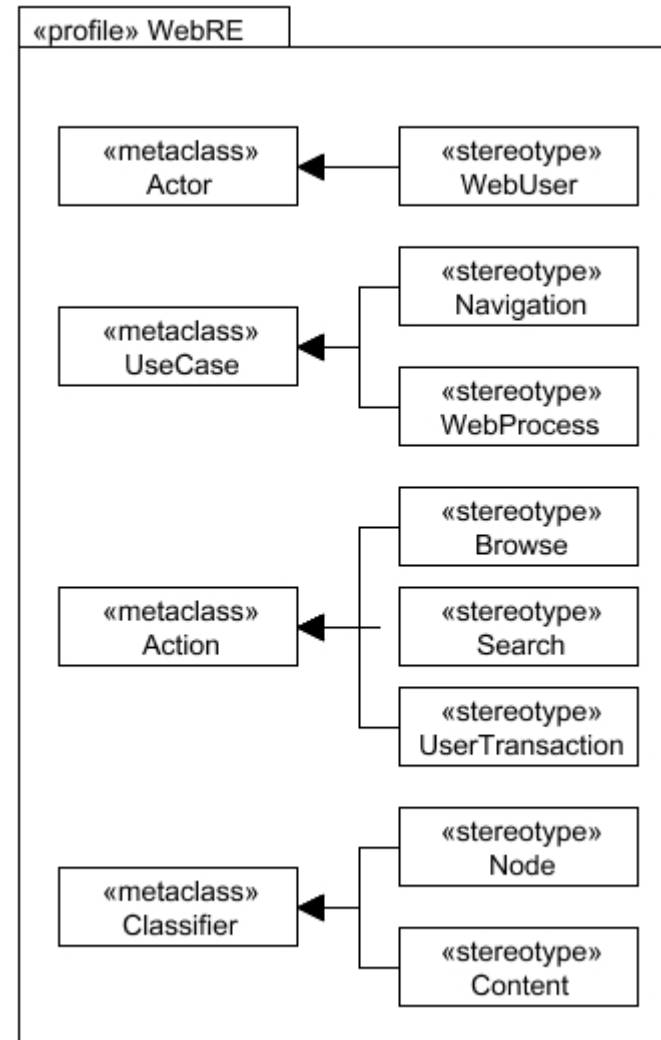
context Search
inv: self.parameters -> forAll
    (p | p.location ->
        includes
            (self.source))
    
```



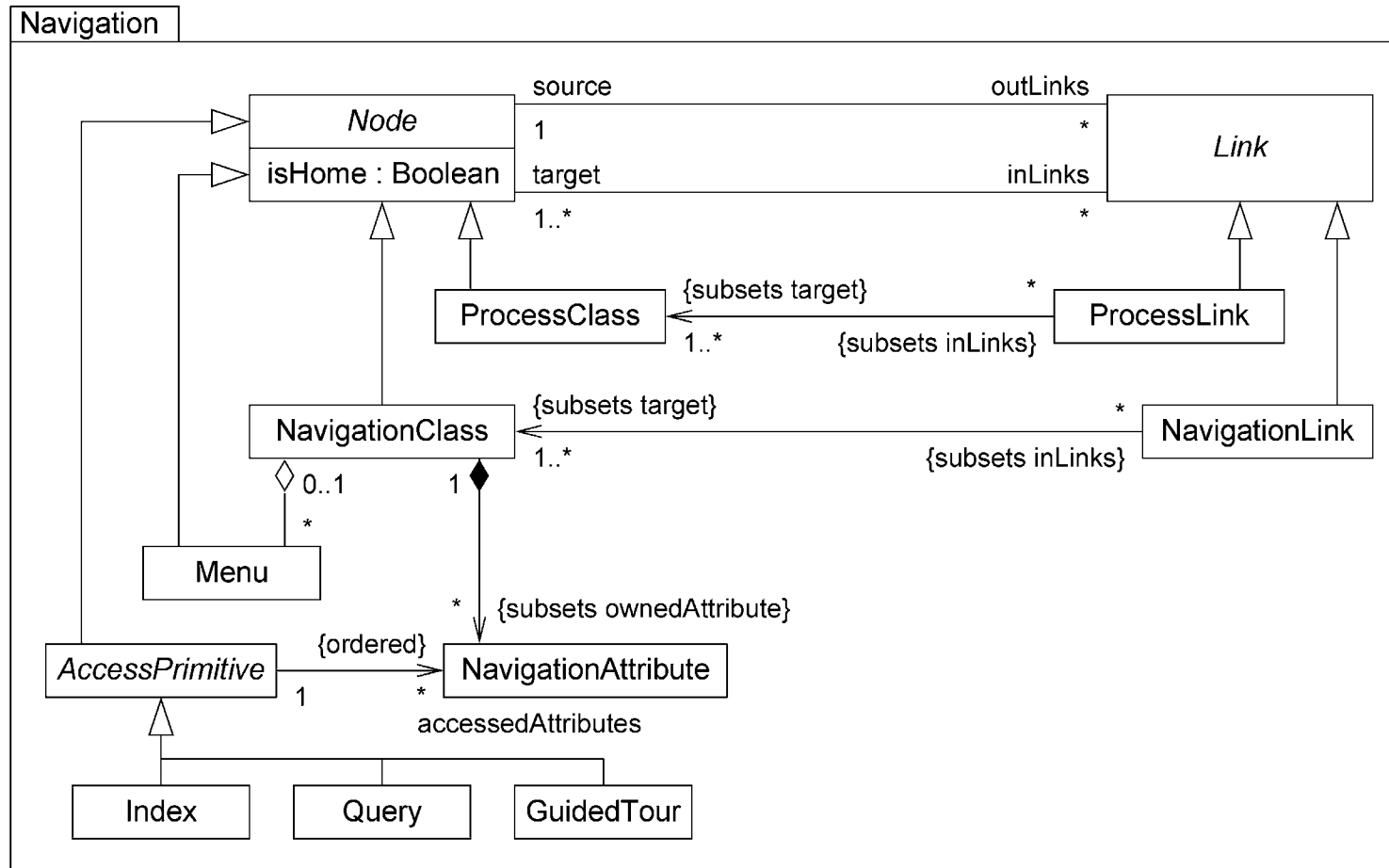
(see stereotypes used in the example)

UWE Profile: Requirements

- UML stereotype for each concept of the requirements package of the UWE metamodel
- Extends relationship
- UML metaclass
- Advantages
 - no need to specify complete semantic of new modelling elements
 - use of all UML CASE tools supporting UML profiles for modelling Web applications



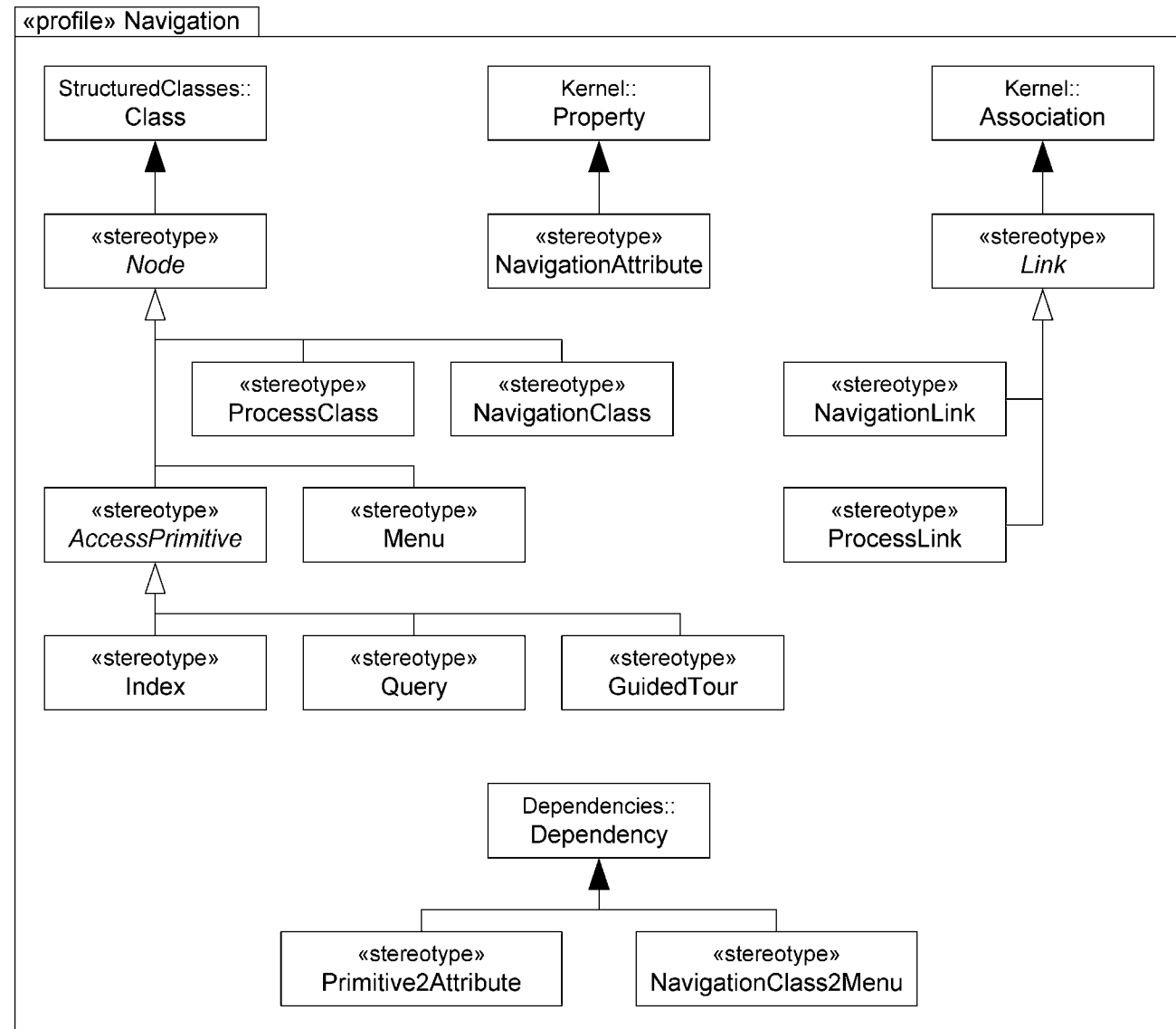
