ITBIS393
Web-Based Information Systems

Chapter 3: Modeling Web Applications
Wieland Schwinger, Nora Koch
Dr. Federico M. Facca
Prof. Dr. Gregor Engels
Prepared by Fadia Hijazie
# Modeling Web Applications

*Wieland Schwinger, Nora Koch*

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Introduction</td>
<td>39</td>
</tr>
<tr>
<td>3.2 Fundamentals</td>
<td>40</td>
</tr>
<tr>
<td>3.3 Modeling Specifics in Web Engineering</td>
<td>41</td>
</tr>
<tr>
<td>3.3.1 Levels</td>
<td>41</td>
</tr>
<tr>
<td>3.3.2 Aspects</td>
<td>42</td>
</tr>
<tr>
<td>3.3.3 Phases</td>
<td>42</td>
</tr>
<tr>
<td>3.3.4 Customization</td>
<td>43</td>
</tr>
<tr>
<td>3.4 Modeling Requirements</td>
<td>43</td>
</tr>
<tr>
<td>3.5 Content Modeling</td>
<td>45</td>
</tr>
<tr>
<td>3.5.1 Objectives</td>
<td>45</td>
</tr>
<tr>
<td>3.5.2 Concepts</td>
<td>45</td>
</tr>
<tr>
<td>3.6 Hypertext Modeling</td>
<td>46</td>
</tr>
<tr>
<td>3.6.1 Objectives</td>
<td>47</td>
</tr>
<tr>
<td>3.6.2 Hypertext Structure Modeling Concepts</td>
<td>47</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>3.6.3 Access Modeling Concepts</td>
<td>49</td>
</tr>
<tr>
<td>3.6.4 Relation to Content Modeling</td>
<td>50</td>
</tr>
<tr>
<td>3.7 Presentation Modeling</td>
<td>51</td>
</tr>
<tr>
<td>3.7.1 Objectives</td>
<td>51</td>
</tr>
<tr>
<td>3.7.2 Concepts</td>
<td>51</td>
</tr>
<tr>
<td>3.7.3 Relation to Hypertext Modeling</td>
<td>52</td>
</tr>
<tr>
<td>3.8 Customization Modeling</td>
<td>53</td>
</tr>
<tr>
<td>3.8.1 Objectives</td>
<td>54</td>
</tr>
<tr>
<td>3.8.2 Concepts</td>
<td>54</td>
</tr>
<tr>
<td>3.8.3 Relation to Content, Hypertext, and Presentation Modeling</td>
<td>58</td>
</tr>
<tr>
<td>3.9 Methods and Tools</td>
<td>58</td>
</tr>
<tr>
<td>3.9.1 Modeling Methods: An Overview</td>
<td>58</td>
</tr>
<tr>
<td>3.9.2 Model-Driven Development</td>
<td>61</td>
</tr>
<tr>
<td>3.9.3 Tool Support</td>
<td>61</td>
</tr>
<tr>
<td>3.10 Outlook</td>
<td>63</td>
</tr>
</tbody>
</table>
Overview

• Introduction
• Fundamentals
• Model Specifications of the Web
  – Levels
  – Aspects
  – Phases
  – Customization
• Modeling Requirements
• Reference Scenario
• Content Modeling
• Hypertext Modeling
• Presentation Modeling
• Customization Modeling
• Modeling Methods And Tools
• Model Driven Development
• Outlook: Wrap-up
Introduction

• A systematic approach and a specification of the Web application to be built in the form of visual models are recommended if we need to develop complex Web applications.
• Emphasis on model-based development of Web applications.
• 3.2 provides an insight into general modeling basics,
• 3.3 discusses the specifics in modeling Web applications.
• 3.4-3.8 describe different models for Web applications, starting from a requirements description. An example of an online conference paper reviewing system throughout these sections.
• 3.9 gives an overview of existing methods and some tools to model Web applications.
• 3.10 gives an overview of future development trends in the field of Web application modeling.
Why Create Models?

