ABSTRACT

LEARNING WEB MODELING LANGUAGE USING MOODLE

Nergis Kılınç

In this thesis, objectives, basic elements, applications developed and the future of Web Modeling Language - new method for designing data-intensive web applications - are taught by using Moodle Learning Management System. Thus, definitions, functions and advantages of Learning Management Systems are exposed.

Necessary technical background to learn Web Modeling Language is presented at the beginning of the course. By taking the advantages of Moodle, the course is presented as weekly lessons by means of slides including one or two topics, dependent on the content. At the end of each week, relevant topic is consolidated by examples and exercises. To evaluate the knowledge level of students', a term examination and a final examination is submitted. Information about developed applications and the future of Web Modeling Language is added to the end of the course.

Keywords: Data-intensive Web Applications, Learning Management Systems, Web Modeling Language, Moodle,

ÖZET

MOODLE KULLANILARAK WEB MODELLEME DİLİ ÖĞRENİMİ

Nergis Kılınç

Bu tez ile, Veri-Yoğunluklu Web Uygulamaları Tasarımı alanında yeni bir yöntem olan Web Modelleme Dili'nin amaçları, temel öğeleri, yapılan uygulamalar ve geleceği ile ilgili bilgilerin verilmesi için Moodle Öğrenim Yönetim Sistemi kullanıldı. Böylece Öğrenim Yönetimi Sistemleri'nin tanımları, işlevleri ve faydaları ortaya konuldu.

Web Modelleme Dili için gerekli olan teknik alt yapı kurs başlangıcında notlar halinde sunuldu. Moodle Moodle Öğrenim Yönetim Sistemi'nin imkanlarından faydalınarak, kurs haftalık ders programları halinde, konunun yoğunluğuna bağlı olarak bir ya da ikişer konu içerecek biçimde, slaytlar şeklinde öğrencilere sunuldu. Her hafta sonunda öğrenilen konu ilgili alıştırma ve örneklerle pekiştirildi. Öğrencilerin bilgi düzeylerini ölçmek için bir vize ve bir final sınavı verildi. Web Modelleme Dili ile ilgili yapılan uygulamalar ve projenin geleceği ile bilgiler kurs sonunda eklendi.

Anahtar kelimeler: Veri-Yoğunluklu Web Uygulamaları Tasarımı, Öğrenim Yönetim Sistemi, Web Modelleme Dili, Moodle.

ACKNOWLEDGEMENTS

I would like to thank my senior supervisor, Prof. Selahattin Kuru, for all the supports he has given to me and his guidance throughout my work and for keeping me motivated with his great advises when I loose my belief to finish this thesis.

Lastly, I would like to thank my parents and Seha my fiancé, for their love, encouragement, and support.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

This thesis aimed to give information about objectives, basic elements and the future of Web Modeling Language - new method for designing data-intensive web applications - using Moodle Learning Management System - a student-centred learning management system - functionalities. Thus, definitions, functions and advantages of Learning Management Systems are introduced.

In Chapter 2, E-Learning concept is considered which Moodle is a member of. Description of e-learning, technologies used and advantages are discussed. The requirements for e-learning are described. E-learning styles are argued in a pedagogical perspective. Teacher-centred learning and students-centred learning which Moodle's style are compared. Conditions which must be supported develop an effective solution for providing an e-learning system are listed. Which technological background and platforms are used to use e-learning were described.

In Chapter 3, Learning Management System description, platforms used, objectives and why it is stands for is described. What the necessities for evaluating a learning management system are listed. LMS standards were introduced and compared. Open, commercial and self-developed Learning Management Systems are listed.

In Chapter 4, there was given information of definition, history, platforms used and the use of Moodle. "Social constructionist pedagogy" which is a philosophy of Moodle is discussed. Features, functionalities and modules of Moodle are introduced.

In Chapter 5, Web Modeling Language is discussed according to what the Web Modeling Language is, why it stands for, when, where it was designed by which team. Objectives, technologies used and architecture are explained.

Basic WebML models (Structural Model, Hypertext Model, Presentation Model and Personalization Model), design of Web applications, development lifecycle, operations and update processes are explained in detail. Each topic explained by giving examples, graphical and textual notations.

Other well-known Web Design Models developed are introduced and compared.

In Chapter 6, WebRatio Site Development Studio a tool which supports the WebML design process is introduced. Features, functions, main aspects and flow diagram are described.

In Chapter 7, explains how WebML is taught using Moodle. Technical background, platforms and tools for teaching WebML using Moodle are listed firstly. Modules and which format used are described. Course is given by using a weekly format. Each week considers one or two main topics. On or two lessons are added to each week. Resources, assignments, glossary, examinations are added to weeks to strengthen of learning of topics.

To discuss and learn the opinions of students about the course, two forums are added.

After students finish lessons and take examinations, grades are explained in the course site.

2 E-LEARNING

2.1 What is E-Learning?

E-learning is an education method offered using electronic delivery methods such as personal computers, CD-ROMs, video conferencing, digital television, P.D.A.s and mobile phones, websites and e-mail. It is often used in distance-learning programs [1,2].

E-learning may also be used to support distance learning through the use of WANs (Wide Area Networks), and may also be considered to be a form of flexible learning where just-in-time learning is possible. Courses can be tailored to specific needs and asynchronous learning is possible. Where learning occurs exclusively online, this is called online education. When learning is distributed to mobile devices such as cell phones or PDAs, it is called M-learning.

As opposed to the computer-based training (CBT) of the 1980s, the term e-learning is most frequently used to refer to computer-based training which incorporates technologies that support interactivity beyond that which would be provided by a single computer [2].

Figure 2.1 shows main functions of an E-Learning System:

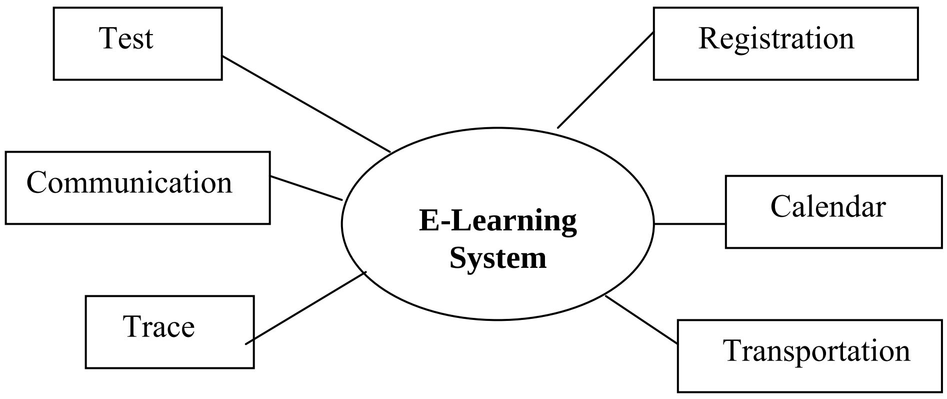


Figure 2.1 Main functions of an E-Learning System

2.2 Supporting Learning Online

Some view e-learning as a means to effective or efficient etc. learning, due to its ease of access and the pace being determined by the learner. Others point out that e-learning software developers tend to limit their focus on course delivery and content, while online education institutions require a much wider range of educational services.

Others are critical of e-learning in the context of education, because the face-to-face human interaction with a teacher has been removed from the process, and thus, some argue, the process is no longer "educational" in the highest philosophical sense. However, these human interactions can be encouraged through web-conferencing programs.

Further, continual advances in technology allow a wider range of learning experiences such as educational animation to be made available to support online learning.

E-learning systems such as Moodle often work towards a student-centred learning solution, building upon a social-constructivist pedagogy [2].

2.3 E-Learning Standards

2.3.1 AICC

The Aviation (All Emcompassing) Industry CBT (Computer-Based Training) Committee (AICC) is an international association of technology-based training professionals. The AICC develops guidelines for aviation industry in the development, delivery, and evaluation of CBT and related training technologies [8].

2.3.2 SCORM

Shareable Content Object Reference Model (SCORM) is a standard for web-based e-learning. It defines how the individual instruction elements are combined on a technical level and sets conditions for the software needed for using the content. The standard uses XML and it is based on the results of work done by AICC, IMS, IEEE and Ariadne [5].

2.3.3 IMS

IMS Global Learning Consortium (IMS) is a non-profit standards organization concerned with establishing interoperability for learning systems and learning content and the enterprise integration of these capabilities. Their mission is to "support the adoption and use of learning technology worldwide" [9].

2.4 Pedagogy of E-Learning

One important point is to help teachers organize their pedagogical perspective.

More recent approaches focus on dialogue, interaction and collaborative activities - courses still contain content but it is of secondary importance or is generated by the students. An open source course management system that makes this approach easier is Moodle. This advocates Social-Constructivism as a pedagogical perspective, whereby learners construct their knowledge through discussions, thereby enhancing their thinking skills [2].

2.4.1 Student-Centred Learning

Student-centered learning is an approach to education focusing on the needs of the students, rather than those of others involved in the educational process, such as teachers and administrators. This approach has many implications for the design of curriculum, course content, and interactivity of courses.

For instance, a student-centered course may address the needs of a particular student audience to learn how to solve some job-related problems using some aspects of mathematics. In contrast, a course focused on learning mathematics might choose areas of mathematics to cover and methods of teaching which would be considered irrelevant by the student [2].

2.4.2 Teacher-Centred Learning

Teacher Centred (TCL) learning styles offer distinct advantages to lecturers in situations where they desire tight control over the learning process and greater predictive capacity over assessment. Furthermore, TCL allows lecturers to control and systematise the structure, content and pace of unit delivery in a way that gives primacy to knowledge base of the lecturer [4].

2.4.3 Student-Centred Learning vs Teacher-Centred Learning

- **Teacher-Centred**

    - Teachers serve as the centre of epistemological knowledge, directing the learning process and controlling student's access to information.
    - Students viewed as 'empty' vessels and learning is viewed as an additive process.
    - Instruction is geared for the 'average' student and everyone is forced to progress at the same rate [3].

- **Student-Centred**

    - Backed by research that students are not empty vessels. They come with their own perceptual frameworks.
    - Students learn in different ways.
    - Learning is an active dynamic process in which connections are constantly changing and their structure is continually reformatted.
    - Students construct their own meaning by talking, listening, writing, reading, and reflecting on content, ideas, issues and concerns [3].

Table 2.1 compares instructional variables associated with teacher and student-centred approaches to teaching and learning. It provides a useful guide to the creation or evaluation of student-centred learning environments.

Table 2.1 Comparison of instructional variables associated with teacher and student-centred approaches to teaching and learning [3]

Instructional Variable	**Instructional Approach**	
	Teacher-Centred	**Student-Centred**
Learning Outcomes	• Discipline-specific verbal information. • Lower order thinking skills, e.g. recall, identify, define. • Memorisation of abstract and isolated facts, figures and formulas.	• Interdisciplinary information and knowledge. • Higher order thinking skills, e.g. problem-solving. • Information processing skills, e.g. access, organise, interpret, communicate information.
Goals and Objectives	• Teacher prescribes learning goals and objectives based on prior experiences, past practices, and state and/or locally mandated standards.	• Students works with teachers to select learning goals and objectives based on authentic problems and students' prior knowledge, interests and experience.
Instructional Strategy	• Instructional strategy prescribed by teacher. • Group-paced, designed for 'average' student. • Information organised and presented primarily by teacher, e.g. lectures, with some supplemental reading assignments.	• Teacher works with students to determine learning strategy. • Self-paced, designed to meet needs of individual student. • Student given direct access to multiple sources of information, e.g. books, online databases, community members.
Assessment	• Assessment used to sort students. • Paper and pencil exams used to assess students acquisition of information. • Teacher sets performance criteria for students. • Students left to find out what teacher wants.	• Assessment is integral part of learning. • Performance based, used to assess students ability to apply knowledge. • Students work with teachers to define performance criteria. • Students develop self-assessment and peer assessment skills.
Teacher's Role	• Teacher organises and presents information to groups of students. • Teachers act as gatekeeper of knowledge, controlling students' access to information. • Teacher directs learning.	• Teacher provides multiple means of accessing information. • Teacher acts as facilitator, helps students access and process information. • Teacher facilitates learning.
Student's Role	• Students expect teachers to teach them what's required to pass the test. • Passive recipients of information. • Reconstructs knowledge and information.	• Students take responsibility for learning. • Active knowledge seekers. • Construct knowledge and meaning.
Learning Environment	• Students sit in rows. • Information presented via lectures, books and films.	• Students work at stations with access to multiple resources. • Students work individually at times but also need to collaborate in small groups.

2.5 Organizing the Content

To develop an effective solution for providing an e-learning system, we must consider the questions below:

- For whom do we create e-learning content?
- What do they need to learn?
- How do we organize the content?
- Which tools/platform do we use? For creating? For delivering?
- Which media do we use (e.g. Text, Pictures, Sound, Video, Educational animation, Interactive exercises) [3]?

2.6 Structure

Most often HTML is used to bind together the different e-learning media. Sometimes XML based files are created which are then rendered to HTML/CSS/JavaScript by using an XSLT transformation. However often proprietary technologies are used like Macromedia Flash, an authoring tool, which leverages the JavaScript-like language called ActionScript to enable advanced functionality and interactivity. Data conferencing applications are sometimes used to share and manipulate e-learning media.

One standard for presenting e-learning content is SCORM whilst other specifications allow for the transporting of "learning objects" [3].

3 LEARNING MANAGEMENT SYSTEM

3.1 What is Learning Management System?

A Learning Management System (or LMS) is a software package, usually on a large scale (that scale is decreasing rapidly), that enables the management and delivery of learning content and resources to students. Most LMS systems are web-based to facilitate "anytime, anywhere" access to learning content and administration.

At a minimum, the LMS usually allows for student registration, the delivery and tracking of e-learning courses and content, and testing, and may also allow for the management of instructor-led training classes.

In the most comprehensive of LMSs, one may find tools such as competency management, skills-gap analysis, succession planning, certifications, virtual live classes, and resource allocation (venues, rooms, textbooks, instructors, etc.). Most systems allow for learner self-service, facilitating self-enrolment, and access to courses.

Some LMS vendors do not distinguish between LMS and LCMS, preferring to refer to both under the term "LMS", but there is a difference. The LCMS, which stands for "Learning Content Management System", facilitates organization of content from authoring tools, and presentation of this content to students via the LMS.

LMSs are based on a variety of development platforms, from J2EE-based architectures to Microsoft .NET, and usually employ the use of a robust database back-end. While most systems are commercially developed, free and open-source models do exist. Other than the most simplistic, basic functionality, all LMSs cater to, and focus on different educational, administrative, and deployment requirements.

Open source LMS is growing fast in the education and business world [6].

3.2 What are the Necessities for Evaluating a Learning Management System?

- **High availability:** the LMS must be robust enough to serve the diverse needs of thousands of learners, administrators, content builders and instructors simultaneously.
- **Scalability:** the infrastructure should be able to expand—or "scale"—to meet future growth, both in terms of the volume of instruction and the size of the student body.
- **Usability:** to support a host of automated and personalized services, such as self-paced and role-specific learning, the access, delivery and presentation of material must be easy-to-use and highly intuitive—like surfing on the Web or shopping on Amazon.com.
- **Interoperability:** to support content from different sources and multiple vendors' hardware/software solutions, the LMS should be based on open industry standards for Web deployments (XML, SOAP or AQ) and support the major learning standards (AICC, SCORM, IMS and IEEE).
- **Stability:** the LMS infrastructure can reliably and effectively manage a large enterprise implementation running 24 x 7.
- **Security:** as with any outward-facing collaborative solution, the LMS can selectively limit and control access to online content, resources and back-end functions, both internally and externally, for its diverse user community [7].

3.3 An Optimum Learning Management System

A learning management system optimally should:

- Consolidate training initiatives on a scalable, low-cost Web-based platform,
- Assemble and deliver learning content rapidly in multiple languages.
- Measure the effectiveness of training initiatives.
- Mix classroom and online learning.
- Integrate with other enterprise application solutions.

- Centralize and automate administration.
- Use self-service and self-guided services as much as possible.
- Support portability and standards: AICC, IMS and SCORM.
- Personalize content and enable knowledge re-use [7].

3.4 Worldwide Learning Management Systems Solutions

Table 2.2 gives short description of commercial and self-developed Learning Management Systems.

Table 2.2 Commercial and self-developed Learning Management Systems [10]

LMS Systems	**Original Nationality**	**Comments**
BlackBoard	American	In 1997, graduate students at Cornell University developed CourseInfo. Blackboard.com now owns CourseInfo, and changed the name of the product to BlackBoard with version 5.
ClassFronter	Norwegian	Classfronter is a Norwegian-developed system that has a very dominant position in Norwegian universities and colleges. The system is available in a number of languages and sold to institutions in several countries.
COM-C	Danish	COM-C is a web-based e-learning system developed by UNI-C, the Danish IT Centre for Research and Education.
Fle3 Future Learning Environment	Finish	Open source system Developed by Learning Environments for Progressive Inquiry Research Group.UIAH Media Lab, University of Art and Design Helsinki.
Ilias	German	Self-developed LMS by Virtus. ILIAS is available as an open source Learning Management system. Its development is coordinated by the Faculty of Economics, Business Administration and Social Sciences at the University of Cologne.
Lotus Learning Space	American	IBM Lotus has updated its e-learning software product family with new updated offerings, and we will be retiring the Lotus LearningSpace family.
Ping Pong	Swedish	PING PONG is a Learning Management System (LMS), developed and designed in Sweden. PING PONG offers integrated Learning Content Management (LCMS) capabilities, and Knowledge, Skills, and Competence Management (KMS & CMS) functionality.
TopClass (WBT Systems)	Irish	TopClass originated as a European Commission project at University College Dublin, in Ireland, before becoming an Irish campus company and then migrating to the United States.

Table 2.3 gives short description of open Learning Management Systems.

Table 2.3 Open Learning Management Systems [11]

Product Name	**Sponser/Developer** **Country of Origin**	**Notes**
ATutor 1.2	University of Toronto Canada	Adaptive Technology Resource Centre (reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=158)
Bazaar 7	University of Athabasca Canada	(reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=152)
CHEF	University of Michigan US	license permits the copying, modification, and distribution of the software as long as original license is included and notice of the changes to the original distribution.
Claroline 1.4	Université catholique de Louvain France	Institute for Education and Multimedia (reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=69)
ClassWeb 2.0	University of California Los Angeles US	(reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=118)
Eledge 1.2	Chuck Wight (University of Utah) US	(reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=73)
Jones e-education V2002	Jones Advisory Group US	Free to post-secondary but not really 'open source' - the software can be modified for an institution's own use, but may not be re- distributed under the license agreement that users must accept.(reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=41)
LearnLoop	ITuniversity in Gothenburg Sweden	Online-Learning community software
OLAT	University of Zuerich Switzerland	OLAT guest account access is availabe at http://www.olat.unizh.ch/
Manhattan Virtual Classroom 0.93	Western New England College US	(reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=139)
MimerDesk 1.5.3.1	Ionstream Finland	(reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=140)
Moodle 1.1	Martin Dougiamas Australia	(reviewed on the edutools site at http://www.edutools.info/course/productinfo/detail.jsp?id=149)
OpenUSS	University of Münster Germany	part of the larger CampusSource project
Stellar	Massachusetts Institute of Technology US	(as far as I know this has not yet been released publicly and is only used internally, but important as a future reference implementation for OKI)
WBT-Master	Graz University of Technology Austria	Institute for Information Processing and Computer Supported New Media,

4 MOODLE

4.1 What is Moodle?

Moodle is a software package for producing internet-based courses and web sites. It's an ongoing development project designed to support a social constructionist framework of education.

Moodle is provided freely as Open Source software (under the GNU Public License). Basically this means Moodle is copyrighted, but that you have additional freedoms. You are allowed to copy, use and modify Moodle provided that you agree to: provide the source to others; not modify or remove the original license and copyrights, and apply this same license to any derivative work. Read the license for full details and please contact the copyright holder directly if you have any questions.

The word Moodle was originally an acronym for Modular Object-Oriented Dynamic Learning Environment, which is mostly useful to programmers and education theorists. It's also a verb that describes the process of lazily meandering through something, doing things as it occurs to you to do them, an enjoyable tinkering that often leads to insight and creativity. As such it applies both to the way Moodle was developed, and to the way a student or teacher might approach studying or teaching an online course. Anyone who uses Moodle is a Moodler [12].

Moodle has been evolving since 1999 (since 2001 with the current architecture). Current version is 1.5, which was released on 6 June 2005. It has been translated into 61 different languages. Major improvements in accessibility and display flexibility have been developed in 1.5.

As of November 2005, nearly 7000 sites from 142 countries have registered their Moodle installation. The real number of current active Moodle installations is unknown, but Moodle is downloaded over 500 times a day. As there are no license fees and growth limit, an institution can add as many Moodle servers as needed. The largest single site has reported over 6,000 courses and over 45,000 students, and the Open University of the UK is building a Moodle installation for their 200,000 users.

The development of Moodle continues as a Free Software project supported by a team of programmers and the user community all over the world. This means, that users are free to download, use, modify and even distribute it (under the terms of the GPL License from GNU).

Moodle runs without modification on Unix, Linux, FreeBSD, Windows, Mac OS X, NetWare and any other systems that support PHP, including most webhost providers. Data is stored in a single database: MySQL and PostgreSQL are best supported, but it can also be used with commercial databases, ADO and generic ODBC database access, since it uses ADOdb [16].

4.2 Philosophy of Moodle

The design and development of Moodle is guided by a particular philosophy of learning, a way of thinking that you may see referred to in shorthand as ”social constructionist pedagogy". This term is explained by four main concepts:

- **Constructivism:** This point of view maintains that people actively construct new knowledge as they interact with their environment.
- **Constructionism:** Constructionism asserts that learning is particularly effective when constructing something for others to experience. This can be anything from a spoken sentence or an internet posting, to more complex artifacts like a painting, a house or a software package.
- **Social Constructivism:** This extends the above ideas into a social group constructing things for one another, collaboratively creating a small culture of shared artifacts with shared meanings. When one is immersed within a culture like this, one is learning all the time about how to be a part of that culture, on many levels.
- **Connected and Separate:** Separate behaviour is when someone tries to remain 'objective' and 'factual', and tends to defend their own ideas using logic to find holes in their opponent's ideas. Connected behaviour is a more empathic approach that accepts subjectivity, trying to listen and ask questions in an effort to understand the other point of view. Constructed behaviour is when a person is sensitive to both of these approaches and is able to choose either of them as appropriate to the current situation [13].

4.3 Features of Moodle

4.3.1 Overall Design

- Promotes a social constructionist pedagogy (collaboration, activities, critical reflection, etc)
- Suitable for 100% online classes as well as supplementing face-to-face learning
- Simple, lightweight, efficient, compatible, low-tech browser interface
- Easy to install on almost any platform that supports PHP. Requires only one database (and can share it).
- Full database abstraction supports all major brands of database (except for initial table definition)
- Course listing shows descriptions for every course on the server, including accessibility to guests.
- Courses can be categorised and searched - one Moodle site can support thousands of courses
- Emphasis on strong security throughout. Forms are all checked, data validated, cookies encrypted etc
- Most text entry areas (resources, forum postings etc) can be edited using an embedded WYSIWYG HTML editor

4.3.2 Site Management

- Site is managed by an admin user, defined during setup
- Plug-in "themes" allow the admin to customise the site colours, fonts, layout etc to suit local needs
- Plug-in activity modules can be added to existing Moodle installations
- Plug-in language packs allow full localisation to any language. These can be edited using a built-in web-based editor. Currently there are language packs for over 70 languages.
- The code is clearly-written PHP under a GPL license - easy to modify to suit your needs

4.3.3 User Management

- Goals are to reduce admin involvement to a minimum, while retaining high security
- Supports a range of authentication mechanisms through plug-in authentication modules, allowing easy integration with existing systems.
- Standard email method: students can create their own login accounts. Email addresses are verified by confirmation.
- LDAP method: account logins can be checked against an LDAP server. Admin can specify which fields to use.
- IMAP, POP3, NNTP: account logins are checked against a mail or news server. SSL, certificates and TLS are supported.
- External database: any database containing at least two fields can be used as an external authentication source.
- Each person requires only one account for the whole server - each account can have different access
- An admin account controls the creation of courses and creates teachers by assigning users to courses
- A course creator account is only allowed to create courses and teach in them
- Teachers may have editing privileges removed so that they can't modify the course (e.g. for part-time tutors)
- Security - teachers can add an "enrolment key" to their courses to keep out non-students. They can give out this key face-to-face or via personal email etc
- Teachers can enrol students manually if desired
- Teachers can unenrol students manually if desired, otherwise they are automatically unenrolled after a certain period of inactivity (set by the admin)
- Students are encouraged to build an online profile including photos, description. Email addresses can be protected from display if required.
- Every user can specify their own timezone, and every date in Moodle is translated to that timezone (e.g. posting dates, assignment due dates etc)
- Every user can choose the language used for the Moodle interface (English, French, German, Spanish, Portuguese etc)

4.3.4 Course Management

- A full teacher has full control over all settings for a course, including restricting other teachers
- Choice of course formats such as by week, by topic or a discussion-focussed social format
- Flexible array of course activities - Forums, Quizzes, Glossaries, Resources, Choices, Surveys, Assignments, Chats, Workshops
- Recent changes to the course since the last login can be displayed on the course home page - helps give sense of community
- Most text entry areas (resources, forum postings etc) can be edited using an embedded WYSIWYG HTML editor
- All grades for Forums, Quizzes and Assignments can be viewed on one page (and downloaded as a spreadsheet file)
- Full user logging and tracking - activity reports for each student are available with graphs and details about each module (last access, number of times read) as well as a detailed "story" of each students involvement including postings etc on one page.
- Mail integration - copies of forum posts, teacher feedback etc can be mailed in HTML or plain text.
- Custom scales - teachers can define their own scales to be used for grading forums and assignments
- Courses can be packaged as a single zip file using the Backup function. These can be restored on any Moodle server.