UWE Metamodel: Navigation



(see stereotypes used in the example)

UWE Profile: Navigation

- UML stereotypes for Web specific concepts used for the specification of the hypertext structure
- Extends relationships
- UML metaclasses



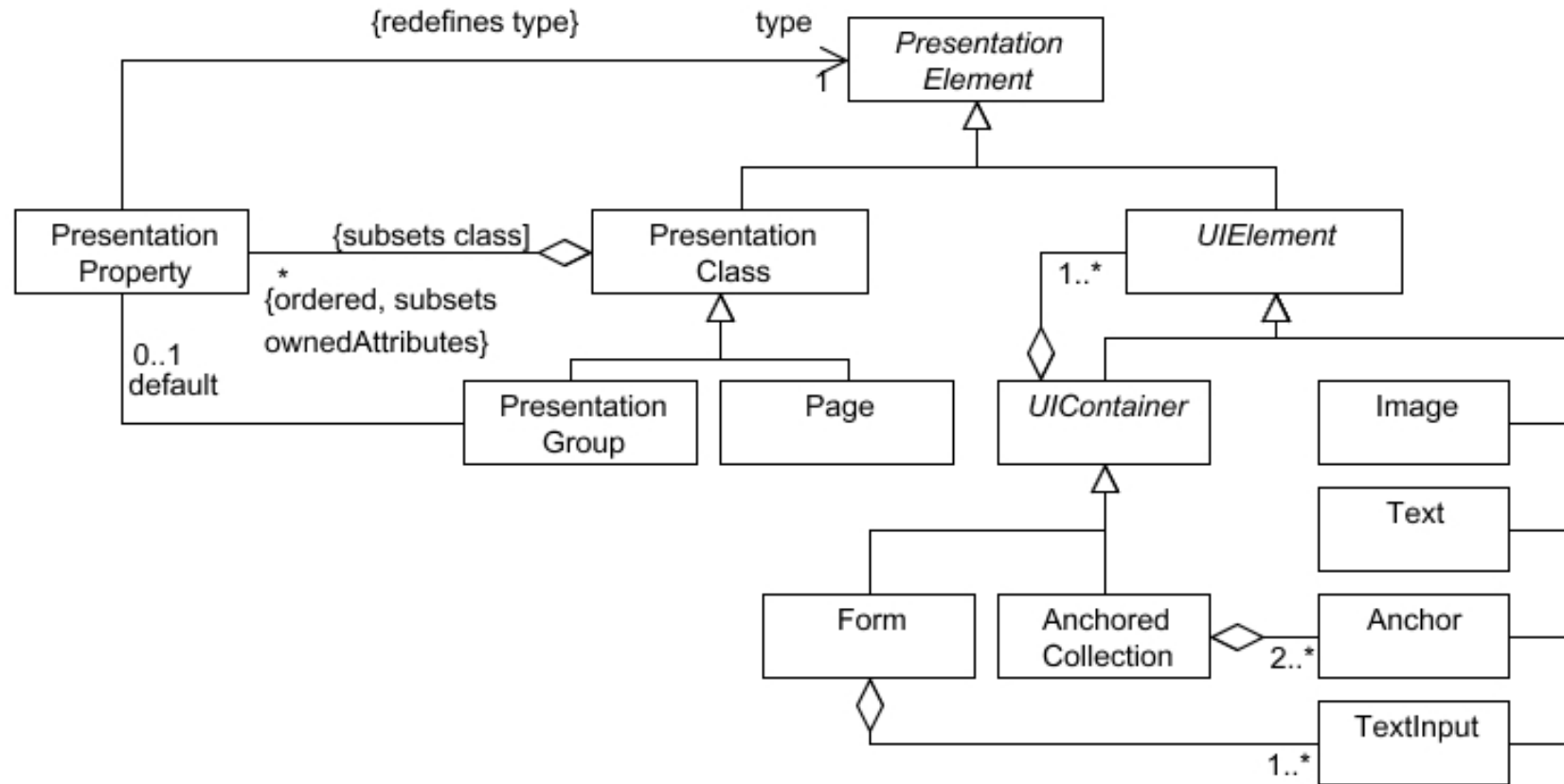
Meta-associations of the Metamodel

- Representation using associations derived from the UML metamodel
 - association {subsets target} for association between NavigationLink and NavigationClass
 - association {subsets ownedAttributes} for composition between NavigationClass and NavigationAttribute