• Define an abstract view of a real-world entity
  – Finding and discovering objects/concepts in a domain
  – Assigning responsibilities to objects

• Tool of thought
  – Reduce complexity
  – Document design decisions

• Means of communication - GUI
Models

Models represent a solid starting point for the implementation of a Web application taking into account:

• static and dynamic aspects of the content,
• hypertext, and
• presentation levels of a Web application.
Content, Hypertext, and Presentation Models

• The content model of a Web application aims at capturing underlying information and application logic and is similar to the corresponding model of a non-Web application.
• The hypertext model represents all kinds of navigation possibilities based on the content.
• The presentation model maps hypertext structures to pages and their links thus represent the graphical user interface GUI.
The Inclusion Of Context Information

• The inclusion of context information, such as user, time, location, and device used, and the adaptation of the Web application which is “derived” from this information, has gained increasing attention in modeling efforts.

• This is undoubtedly a consequence of ubiquitous Web applications that have become increasingly popular.
The object of modeling is the application to be created of which its scope spans along three orthogonal dimensions.

**Levels** – in the sense of an encapsulation of the “how” & “what” of an application → inside, outside

**Aspects** – objects, attributes, and relationships; function and processes → structure, behavior

**Phases** – Development cycle (during which application has to be gradually refined and expanded → refinement steps)
Standard Development Approaches

• Modeling Language
  – Unified Modeling Language (UML 2.0)

• Implementation Language
  – Java, C++, C#, …
  – GUI Builder

• Development Process
  – RUP (Rational Unified Process)
  – Agile Process, eXtreme Programming
• **Levels** – Information, node/link structure, UI and page layout separate.
• **Aspects** – (structure, behavior) Same as Software Applications
• **Phases** – Approach depends upon type of application
• **Customization** – Context information
Why adding a new layer?

The tools of the trade in Web application modeling are basically not new; however, traditional modeling methods (such as UML) are not adopted for the web since they do not provide appropriate concepts for the specification aspects of hyperlinks.

To model Web applications, the document-like character of its content as well as its non-linear hypertext navigation has to be taken into account. Hence a new layer is needed! Why?

- Corresponds to the status of web applications
- Improve Reuse

Content, is the content as displayed in the web application → It is the information structure.

Several pages access the same content → If you model it you can reuse it.
Benefits of Level Separation

• model evolution
• layered model stack
  – different hypertext structure on top of the same content
  – different presentation models on top of the same hypertext model
• different modeling objectives
  – content: no redundancy vs. indepedence
  – hypertext: planned redundancy, i.e., information may be retrieved via different navigation paths
Levels

**Content** is the information structure and application logics underneath the Web application. The aim of a content model is the explicit definition of the information structure.

- **structure**: domain-dependent data
- **behavior**: domain-dependent application logic

**Hypertext**: the structuring of the content into nodes and links between these nodes

- **structure**: page compositions and navigational relationships
- **behavior**: run-time behavior of hypertext

**Presentation GUI**

- **structure**: user interface design elements and their composition (page layout)
- **behavior**: reactions to input events, interaction and synchronization between user interface elements

A clear separation of these three levels allows **Reuse** and helps to **reduce complexity**.
Aspects

- Aspects refer to the levels of the application logic structure as well as application logic behavior.
- The relevance of the structure and behavior models depends on the type of Web application to be implemented.
- Web applications which make mainly static information available require less behavior modeling compared with highly interactive Web applications e.g. e-commerce
- Its modeling formalism has to cope with the specific characteristics of each of the three levels.
- **different amount of structure and behavior**
  - less behavior in static website applications
  - focus on behavior in highly interactive applications
- **unified modeling approach recommendable**
Phases

• No consensus about a general modeling approach for the development of Web applications - no general agreed process for web applications yet

• The sequence of steps to model the levels should be decided by the modeler

There are different approaches:

• **Information-driven** ("content first")

• **Presentation-driven** ("layout first") → start with modeling of the application’s presentation aspects.