4.3.5 Assignment Module

- Assignments can be specified with a due date and a maximum grade.
- Students can upload their assignments (any file format) to the server - they are date-stamped.
- Late assignments are allowed, but the amount of lateness is shown clearly to the teacher
- For each particular assignment, the whole class can be assessed (grade and comment) on one page in one form.

- Teacher feedback is appended to the assignment page for each student, and notification is mailed out.
- The teacher can choose to allow resubmission of assignments after grading (for regrading)

4.3.6 Chat Module

- Allows smooth, synchronous text interaction
- Includes profile pictures in the chat window
- Supports URLs, smilies, embedded HTML, images etc
- All sessions are logged for later viewing, and these can also be made available to students

4.3.7 Choice Module

- Like a poll. Can either be used to vote on something, or to get feedback from every student (e.g. research consent)
- Teacher sees intuitive table view of who chose what
- Students can optionally be allowed to see an up-to-date graph of results

4.3.8 Forum Module

- Different types of forums are available, such as teacher-only, course news, open-to-all, and one-thread-per-user.
- All postings have the authors' photo attached.
- Discussions can be viewed nested, flat or threaded, oldest or newest first.
- Individual forums can be subscribed to by each person so that copies are forwarded via email, or the teacher can force subscription for all
- The teacher can choose not to allow replies (eg for an announcements-only forum)
- Discussion threads can be easily moved between forums by the teacher
- Attached images are shown inline
- If forum ratings are being used, these can be restricted to a range of dates

4.3.9 Quiz Module

- Teachers can define a database of questions for re-use in different quizzes
- Questions can be stored in categories for easy access, and these categories can be "published" to make them accessible from any course on the site.
- Quizzes are automatically graded, and can be re-graded if questions are modified
- Quizzes can have a limited time window outside of which they are not available
- At the teacher's option, quizzes can be attempted multiple times, and can show feedback and/or correct answers
- Quiz questions and quiz answers can be shuffled (randomized) to reduce cheating
- Questions allow HTML and images
- Questions can be imported from external text files
- Quizzes can be attempted multiple times, if desired
- Attempts can be cumulative, if desired, and finished over several sessions
- Multiple-choice questions supporting single or multiple answers
- Short Answer questions (words or phrases)
- True-False questions
- Matching questions
- Random questions
- Numerical questions (with allowable ranges)
- Embedded-answer questions (cloze style) with answers within passages of text
- Embedded descriptive text and graphics

4.3.10 Resource Module

- Supports display of any electronic content, Word, Powerpoint, Flash, Video, Sounds etc. that are stored locally, or remotely
- Files can be uploaded and managed (zipped, unzipped, renamed, moved, etc..) on the server
- Folders can be created and managed on the server and linked to
- Internal web pages (html formatted) can be created with WYSIWYG editor and linked to

- Internal text pages (no formatting) can be created and linked to
- External content on the web can be linked to or seamlessly included within the course interface.
- External web applications can be linked to with data passed to them
- Linked MP3 audio files will display with elegant flash player

4.3.11 Survey Module

- Built-in surveys (COLLES, ATTLS) have been proven as instruments for analysing online classes
- Online survey reports always available, including many graphs. Data is downloadable as an Excel spreadsheet or CSV text file.
- Survey interface prevents partly-finished surveys.
- Feedback is provided to the student of their results compared to the class averages

4.3.12 Workshop Module

- Allows peer assessment of documents, and the teacher can manage and grade the assessment.
- Supports a wide range of possible grading scales
- Teacher can provide sample documents for students to practice grading
- Very flexible with many options [14].

4.4 Case for Moodle

As I discussed in Chapter 4.2 - "What the necessities are for evaluating a Learning Management System" –, Moodle provides all the requirements [15].

5 WEB MODELING LANGUAGE

5.1 Introduction

Today, data-intensive Web applications are the predominant kind of application found on the Web; sites for online trading and e-commerce, institutional Web sites of private and public organizations, digital libraries, corporate portals, and community sites are all examples of data-intensive Web applications.

The development of a data-intensive Web application is a multi-disciplinary activity, which requires a variety of skills, necessary to address very heterogeneous tasks, like the design of data structures for storing content, the conception of hypertext interfaces for information browsing and content management, the creation of effective presentation styles, the assembly of robust and high performance architectures, and the integration with legacy applications and external services. The development and maintenance of data-intensive Web applications requires all the tools and techniques of software engineering, including a well-organized software development process, appropriate design concepts and notations, and guidelines on how to conduct the various activities [17].

State-of-the-practice Web development tools help simplify the generation and deployment of data-intensive Web applications by means of page generators, such as Microsoft's Active Server Pages or JavaSoft's Java Server Pages. Even if these systems are very productive implementation tools, they offer scarce support to bridge the gap between requirements collection and the subsequent phases of the development process [18].

By looking at the way in which data-intensive Web applications are built today and at the tools available to developers, one realizes soon that the software engineering principles and pragmatics are not exploited to their full potential.

This gap is particularly apparent in the design concepts and notations: when it comes to specifying the front-end of their Web application, development teams resort to rather rudimentary tools, like paper and pencil or HTML mock-ups. This situation, which we have frequently witnessed also in very large organizations well equipped with software engineering tools, demands for an adaptation of the software development process, capable of addressing the characterizing features of Web applications.

WebML consists of simple visual concepts for expressing a hypertext as a set of pages made up of linked content units and operations, and for binding such content units and operations to the data they refer to.

All the proposed notations fit perfectly in the commercial tool suites popular among developers, like Entity-Relationship and UML editors and code generators. In particular, WebML can be easily supported, either by representing WebML diagrams using UML, or by exploiting WebML-aware tools [17].

The Web Modeling Language (WebML) allows Web sites to be conceptually described. The focus of WebML is primarily from the user's view and the data modeling. Our model derived from the software is complementary to the solutions proposed by WebML [19].

While the language WebML is capable of modeling simple dynamic features of a data-intensive web application by providing operation units for creating, deleting and modifying entities, it does not support more complex structures such as modular, nestable dialog sequences [20]. WebML is more suited for the high level specification of web application than for modeling the actual implementation because it lacks the concepts needed to model control flow [21].

5.2 History of WebML

The model-driven approach to Web application development is the result of more than five years of research at Politecnico di Milano, an Italian IT School in Italy. The first research prototype of a model-driven CASE tool for Web applications, called AutoWeb, was designed by Piero Fraternali and Paolo Paolini between 1996 and 1998. The tool, operational since 1997, has been used to develop several Web applications, and has demonstrated the possibility of automating the construction of data-intensive Web sites specified with a high level conceptual language.

WebML was conceived in the context of the Esprit project “Web-Based Intelligent Information Infrastructures” (W3I3, 1998–2000), supported by the European Community, with the participation of five partners (Politecnico di Milano and TXT e-solutions from Italy, KPN Research from Holland, Digia Inc. from Finland, Otto Versand from Germany); the project delivered a prototype development environment, called ToriiSoft. Since 1999, WebML has been used for the development of industrial Web applications, both inside research contracts with companies such as Microsoft and Cisco Systems, and in industrial projects with companies like TXT e-solutions and Acer Europe. In the fall 2001, a team of WebML designers and developers founded a start-up company with the goal of further developing, distributing, and marketing WebRatio Site Development Studio, a tool suite based on WebML [17].

5.3 What is Web Modeling Language?

The Web Modeling Language (WebML) is a visual notation for specifying the composition and navigation features of hypertext applications.

Building on such popular standards as Entity-Relationship and UML, the Web Modeling Language lets us to specify complex Web applications in a platform-independent way. WebML is field-tested in tens of real applications [22].

WebML enables the high-level description of a Web site under distinct orthogonal dimensions: its data content (structural model), the pages that compose it (composition model), the topology of links between pages (navigation model), the layout and graphic requirements for page rendering (presentation model), and the customization features for one-to-one content delivery (personalization model).

All the concepts of WebML are associated with a graphic notation and a textual XML syntax. WebML specifications are independent of both the client-side language used for delivering the application to users, and of the server-side platform used to bind data to pages [18].

WebML follows the style of well-known conceptual modeling languages like Entity-Relationship and UML: every concept has a graphical representation, and specifications are diagrams. Therefore, the reader should not worry about the need to learn yet another language. As for the Entity-Relationship constructs, also WebML diagrams could be represented using the UML syntax, possibly with some loss of conciseness, but not of expressive power [17].

5.4 Architecture of WebML Web Applications

Figure 5.1 shows a diagram about the work flow and main parts of WebML Web applications.

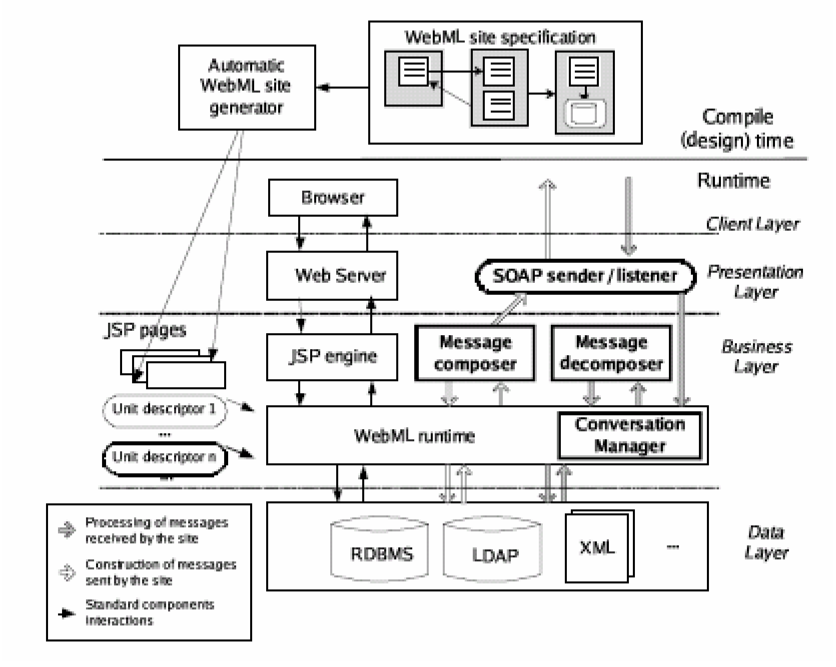


Figure 5.1 Architecture of WebML Web applications [24]

5.5 Objectives of WebML

WebML provides graphical, yet formal, specifications, embodied in a complete design process, which can be assisted by visual design tools. The main objectives of the WebML design process are:

- Expressing the structure of a Web application with a high-level description, which can be used for querying, evolution, and maintenance;
- Providing multiple views of the same content;

- Separating the information content from its composition into pages, navigation, and presentation, which can be defined and evolved independently;
- Storing the meta-information collected during the design process within a repository, which can be used during the lifetime of the application for dynamically generating Web pages;
- Modeling users and communities explicitly in the repository, to permit the specification of personalization policies and one-to-one applications;
- Enabling the specification of data manipulation operations for updating the site content or interacting with arbitrary external services [23].

5.6 Other Web Design Models

The typical modeling layers used in the process of designing an application include the conceptual or structural level (information domain structure and design), the hypertext level (composition and navigation structure of the application), the presentational level (user interface or application look-and-feel design), personalization level (customization design), and the implementation level. The extent of coverage of these layers varies from one design model to another, and most of them formally focus on three layers, that is the conceptual, hypertext and presentational levels. At the conceptual level, the information domain is captured and modeled using three main design techniques:

- **Entity-Relationship:** information objects and data structure described by means of entities and relationships,
- **Object-Oriented:** information objects modeled as objects/classes,
- **Ontology-based:** information objects modeled as ontology classes.

The concept of views or perspectives is used at the hypertext level to enable the modeling of different applications, providing static views over the same conceptual model. A few methods such as WSDM, OOHDM and WebML provide support for more flexible personalization features (content, link, structure or context customizations). The compositional and navigational structure of an application is built on nodes (pages, navigation units, content units, slices or cards) and different types of links (perspective, structural, application link, etc) between them. The navigation units (nodes units) are mapped to conceptual units (entities or classes) to display the information or data at rendering/presentation time. Although the design methods and models share at a higher level the similarities, they have several differences, including their main application domains, the level of coverage of the design process, the level of support provided at different stages.

- **HDM (Hypermedia Design Model)**

It is an early E/R-based design model proposed by Garzotto et al. to define the structure and interactions in large scale and read-only hypermedia systems. The model is suitable for domains with a high level of organisation, modularity and consistency.

It focuses on hierarchically describing the information objects in terms of entities made of components containing units of information as well as the navigation structure, independently of their implementation. This navigation structure comprises perspective links between units, structural links between components, application links between entities, components or units, index and guided tours. The presentation design consists of slots (unit of information) and frames (grouping of slots).

- **RMM (Relationship Management Methodology)**

It is E/R-based, suitable for structuredhypermedia applications and its design process consists of seven steps: entity-relationship design; slice design (grouping entitie's attributes as node/presentation units called slice or Mslice); navigational design (access methods: link, menus, index, guided tour, indexed guided tour); protocol conversion design (converting design components into physical objects); user interface design (screen layouts); run-time behaviour design and construction and testing.

- **OOHDM (Object Oriented Hypermedia Design Model)**

It is an OO-based design model that allows the specification of hypermedia applications as navigational views over the conceptual model. Its design process consists of four main dimensions and has recently been extended to formally cover requirements gathering and personalisation modelling. Navigation units or nodes are mapped to conceptual classes, and the design and generation of OOHDM-based read-only web sites is supported by a CASE tool called OOHDM-Web.

- **EORM (The Enhanced Object-Relationship Model)**

It is an OO-based methodology whose major characteristic is the representation of relationships between objects (links) as separate objects. Links are therefore first class objects stored in reusable libraries, facilitating the mapping of relations into link class. The method is based on three frameworks: Class framework, Composition framework, GUI framework. EORM has early prototyping of user interfaces, a richer typology of links and a CASE tool (ONTOS Studio) to support the modelling process.

- **SOHDM (Scenario-based Object-oriented Hypermedia Design Methodology)**

It is another OObased approach focusing on process-oriented hypermedia systems to support organisational processes. Scenarios are defined during the domain analysis and serve as the basis for object modelling and navigational design. Different types of OO views can be generated from the domain object model to compose a new application. It consists of six phases.

- **WDSM (The Web Site Design Method)**

It is a user-centred approach, as the application model is based on the user model, identifying user classes and their preferences and views. The design process for a reead-only web site comprises three main stages. The conceptual design consists of both the object modelling (which can be E/R or OO based) and navigational design.

- **OntoWebber**

It is an ontology-based approach to building read-only web sites, focusing on integrating heterogeneous data sources to build data-intensive web portals. A Domain Ontology serves as a reference ontology for data integration and content modelling, and as the starting point for the entire web site design. The site view modelling or site view graph consists of the navigation, content and presentation models; and further steps include personalisation and maintenance models. Nodes or navigation units are called cards which are mapped to ontology classes via the content model and the overall design is represented by an XMLbased meta-schema using RDF and DAML+OIL [25].

Table 5.1 makes a comparison about well-known Web Design Model features.

Table 5.1 Web Design Model features [25]

	Design process	**Navigation structure: Nodes and navigation units (nu)**	**Interactivity**	**Modeling Technique**
HDM	1. Authoring-in-the-large 2. Authoring-in-the-small	Entities, components, units (nu)	Read-Only	E-R
RMM	1. E-R design 2. Slice design 3. Navigational design 4. Conversion protocol design 5. IU screen design 6. Runtime behavior design 7. Construction and testing	outlines, slices (nu)	Read-Only	E-R
WebML	1. Structural model 2. Hypertext model 3. Composition model 4. Navigation model 5. Presentation model 6. Personalization model	Pages, Content units (nu)	Read-Write	E-R / OO
OOHDM	1. Conceptual design 2. Navigational design 3. Abstract interface design 4. Implementation	Navigation classes (nu)	Read-Only	OO
EORM	1. Class framework 2. Composition framework 3. GUI framework	Navigation classes (nu)	Read-Only	OO
SOHDM	1. Domain analysis 2. OO modeling 3. View design 4. Navigational design 5. Implementation design 6. Construction	classes	Read-Only	OO
WSDM	1. User modeling 2. Conceptual design Object modeling Navigational design 3. Implementation design 4. Implementation	Navigation classes (nu)	Read-Only	E-R / OO
Onto-Webber	1. Domain ontology 2. Site View modeling, navigation model, content model, presentation model 3. Personalization model 4. Maintenance model	Pages, Cards (nu)	Read-Only	Ontology

5.7 Preview of WebML Models

The specification of a site in WebML consists of four orthogonal perspectives:

- **Structural Model:** Expresses the data content of the site, in terms of the relevant entities and relationships. WebML does not propose yet another language for data modeling, but is compatible with classical notations like the E/R model, the ODMG object-oriented model, and UML class diagrams. To cope with the requirement of expressing redundant and calculated information, the structural model also offers a simplified, OQL-like query language, by which it is possible to specify derived information.

- **Hypertext Model:** Describes one or more hypertexts that can be published in the site. Each different hypertext defines a so-called site view. Site view descriptions in turn consist of two sub-models.

    - **Composition Model:** Specifies which pages compose the hypertext, and which content units make up a page. Six types of content units can be used to compose pages: data, multi-data, index, filter, scroller and direct units. Data units are used to publish the information of a single object (e.g., a music album), whereas the remaining types of units represent alternative ways to browse a set of objects (e.g., the set of tracks of an album). Composition units are defined on top of the structure schema of the site; the designer dictates the underlying entity or relationship on which the content of each unit is based.

    - **Navigation Model:** Expresses how pages and content units are linked to form the hypertext. Links are either non-contextual, when they connect semantically independent pages (e.g., the page of an artist to the home page of the site), or contextual, when the content of the destination unit of the link depends on the content of the source unit. For example, the page showing an artist's data is linked by a contextual link to the page showing the index of reviews of that specific artist. Contextual links are based on the structure schema, because they connect content units whose underlying entities are associated by relationships in the structure schema.

- **Presentation Model:** Expresses the layout and graphic appearance of pages, independently of the output device and of the rendition language, by means of an abstract XML syntax. Presentation specifications are either page-specific or generic. In the former case they dictate the presentation of a specific page and include explicit references to page content (e.g., they dictate the layout and the graphic appearance of the title and cover data of albums); in the latter, they are based on predefined models independent of the specific content of the page and include references to generic content elements (for instance, they dictate the layout and graphic appearance of all attributes of a generic object included in the page).

- **Personalization Model:** Users and user groups are explicitly modeled in the structure schema in the form of predefined entities called User and Group. The features of these entities can be used for storing group-specific or individual content, like shopping suggestions, list of favorites, and resources for graphic customization. Then, OQL-like declarative expressions can be added to the structure schema, which define derived content based on the profile data stored in the User and Group entities. This personalized content can be used both in the composition of units or in the definition of presentation specifications. Moreover, high-level business rules, written using a simple XML syntax, can be defined for reacting to site-related events, like user clicks and content updates. Business rules typically produce new user- related information (e.g., shopping histories) or update the site content (e.g., inserting new offers matching users' preferences). Queries and business rules provide two alternative paradigms (a declarative and a procedural one) for effectively expressing and managing personalization requirements [18].

5.8 Structural (Data) Model

The purpose of data modeling is enabling the specification of the data used by the application, in a formal yet intuitive way. The result of data modeling is a conceptual schema, which conveys in a simple and readable way the available knowledge about the application data.

Data modeling is one of the most traditional disciplines of Information Technology, for which well-established modeling languages exist. Entity-Relationship (E-R) model is used to express Structural Model.

The essential ingredients of the Entity-Relationship model are entities, defined as containers of structured data, and relationships, representing semantic associations between entities. Entities are described by means of typed attributes, and can be organized in generalization hierarchies, which express the derivation of a specific concept from a more general one. Relationships are characterized by cardinality constraints, which impose restrictions on the number of relationship instances an object may take part in.

5.8.1 Entities

An entity represents a description of the common features of set of objects of the real world. Examples of entities are Person, Car, Artist, and Album. An entity has a population, which is the set of objects that are described by the entity. These objects are also called the instances of the entity. For example, the population of entity Person is a specific set of persons, and the population of entity Car is a specific set of cars, and so on.

In Figure 5.2, an example of graphic notation for entities is given.

Album
Artist

Figure 5.2 Graphic notation for entities.

Attributes

Attributes represent the properties of real-world objects that are relevant for the application purposes. Examples of attributes are the name, address, and photo of a person. Entity is a descriptor of the common properties of a set of objects, and such properties are expressed as attributes. Attributes are graphically represented inside the entity box.

In Figure 5.3, an example of graphic notation for entities and attributes is given.

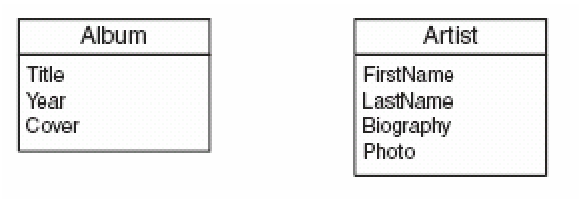


Figure 5.3 Graphic notation for entities and attributes.

All the instances of an entity must be distinguishable, by means of a unique identity that permits their unambiguous identification. To express the unique identity of entity instances, one or more attributes can be defined as the primary key of the entity. Primary key attributes must satisfy a few restrictions, not requested for regular attributes. Their value must be defined for every instance of the entity, and unique, which means that there should not exist two entity instances with the same value of the key attributes.

It is good practice to define the primary key of entities using a single special purpose attribute, called Object Identifier (OID), whose sole purpose is to assign a distinct identifier to each instance of an entity. Assume that the OID property is implicitly defined for all entities, and omit it from the Entity-Relationship diagrams.

Figure 5.3 shows an example of graphic notation for primary keys.

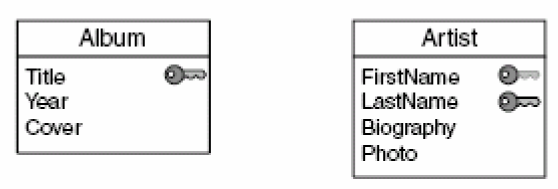


Figure 5.4 Graphic notation for primary keys.

The Entity-Relationship model does not prescribe any specific set of types for attributes, but it is good practice to express attribute types in the data model, both for making the specification more expressive, and for directing the data implementation.

In Table 5.2, we see a list of built-in data types.

Table 5.2 Typical built-in data types.

Data type	Description
String	A "short" sequence of characters
Text	A "long" sequence of characters. Text types can be further refined by expressing their MIME type (for example, text/html)
Integer	An integer numerical type
Float	A floating point numerical type
Date	A calendar date
Time	A temporal instant of time
Boolean	A true or false value
Enumeration	A sequence of user-defined values
BLOB	Binary Large OBject, for example an image or a video, which must be handled in a special way because of its size. BLOB types can be further refined by expressing their MIME type (for example image/gif)
URL	Uniform Resource Locator of a Web resource

Attribute types can be represented graphically, by means of a label positioned besides the attribute declaration in the entity box as in Figure 5.5.

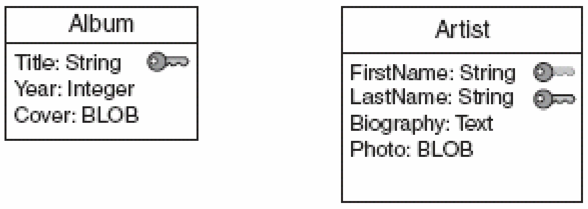


Figure 5.5 Graphic notation for attribute types.

Generalization Hierarchies

The Entity-Relationship model permits the designer to organize entities into a hierarchy, where they share some common features. The basic generalization hierarchy (also called IS-A hierarchy) has one super-entity and one or more sub-entities. Each sub-entity inherits all attributes and relationships defined in the super-entity and may add locally defined attributes and relationships.

- Each entity is defined as the specialization of at most one super-entity. In technical terms, “multiple inheritance” is avoided.
- Each instance of a super-entity is specialized exclusively into one sub-entity.
- Each entity appears in at most one generalization hierarchy.

Figure 5.6 shows a simple graphic notation for IS-A hierarchies.

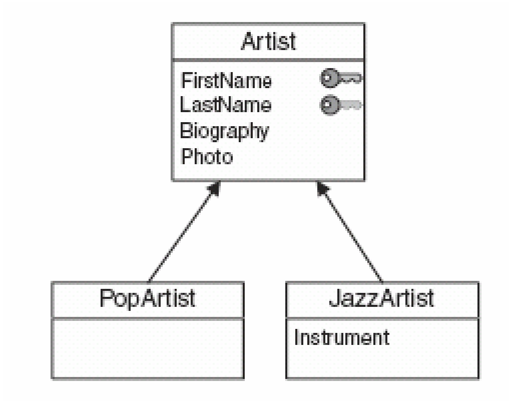


Figure 5.6 Graphic notation for IS-A hierarchies.