- Representation stereotyping UML metaclass Dependency
 - aggregation between NavigationClass and Menu
 - association between AccessPrimitive and NavigationAttribute, leading to the following constraint

```
context Dependency
inv: self.stereotypes->
    includes("Primitive2Attribute") implies
    (self.client.stereotypes->
        includes("AccessPrimitive") and
    self.supplier.stereotypes->
        includes("NavigationAttribute"))
```

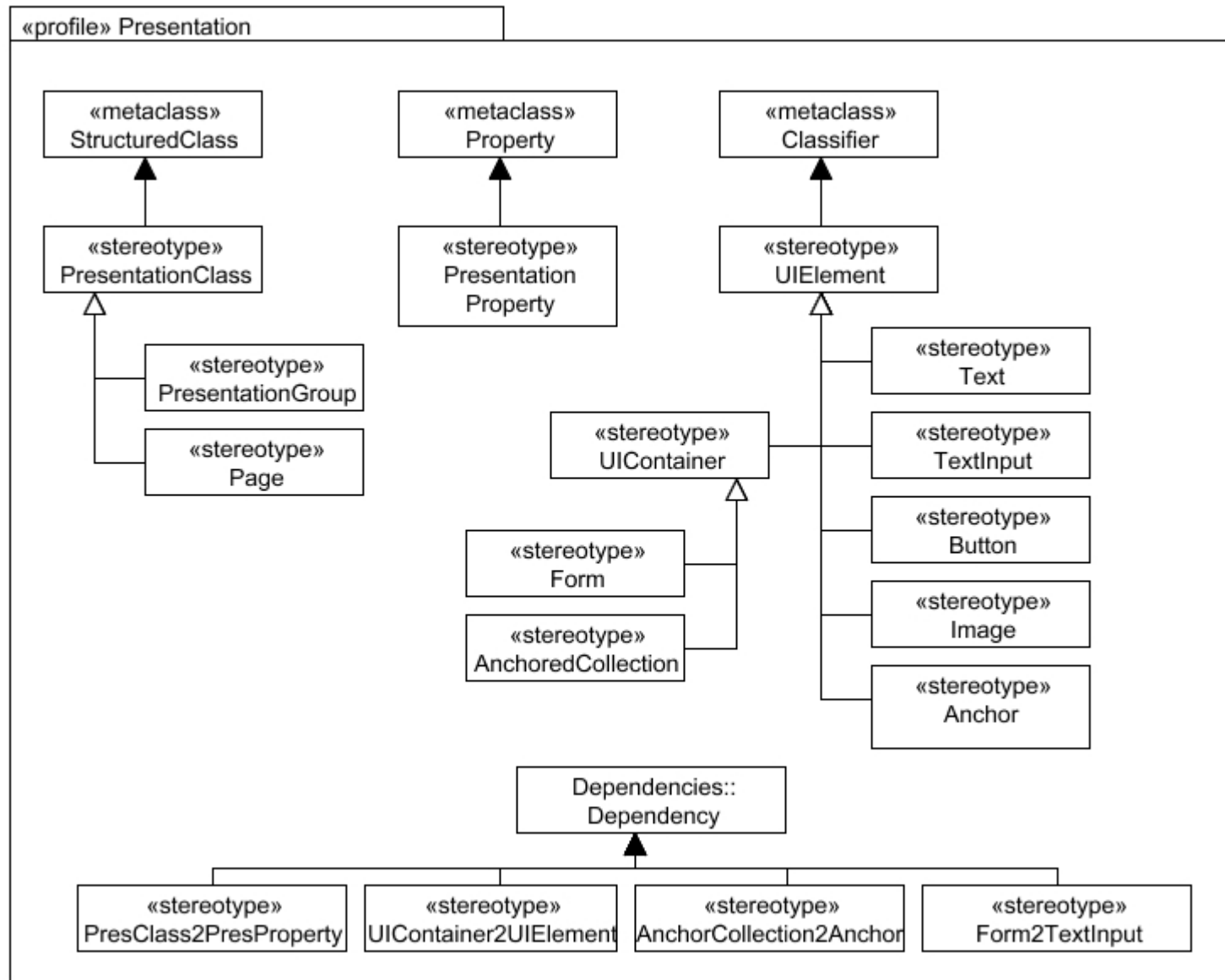
where client and supplier denote the ends of the Dependency relationship

UWE Metamodel: Presentation