• **Logic based** or prototype-driven, agile ("test first")
Customization

• It considers the context, e.g., users’ preferences, device characteristics, or bandwidth restrictions, and allows to adapt the Web application accordingly.

• It influences all three Web modeling dimensions of content, hypertext, and presentation with respect to structure and behavior.

• All phases of the development process should be taken into account adaptation (customization), evolution
  – on all levels
  – of all aspects
  – during all phases to be taken into account!
Modeling Requirements

Various techniques can be used to identify, analyze, describe, evaluate, and manage Web application requirements.

• Use cases are the preferred modeling technique for **functional requirements** since they can be represented graphically. The overall functionality of a Web application is modeled as a set of use cases, which describe the Web application requirements.

• Use cases can be supplemented by UML activity diagrams to describe the functional requirements in more detail.

• Among the Web application requirements is **navigation functionality**. (Baresi et al. 2001) suggests separating the functional from the navigational use cases, creating two distinct models.

• **UWE**, “*Online Conference Paper Reviewing System*”, creates one single use case model, which uses the UML navigation stereotype to denote the difference between functional and hypertext-specific use cases.

• **Navigational requirements** supplementing functional requirements are made explicit through the stereotype navigation in the use case diagram.
Use case diagram of the reviewing system
Activity Diagram Of The Submission Process
A simple walkthrough case study

Reference SCENARIO
The Conference Review System

• This case study was presented at IWWOST (International Workshop on Web-Oriented Software Technology) 2001 to compare different Web application modeling methods.

• The purpose of the system is to support the process of submission, evaluation, and selection of papers for a conference.

• Reviewed methods includes:
  – UWE (UML Web Engineering)
  – OOHDM (Object-Oriented Hypermedia Design Method)
  – WSDM (Web Services Distributed Management)
  – WEBML (Web Modeling Language)
The Conference Review System - Methods

- **UWE** modeling methods, which defines
  - Navigational links (between nodes)
  - Process links (point to start point of a process)
  - External links (point to a node not directly belonging to the application)
- **OOHDM** is an approach to model scenarios where the Hypertext structure model can be built directly from the navigational requirements identified by these scenarios)
- **WSDM** pronounced *wisdom* is a web service standard for managing and monitoring the status of other services.
- **WEBML** is a visual notation for designing complex data-intensive Web applications. It provides graphical, yet formal, specifications, embodied in a complete design process, which can be assisted by visual design tools, like WebRatio. This method has five models: structure, derivation, composition, navigation and presentation. These models are developed in an iterative process.
Actors I

• PC Chair
  – creating the conference
  – determining the conference topics (or tracks) and subjects
  – establishing the Program Committee
  – defining the final list of accepted and rejected papers
  – defining the conference deadlines: submission, review, and notification.

• PC Member
  – evaluating a set of papers assigned to him
  – indicating another person as a reviewer of a paper
  – advising the PC Chair for the final list of accepted papers
Actors II

• Reviewer
  – responsible for reviewing a paper

• Author
  – submitting a paper for acceptance at the conference
  – PC Members and Reviewers may also be Authors, they must have different Ids for each role
Functions I: Paper Submission

• Any registered author may submit a paper
  – The author must register: the title, the abstract, the conference track, and a set of subjects chosen from a list previously determined by the PC Chair, if there is one
  – The system, after checking the authors’ registrations, assigns a paper ID to the new paper, and allows the user to submit it by uploading a file
  – At any moment, an author is allowed to check the data about his submitted papers. Until the submission deadline, the author is also allowed to substitute the uploaded file by a new one, or to change any of the informed data about the paper
Functions II: Assignment of Papers to PC Members

• The PC Chair may indicate potential conflicts of interest between PC Members and submitted papers
• Once the submission deadline has been reached
  – PC Members may indicate their interest and also conflicts of interest with some papers
  – In case of conflict of interest, the PC Member will not see any information about the paper
  – The PC Chair assigns papers to PC Members for reviewing, an email message with the list of papers, and a URL to a page where he can access these papers is sent
Functions III: Entering a Review