5.8.2 Relationships

Relationships represent semantic connections between entities, like the association between an artist and his/her album, or between an artist and his/her reviews. The meaning of the association is conveyed by the relationship’s name, which is established by the designer. For example, the relationship between an artist and the albums he/she has published could be named Publication. The simplest form of relationship is the binary relationship, which connects two entities. Relationships involving more than two entities, called N-ary relationships, are allowed; however, the use of N-ary relationships is discouraged, because they can be equivalently expressed by means of multiple binary relationships.

Relationship roles can be annotated with minimum and maximum cardinality constraints, respectively denoting the minimum and maximum number of objects of the destination entity to which any object of the source entity can be related as seen in Figure 5.7.

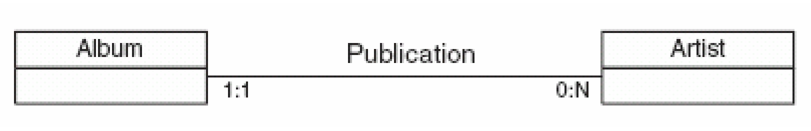


Figure 5.7 Graphic notation for relationships.

Each binary relationship is characterized by two relationship roles, each one expressing the function that one of the participating entities plays in the relationship. For example, in Figure 5.8, the relationship Publication between an artist and his/her album can be decomposed into two relationship roles, one from artist to album, named Publishes, and one from album to artist, named Published_By.

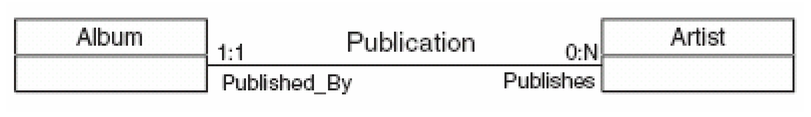


Figure 5.8 Graphic notation for relationship roles.

- Relevant values for the minimum cardinality are zero or one; a relationship is said to be optional for its source entity if the minimum cardinality is zero and mandatory otherwise. Mandatory relationships introduce existential dependencies between entities, because an object of the source entity cannot exist without being associated with at least one object of the destination entity.

- Relevant values for maximum cardinalities are one or many, the latter option being denoted as “N.”

Based on their maximum cardinality constraints, relationships are called “one-to-one,” if both relationship roles have maximum cardinality 1, “one-to-many,” if one relationship role has maximum cardinality 1 and the other role has maximum cardinality N, or “many-to-many,” if both relationship roles have maximum cardinality N.

Figure 5.9 shows N-ary relationships are expressed as a primitive construct using binary relationships and entities.

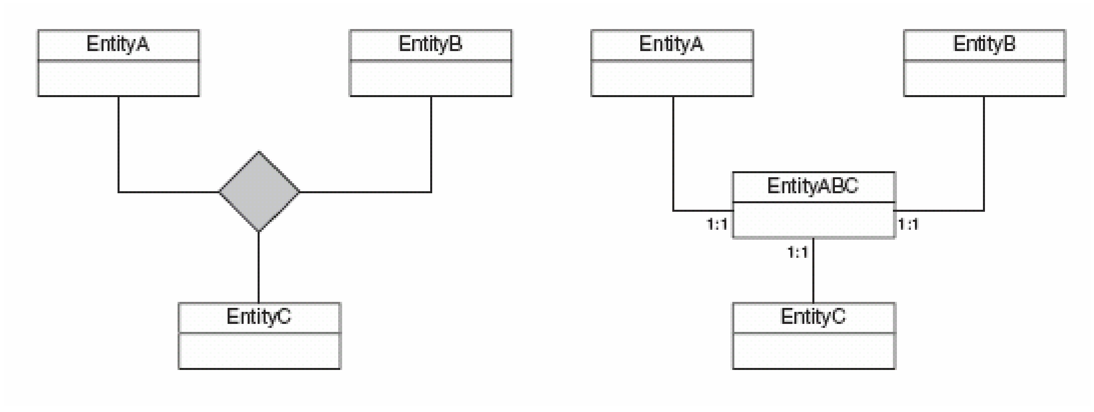


Figure 5.9 On the left, N-ary relationships expressed as a primitive construct. On the right, using binary relationships and entities.

5.8.3 Derived Information

The value of some attribute or relationship of an entity can be determined from the value of some other elements of the schema. Attributes and relationships that can be calculated are called derived.

The Entity-Relationship model does not include a standard notation for characterizing attributes and relationships as derived, nor a language for expressing their computation rule. However, the specification of an attribute or of a relationship can be easily extended, to support the modeling of derived information:

- An attribute or relationship is denoted as derived by adding a slash character (/) in front of the attribute or relationship name.
- The computation rule that defines the derived attribute or relationship is specified as an expression added to the declaration of the attribute or relationship.

The language used for writing derivation rules may be any language supporting expressions built from the attributes of an entity and path expressions denoting the traversal of relationships. In the following examples, the Object Constraint Language (OCL) will be used defined in the Unified Modeling Language (UML).

In Figure 5.10 and 5.11, examples of expression of derived attributes and derived relationships are shown respectively.

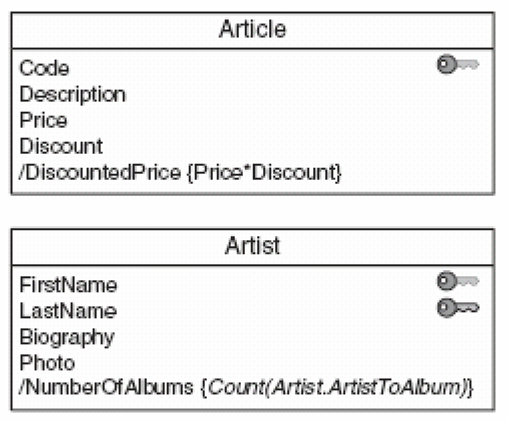

Article
Code
Description
Price
Discount
/DiscountedPrice {Price*Discount}
Artist
FirstName
LastName
Biography
Photo
/NumberOfAlbums {Count(Artist.ArtistToAlbum)}

Figure 5.10 Derived attributes.

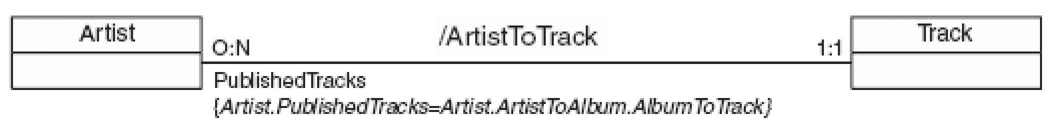


Figure 5.11 Derived relationship.

The following XML code represents derived information of the WebML specification of a structural schema representing information about musical albums, which are composed by artists, about whom are provided some reviews. Each album can contain several tracks:

```
<DOMAIN id="SupportType" values="CD Tape Vinyl">;

<ENTITY id="Album">
  <ATTRIBUTE id="title" type="String"/>
  <ATTRIBUTE id="cover" type="Image"/>
  <ATTRIBUTE id="year" type="Integer"/>
  <COMPONENT id="Support" minCard="1" maxCard="N">
    <ATTRIBUTE id="type" userType="SupportType"/>
    <ATTRIBUTE id="listPrice" type="Float"/>
    <ATTRIBUTE id="discountPercentage" type="Integer"/>
    <ATTRIBUTE id="currentPrice" type="Float"
              value="Self.listPrice *
                    (1 - (Self.discountPercentage / 100))"/>
  </COMPONENT>
  <RELATIONSHIP id="Album2Artist" to="Artist" inverse="ArtistToAlbum"
              minCard="1" maxCard="1"/>
  <RELATIONSHIP id="Album2Track to="Track" inverse="Track2Album"
              minCard="1" maxCard="N"/>
</ENTITY>

<ENTITY id="Artist">
  <ATTRIBUTE id="firstName" type="String"/>
  <ATTRIBUTE id="lastName" type="String"/>
  <ATTRIBUTE id="birthDate" type="Date"/>
  <ATTRIBUTE id="birthPlace" type="String"/>
  <ATTRIBUTE id="photo" type="Image"/>
              <ATTRIBUTE id="biographicInfo" type="Text"/>
              <RELATIONSHIP id="Artist2Album" to="Album" inverse="Album2Artist"
                          minCard="1" maxCard="N"/>
              <RELATIONSHIP id="Artist2Review" to="Review" inverse="Review2Artist"
                          minCard="0" maxCard="N"/>
             </ENTITY>

             <ENTITY id="Track">
               <ATTRIBUTE id="number" type="Integer"/>
               <ATTRIBUTE id="title" type="String"/>
               <ATTRIBUTE id="mpeg" type="URL"/>
               <ATTRIBUTE id="hqMpeg" type="URL"/>
               <RELATIONSHIP id="Track2Album" to="Album" inverse="Album2Track"
                          minCard="1" maxCard="1"/>
```

```
</ENTITY>

<ENTITY id="Review">
  <ATTRIBUTE id="text" type="Text"/>
  <ATTRIBUTE id="autho" type="String/>
  <RELATIONSHIP id="Review2Artist" to="Artist" inverse="Artist2Review"
          minCard="1" maxCard="1"/>
</ENTITY>
```

5.9 Hypertext Model

The goal of hypertext modeling is to specify the organization of the front-end interfaces of a Web application.

The hypertext model should be at the right level of abstraction; the specification of the hypertext should be maintained at the conceptual level, which means that it should not commit too much to design and implementation details, such as the actual distribution of functionality between the various tiers of a Web application.

Unlike data modeling, which is a very consolidated activity, hypertext modeling is a younger discipline, still lacking a well-established base of concepts, notations, and design methods.

The key ingredients of WebML are pages, units, and links, organized into modularization constructs called areas and site views.

5.9.1 Composition Model

The purpose of composition modeling is to define which nodes make up the hypertext contained in the Web site. Composition modeling specifies content units (units for short), i.e., the atomic information elements that may appear in the Web site, and pages, i.e., containers by means of which information is actually clustered for delivery to the user. In a concrete setting, e.g., an HTML implementation of a WebML site, pages and units are mapped to suitable constructs in the delivery language, e.g., units may map to HTML files and pages to HTML frames organizing such files on the screen [18].

Units

Units are the atomic elements for specifying the content of a Web page. WebML supports five types of units:

- **Data units:** show information about a single object.
- **Multidata units:** present information about a set of objects.
- **Index units:** show a list of descriptive properties of some objects, without presenting their detailed information.
- **Scroller units:** enable the browsing of an ordered set of objects, by providing commands for accessing the first, last, previous, and next element of a sequence.
- **Entry units:** model entry forms, whose fields allow gathering input, needed to perform searches or to feed update operations.

These basic types of content units can be combined to represent Web pages of arbitrary complexity. Data, multidata, index, and scroller units present content extracted from the data schema; therefore, it is necessary to specify where their content comes from. WebML uses two concepts for expressing the origin of a unit’s content: the source and the selector.

- The source is the name of the entity from which the unit’s content is extracted. Thus, the source entity tells the type of the objects used to compute the unit’s content. A content unit can be associated with one source entity, which is the most common case, or with multiple source entities.

- The selector is a predicate, used for determining the actual objects of the source entity that contribute to the unit’s content. Selectors are the conjunction of elementary conditions, built from the entity attributes and from the relationship roles in which the entity is involved, and from constant or variable terms. Variable terms are constructed using parameters associated with the input links of the unit. Selectors whose conditions use parameters are called parametric selectors.

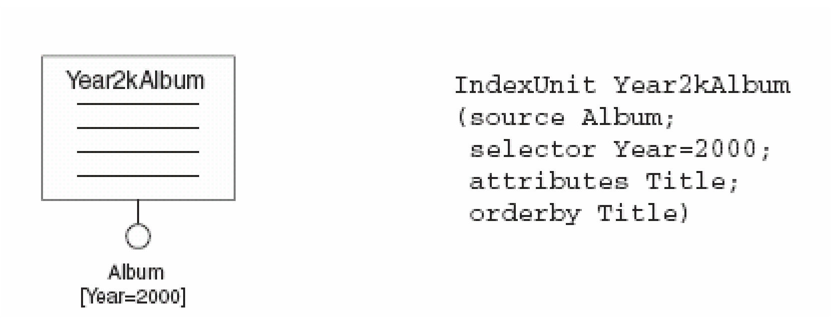
Data Units

Data units are defined to select a mix of information, which provides a meaningful view of a given concept of the structure schema. More than one unit can be defined for the same entity or component, to offer alternative points of view (e.g., a short or long, textual or multimedia version of the object) [18].

A data unit is characterized by the following properties:

- **Name:** the user-defined name for the data unit.
- **Source:** the entity providing the content to the unit.
- **Selector (optional):** a predicate identifying a unique object, which is displayed by the data unit. The selector of a data unit is optional, but it can be omitted only in the case in which the source entity has a single instance; otherwise, the object to be displayed in the data unit remains undefined.
- **Included attributes:** the set of attributes of the source entity to be visualized.

Figure 5.12 shows a simple example about graphical and textual notation for a content unit.



```
IndexUnit Year2kAlbum
(source Album;
 selector Year=2000;
 attributes Title;
 orderby Title)
```

Figure 5.12 Graphical and textual notation for a content unit.

Syntactically, data units are defined using the DATAUNIT element, which provides tags and attributes for the various aspects of unit definition. The selective inclusion of content in a unit is specified using the element INCLUDE. Included attributes must be chosen among those declared in the structure schema for the entity or component. The INCLUDEALL element can be used to specify that all attributes are included.

Following definitions introduce two units for presenting the Artist entity. The goal of these definitions is to provide a short view of artists [18].

```
<DATAUNIT id="ShortArtist" entity="Artist">
  <INCLUDE attribute="firstName"/>
  <INCLUDE attribute="lastName"/>
  <INCLUDE attribute="photo"/>
</DATAUNIT>

<DATAUNIT id="BiographyUnit" entity="Artist">
  <INCLUDEALL/>
</DATAUNIT>
```

As we see in Figure 5.13, there is a simple example about graphical and textual notation for a data unit and rendition in HTML.

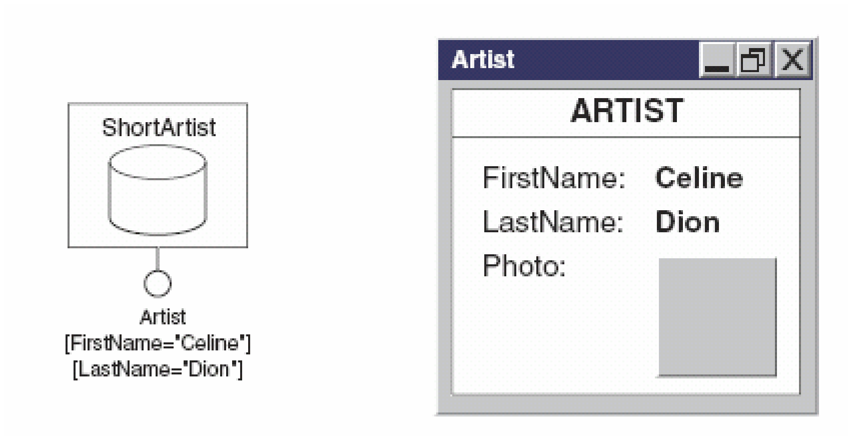


Figure 5.13 WebML graphic and textual notation for data units, and rendition in HTML.

Multidata Units

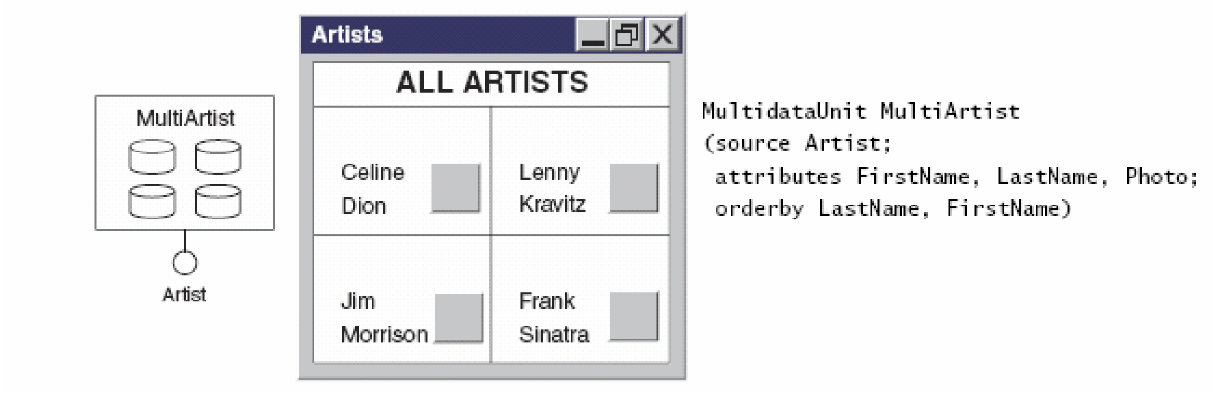
Multi-data units present multiple instances of an entity or component together, by repeating the presentation of several, identical data units [18].

A multidata unit is characterized by the following properties:

- **Name:** the user-defined name for the multidata unit.

- **Source:** the entity providing the content to the unit.
- **Selector (optional):** a selection predicate determining the objects displayed by the multidata unit. If the selector is missing, all objects are considered.
- **Included attributes:** the set of attributes of the source entity to be visualized for each object displayed by the multidata unit.
- **Order clause (optional):** the set of attributes used to sort the objects of the multidata unit and the sorting criterion to be applied, which can be ascending or descending. Ascending is assumed as default.

Figure 5.14 explains WebML graphic notation for multidata units, rendition in HTML, and a textual definition.



```
MultidataUnit MultiArtist
(source Artist;
 attributes FirstName, LastName, Photo;
 orderby LastName, FirstName)
```

Figure 5.14 WebML graphic notation for multidata units, rendition in HTML, and a textual definition.

Syntactically, a multi-data unit is represented by a MULTIDATAUNIT element, which includes a nested DATAUNIT element. The container is an argument of the external MULTIDATAUNIT element. The following example shows how all albums can be shown in the same multidata unit, by displaying all attributes of each individual album.

```
<MULTIDATAUNIT id="MultiAlbumUnit" entity="Album">
  <DATAUNIT id="AlbumUnit" entity="Album">
    <INCLUDEALL/>
  </DATAUNIT>
</MULTIDATAUNIT>
```

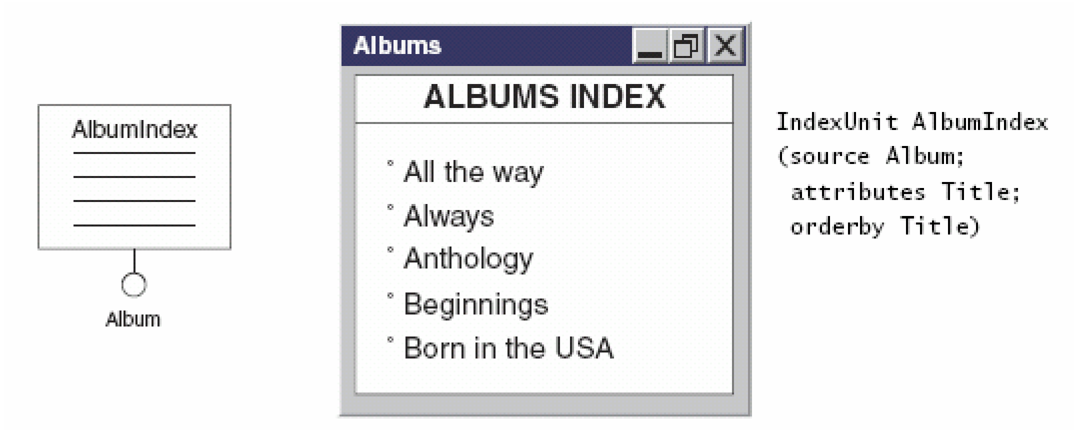
Index Units

Index units present multiple instances of an entity or component as a list, by denoting each object as an entry in the list.

An index unit specification includes the following properties:

- **Name:** the user-defined name for the index unit.
- **Source:** the entity providing the content to the unit.
- **Selector (optional):** a selection predicate determining the objects displayed by the unit. If the selector is missing, all objects are considered.
- **Included attributes:** the set of attributes of the source entity used to display the index entries.
- **Order clause (optional):** the set of attributes used to sort the objects of the index unit and the sorting criterion to be applied, which can be ascending or descending. Ascending is assumed as default.

In Figure 5.15, we see a WebML graphic notation for index units, a rendition in HTML, and a textual definition.



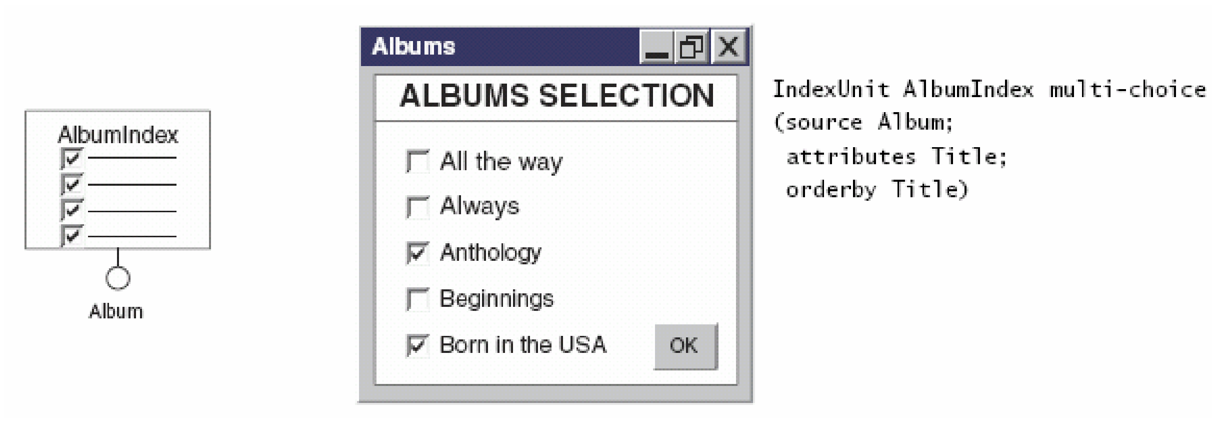
```
IndexUnit AlbumIndex
(source Album;
 attributes Title;
 orderby Title)
```

Figure 5.15 WebML graphic notation for index units, rendition in HTML, and a textual definition.

Syntactically, an INDEXUNIT element is used, which includes a nested DESCRIPTION element. The following example shows how all albums can be shown in a list, by displaying only the title of each individual album.

```
<INDEXUNIT id="AlbumIndex" entity="Album">
  <DESCRIPTION Key="title"/>
</INDEXUNIT>
```

In Figure 5.16, album instances are denoted by the title and listed in ascending order by title, as specified in the following textual definition, where the keyword multi-choice is added to the declaration of the index unit.



```
IndexUnit AlbumIndex multi-choice
(source Album;
 attributes Title;
 orderby Title)
```

Figure 5.16 WebML graphic notation for multi-choice indexes, rendition in HTML, and a textual definition.

Scroller Units

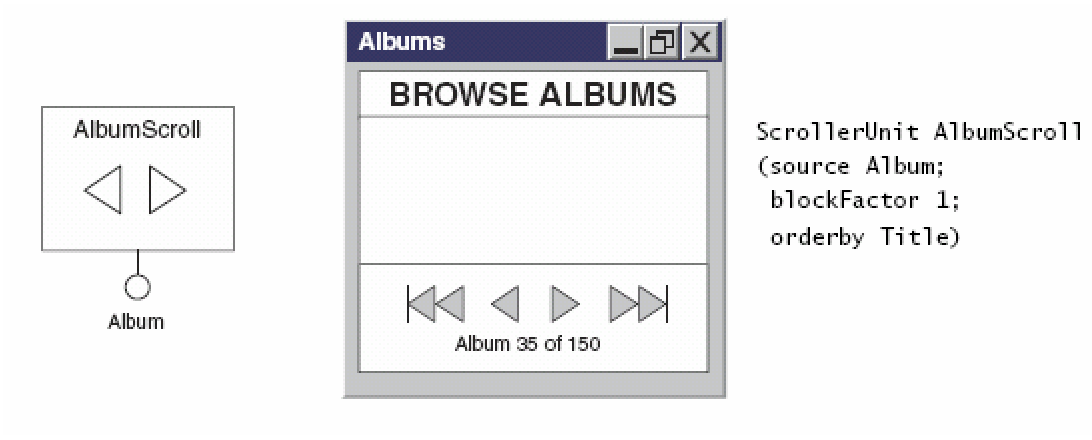
Scroller units provide commands to scroll through the objects in a container, e.g., all the instances of an entity or all the objects associated to another object via a relationship. A scroller unit is normally used in conjunction with a data unit, which represents the currently visualized element of the container [18].

A scroller unit specification is characterized by the properties:

- **Name:** the user-defined name for the scroller unit.

- **Source:** the entity providing the content to the unit.
- **Selector (optional):** a selection predicate determining the objects scrolled by the unit. If the selector is missing, all objects are considered.
- **Block factor:** the number of objects that are scrolled together. By default, the block factor is 1, which means that objects are scrolled one at a time.
- **Order clause (optional):** the set of attributes used to sort the objects of the scroller unit and the sorting criterion to be applied, which can be ascending or descending. Ascending is assumed as default.