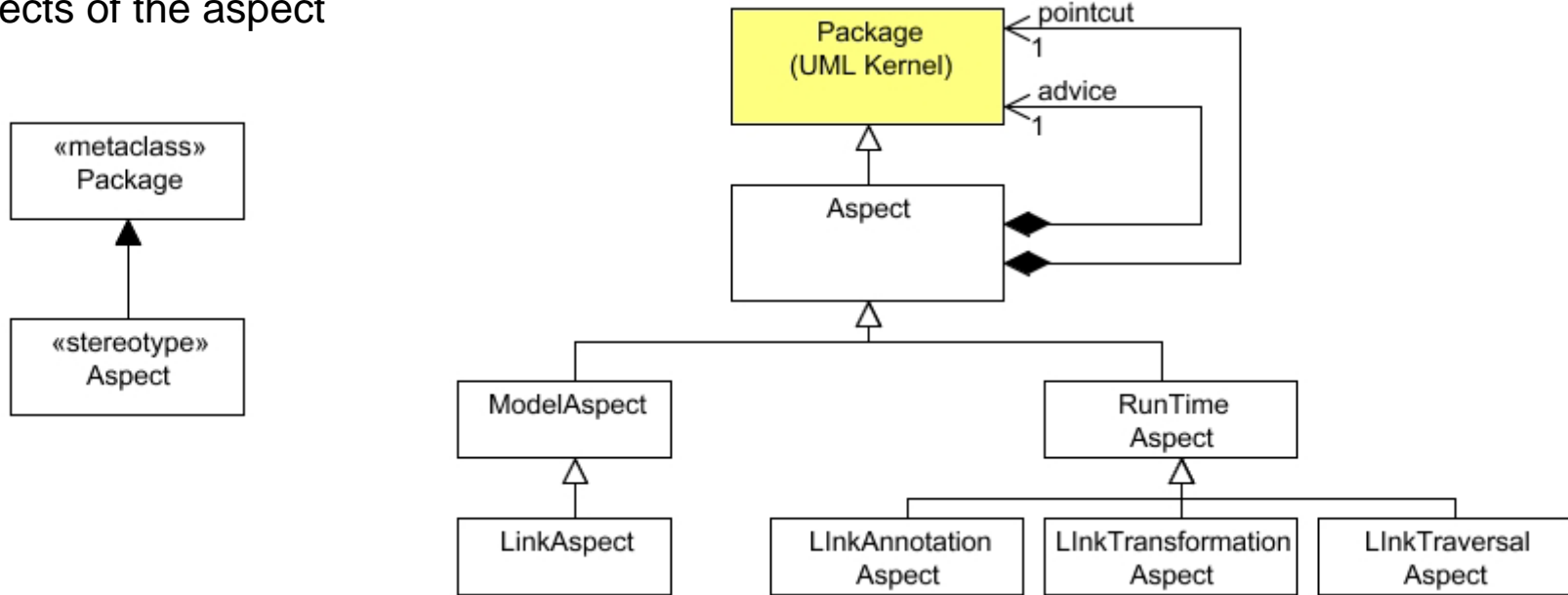
(see stereotypes used in the example)

UWE Profile: Presentation



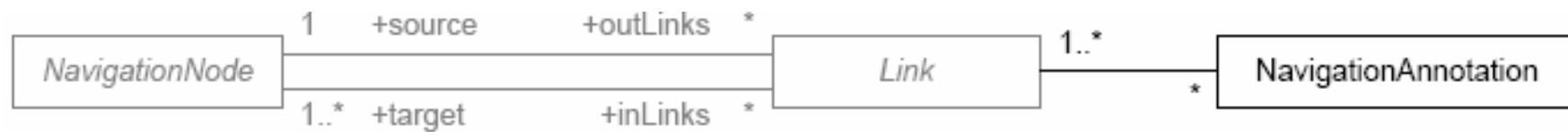
UML Extension for Aspects

- lightweight extension of UML
- an **Aspect stereotype** as an extension of UML metaclass **Package** composing
 - **Pointcut package** that references to all model elements on which
 - **Advice package** is applied
- Pointcut and Advice packages may contain OCL constraints detailing
 - conditions for the application of the aspect
 - effects of the aspect



UWE Metamodel: Adaptivity