• A PC Member, or a Reviewer, may enter a review for a paper assigned to him.
• The review is entered by accessing a form containing all the evaluation items.
• A PC Member may see other reviews (entered by others) for any of the papers he is reviewing, but only after he has entered his own review.
• The PC Chair has full access to all papers and all reviews.
Function IV: Choosing Accepted and Rejected Papers

• Once the review deadline has been reached, the review process is closed

• The PC Chair, taking into account the recommendations of the PC Members and reviewers, chooses the papers that will be accepted and rejected

• Once the process is marked as finalized by the PC Chair, the system issues a notification message to paper authors, which includes the appropriate parts of the reviews submitted by the PC Members and reviewers
How to model the data underlying a Web application

CONTENT MODELING
Introduction I

• Purpose: To model the information requirements of a Web application
  – Diagramming the structural (i.e., information objects) and the behavioral aspects of the information.
  – NOT concerned with navigation.
• Primary Models
  – Class diagrams – enough for static applications.
  – State machine diagrams – captures dynamic aspects

Remark. Structure of the information is fixed even if it changes \(\rightarrow\) Consistency \(\rightarrow\) to keep cognitive stress on users low
Introduction II

Content modeling includes the creation of the problem domain model, consisting of static and dynamic aspects (Traditional Software Engineering). In addition the following Web application characteristics have to be taken into account:

• *Document-centric character and multimedia: It is necessary to take all kinds of different media formats into account when modeling the content, including the structures the information is based on.*

• *Integration of existing data repositories and software components* which were not created for Web applications originally. Content modeling has to satisfy two potentially contradicting objectives
  — it should cover the content requirements of the Web application
  — it should include existing data structures and software components.
Class Diagram For The Reviewing System

- Followed is a diagram that models a conference to be held on a number of topics.
- Users can sign in to the conference and submit their papers.
- A paper is subject to a review by three reviewers.
- Notice the invariant attached to the class “Paper”: it ensures that authors won’t be able to review their own papers.
- This class diagram will later serve as the basis to model the hypertext and the presentation for the example application.
Content Structure Model

Subject
  name
  description

Composition
  1..* 1

Conference
  name
  submissionDate
  reviewDate
  notificationDate
  status
  conferenceDate
  newConference()
  closeReviewing()

User
  name
  organization
  email
  login
  passwd
  loggedIn
  login()
  register()

Paper
  paperID
  title
  abstract
  url
  status
  /result
  upload()
  assign()
  updateStatus()

Review
  originality
  technicalQuality
  relevance
  overallRating
  enterReview()

Relation
  +users
  +reviewers
  * 1..*

Class name
Class attributes
Class operations
Invariant
Derived attribute
A submitted paper will be assigned to three reviewers for review after the submission deadline has expired. If a pre-set threshold value is reached, the paper is accepted; otherwise, it is rejected.
How to model the hypertext of a Web application
Introduction

• Purpose: To model the navigation paths available to users

• UWE Artifacts (work of art)
  – Hypertext Structure Model – navigating among classes
  – Access Model – UML-compliant site map

• Focuses on the structure of the hypertext & access elements

• Use “<<navigation class>>” annotation to distinguish from content classes

• Considered as a view on the content models → sometimes called navigational view
Hypertext Modeling

• Should rely on recurring navigation patterns

• The non-linearity of hypertext is one of the most important properties to be taken into account when modeling Web applications.