Figure 5.17 shows an example of WebML graphic notation for scroller indexes, rendition in HTML, and a textual definition.



```
ScrollerUnit AlbumScroll
(source Album;
 blockFactor 1;
 orderby Title)
```

Figure 5.17 WebML graphic notation for scroller indexes, rendition in HTML, and a textual definition.

Syntactically, the SCROLLERUNIT element is used, which specifies the container (entity, relationship, or component) providing the set of objects to scroll, and suitable attributes to express which scrolling commands to use. For example, the following declaration introduces a unit for moving along the set of reviews of an artist, whereby it is possible to move to the first, previous, next and last review [18].

```
<SCROLLERUNIT id="AlbumScroll" entity="Album" first="yes"
          last="yes" previous="yes" next="yes"/>
```

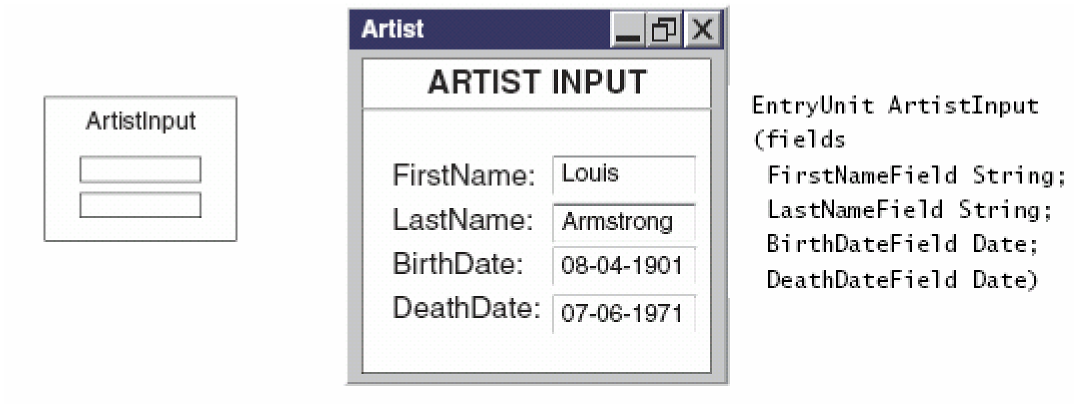
Filter Units

Filter units provide input fields to search the objects in a container, e.g., all the instances of an entity whose attributes contain a given string. A filter unit is normally used in conjunction with an index or multidata unit, which present object matching the search condition.

Entry units are characterized by the following properties:

- **Name:** the user-defined name for the entry unit.
- **Fields:** the set of fields for inputting values.

An example of WebML graphic notation for entry unit, rendition in HTML, and a textual definition is expressed in Figure 5.18.



FirstName:	Louis
LastName:	Armstrong
BirthDate:	08-04-1901
DeathDate:	07-06-1971

```
EntryUnit ArtistInput
(fields
 FirstNameField String;
 LastNameField String;
 BirthDateField Date;
 DeathDateField Date)
```

Figure 5.18 WebML graphic notation for entry unit, rendition in HTML, and a textual definition.

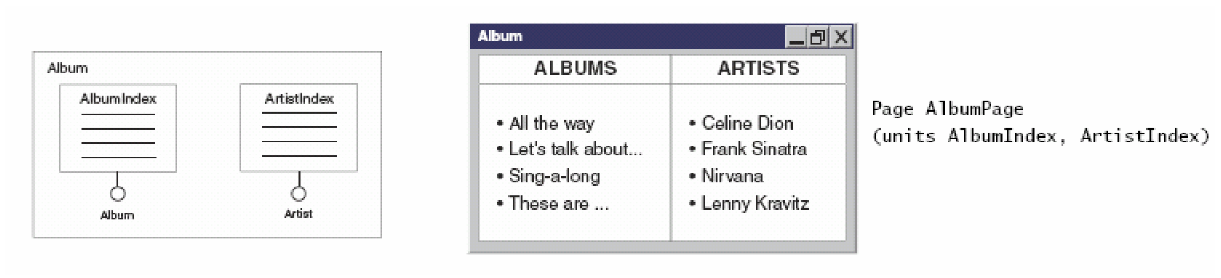
Syntactically, the FILTERUNIT element is used, which specifies the container (entity, relationship, or component) providing the set of objects to search. Inside the FILTERUNIT element, the SEARCHATTRIBUTE element is used to specify a search predicate on the value of a specific attribute. This element tells the attribute on which the search has to be performed and the comparison operator to use. In the following example, the AlbumFilter unit specifies a search form over the set of all albums. The form includes two input fields: the former for inputting a string to be located in the album's title, the latter for inputting the publication time interval of the album [18].

```
<FILTERUNIT id="AlbumFilter" entity="Album"/>
  </SEARCHATTRIBUTE name="title" predicate="like">
  </SEARCHATTRIBUTE name="year" predicate="between">
</FILTERUNIT>
```

Pages

Pages are the actual interface elements delivered to the user, who browses the hypertext by accessing its pages in the desired sequence. A page typically consists of several units, grouped together to accomplish a well-defined communication purpose.

There is a simple example about WebML graphic notation for pages, rendition in HTML, and a textual definition in Figure 5.19.



```
Page AlbumPage
(units AlbumIndex, ArtistIndex)
```

Figure 5.19 WebML graphic notation for pages, rendition in HTML, and a textual definition.

Syntactically, the organization of units into pages is specified using the PAGE element. A possible definition is shown in the following example [18]:

```
<PAGE id="outermost">
<PAGE id="leftmost"><UNIT id="pastIndex"/><UNIT id="thisYearIndex"/></PAGE >
<PAGE id="rightmost"><UNIT id="AlbumInfo"/></PAGE >
</PAGE>
```

5.9.2 Navigation Model

Units and pages do not exist in isolation, but must be connected to form a hypertext structure. The purpose of navigation modeling is to specify the way in which the units and pages are linked to form a hypertext. To this purpose, WebML provides the notion of link.

Links

The central notions of navigation modeling are the concepts of link, link parameters, and parametric selectors:

- A link is an oriented connection between two units or pages.
- A link parameter is the specification of a piece of information, which is transported from the source to the destination of the link.
- A parametric selector is a unit selector whose predicates contain a reference to a link parameter.

Links abstract and generalize the fundamental notion of hypertexts: the concept of anchor. An anchor is an active device, whereby the user can interact with the hypertext.

The following practical cases, referred to an HTML-based hypertext, are all examples of what can be considered an anchor:

- An HTML anchor tag with an href attribute that refers to another page.
- Clicking on the anchor replaces the currently visualized page with the page referred to by the tag anchor.
- An HTML anchor tag with an href attribute that refers to the same page.
- Clicking on the anchor redisplays the currently visualized page, possibly with some new content; for example, due to a selection in some index, which causes the details of a new object to be displayed.
- The confirmation button of an HTML form used for searching. Inserting input in the form and pressing the button causes a new page or the same page to be redisplayed with the search results.

- The confirmation button of an HTML form used for sending input to an operation, for example for logging into a password-protected site.

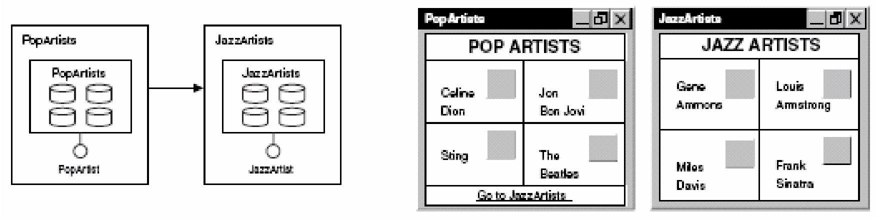
Links crossing the boundaries of pages are called inter-page links, whereas links with the source and destination inside the same page are called intra-page.

There are two variants of links:

- **Contextual links:** they connect units in a way coherent to the semantics expressed by the structure schema of the application. A contextual link carries some information (called context) from the source unit to the destination unit. Context is used to determine the actual object or set of objects to be shown in the destination unit.

- **Non-contextual links:** they connect pages in a totally free way, i.e., independently of the units they contain and of the semantic relations between the structural concepts included in those units.

Graphic notation, rendition in HTML, and textual definition of inter-page non-contextual links are expressed as in Figure 5.20



```
link PopToJazz
(from PopArtists to JazzArtists)
```

Figure 5.20 WebML graphic notation for inter-page non-contextual, rendition in HTML, and a textual definition.

Syntactically, contextual and non-contextual links are denoted by element INFOLINK and HYPERLINK, respectively nested within units and pages. Graphically links are represented by oriented arcs, which connect the source unit or page to the destination unit or page.

Possible definition of contextual link is shown in the following example:

```
<DATAUNIT id="ArtistUnit" entity="Artist">
  <INCLUDEALL/>
  <INFOLINK id="link1" to="AlbumIndex"/>
</DATAUNIT>

<INDEXUNIT id="AlbumIndex" relation="Artist2Album>
  <DESCRIPTION key="title"/>
  <INFOLINK id="link2" to="AlbumUnit"/>
</INDEXUNIT>

<DATAUNIT id="AlbumUnit" entity="Album">
  <INCLUDEALL/>
</DATAUNIT>
```

Possible definition of non-contextual link is shown in the following example:

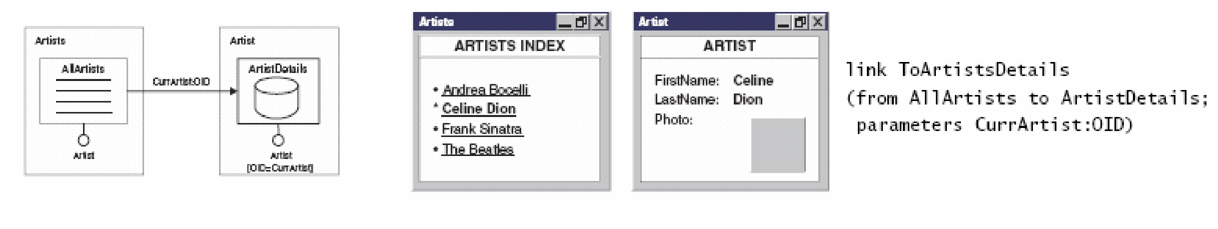
```
<DATAUNIT id="ArtistUnit" entity="Artist">
  <INCLUDEALL/>
</DATAUNIT>
<INDEXUNIT id="AllAlbums" entity="Album>
          <DESCRIPTION  key="title"/>
</INDEXUNIT>

<PAGE id="ArtistPage">
  <UNIT id="ArtistUnit"/>
  <HYPERLINK id="link1" to="AllAlbumsPage"/>
</PAGE>
<PAGE id="AllAlbumsPage">
 <UNIT id="AllAlbums"/>
</PAGE>
```

Link Parameters and Parametric Selectors

The binding between the source unit and the destination unit of the link is formally represented by a link parameter defined over the link, and by a parametric selector, defined in the destination unit.

A link parameter is a value associated with a link between units, which is transported, as an effect of the link navigation, from the source unit to the destination unit. A link may be associated with as many link parameters as required by the destination unit. A parametric selector is a unit selector whose condition mentions one or more parameters.

Graphic notation, rendition in HTML, and textual definition of inter-page contextual links are expressed as in Figure 5.21



```
link ToArtistsDetails
(from AllArtists to ArtistDetails;
 parameters CurrArtist:OID)
```

Figure 5.21 WebML graphic notation for inter-page contextual link, rendition in HTML, and a textual definition with associated link parameter.

A link parameter can be single-valued which stores the OID of the single instance selected from an index unit, or set-valued; for example it may hold the set of OIDs of the objects selected from a multi-choice index unit.

As Table 5.3 clearly indicates, multidata units, index units, hierarchical index units, and multi-choice index units differ in the meaning of the link parameter associated with their outgoing link: for index units and hierarchical index units, the outgoing link permits the selection of an individual object, for multi-choice index units it permits the selection of a subset of objects, for multidata units it permits only the passage of the entire set of visualized objects.

To make hypertext diagrams more readable, the link parameter specification can be omitted, when the parameters associated with the link are deducible from the context. To help this simplification, for each unit a default output parameter is defined, so that, when a link is associated with the default output of its source unit, the link parameter specification can be omitted without loss of information. Table 5.4 summarizes the default output of units.

Table 5.3 Link parameters provided in output by content units.

Source unit	Link parameters (for outgoing links)	Labels of link parameters
Data unit	Any attribute (including the OID) of the object displayed by the unit	SourceEntity.attributeName
Multidata unit	The set of values of any attribute (including the OID) of the objects displayed by the unit	{SourceEntity.attributName}
Index unit	The value of any attribute (including the OID) of the object selected by the user clicking on an anchor of the index unit	SourceEntity.attributeName
Hierarchical index unit	The value of any attribute (including the OID) of the object selected by the user clicking on an anchor in the unit. All the objects displayed at all the levels of the hierarchy can be selected	SourceEntity.attributeName
Multi-choice index unit	The set of values of any attribute (including the OID) of the multiple objects selected using the check boxes of the unit	{SourceEntity.attributName}
Scroller unit	The set of values of any attribute (including the OID) of the block of objects selected by clicking on an anchor in the unit	{SourceEntity.attributName}
Entry unit	The value input by the user in each field	fieldName

Table 5.4 Default link parameters of units.

Source unit	**Default link parameters of outgoing links**
Data unit	The OID of the object displayed by the unit
Multidata unit	The set OIDs of the objects displayed by the unit
Index unit	The OID of the single object selected from the unit
Hierarchical index unit	The OID of the single object selected from the unit. If the unit has multiple nested entities, the OID refers to an instance of the entity at the top of the hierarchy. Parameters associated with the objects of entities nested at inner levels must be specified explicitly
Multi-choice index unit	The set of OIDs of the multiple objects selected from the unit
Scroller unit	The set of OIDs of the selected block of objects, or a single OID if the block factor is 1

A similar simplification can be done also for selector conditions. Table 5.5 shows the default selectors of units with incoming links; when a unit has an incoming link and no selector, the default selector specified in Table 5.5 is implicitly assumed.

Table 5.5 The default selectors of units with incoming links

Destination unit	**Default selector condition when an input link is specified and no explicit selector is mentioned**
Data unit	OID = <link parameter of type OID of the input link>
Multidata, index, multi-choice index, scroller unit	OID IN <link parameter of the type {OID} of the input link>
Hierarchical index unit	The default selector is defined only for the entity at the top of the hierarchy and is: OID IN <link parameter of the type {OID}of the input link>

Figure 5.22 and Figure 5.23 show a notation and an example of default link parameters and unit selectors respectively.

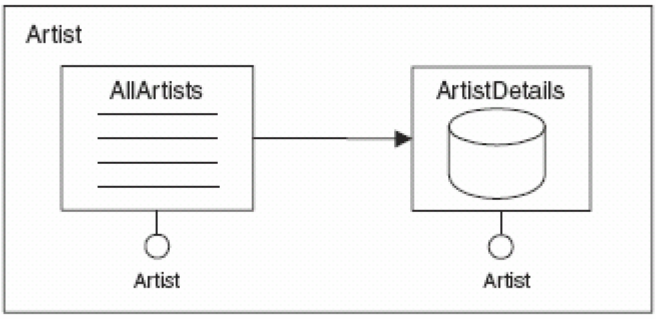


Figure 5.22 Simplified notation exploiting default link parameters and unit selectors.

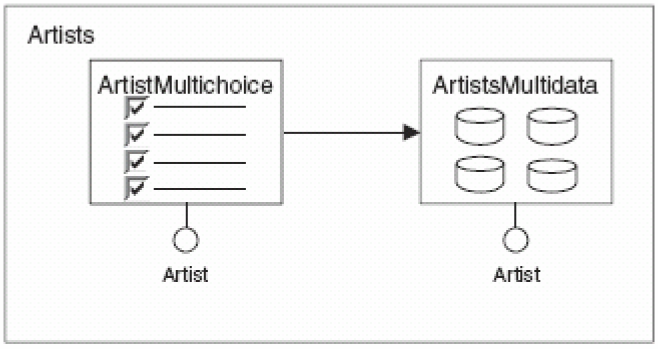


Figure 5.23 Example of use of a default link parameters and selectors.

Figure 5.24 shows a contextual link with LINK parameter associated with an input field.

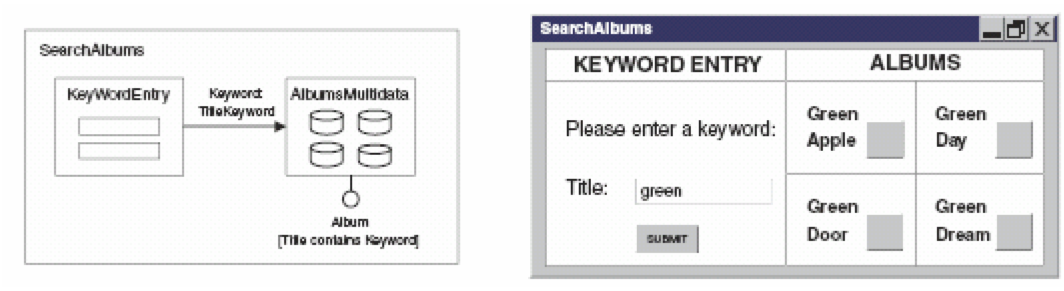


Figure 5.24 Contextual link with LINK parameter associated with an input field.

Units may have multiple outgoing links, possibly associated with different parameters as shown in Figure 5.25.

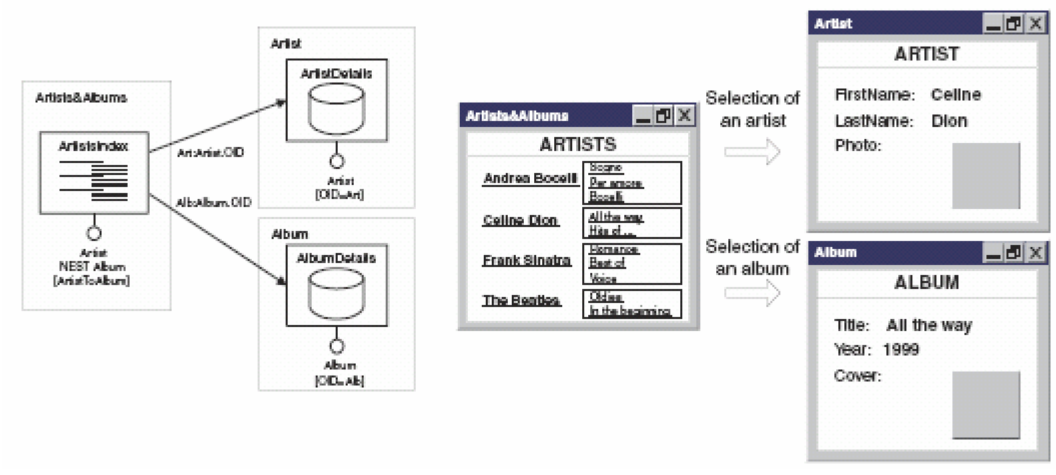


Figure 5.25 Hierarchical index with two outgoing links with different link parameters.

Automatic and Transport Links

Besides supporting user navigation, links can be used also to specify a particular kind of information flow between units, which takes place in absence of user intervention.

In some applications, it may be necessary to specify a different behavior, whereby the content of some unit is displayed as soon as the page is accessed, even if the user has not navigated its incoming link. This effect can be achieved by using automatic links. An automatic link is a link that is “navigated” in absence of user’s interaction, when the page that contains the source unit of the link is accessed. Figure 5.26 shows a simple example.

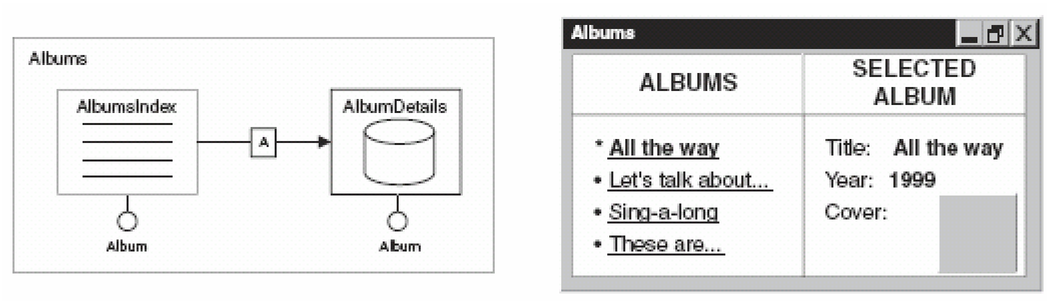


Figure 5.26 Example of automatic link and its HTML rendition at page access.

All the links seen so far are rendered by means of anchors or confirmation buttons. However, there are cases in which a link is used only for passing context information from one unit to another one, and thus is not rendered as an anchor. This type of link is called transport link, to highlight the fact that the link enables only parameter passing, and not user navigation. Figure 5.27 shows a simple example about transport links.

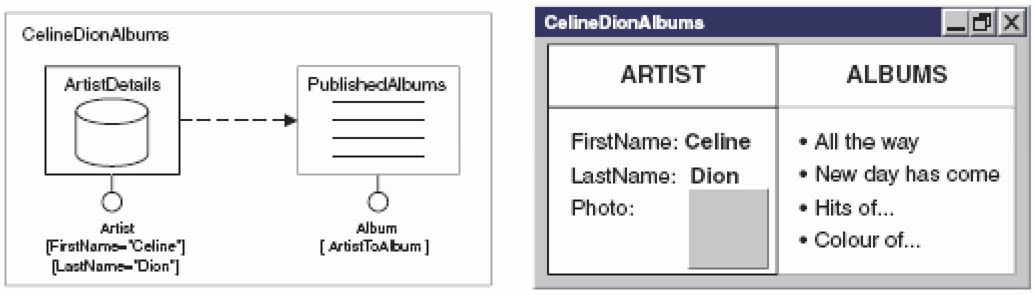


Figure 5.27 Example of transport link.

Global Parameters

In the previous examples, the context information needed to calculate units is associated to links, which go from one point of the hypertext to another one. However, there are situations in which context information is not transferred point-to-point during navigation, but must be available “globally” to all the pages of a site.

WebML offers the notion of global parameter for storing information available to multiple pages. A global parameter is a piece of information, either the OID of an object or a typed value, which can be explicitly set at some point during hypertext navigation, and then retrieved to compute the content of some unit, later during the navigation. The value of the global parameter is associated with the user's session, so that distinct users may have different values for the same global parameter; for instance, two users may browse the same multi-lingual application, but receive content in different languages, due to the different value of the global parameter representing the selected country.
Using a global parameter requires three steps: declaring it, setting its value, and then getting the value.

The declaration of a global parameter requires the definition of the following:

- A user-defined name for the parameter.
- The type of the value stored in the parameter.
- A possible default value, which is a constant value initially assigned to the parameter.

A possible example of the textual definition of the global parameters is as follows:

globalParameter CountryOID
(type OID;
entity Country)

The value of a parameter is assigned by means of an ad hoc unit, called set unit. A set unit has only one input link, which is associated with a link parameter holding the value to be assigned to the global parameter as shown in Figure 5.28.

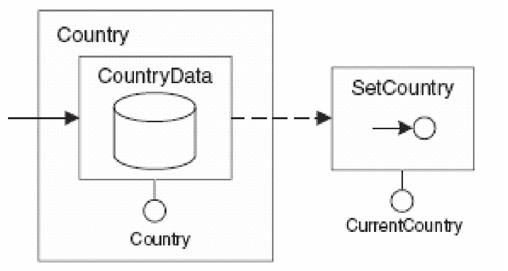


Figure 5.28 WebML graphic notation for set units.

The textual definition of the set unit and of its input link is:

link CountryDataToSetCountry transport
(from CountryData to SetCountry)

setUnit SetCountry
(parameter CurrentCountry)

A global parameter is retrieved by means of the get unit, which can be considered the dual operation with respect to the set unit.

As explained in Figure 5.29, a get unit has no incoming links, and has only one outgoing link, transporting the value of the retrieved parameter; the unit is placed inside the page where the global parameter value is used, to show the fact that the parameter is retrieved to help the computation of some unit local to the page.

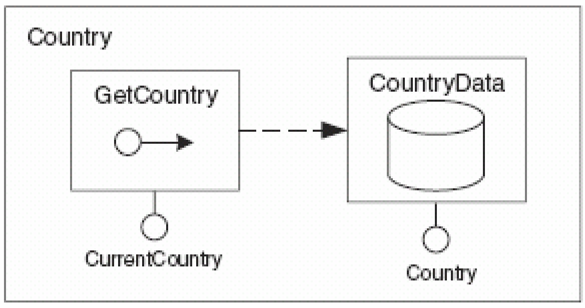


Figure 5.29 WebML graphic notation for get units.

The textual definition of the get unit and of its output link is as follows:

getUnit GetCountry
(parameter CurrentCountry)

link GetCountryToCountryData transport
(from GetCountry to CountryData)

Hypertext Organization

Site Views

A WebML hypertext is packaged into an application to be delivered to users by enclosing its linked pages and units into a modularization construct called site view.