- **NavigationAnnotation** stereotype to model link adaptation
 - related to **Link**
 - navigation **annotations** are attached to navigation **links**



Literature

- **Web Engineering: Systematic Development of Web Applications**
Gerti Kappel, Birgid Pröll, Siegfried Reich, Werner Retschitzegger (eds.)
dpunkt-verlag (German version), 2003,
John Wiley & Sons (English version), 2006.
- **Web Engineering: Modelling and Implementing Web Applications**
G. Rossi, O. Pastor, D. Schwabe, L. Olsina (eds.), Springer (2007), to appear.
- **Metamodelling the Requirements of Web Systems**
María José Escalona and Nora Koch
2nd International Conference on Web Information Systems and Technologies (WebIST'06), Setubal, Portugal, pages 310-317. INSTICC, 2006.
- **Modelling Adaptivity with Aspects**
Hubert Baumeister, Alexander Knapp, Nora Koch and Gefei Zhang
5th International Conference on Web Engineering (ICWE 2005), Sydney, Australia, LNCS 3579, 406-416, 2005.
- **Modelling Business Processes in Web Applications with ArgoUWE**
Alexander Knapp, Nora Koch and Hanns-Martin Hassler
7th International Conference on the Unified Modeling Language (UML 2004), Lisboa, Portugal, LNCS 3273, 69-83, 2004.