• The design should be achieved using suitable access structures, i.e., navigation options, to avoid the risk of users getting lost and putting them under excessive cognitive stress
Navigation Structure Model

Hypertext structure model of the PC’s view on the viewing system
Different Models… Different Links…

• HDM (Hypertext Design Model)
  – Structural links – connects elements of the same node
  – Perspective links – put various views in a relation to each other (PS, PDF)
  – Application links – put different nodes in relation to each other

• WebML (Web Modeling Language)
  – Contextual links – carry context link
  – Non-contextual links – have no associated context information
  – Intra-page links
  – Inter-page links
Navigation Access Model

- Hypertext structure models describe navigation, but not orientation.
- Access models describe both through Navigation patterns, used to consistently describe conventional elements.
  - <<menu>> (list of heterogeneous objects)
  - <<index>> (list of objects of the same type)
  - <<guided-tour>> (sequential links)
  - <<query>>

Note. UWE uses UML stereotypes i.e. <<menu>> (e.g. Conference), <<index>> (e.g. “Reviewing status”), <<query>> (e.g. “SearchPaperByTitle”) and <<guided-tour>> to specify the menu, index, query and guided tour access structure.
Navigation Access Model

Simplified access model of the hypertext model
Remarks

• Is it clear to what page each component belongs?
• I.e. what the relation with the previous schema?
• Page identity is missing! The models cannot be mixed together: this is not intuitive.
How to model the look & feel of a Web application
Introduction

• Purpose: To model the structure and behavior of the user interface and thus the look & feel of the Web application at the page (central element) level.
• The design should aim for simple and self-explanatory web application.
• It Describes presentation structure:
  – Composition and design of each page
  – Identify recurring elements (headers/footers)
• It Describes presentation behavior:
  – Elements => Events
Levels of Presentation Models

• Presentation Page – “root” element; equivalent to a page container.
• Presentation Unit
  – A fragment of the page logically defined by grouping related elements.
  – Represents a hypertext model node
• Presentation Elements - the basic building block
  – Represents a unit’s informational components (node’s set of information)
  – Can include Text, images, buttons, fields
Presentation Structure Model

Presentation pages of the reviewing system
Interaction overview diagram of the reviewing system
Presentation Behavior Model

Sequence diagram for retrieving a list of assigned papers
Presentation Behavior Model

Sequence diagram for displaying selected papers
How to model the representation of context information adapted by Web application
Customization Modeling

- Relevant proposals for customization originate from the fields of personalization
- Customization modeling is still a very young field, and only a few existing methods for Web application modeling offer a form of customization modeling
- For the UWE methodology have recently proposed an aspect-oriented approach to deal with customization.
Customization Modeling - Objective

• Aim to explicitly represent context information, and the adaptations derived from it.
• Depending on the modeling method the result is not always an explicit customization model.
• In most cases, customization modeling is intermingled with content, hypertext, and presentation models.
• It considers context information that can be predicted at modeling time which can assume different values when the Web application is run.
• Customization requires examining the Web application’s usage situation, i.e., dealing with the questions of “what” should be adapted and “when”.
Dynamic Adaptation Of An Index In The Hypertext Model.
Dynamic Adaptation Of A Page In The Presentation Model
Modeling Methods and Tools - An Overview