Site views are characterized by a user-defined name and contain a set of pages and/or areas. A WebML site view comprises a set of pages and links, with associated presentation styles, all reachable from a designated home page. All the site views of a given Web application share the same structural schema, which represents at high level the data sources underlying the site. The structural schema, in turn, is mapped to one or more data sources, possibly embodied within legacy systems.

In Figure 5.30, we see an example of a site view.

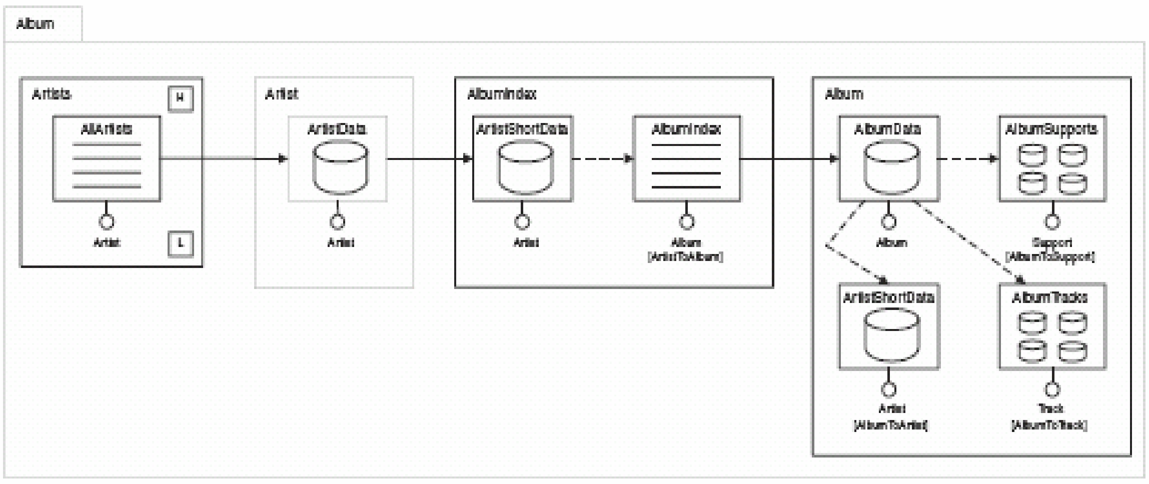


Figure 5.30 Site view in WebML.

The textual representation of the site view contains the site view name and the list of its pages and/or areas:

siteview Album
(pages Artists, AlbumIndex, Album, Artist)

Areas, Landmarks and Homepages

WebML provides primitives for improving the organization of site views and pages: areas, landmarks, and homes.

Areas are containers of pages or, recursively, other sub-areas, which can be used to give a hierarchical organization to a site view. Most real-life Web sites are partitioned into areas. Areas can be nested, so that sub-areas can be defined inside areas.

Each area should have a DEFAULT PAGE or a DEFAULT SUB-AREA, to give a meaning to landmark areas and non-contextual links pointing to areas.

Examples of areas:

- Sections of a portal: Sport, Music, Technology …
- Elements of a data-management system: Products management, News management …

Pages and areas are characterized by some distinguishing properties, which highlight their "importance" in the Web site. In particular, pages inside an area or site view may have the following three properties (Figure 5.31):

- The home page is the page at the default address of the site or presented after the user logs in to the application. The home page must be unique across the site view. In the graphic specification, an "H" inside the page icon denotes the home property of a page; in the textual declaration, the keyword home is added to the page specification.
- The default page is the one presented by default when its enclosing area is accessed. The default page inside an area must be unique. In the graphic specification, a "D" inside the page icon denotes the default property of a page; in the textual declaration, the default keyword is added.
- A landmark page is reachable from all the other pages or areas within its enclosing module (the site view or a super-area). In the graphic specification, an "L" inside the page icon denotes the landmark property; in the textual declaration, the landmark keyword is added.

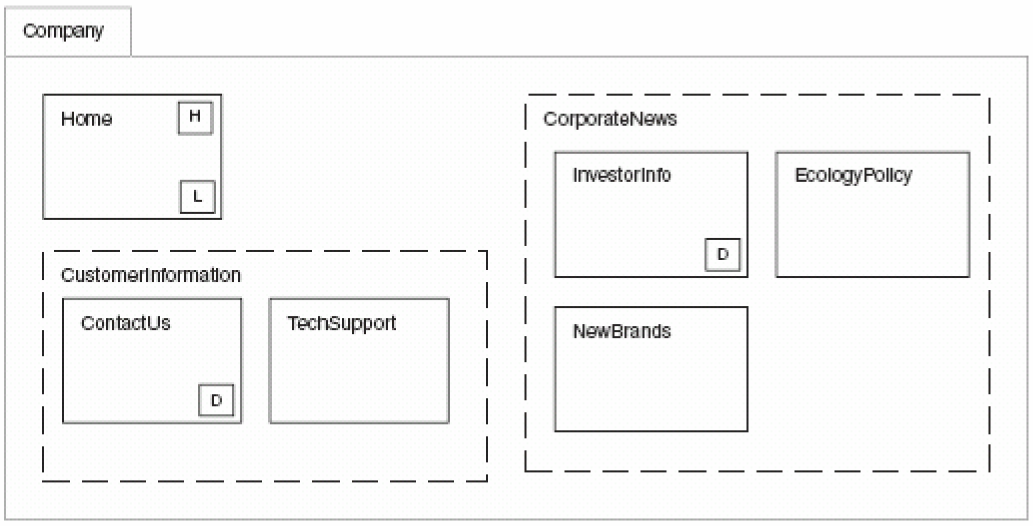


Figure 5.31 Two-level decomposition of site views into areas.

Areas can be associated with the landmark and default properties (Figure 5.32, 5.33):

- The default area is the sub-area accessed by default when its enclosing super-area is accessed. If the user navigates a link pointing to the superarea, he/she is redirected to the default page of the sub-area. The default page of the sub-area is defined recursively: it is either the default page defined locally inside the sub-area, or a default page recursively nested inside an arbitrary number of default sub-sub-areas.
- A landmark area is an area implicitly reachable from all other pages or areas of the enclosing site view or super-area.

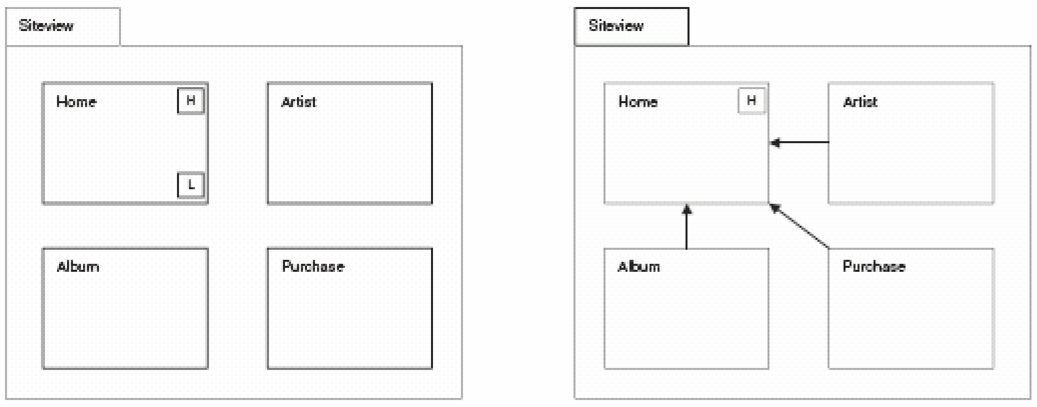


Figure 5.32 Home and landmark pages (left) and equivalent diagram with explicit links (right).

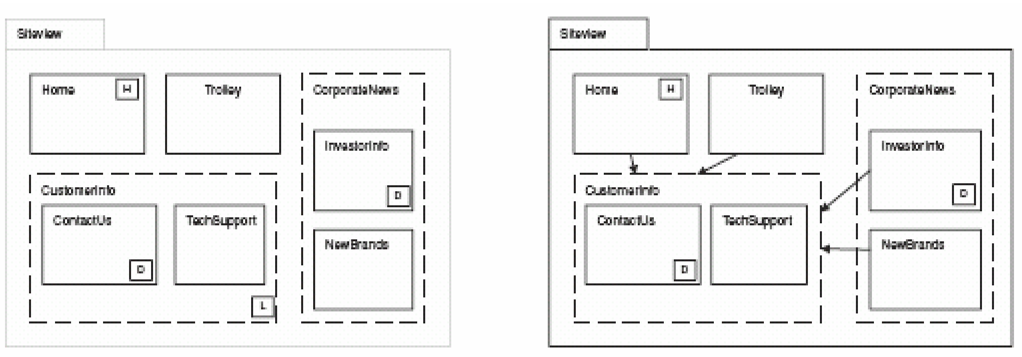


Figure 5.33 Landmark area (left) and equivalent diagram with explicit links (right).

5.9.3 Default Hypertext

To enable fast prototyping, WebML includes several shortcuts to obtain a running application from incomplete specifications. In particular, given the structural model, WebML supports the notion of default hypertext, automatically generated according to the following rules:

- For each entity, a data unit is generated which includes all attributes.
- For each one-to-many or many-to-many relationship R between an entity A and an entity B, an index unit P is provided over the relationship R, based on the primary key of the entity B; two contextual links are established from the data unit of A to P and from P to the data unit of B.
- For each one-to-one or many-to-one relationship R between an entity A and an entity B, a direct unit P is provided over relationship R; two contextual links are established from the data unit of A to P and from P to the data unit of B.
- For each component C of an entity A, a multi-data unit P is provided over component C, and a contextual link is established from the data unit of A to P.
- For each entity, an index unit is created on the entity's primary key, which includes the list of all instances of the entity.

The default hypertext maps every concept of the structural model into exactly one unit and provides default indexes over all the defined entities. Given the default hypertext, a default site view is defined by associating units to pages as follows:

- Data unit over entities and index pages over relationships are put in distinct pages.
- Component multi-data units are kept in the same page as the data unit of the entity or component that encloses them.
- Index units over entities are put in distinct pages. An empty page is created as the home page, and connected by means of non-contextual links to the all these pages.

5.9.4 Validity of WebML Hypertext

A valid logical hypertext is defined by the following constructive rules:

- A logical hypertext constituted by a navigation chain over an entity followed by a data unit on such entity is valid.
- The logical hypertext obtained by adding a linked sequence of container units over a relationship or component of the structural schema, from a data unit of a valid hypertext to another data unit (an existing one or a new one) is valid.

Given a valid logical hypertext, a valid physical hypertext is obtained by grouping units within pages and adding non-contextual links between pages according to the following rules:

- **Reachability:** there must not be pages (with the exception of the home page) without any incoming link (contextual or non-contextual);
- **Context flow:** if a page contains a unit that needs context information, then such context information must be supplied by a contextual link. There are two sub-cases:
  - The unit may receive context from another unit in the same page that does not require context information (e.g., an index unit over an entity).
  - If the above case does not hold, the unit must receive its context from all the entry units in the same page, where an entry unit is any unit which is the destination of contextual links coming from outside the page.

- **Uniqueness of context:** if a unit in the page has more than one path from which it receives context information from another unit inside the same page, then only for one such paths the initial filling option should be enabled [18].

5.10 Presentation Model

Purpose of Presentation Model is to express the layout and graphic appearance of pages, independently of the output device and of the rendition language, by means of an abstract XML syntax.

Presentation modeling is concerned with the actual look and feel of the pages identified by composition modeling. WebML pages are rendered according to a style sheet. A style sheet dictates the layout of pages and the content elements to be inserted into such layout, and is independent of the actual language used for page rendition.

For better reusability, two categories of style sheets are provided:

- Untyped style sheets (also called models) describe the page layout independently of its content, and thus can be applied regardless of the mapping of the page to a given concept.
- Typed style sheets are specified at a finer granularity and thus apply only to pages describing specific concepts.

Presentation Model is addressed in WebML by attaching XSL style sheets to site views, pages, units and unit subelements.

An implementation of WebML may include :

- Several pre-defined presentation style sheets.
- The server-side components supporting the data access queries needed to populate the content of the page templates produced by the XSL style sheets.

5.11 Personalization Model

Personalization is the definition of content or presentation style based on user profile data. In WebML, units, pages, their presentation styles, and site views can be defined so to take user- or group-specific data into account.

In reserved Web sites commitment wants

- One or more public pages readable from anyone.
- A set of private page accessed only after login, which contains personal content and personal services.
- Personalization in terms of delivered pages (which pages user can access) and delivered content (which content user needs/can see).

Purpose of Personalization Model is explicitly modeling users and user groups for

- Storing group-specific or individual content, like shopping suggestions, list of favorites, and resources for graphic customization.
- Business rules typically produce new user- related information (e.g., shopping histories) or update the site content (e.g., inserting new offers matching users' preferences).

5.11.1 Declarative Personalization

The designer defines derived concepts (e.g., entities, attributes, multi-valued components) whose definition depends on user-specific data. In this way, customization is specified declaratively; the system fills in the information relative to each user when computing the content of units.

As an example of declarative personalization, the computation of an album's discounted price could be based on personalized discounts, associated with users or user groups.

5.11.2 Procedural Personalization

WebML includes an XML syntax for writing business rules that compute and store user-specific information. A business rule is a triple event-condition-action, which specifies the event to be monitored, the precondition to be checked when the event occurs, and the action to be taken when the condition is found true. Typical tasks performed by business rules are the assignment of users to user groups based on dynamically collected information (e.g., the purchase history, or the access device), the notification of messages to users upon the update of the information base (push technology), the logging of user actions into user-specific data structures (user tracking), and so on.

As an example of procedural personalization, a business rule could assign a customer to the "best buyer" group, based on his/her purchase history.

5.12 Operations

Web applications often perform operations on data. WebML operation units (operations, for short) are a new kind of units, which can be placed outside of pages and linked to other operations, or to content units defined inside pages. Unlike content units, operations do not display content, which justifies the fact that they are placed outside pages; rather, they perform an action. Like content units, operations may have a source object (either an entity or a relationship) and selectors, may receive parameters from their input links, and may provide values to be used as parameters of their output links. Operations can be either predefined or generic.

Regardless of their type, WebML operations obey the following design principles:

- An operation may have multiple incoming links, providing values for its input parameters.
- Operations can be linked to form a sequence. Firing the first operation of the sequence implies executing also the remaining operations. One of the incoming links must be a regular link, and all the remaining links should be transport links. The operation is executed by navigating the regular link and uses the parameters associated with all the input links.
- Each operation has one OK link and one KO link; the former is followed when the operation succeeds; the latter when the operation fails. The selection of the link to follow (OK or KO) is based on the outcome of operation execution and is under the responsibility of the operation implementation.
- An operation may have any number of outgoing transport links, which are used to specify link parameters needed by content units or other operation units. The

specification of transport links does not alter the execution sequence of operations, which is based only on the OK and KO links, but permits the designer to use the various outputs of operations as input in other units.

Operations do not display content, but execute some processing as a side effect of the navigation of a link. However, the result of executing an operation can be displayed in a page, by linking an operation to an appropriate content unit, which accepts input parameters from the operation and uses such parameters to retrieve and display the relevant information.

5.12.1 Predefined Operations

WebML provides a number of built-in operations, whose meaning is predefined in the language.

Object Creation

Create unit performs the creation of a new entity instance. Each create unit is characterized by the following:

- A user-defined name.
- The source entity to which the operation applies.
- A set of assignments, binding the attributes of the object to be created to the parameters values coming from the input link, or to some constant values.

As Figure 5.34 shows, the input of a create unit is a set of attribute values, typically coming from one input link exiting from an entry unit. These values are used by the create operation to construct the new object; if some attributes have no associated input value, they are set to null, with the exception of the OID, which is treated differently: if no value is supplied, a new value, unique with respect to the entity instances, is generated by the operation. The output produced by the create operation is the set of attribute values, including the OID, of the newly created object. This output is defined only when the operation succeeds, and thus can be meaningfully associated as a link parameter only to the OK link, and not to the KO link. The default output of the create unit is the value of the OID attribute, which is assumed as the implicit link parameter of the OK link, if no parameter is specified explicitly.

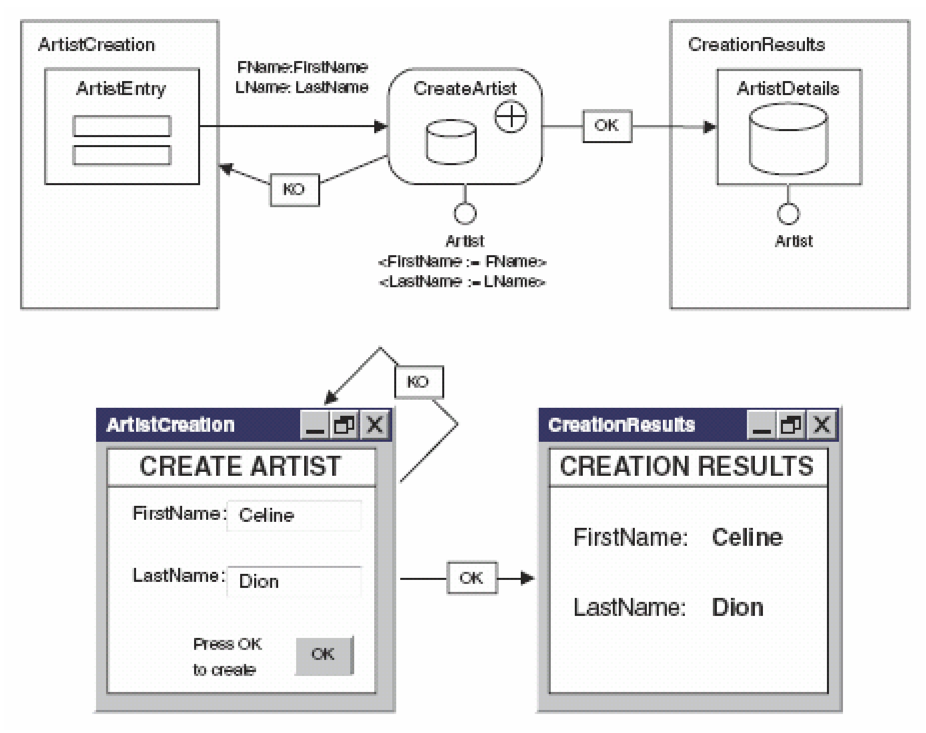


Figure 5.34 WebML graphic notation for create units, and a possible rendition in HTML.

The complete textual description of the example of Figure 5.34 as follows:

link toCreateUnit
(from ArtistEntry to CreateArtist;
parameters FName:FirstName, LName:LastName)

CreateUnit CreateArtist
(source Artist;
FirstName:=FName, LastName:=LName)

OKLink createOKlink
(from CreateArtist to ArtistDetails;)

KOLink createKOlink
(from CreateArtist to ArtistCreation;)

DataUnit Artist details
(Source Artist;
attribute FirstName, LastName)

Object Deletion

The delete unit is used to delete one or more objects of a given entity. Each delete unit is characterized by the following:

- A user-defined name.
- The source entity and the selector, which determine the object or set of objects to which the operation applies. The objects to delete are those that satisfy the selector condition.

The delete unit has a default selector, based upon the cardinality of the set of OIDs received in input: it is [OID=<link parameter>] if the input link parameter is single-valued, or [OID IN <link parameter>] if the input link parameter is multi-valued. As usual, the default selector can be inferred and need not be expressed in the graphical and textual notation (Figure 5.35).

The OK link is followed when all the objects determined by the selector have been deleted, and has no link parameters. The KO link is followed when at least one of the objects has not been deleted, and is associated with a link parameter holding the OID or set of OIDs of the objects that could not be deleted.

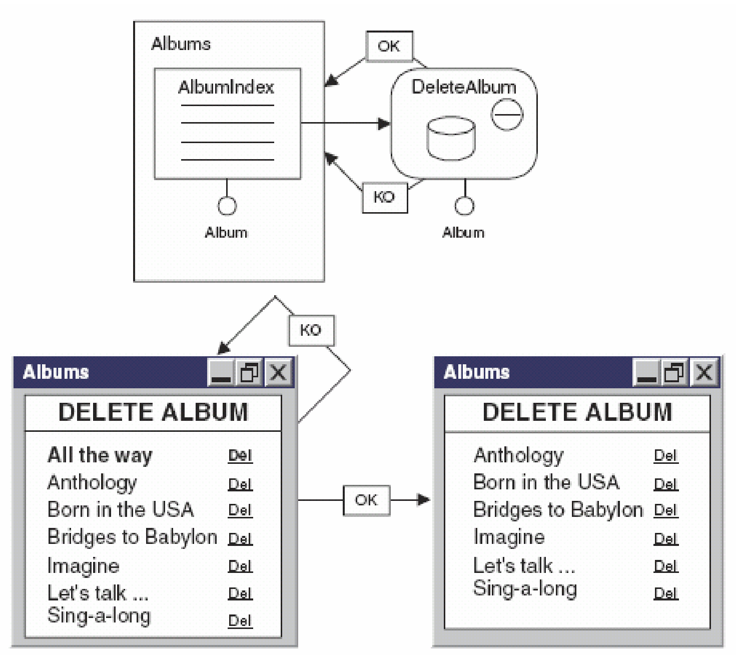


Figure 5.35 WebML graphic notation for delete unit, and rendition in HTML.

The textual specification of the delete unit of Figure 5.35 is simply:

DeleteUnit DeleteAlbum
(source Album;)

Object Modification

The modify unit is used to update one or more objects of a given entity. Each modify unit is characterized by the following:

- A user-defined name.
- The source entity and the selector, which identify the object or set of objects to which the operation applies; the objects to modify are the set of objects that satisfy the selector.
- A set of assignments, binding the new values to the attributes of the objects to be modified.

A modify unit must be properly linked to other units, to obtain the needed inputs (Figure 5.36):

- The new attribute values: these are typically defined as parameters of an input link coming from an entry unit.
- The objects to modify: these are usually specified as a parameter of an input link, holding one OID or a set of OIDs, which is used in the selector of the modify unit. The modify unit has a default selector of the form [OID=<link parameter>], if the OID parameter is single-valued, and [OID IN <link parameter>], if the link parameter is multi-valued.
- Alternatively to the usage of link parameters of type OID, the objects to modify can be identified by means of attribute-based or relationshipbased selectors inside the modify unit, possibly exploiting parameters associated with input links.

The OK link of a modify unit is followed when all the objects have been successfully modified: in this case the OK link has a default parameter holding the set of modified objects.

The KO link is followed when at least one of the objects could not be modified, and has as a default parameter the set of OIDs of the objects that were not modified.

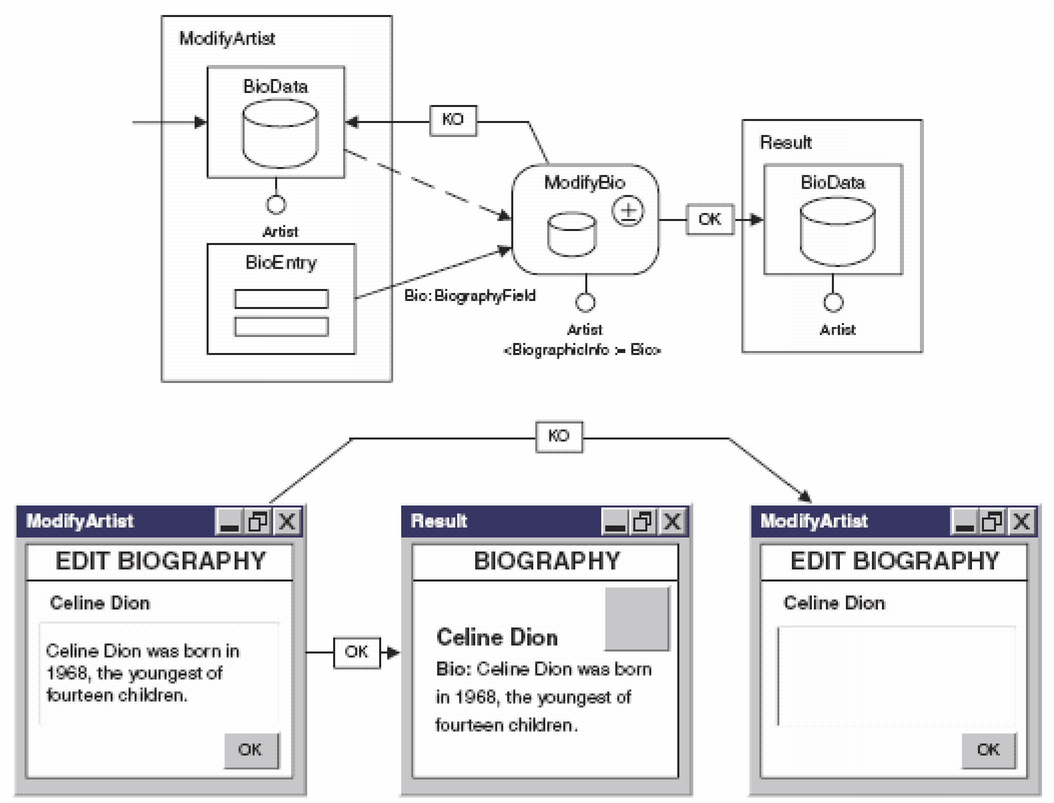


Figure 5.36 Modify unit, and rendition in HTML.