- Methods available for Web application modeling are normally based on traditional methods, such as ER, or they enhance an object-oriented modeling language, e.g., UML. More recent methods usually build on the strengths of earlier methods.
- Modeling methods follow different paradigms, depending on their origin and focus:
  - **Data-oriented** originated from the field of database systems. Examples Relationship Management Methodology (RMM), Hera, and the Web Modeling Language (WebML).
  - **Hypertext-oriented** based on the hypertext character of Web applications. Example Hypertext Design Model (HDM), extended into W2000 and HDM-lite or the Web Site Design Method (WSDM)
  - **Object-oriented** based on either OMT or UML. UML is the preferred notation when a standard language for modeling is selected. This category includes the Object-Oriented Hypermedia Design Method (OOHDM), UML-based Web Engineering (UWE), Object-Oriented Web Solutions (OOWS) and the Object-Oriented Hypermedia (OO-H) method
  - **Software-oriented** that look at Web applications mainly from the perspective of traditional software development, using techniques that strongly follow classical Software
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>HDM-lite</td>
<td>HT</td>
<td>ER + own notation</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>v</td>
<td>×</td>
<td>s</td>
<td>own</td>
<td>generation tools</td>
<td>auto</td>
<td>process for model transformation, automatic generation</td>
<td></td>
</tr>
<tr>
<td>Hera</td>
<td>DB</td>
<td>ER + RMM + own notation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>s + b</td>
<td>own</td>
<td>authoring &amp; generation tool</td>
<td>semi</td>
<td>model-driven development</td>
<td></td>
</tr>
<tr>
<td>OO-H</td>
<td>OO</td>
<td>UML + own notation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>pers</td>
<td>s + b</td>
<td>own</td>
<td>modeling- &amp; generation tool</td>
<td>auto</td>
<td>tool for automatic generation</td>
</tr>
<tr>
<td>OOHDM</td>
<td>OO</td>
<td>UML + own notation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>pers</td>
<td>s + b</td>
<td>own</td>
<td>×</td>
<td>×</td>
<td>powerful concepts for contextual navigation, personalization</td>
</tr>
<tr>
<td>UWE</td>
<td>OO</td>
<td>UML</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>pers</td>
<td>s + b</td>
<td>RUP</td>
<td>extended UML tool &amp; generation tools</td>
<td>semi</td>
<td>UML-based method, model-driven development, aspect-oriented customization</td>
</tr>
<tr>
<td>W2000 (HDM)</td>
<td>HT</td>
<td>UML</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>pers</td>
<td>s</td>
<td>×</td>
<td>extended UML tool</td>
<td>×</td>
<td>user-centric hypertext modeling</td>
</tr>
<tr>
<td>WAE2 (WAE)</td>
<td>SW</td>
<td>UML</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>pers</td>
<td>s + b</td>
<td>RUP</td>
<td>standard UML tools</td>
<td>×</td>
<td>implementation design, architectural design</td>
</tr>
<tr>
<td>WebML</td>
<td>DB</td>
<td>ER, UML</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>pers</td>
<td>s + b</td>
<td>own</td>
<td>modeling- &amp; generation tool</td>
<td>auto</td>
<td>well-elaborated notation, database integration, generation</td>
</tr>
<tr>
<td>WS DM</td>
<td>HT</td>
<td>own notation</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>s + b</td>
<td>own</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>user-centric approach for analysis</td>
</tr>
</tbody>
</table>

- ✓ supported
- × not supported
- s structure modeling
- b behavior modeling
- auto automatic generation
- semi semi-automatic generation
- RUP Rational Unified Process
- own own process model / approach
- HT hypertext-oriented
- OO object-oriented
- DB data-oriented
- SW software-oriented
Historical development of methods for Web application modeling.
Genre
(from g:Genre)
name: xsd:string
artists:Idx Artists By Genre(self)

Artist
(from a:Artist)
name: xsd:string
tracks:Idx Tracks By Artist(self)
genres:Idx Genres By Artist(self)
isGenreOf 1..*

Track
(from t:Track)
title: xsd:string
artistName: anchor (Ctx Artist Alpha (a:Artist where a plays t))
plays 1..*

Menu
Artists
Genres
Tracks
Artists
Alphabetic
By Genre
Tracks
Alphabetic
By Artist
WebML
Web Modeling

• Modeling static and dynamic aspects of content, hypertext, and presentation

• We focus on object-oriented analysis and design
  – Analysis: Finding and discovering classes of objects/concepts in a domain
  – Design: Defining software objects and how they interact to fulfill requirements.
Objects

• Software entities – like real-world entities - that consist of states and behaviors
  • States:
    – Variables store the states of an object’s properties
    – Hidden from the outside world (data encapsulation)
  • Behaviors:
    • Methods define the object’s behaviors
    • Used by objects to communicate with other objects

• Classes
  – blueprints for creating objects of a particular type
Discovering Objects in a Domain

• The way we represent a domain’s software model should resemble the physical model as closely as possible.
• To find key classes of objects:
  – Reuse existing models, if they exist
  – Make a category list
    • People, places, things
    • Transactions
    • Events
  – Identify noun phrases
• When naming classes, use terms that are commonly used in the domain
  – i.e., terms users would understand

We saw something about “sharing” and “reusing” knowledge. Where? Can it help?
Assigning Responsibilities

• Responsibilities are an object’s obligations, or behaviors related to its role in the system

• What does an object do?
  – Doing something (to) itself
  – Pass actions (messages) to other objects
  – Controlling and coordinating the activities in other objects

• What does an object know?
  – Private, encapsulated data
  – Its related objects
  – Items it can derive or calculate
• From sketch models to code models
Why Models at All?

• When it comes down to it, the real point of software development is cutting code
• Diagrams are, after all, just pretty pictures
• No user is going to thank you for pretty pictures; what a user wants is software that executes
Remarks

Provision of an expressive modeling language
... to allow the specification, construction, visualization, and
documentation of different artifacts of a software system
... to build different types of diagrams
... to provide for the exchange of models

Meaning of the term »Unified«
Support of the whole development process
Flexibility with respect to process models
Independence from development tools/platforms and
programming languages
Employment for various application areas
Genericity of language concepts defined in the meta-model
Integration of »Best Practices«
Unified Modeling Language (UML)

• “The Unified Modeling Language is a visual language for specifying and documenting the artifacts of systems.”

• Language of choice (and ISO standard) for diagramming notation in OO development
  – Structural – Class diagrams (domain models)
  – Behavioral – Use Cases, Sequence diagrams

• Currently at version 2.0, although many analysts and designers still use 1.0
The Role of Model in the Development

- **Models as sketch**
  - For communicating ideas and alternatives
  - Essence: Selectivity
  - "Sketchers" don’t have to care much about

- **Models as blueprint**
  - All design decisions (maybe of a particular area) are laid out
  - Essence: Completeness – programming should be pretty straightforward
  - Issue of reverse engineering

- **Models as program**
  - Applications are automatically generated
  - In MDA separation of Platform Independent Model (PIM) and Platform Specific Model (PSM);
    ideally the transformation is "parameterized" by a Platform Description Model (PDM)
  - Essence: models become the source code
  - Semantics and transformations (QVT, ATL) the holy grail of MDA
Model - Code Interplay

Code only

Code Visualization

Model

Code

Roundtrip Engineering

Model

Code

Model-centric / Model-driven

Model

Code

"Models as Code"

Model only

Model
Model-Driven ...

• Systematic development on basis of models
• Models become the first hand artifacts in the software development cycle

• Key concepts
  – abstraction from implementation detail
  – systematic transformations

• Related Terminology
  – Model Driven [Software] Engineering (MDE),
  – Model Driven [Software] Development (MDD/MDSD),
  – Model Driven Architecture (MDA)
  – Model Driven Web Engineering (MDWE)
Model-Driven Development (MDD)

The Vision

- Should go far beyond the notion of CASE tools of the 80’s
- Reduced gap between problem and realization domain
  - models as primary artefact throughout the lifecycle instead of code
  - models as program instead of models as sketch/blueprint
- Systematic transformations of abstract models to concrete implementations
  - multiple levels of abstractions, (e.g., OMG’s PIM, PSM and PDM)
  - horizontal (M2M) and vertical (M2C) transformations
- Standards for uniform storage, exchange, and transformation of models, e.g., OMG’s
  - MOF (Meta Object Facility) and Eclipse’s realization “Ecore”
  - XMI (XML Metadata Interchange) and
  - OCL (Object Constraint Language)
Model-Driven Development (MDD)
Model-Driven Engineering (MDE)
Modeling Objectives

detailed specification
- as base for automatic model transformation
- as input for realization / coding

- reducing the complexity
- documentation of design decisions
- readable description of system structure and functionality
- visualization of relevant system aspects
Technological Obsolescence

We don't want anymore to pay such a high price for simply moving our information system to a new middleware platform (COM, CORBA, Java, HTML, XML, DotNet, etc.) when our business system stays stable.