The textual specification of the modify unit, with the default selector omitted, is the following:

ModifyUnit ModifyBio
(source Artist;
BiographicInfo:=Bio)

Relationship Creation

A connect unit is used to create new instances of a relationship. More precisely, a connect unit applies to one of the two possible roles of a relationship, and creates one or more instances of the relationship role connecting some objects of the source entity to some objects of the destination entity. The properties of the connect unit are the following:

- A user-defined name.
- The source relationship role, that is, the role to which the operation applies.
- Two selectors, one for locating the objects of the source entity and one for the objects of the destination entity. To distinguish the conditions applied to the source and destination entity, the attributes and relationship roles used in the selector predicates can be prefixed with the name of the entity to which they refer.

The connect operation creates one instance of the source relationship role for each pair of objects of the source and destination entities retrieved by evaluating the two selectors; it provides in output two values, respectively holding the OIDs of the objects of the source and of the destination entity retrieved by the selector. These values can be used to define parameters in the OK, KO, and transport links of the operation. The KO link is followed if the creation of at least one relationship instance fails, whereas the OK link is followed if all the connections can be created. If the connect operation attempts at connecting two objects for which a relationship instance already exists in the database, then the operation execution does not introduce a duplicate relationship instance, but is still considered successful (Figure 5.37).

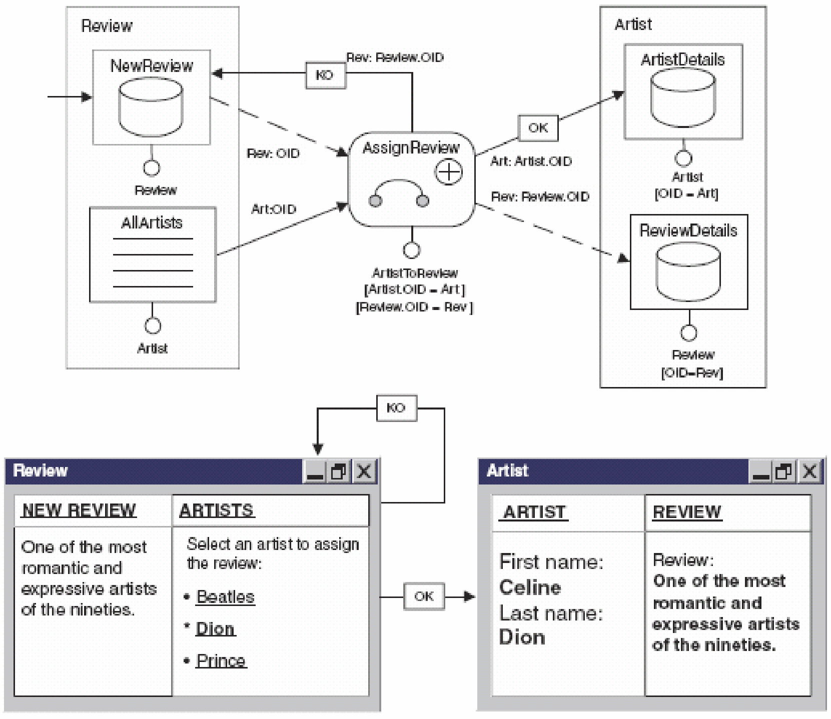


Figure 5.37 Connect unit, and rendition in HTML.

The textual description of the connect unit and of its input links is the following:

```
link toConnectDestination transport
(from NewReview to AssignReview;
parameters Rev:OID)
```

Relationship Deletion

A disconnect unit is used to delete instances of a relationship. More precisely, a disconnect unit is applied to one of the two possible roles of a relationship, and deletes the connection between some objects of the source entity and some objects of the destination entity. The properties of the connect unit are the following:

- A user-defined name.
- The source relationship role, that is, the role to which the operation applies.
- Two selectors, one for locating the objects of the source entity and one for the object of the destination entity.

The operation deletes one instance of the source relationship role for each pair of objects of the source and destination entities identified by the two selectors. As for the connect unit, the disconnect operation provides in output two values respectively holding the OIDs of the objects of the source and of the destination entity retrieved by the selector. These values can be used to define parameters in the OK, KO, and transport links. The KO link is followed if the deletion of at least one relationship instance fails, whereas the OK link is followed if all the connections can be deleted.

If two objects to be disconnected are not linked by a relationship instance, then no disconnection occurs but the operation is still considered successful. Also for disconnect units, when parameters and selectors are implied from the context, they can be omitted from the diagram (Figure 5.38).

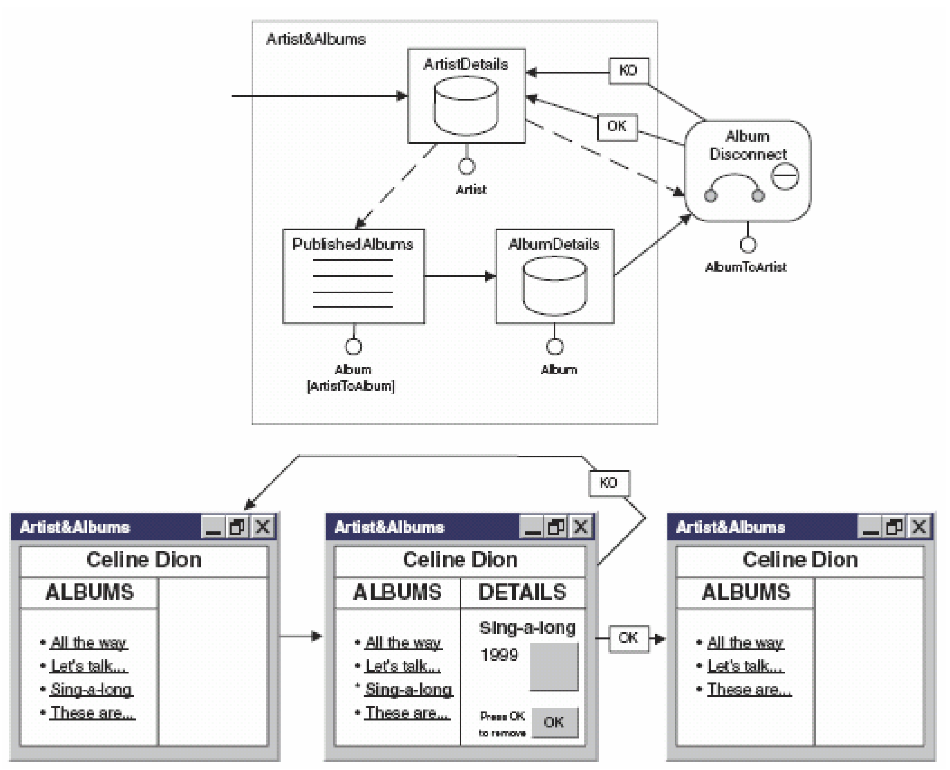


Figure 5.38 Disconnect unit, and rendition in HTML.

The textual description of the disconnect unit of Figure 5.38 and of its input links, with default parameters and selectors omitted, is the following:

link ArtisDetailsToDisconnect transport
(from ArtistDetails to AlbumDisconnect;)

link AlbumDetailsToDisconnect
(from AlbumDetails to AlbumDisconnect;)

DisconnectUnit AlbumDisconnect
(source AlbumToArtist;)

Transactions

A transaction is an atomically executed sequence of operations; that is, either all operations execute successfully, or the entire sequence is rolled back. Transactions are a fundamental concept of database systems, which grant the correct synchronization of the work of multiple concurrent users operating on the same content; they are natively supported in most database management products, and also offered by latest-generation middleware systems. In particular, a transaction guarantees the “acid” properties (so-called by taking the initials of each property as an acronym):

- **Atomicity:** either all the involved operations are successfully completed, producing a new database state, or the initial database state is left unchanged.
- **Consistency:** carrying out a transaction should not violate the integrity of data.
- **Isolation:** each transaction execution is independent of the simultaneous execution of other transactions.
- **Durability:** effects of transactions that complete successfully are recorded persistently.

Graphically, a transaction is represented as a named dashed box surrounding the involved operations; when transaction boxes are omitted, each single operation is considered as a transaction. The operations of a transaction are chained by a sequence of OK links, connecting the first operation to the second, the second to the third one, and so on. The last OK link leads to the (unique) hypertext page shown after the successful completion of the transaction. Each operation can independently fail, therefore it can have a different KO link; however, the overall transaction can have a unique KO link, represented as an arrow exiting from the transaction box, which means that the destination page shown after the failure of any of the operations of the transaction is the same. KO links could be associated both to the entire transaction and to some of its operations, in which case the locally defined KO link associated with the individual operation prevails over the KO link specified for the whole transaction. Transactions are exemplified in the next section, dedicated to content management patterns.

5.13 Operations for Access Control

Login Operation

To implement access control and to verify the identity of a user accessing the site, WebML provides a predefined operation called login. The operation has two fixed parameters (username and password), whose values must be passed in input by a link, typically exiting from an entry unit.

The login operation checks the validity of the identity of the user, and if the verification succeeds, forwards him/her to a default page. If the credentials are invalid, the login operation forwards the user to the error page pointed by the KO link. Figure 5.39 shows an example of login unit.

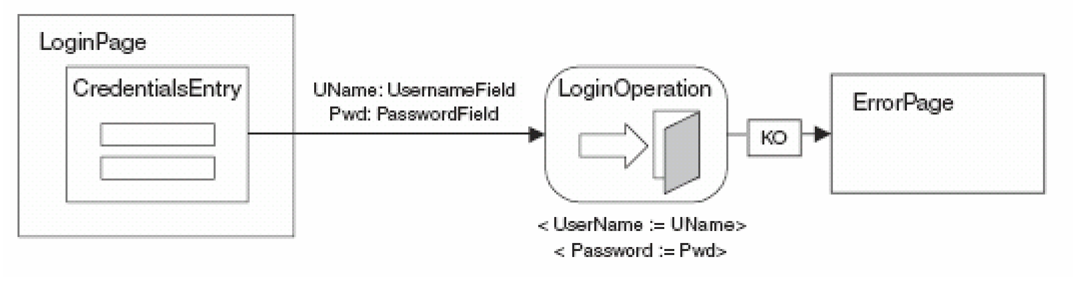


Figure 5.39 WebML login unit, preceded by an entry unit for credential input.

The textual description of the login operation of Figure 5.39 is the following:

login LoginOperation
(parameters UserName :=UName, Password :=Pwd)

Logout Operation

The logout operation is used to "forget" the session of a logged user, and forward him/her to a default page with no access control. The logout operation has no input and output, and can be invoked by a simple non-contextual link.

The textual description of a logout operation is as follows:

logout LogoutOperation

5.14 Design of Web Applications

Developing a Web application, as with any other kind of software system, is a complex achievement that requires the ability to master a broad spectrum of tasks, jointly performed by a number of persons with different skills.

5.14.1 Development Roles

Web application development involves different actors with complementary skills and goals as follows (Figure 5.40):

- The application analyst collects the business requirements and turns them into a specification of the application requirements. In doing this, he/she interprets the long-term strategic business goals and constraints and transforms them into short-term, concrete, application requirements.
- The data architect focuses on the part of the application requirements that deals with the data, and produces the conceptual data model.
- The application architect focuses on the part of the application requirements dealing with the services to be delivered, and designs the application hypertexts by specifying site views built on top of the data schema produced by the data architect.
- The graphic designer conceives the presentation styles of pages, based on the business requirements that deal with the visual identity and communication standards of the organization.
- The developer and site administrator is responsible for site implementation, architecture design and tuning, and site view deployment and evolution. In particular, he/she focuses on the nonfunctional requirements of performance, availability, security, scalability, and manageability, and is responsible for ensuring the appropriate level of service.

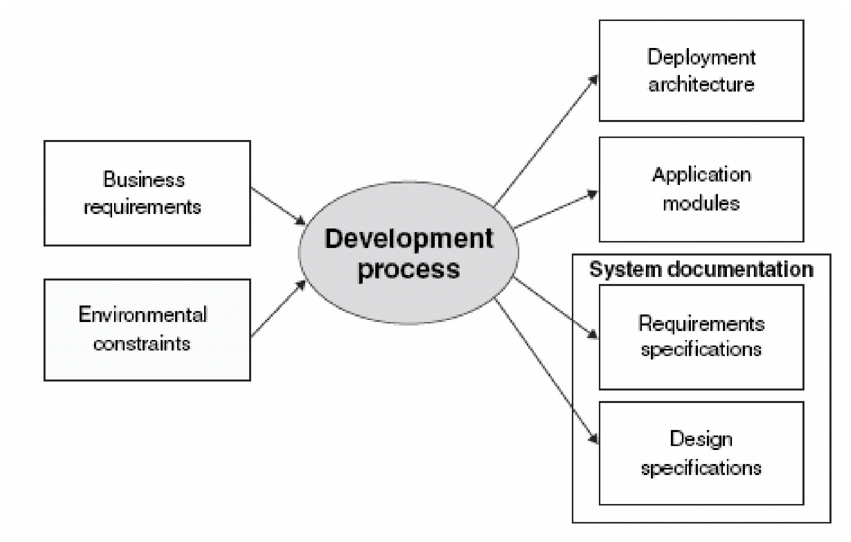


Figure 5.40 Inputs and outputs of the Web application development process.

5.14.2 Development Lifecycle

Application development undergoes several cycles of problem discovery/design refinement/implementation, and each iteration produces a prototype or partial version of the system. At each iteration, the current version of the system is tested and evaluated, and then extended or modified. Such an iterative and incremental approach is not exclusive of Web application development, but appears particularly appropriate for this class of systems, because Web applications must be deployed quickly and their requirements are often subject to changes.

Figure 5.41 shows phases in the development process of data-intensive Web applications.

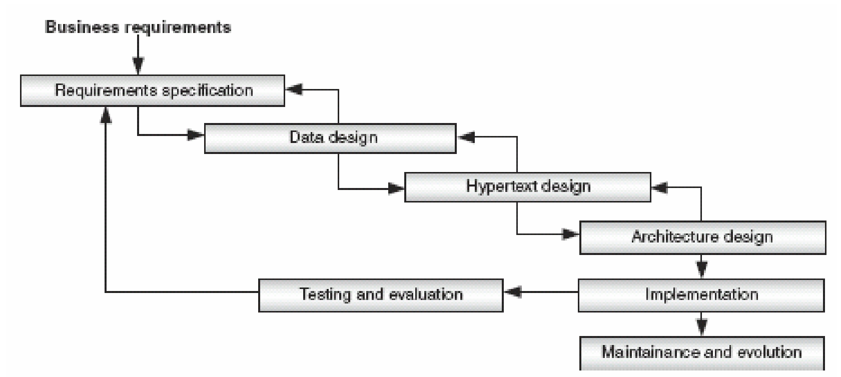


Figure 5.41 Phases in the development process of data-intensive Web applications.

Requirements Specification

Requirements specification is the activity in which the application analyst collects and formalizes the essential information about the application domain and expected functions.

Data Design

Data design is the phase in which the data expert organizes the main information objects identified during requirements specification into a comprehensive and coherent conceptual data model.

Hypertext Design

Hypertext design is the activity that transforms the functional requirements identified during requirements specification into one or more site views embodying the needed information delivery and data manipulation services.

Architecture Design

Architecture design is the definition of the hardware, network, and software components that make up the architecture on which the application delivers its services to users. The goal of architecture design is to find the mix of these components that best meets the application requirements in terms of performance, security, availability, and scalability, and at the same time respects the technical and economic constraints of the application project.

Implementation

Implementation is the activity of producing the software modules necessary to transform the data and hypertext design into an application running on the selected architecture.

Testing and Evaluation

Testing and evaluation is the activity of verifying the conformance of the implemented application to the functional and nonfunctional requirements.

6 WEBRATIO - A TOOL FOR MODEL-BASED DEVELOPMENT OF WEB APPLICATIONS

WebRatio Site Development Studio (WebRatio, for short), which supports the WebML design process covers the phases of data design and hypertext design, and supports implementation by automating the production of the relational database and of the application page templates.

WebRatio focuses on five main aspects:

- **Data design:** supports the design of Entity-Relationship data schemas, with a graphical user interface for drawing and specifying the properties of entities, relationships, attributes, and generalization hierarchies.
- **Hypertext design:** assists the design of site views, providing functions for drawing and specifying the properties of areas, pages, units, and links.
- **Data Mapping:** permits declaring the set of data sources to which the conceptual data schema has to be mapped, and automatically translates. Entity-Relationship diagrams and OCL expressions into relational databases and views.
- **Presentation design:** offers functionality for defining the presentation style of the application, allowing the designer to create XSL style sheets and associate them with pages, and organize page layout, by arranging the relative position of content units in the page.
- **Code generation:** automatically translates site views into running Web applications built on top of the Java2EE, Struts, and .NET platforms.

Unlike traditional prototyping tools, which generate application mock-ups, the WebRatio code generator produces application modules running on state-of-the-art architectures, and can be used for implementation, maintenance, and evolution.

WebRatio internally uses XML and XSL as the formats for encoding both the specifications and the code generators: XML is used for describing data and hypertext schemas, whereas XSL is used for generating the graphic properties and layout of the page templates, for validity checking, and for automatic project documentation.

Figure 6.1 shows a design flow diagram of WebRatio.

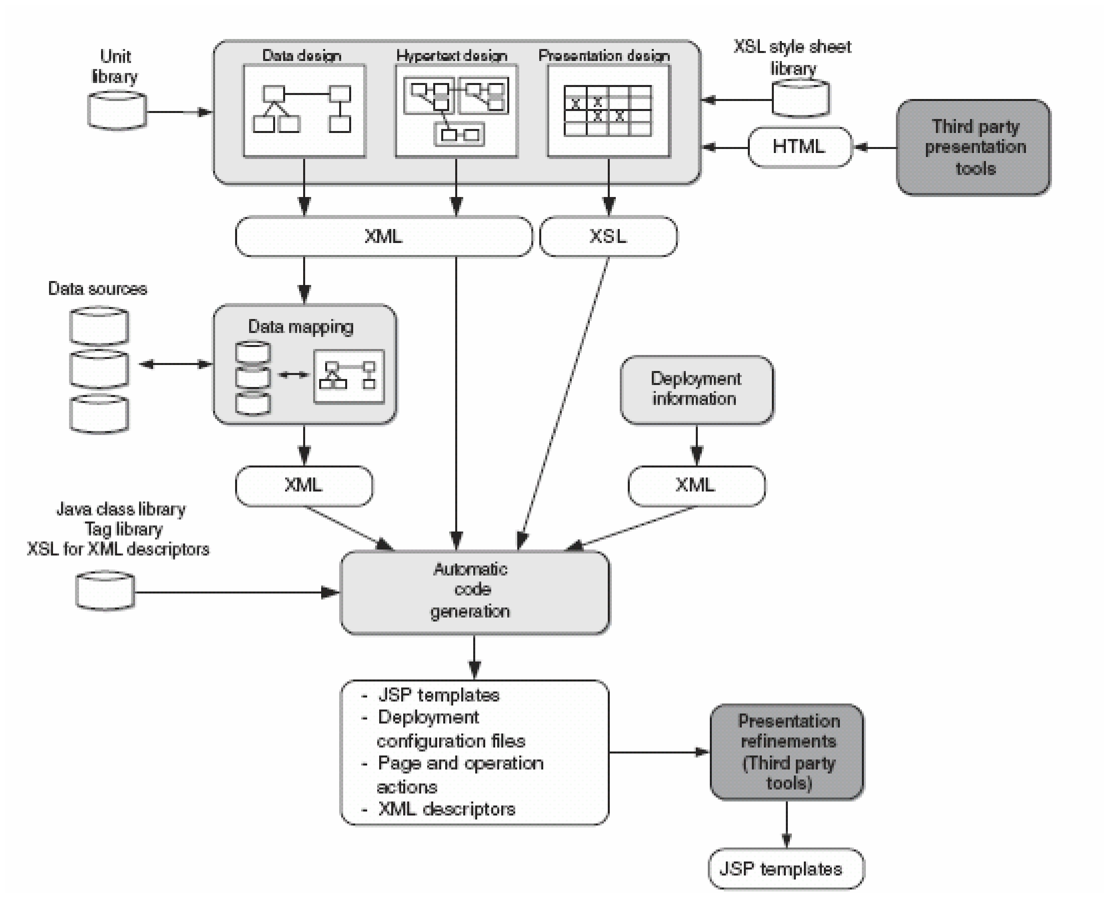


Figure 6.1 Design flow diagram of WebRatio.

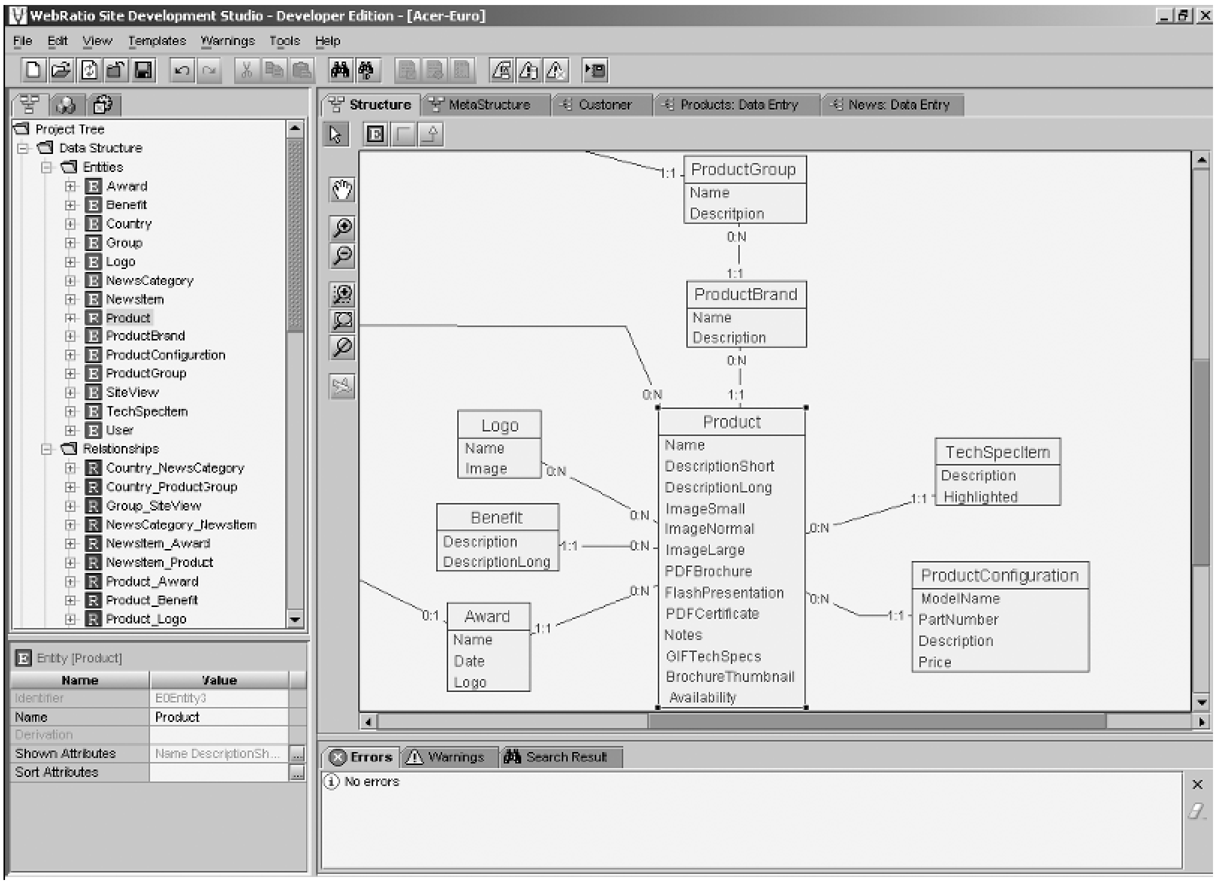
6.1 Data and Hypertext Design

WebRatio provides a graphical user interface, which allows designers to compose both the Entity-Relationship diagram and the site views of the application.

Figure 6.2 shows a snapshot of the WebRatio user interface, which is organized into the typical four areas of application development tools:

- A project tree (upper left frame), organizing all the elements of the application project.
- A work area (upper right frame), where the specifications are visually edited.
- A property frame (lower left frame), where the properties of individual elements can be set.
- A message area (lower right frame), where messages and warnings are displayed.

In particular, Figure 6.2 shows a portion of the Entity-Relationship diagram of the running example. The work area visualizes the data schema, and the designer can define entities, attributes, relationships, and generalizations. The elements displayed in the diagram are also presented in the project tree, where they are hierarchically organized in folders. The properties of the currently selected element of the schema are displayed and can be edited in the property frame. The same organization of the graphical user interface supports also the editing of the site view diagrams.



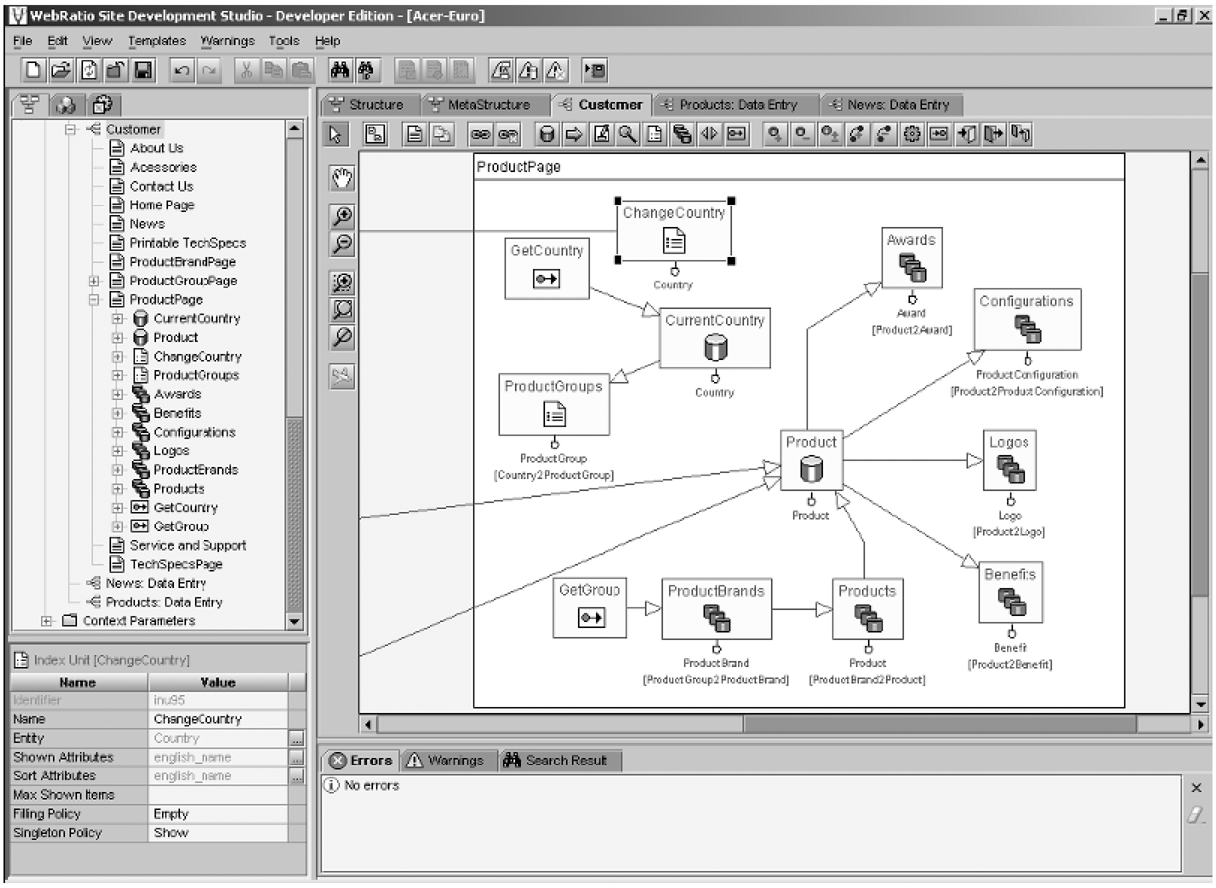
Entity [Product]

Name	Value
Identifier	E0Entity3
Name	Product
Derivation	
Shown Attributes	Name DescriptionSh...
Sort Attributes	

No errors

Figure 6.2 Data design in WebRatio.

A WebRatio application project consists of a single Entity-Relationship diagram and of a set of site views. A default structure schema consisting of the User and Group entities and their standard relationships is automatically added to each project, and the developer can extend it with additional entities and relationships. The design of a site view is accomplished by visually manipulating hypertext elements such as units, pages, areas, links, selectors, and context parameters. Figure 6.3 shows the site view work area, with the focus on the Product page, which includes multiple units; these are also displayed in the project tree, and the properties of the currently selected unit (unit ChangeCountry in Figure 6.3) appear in the property frame.



Index Unit [ChangeCountry]

Name	Value
Identifier	inu95
Name	ChangeCountry
Entity	Country
Shown Attributes	english_name
Sort Attributes	english_name
Max Shown Items	
Filling Policy	Empty
Singleton Policy	Show

No errors

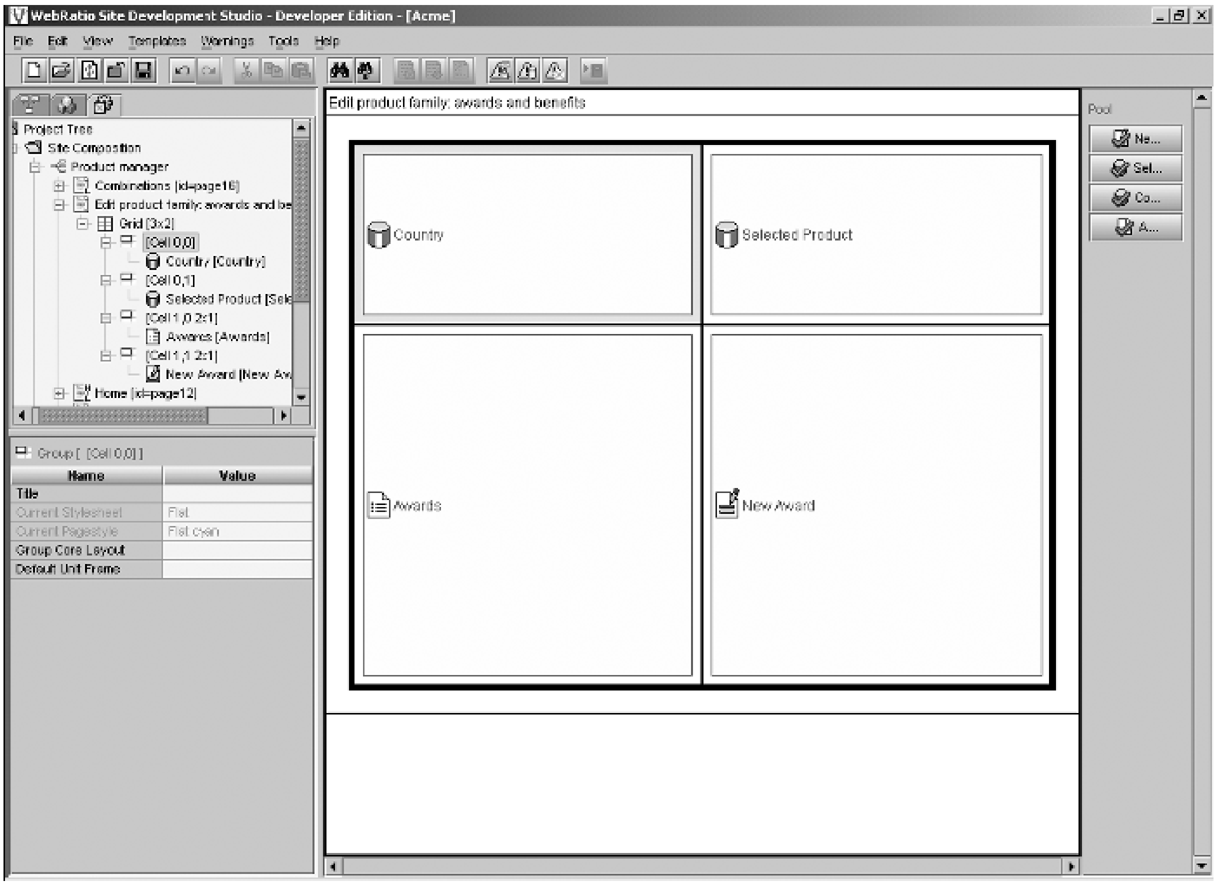
Figure 6.3 Hypertext design in WebRatio.

WebRatio supports also the visual definition of derived data. A wizard can be invoked to specify the expression for computing a derived entity, attribute, or relationship. Such expression, written in a subset of the OCL language, is automatically translated into a SQL view, and included into the application database.

6.2 Presentation Design

Presentation design addresses the definition of XSL style sheets, embodying the presentation rules needed by the code generator to produce the page templates. WebRatio provides functionalities both for selecting from a library already available presentation styles and associating them to application pages, and for automatically transforming HTML mockups designed by graphic artists into XSL style sheets.

An XSL style sheet encompasses a set of XSL rules that govern the way in which the page layout and the various kinds of units are rendered. To make XSL style sheets reusable across multiple pages with different content, the XSL rules do not reference the units of the individual pages, but include the specification of the positions in the page layout where units can be placed. Once a style sheet is selected for a page, WebRatio assists the coupling of page units to the locations exposed by the style sheet, with the drag-and-drop interface shown in Figure 6.4.



Group [[Cell 0,0]]

Name	Value
Title	
Current Stylesheet	Flat
Current Pagestyle	Flat cyan
Group Core Layout	
Default Unit Frame	

Figure 6.4 Positioning page units in the locations exposed by the XSL style sheet.

For each unit positioned in the page, and even for each attribute, contextual link, and field contained in a unit, a different XSL style sheet can be selected, which defines the specific presentation style to be used for rendering the element.

The XSL style sheets of pages and units may be handwritten by the XSL programmer, or automatically generated from HTML mockups.

6.3 Code Generation

After specifying the Entity-Relationship schema and site view diagrams, mapping the data model to the data sources, and assigning style sheets to pages, it is possible to launch automatic code generation, transforming the site views into modules for the selected deployment platform, which may be JSP, Struts, and Microsoft .NET. Before generating the application code, the target platform and the deployment host must be set.

The code generator implements the Model-View-Controller software architecture. For instance, by choosing HTML as mark-up language and Struts as a deployment platform, the output of code generation includes:

- A set of JSP page templates for the View, including HTML code and JSP custom tags. Two tag libraries can be used: the standard tag library of JSP (JSTL) or a WebML-specific library (WebML Taglib).
- A set of page and operation actions, to be deployed in the Model.
- The configuration file of the Controller.
- A set of XML descriptors, which specify the properties of pages, units, and links for the generic page, unit, and operation services. Normally, the developer is not required to edit descriptors; however, if this need arises, for instance to optimize a SQL query, the updated descriptor can be stored in a special directory and will not be overwritten by subsequent invocations of the code generator.

The produced templates may use any mark-up language. Therefore, the code generator can be used to effectively deploy multi-device applications, in which the same content is served to multiple delivery channels, for instance to HTML browsers.

Figure 6.5 shows an extended design flow of WebRatio with additional features explaining code generation.

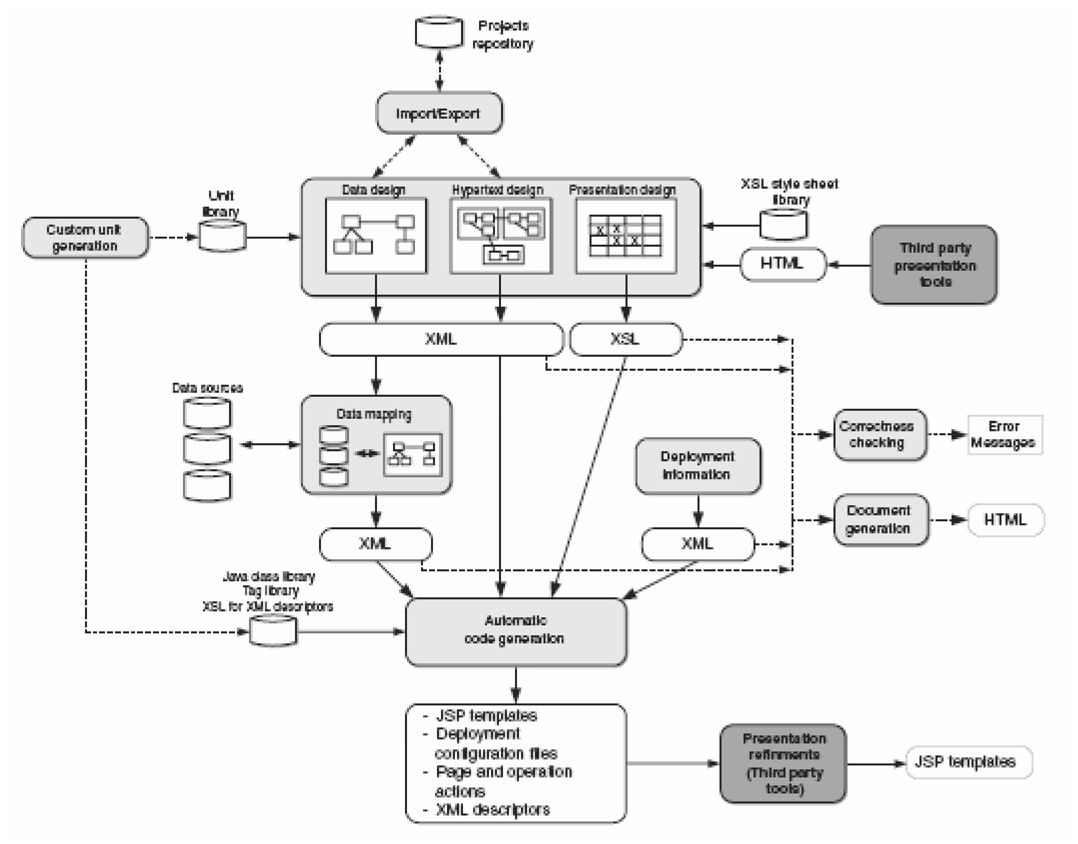


Figure 6.5 Design flow of WebRatio extended with additional features explaining code generation.

7 LEARNING WEBML USING MOODLE

In order to use Moodle, an Apache server, PHP and MySQL are needed to be installed. Thus, EasyPHP package was used to install the technologies mentioned.

EasyPHP is a complete software package allowing to use the dynamic language PHP and the efficient use of databases under Windows. Package includes an Apache server, a MySQL database, a fully PHP execution, as well as easy development tools for web site or applications.

After Moodle was installed, necessary site and administration settings were configurated.

A course "Introduction to WebML" which is aimed to teach was created and a teacher was assigned to the course. Users were created and enrolled as students of the course.

The course was determined as weekly outlines, each week containing one or two main topics of WebML. Necessary technological background to learn the lesson, glossary of terms, books, assignments and examinations were added to strengthen learning and the content of the course. Each lesson contains a table of contents, objectives list, definitions and examples.

7.1 Homepage

When a user logs in a course, the site is opened in a default home page. Home page of a course consists of some default modules which help the user to access some areas of the course quickly. These modules can be modified by the site admin at any time.

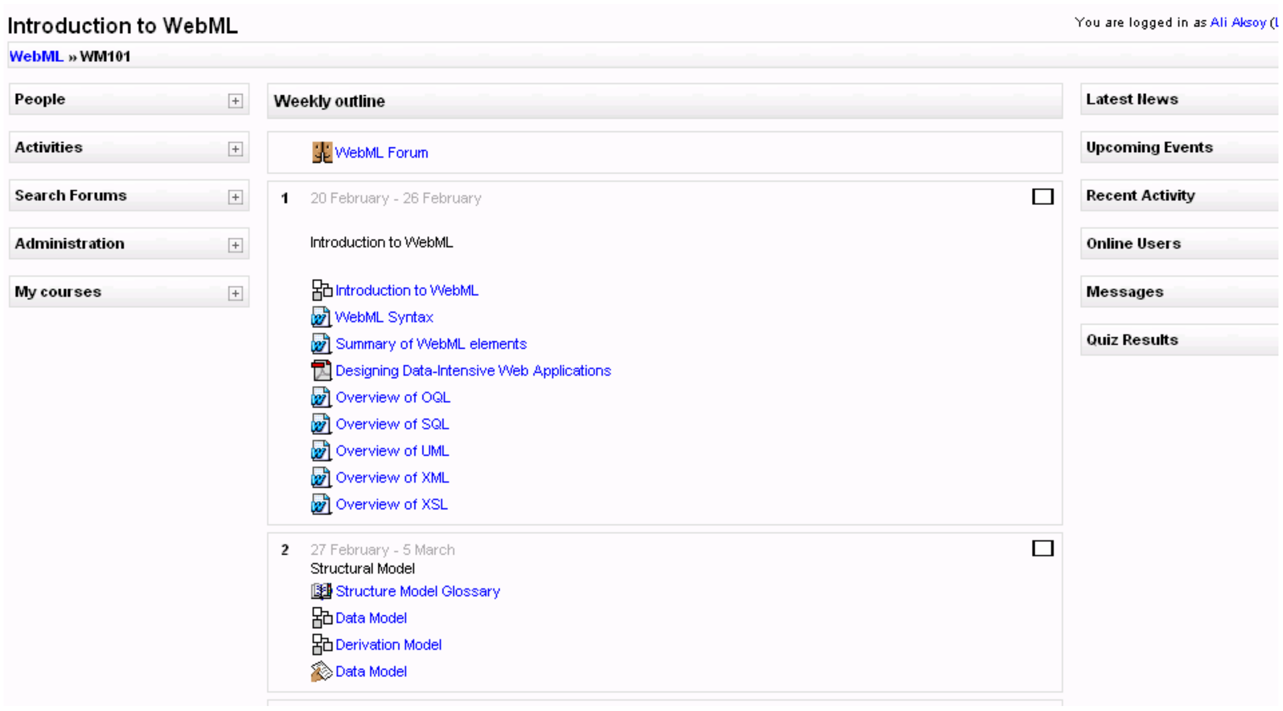
In the left frame of home page, there are five modules (Figure 7.1):

- **People:** Shows the participants of the course as students and teacher.
- **Activities:** There are Assignments, Forums, Glossaries, Lessons, Quizzes, Resources links which students and teacher may follow as a shortcut.
- **Search Forums:** Students and teacher may search forums by entering specific words or make an advanced search.

- **Administration**
  - **Students administration module:** Consists of Grades, Activity report, Edit profile, Change password links.
  - **Teacher administration module:** Consists of Settings, Edit profile, Teachers, Students, Groups, Backup, Restore, Import course data, Scales, Grades, Logs, Files, Help, Teacher forum.
- **Courses:** Shows the courses which students enrolled or teachers assigned to.

In the right frame of home page, there are six modules (Figure 7.1):

- **Latest News:** shows discussion topics or news about the course.
- **Upcoming Events:** shows activities which are on the way according to their start and end dates.
- **Recent Activity:** shows activities which are current to their start and end dates.
- **Online Users:** shows online users.
- **Messages:** shows messages posted by other participants of the course.
- **Quiz Results:** shows examination results after the examination date.



Introduction to WebML

You are logged in as Ali Aksoy (

WebML » WM101

People

Activities

Search Forums

Administration

My courses

Weekly outline

WebML Forum

1 20 February - 26 February

Introduction to WebML

Introduction to WebML
WebML Syntax
Summary of WebML elements
Designing Data-Intensive Web Applications
Overview of OQL
Overview of SQL
Overview of UML
Overview of XML
Overview of XSL

2 27 February - 5 March
Structural Model
Structure Model Glossary
Data Model
Derivation Model
Data Model

Latest News

Upcoming Events

Recent Activity

Online Users

Messages

Quiz Results

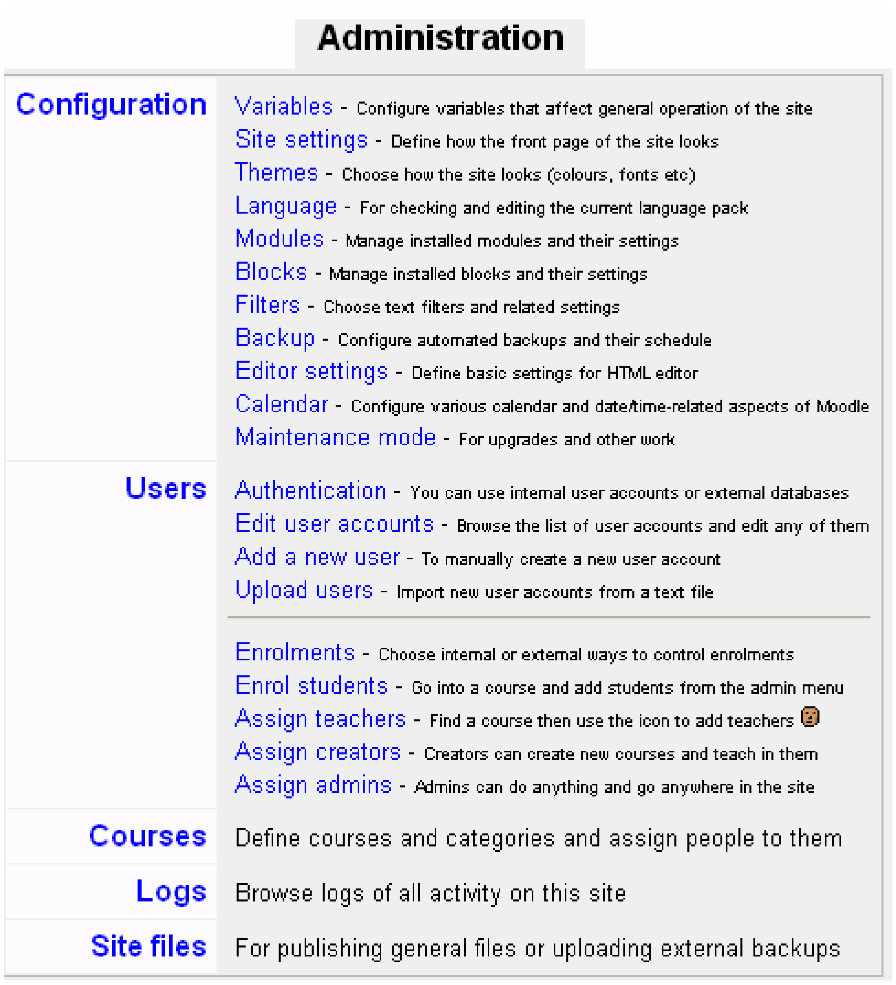
Figure 7.1 General appearance of home page

In the middle frame, outlines of the course are listed containing weekly lessons. Students can enter the activites of each week by using the links.

7.2 Administration Settings

By using full authority of Site Admin of Moodle, necessary settings were configurated from the administration page shown in Figure 7.2.

Administration page consists of five modules including Configuration, Users, Courses, Logs and Site Files. Each module has sub-pages for easy access and use. Each setting can be changed at anytime needed.



Administration

Configuration	Variables - Configure variables that affect general operation of the site
	Site settings - Define how the front page of the site looks
	Themes - Choose how the site looks (colours, fonts etc)
	Language - For checking and editing the current language pack
	Modules - Manage installed modules and their settings
	Blocks - Manage installed blocks and their settings
	Filters - Choose text filters and related settings
	Backup - Configure automated backups and their schedule
	Editor settings - Define basic settings for HTML editor
	Calendar - Configure various calendar and date/time-related aspects of Moodle
	Maintenance mode - For upgrades and other work
Users	Authentication - You can use internal user accounts or external databases
	Edit user accounts - Browse the list of user accounts and edit any of them
	Add a new user - To manually create a new user account
	Upload users - Import new user accounts from a text file
	Enrolments - Choose internal or external ways to control enrolments
	Enrol students - Go into a course and add students from the admin menu
	Assign teachers - Find a course then use the icon to add teachers
	Assign creators - Creators can create new courses and teach in them
	Assign admins - Admins can do anything and go anywhere in the site
Courses	Define courses and categories and assign people to them
Logs	Browse logs of all activity on this site
Site files	For publishing general files or uploading external backups

Figure 7.2 Site administration page.

7.2.1 Configuration

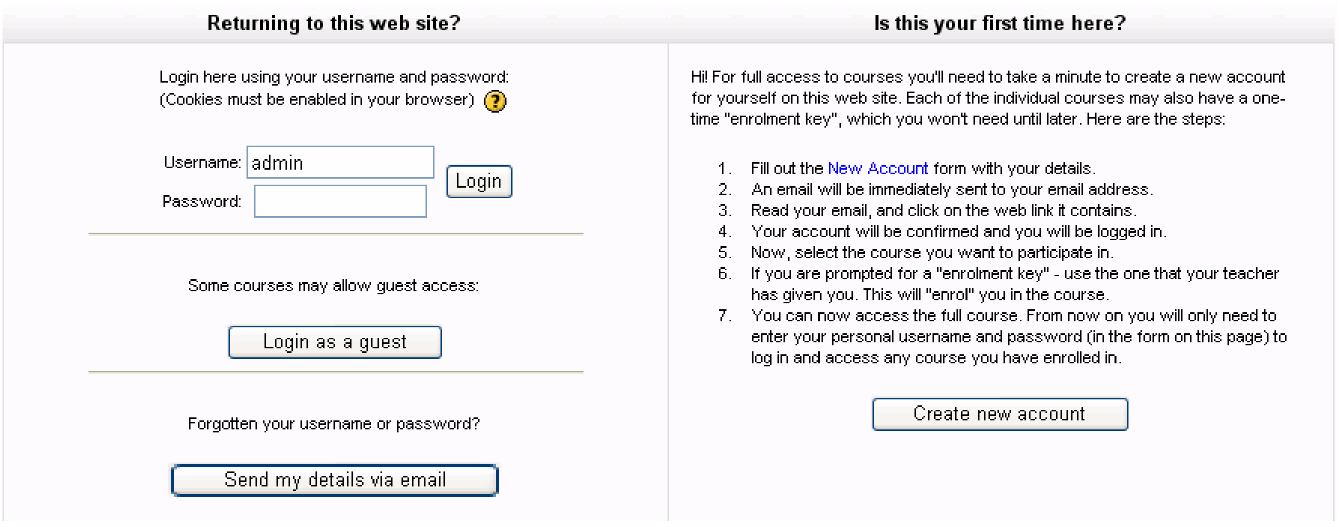
From the configuration section, appropriate settings for Interface, Security, Operating System, Maintenance, Mail, general User, Permissions, and Miscellaneous were formed. Themes, Language, Modules, and Backup settings were updated as needed.