We are prepared to pay a last price for building the abstract models of our business and services that will guarantee us against technological obsolescence.

From there, any platform provider will also have to provide the mapping solutions from standard business models before we buy.
What is Model Driven Architecture?

• MDA is defined and developed by the Object Management Group (OMG) since March 2001
• MDA is:
  – "Model-Driven …"-framework for software development, defined by the OMG
  – open, vendor-neutral approach to interoperability using OMG's modeling specifications:
    • Unified Modelling Language (UML), Meta-Object Facility (MOF) and Common Warehouse Model (CWM)
• Main ideas:
  – Addresses the complete system development life cycle
  – Separate specification from implementation
  – Specify a system that is independent of a platform
  – Transform a platform-independent (PIM) system specification into a specific platform (PSM)
  – Code generation
Remarks

The MDA is an innovative approach to constructing an enterprise architecture. It is an initiative created by the Object Management Group. OMG’s focus has always been, and still is, to help users solve integration problems, in a productive way, by supplying open, vendor-neutral interoperability specifications, which is being achieved by the widely adopted CORBA standard.

The MDA is OMG’s next step in solving integration problems, which is upward from application implementation to the level of application design, which takes full advantage of the successful Unified Modeling Language (UML) standard, also developed by OMG.
Model-Driven Architecture (MDA)
Multi-target Code Generation

PIM

Platform-Independent Model

CORBA
Java/EJB
C#/DotNet
Web/XML/SOAP

SMIL/Flash

data grid computing pervasive computing cluster computing

+ SVG, GML, Delphi, ASP, MySQL, PHP, etc.

eetc.
MDA™: OMG Vision

The OMG (Object Management Group) is in the ideal position to provide the model-based standards that are necessary to extend integration beyond the middleware approach...Now is the time to put this plan into effect. Now is the time for the Model Driven Architecture.

Richard Soley and the OMG staff, MDA Whitepaper. November 27, 2000
MDA™ Logo
Developing in the MDA

• **PIM**
  – Platform Independent Model (PIM) represents business functionality and behavior without technology details

• **PSM**
  – Applies a standard mapping to create or generate a Platform Specific Model (PSM) from the PIM

• **Code Model**
  – Create or generate the code for PSM
Modeling Methods (not all are MDA)

- ER
- OMT
- HDM
- HDM-lite
- WEBML
- W2000
- OOHDM
- UWE
- WAE
- WAE2
- HERA
- OO-H
- OOWS
- WSDM
- MDA
- OOHDM

- Data-oriented
- Hypertext-oriented
- Object-oriented
- SW-oriented

Academic 2009-2010 1st Semester
UML for Web Engineering

• The book adopts the UML Web Engineering (UWE) notation
  – UML-compliant
  – Comprehensive modeling tool
  – Download it here:
    http://www.pst.ifi.lmu.de/projekte/uwe/
  – Requires MagicDraw UML (the free edition is enough)

• In the next lecture we will see WebML in details
Things to keep in mind (Outlook or WRAP - UP)

• Modeling is fundamental
  – Helps development
  – Support communication

• Model Driven Design and Development
  – Automatic code generation of Web applications

• One model for each layer
  – Content
  – Navigation
  – Presentation

• Different methods have different expressive power
Resources

**Mandatory reading**
- Web Engineering
  - Chapter 3

**Suggested**
- First International Workshop on Web-Oriented Software Technology
  - [http://www.dsic.upv.es/~west/iwwost01/](http://www.dsic.upv.es/~west/iwwost01/)
- UML-based Web Engineering
  - [http://www.pst.informatik.uni-muenchen.de/projekte/uwe/index.html](http://www.pst.informatik.uni-muenchen.de/projekte/uwe/index.html)