7.2.2 Users

From the "Assign teachers" section, the user "Nergis Kilinc" was assigned as the teacher of the course "Introduction to WebML".

From the "Add a new user" section, new users were added. These users would be the students of the course in the future, when enrolled to the course from the "Enrol students" section.

From the "login page" in Figure 7.3 of the site, users could create new accounts by themselves. Then, admin of the site could confirm the registration.



Returning to this web site?

Login here using your username and password:
(Cookies must be enabled in your browser)

Username: admin
Password:

Some courses may allow guest access:

Forgotten your username or password?

Is this your first time here?

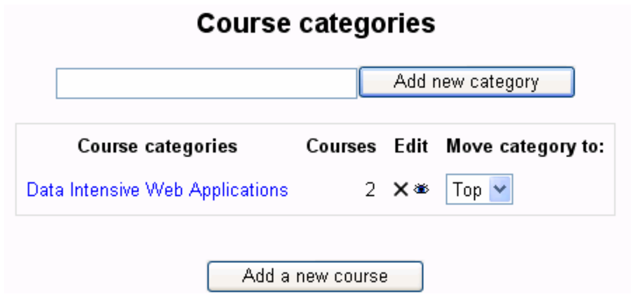
Hi! For full access to courses you'll need to take a minute to create a new account for yourself on this web site. Each of the individual courses may also have a one-time "enrolment key", which you won't need until later. Here are the steps:

1. Fill out the New Account form with your details.
2. An email will be immediately sent to your email address.
3. Read your email, and click on the web link it contains.
4. Your account will be confirmed and you will be logged in.
5. Now, select the course you want to participate in.
6. If you are prompted for a "enrolment key" - use the one that your teacher has given you. This will "enrol" you in the course.
7. You can now access the full course. From now on you will only need to enter your personal username and password (in the form on this page) to log in and access any course you have enrolled in.

Figure 7.3 Login page.

7.2.3 Courses

In order to create a new course, first a category is needed to be chosen to add the course to. In the "Course categories" section shown in Figure 7.4, a new category was created as "Data Intensive Web Applications".

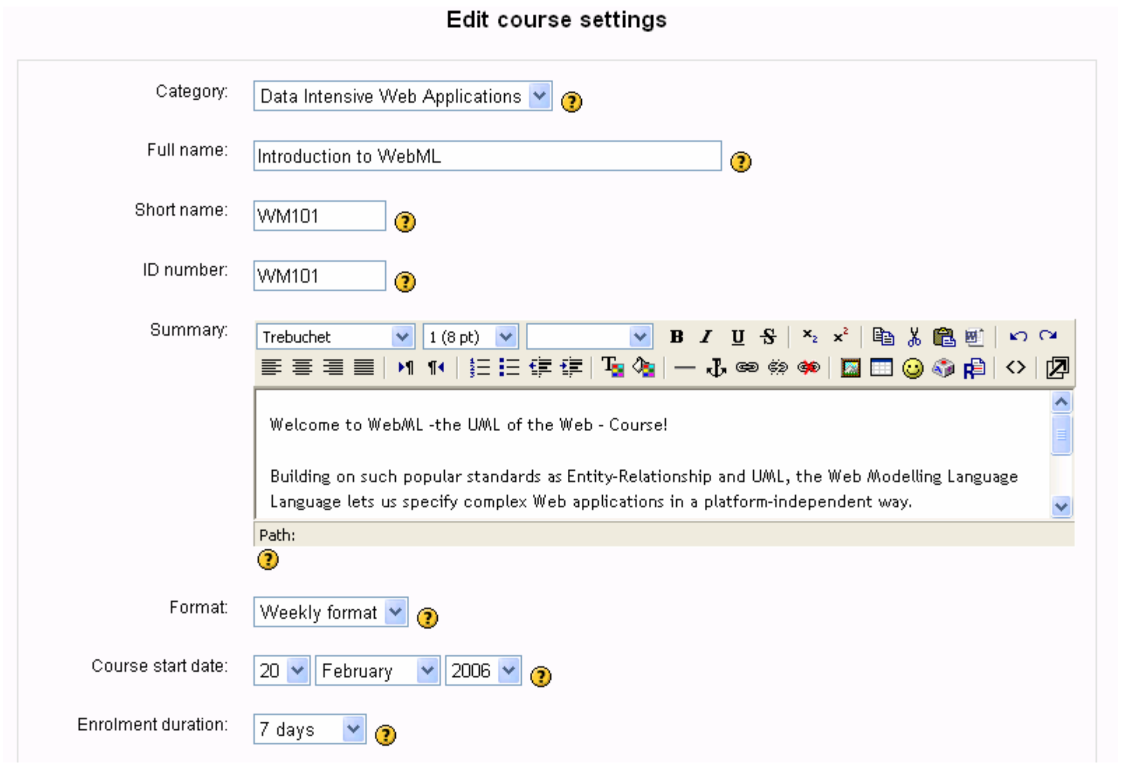


Course categories

Course categories	Courses	Edit	Move category to:
Data Intensive Web Applications	2		Top

Figure 7.4 Course category page.

The course "Introduction to WebML" - coded as WM101 - was created under the "Data Intensive Web Applications" category from the "Edit course settings" page as shown in Figure 7.5.



Edit course settings

Category:	Data Intensive Web Applications
Full name:	Introduction to WebML
Short name:	WM101
ID number:	WM101
Summary:	Welcome to WebML -the UML of the Web - Course! Building on such popular standards as Entity-Relationship and UML, the Web Modelling Language Language lets us specify complex Web applications in a platform-independent way.
Format:	Weekly format
Course start date:	20 February 2006
Enrolment duration:	7 days

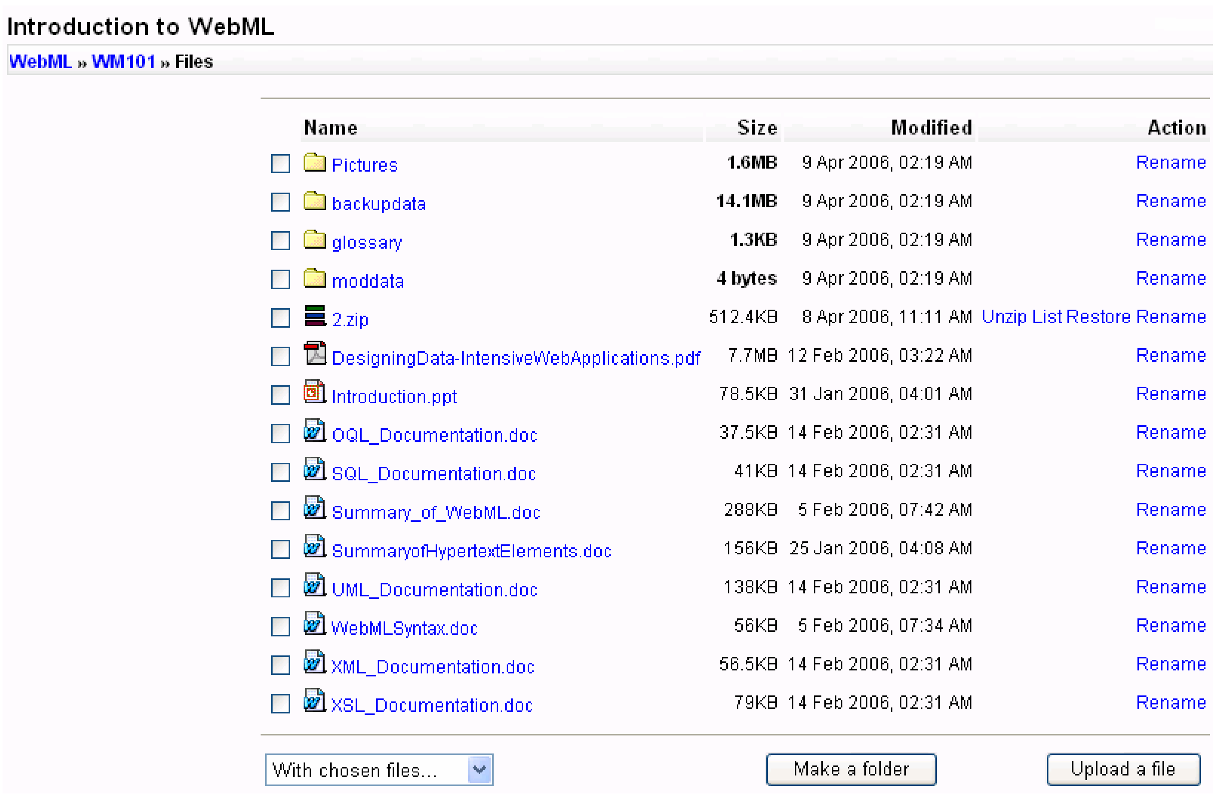
Figure 7.5 Edit course settings page.

In the section “Edit course settings”, a short description of the lesson, format, enrolment duration, course start date, guest access choice, maximum upload setting, language, and how to address teachers and students were chosen. Only enrolled students can view the course. Settings can be changed at anytime needed.

7.2.4 Site Files

The files of overview of some topics which will help students about the technology and concept mentioned in the course were uploaded. Also, the pictures used in the slides of the lessons to show definitions and examples were uploaded by organizing various directories. Maximum upload size changed to 8 MB.

Figue 7.6 shows the site files uploaded containing necessary information for the course and back up data.



Introduction to WebML

WebML » WM101 » Files

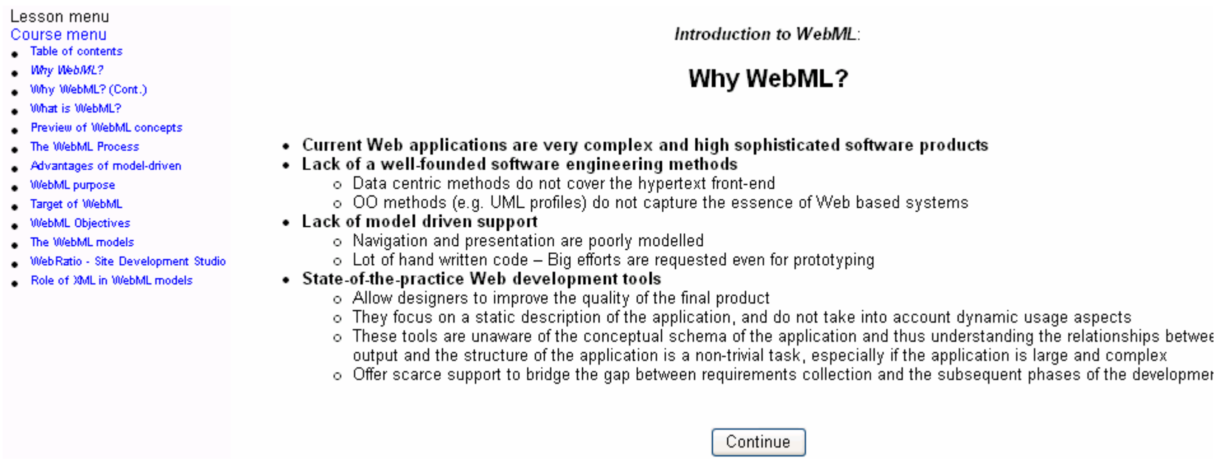
Name	Size	Modified	Action
Pictures	1.6MB	9 Apr 2006, 02:19 AM	Rename
backupdata	14.1MB	9 Apr 2006, 02:19 AM	Rename
glossary	1.3KB	9 Apr 2006, 02:19 AM	Rename
moddata	4 bytes	9 Apr 2006, 02:19 AM	Rename
2.zip	512.4KB	8 Apr 2006, 11:11 AM	Unzip List Restore Rename
DesigningData-IntensiveWebApplications.pdf	7.7MB	12 Feb 2006, 03:22 AM	Rename
Introduction.ppt	78.5KB	31 Jan 2006, 04:01 AM	Rename
OQL_Documentation.doc	37.5KB	14 Feb 2006, 02:31 AM	Rename
SQL_Documentation.doc	41KB	14 Feb 2006, 02:31 AM	Rename
Summary_of_WebML.doc	288KB	5 Feb 2006, 07:42 AM	Rename
SummaryofHypertextElements.doc	156KB	25 Jan 2006, 04:08 AM	Rename
UML_Documentation.doc	138KB	14 Feb 2006, 02:31 AM	Rename
WebMLSyntax.doc	56KB	5 Feb 2006, 07:34 AM	Rename
XML_Documentation.doc	56.5KB	14 Feb 2006, 02:31 AM	Rename
XSL_Documentation.doc	79KB	14 Feb 2006, 02:31 AM	Rename

Figure 7.6 The page of site files of Introduction to WebML course.

7.2.5 Lessons

Lessons include the topics discussed in Chapter 5. Each lesson includes one or more main topics of Web Modeling Language.

Content of the lessons were consolidated by help of definitions, explanations, examples and pictures. Lessons are shown as slides and students may go to the next slide by clicking the "Continue" button. On the left frame, a lesson menu shows all slide links, thus students may choose a any slide he/she is interested in. Figure 7.7 shows an example of a lesson page.



Lesson menu
Course menu
- Table of contents
- *Why WebML?*
- Why WebML? (Cont.)
- What is WebML?
- Preview of WebML concepts
- The WebML Process
- Advantages of model-driven
- WebML purpose
- Target of WebML
- WebML Objectives
- The WebML models
- WebRatio - Site Development Studio
- Role of XML in WebML models

Introduction to WebML:

Why WebML?

- **Current Web applications are very complex and high sophisticated software products**
- **Lack of a well-founded software engineering methods**
  - Data centric methods do not cover the hypertext front-end
  - OO methods (e.g. UML profiles) do not capture the essence of Web based systems
- **Lack of model driven support**
  - Navigation and presentation are poorly modelled
  - Lot of hand written code – Big efforts are requested even for prototyping
- **State-of-the-practice Web development tools**
  - Allow designers to improve the quality of the final product
  - They focus on a static description of the application, and do not take into account dynamic usage aspects
  - These tools are unaware of the conceptual schema of the application and thus understanding the relationships betwee output and the structure of the application is a non-trivial task, especially if the application is large and complex
  - Offer scarce support to bridge the gap between requirements collection and the subsequent phases of the developmer

Figure 7.7 An example of a lesson page.

7.3 Overview of the Introduction to WebML Course

This course was given for teaching basics of Web Modeling Language. The course was organized as nine weekly outlines, each week containing a lesson of one or two main topics and some auxiliary materials such as resources. During the learning period of lessons, the resources guide students to get information of the content.

Students can not take lessons unless current date is in the interval of start and end date. In other words, lessons are opened between start and end dates. Lessons can be reviewed in this time interval.

Assignments are given at the end of some lessons to strengthen the knowledge of students. Each asignment has a due date and thus assignments which are submitted late are low graded.

In the middle of the semester, a term examination is given to test knowledge level of students. Students can learn grades immediately after finishing the test.

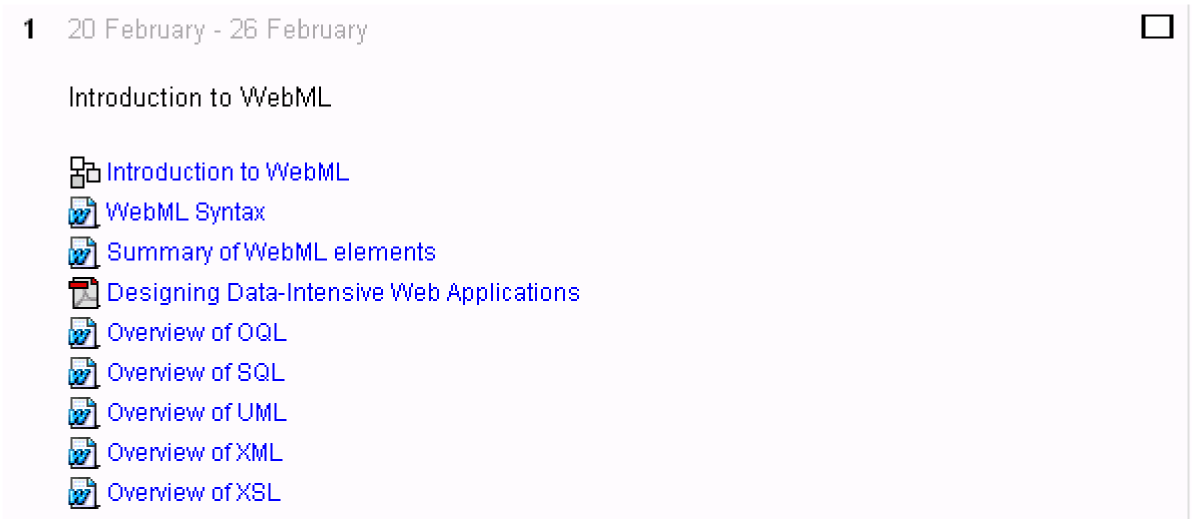
Real life examples of applications are introduced to give an idea of use of WebML by brief explanation.

Forums and polls are used for opinion research. A poll was released about whether the final ezamination should be given as a term project or a traditional examination.

At the end of the course, a final examination was given containing whole topics as a term project. Thus students' ability to implement the WebML models.

7.3.1 Week 1: Introduction to WebML

Fist week is aimed to give an overview information of WebML. Figure 7.8 shows the contents of the fist week.



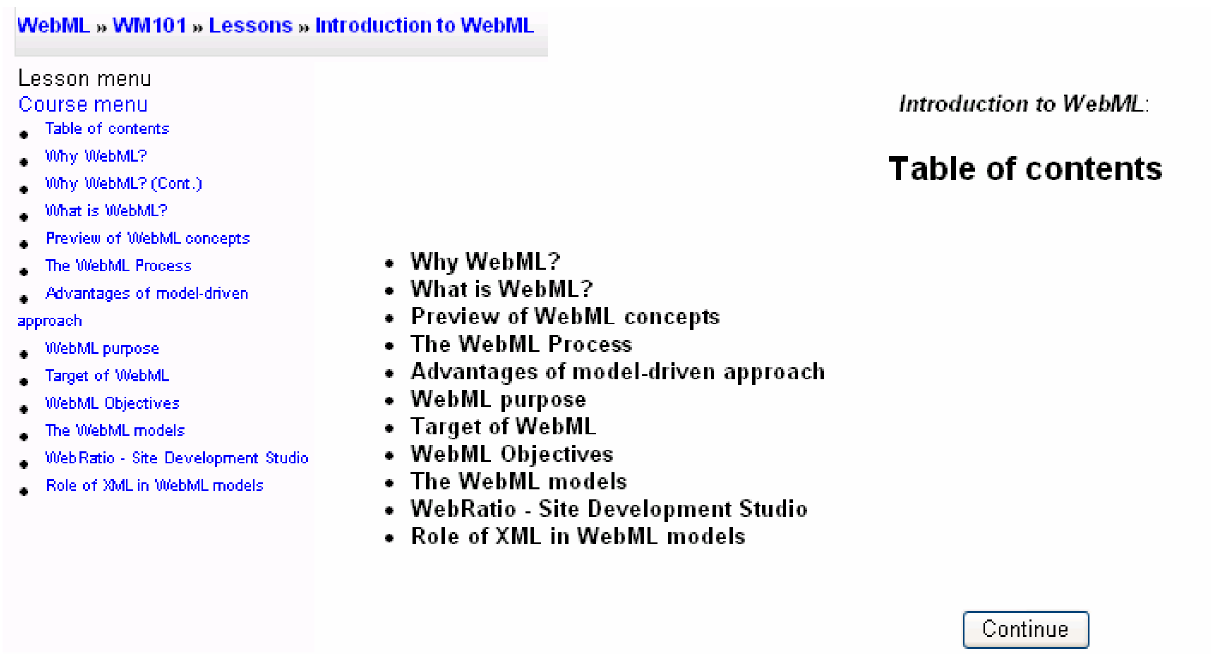
1 20 February - 26 February

Introduction to WebML

- Introduction to WebML
- WebML Syntax
- Summary of WebML elements
- Designing Data-Intensive Web Applications
- Overview of OQL
- Overview of SQL
- Overview of UML
- Overview of XML
- Overview of XSL

Figure 7.8 Content of Week 1.

The first lesson named “Introduction to WebML” was added which contains an overview information of WebML. Definition, advantages and objectives of WebML, preview of WebML concepts and WebRatio is are explained. Figure 7.9 shows the first slide of lesson “Introduction to WebML”.


WebML » WM101 » Lessons » Introduction to WebML

Lesson menu
Course menu

- Table of contents
- Why WebML?
- Why WebML? (Cont.)
- What is WebML?
- Preview of WebML concepts
- The WebML Process
- Advantages of model-driven approach
- WebML purpose
- Target of WebML
- WebML Objectives
- The WebML models
- WebRatio - Site Development Studio
- Role of XML in WebML models

Introduction to WebML:

Table of contents

- Why WebML?
- What is WebML?
- Preview of WebML concepts
- The WebML Process
- Advantages of model-driven approach
- WebML purpose
- Target of WebML
- WebML Objectives
- The WebML models
- WebRatio - Site Development Studio
- Role of XML in WebML models

Continue

Figure 7.9 Table of contents of lesson “Introduction to WebML”.

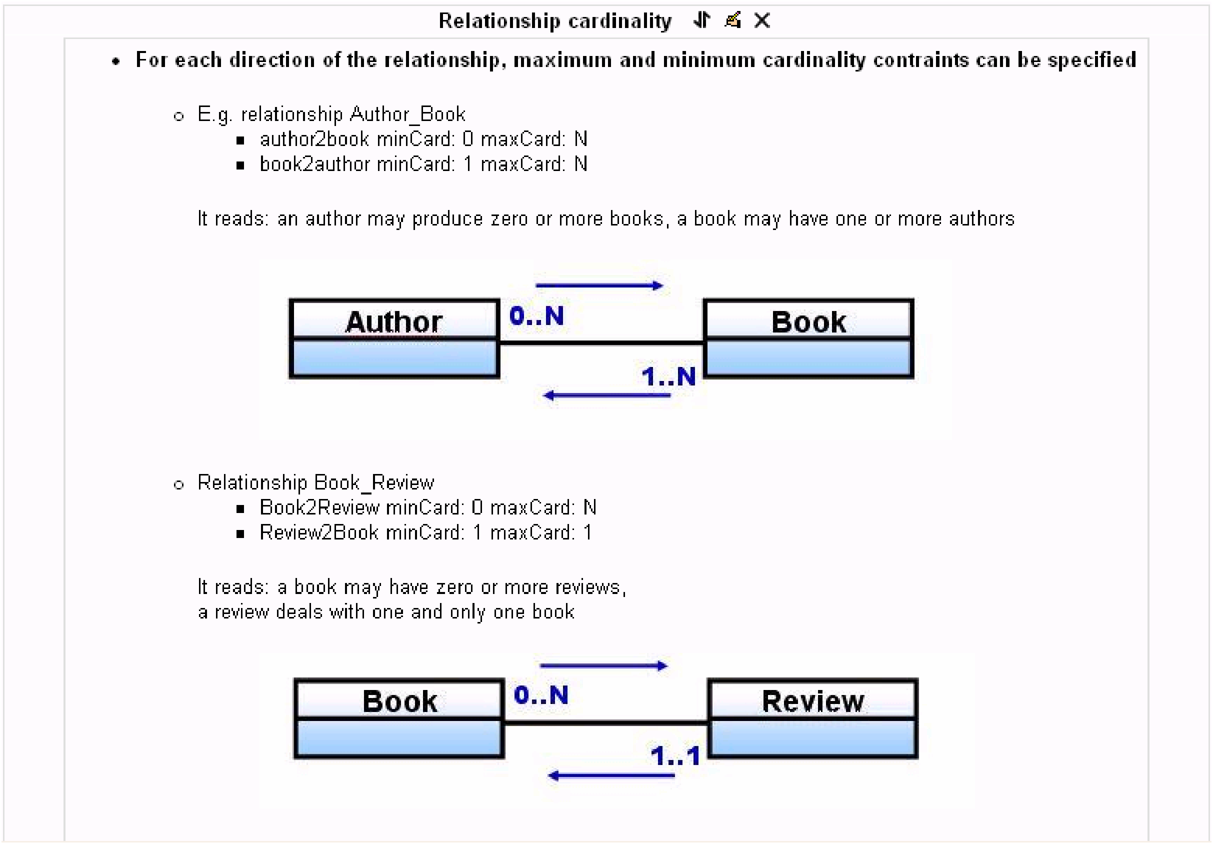
There are some concepts which students should know before taking the lesson of Week 1. Some brief explanation about SQL, OQL, UML, XML and XSL were given as adding resources. “WebML Syntax” and “Summary of WebML elements” resources.

The resource “Designing Data-Intensive Web Applications” is the source book of the course written by Stefano Ceri and Piero Fraternali who are professors at Politecnico di Milano.

7.3.2 Week 2: Structural Model

Second week mentions about Data and Derivation Models. Two lessons added named “Data Model ” and “Derivation Model”.

The lesson “Data Model” gives information about one of the WebML models, Data Model. Purpose, primitives of Data Model are explained at the beginning of the lesson. Then entity , attribute, relationship between entities, relationship types and structure patterns were mentioned. Each concept is supported by the help of examples. For example, in the figure 7.10, “Relationship cardinality” concept is defined and there was given some examples using a book store example. Cardinality constraints of relationship between authors and their books are seen. In the same way, cardinality constraints of relationship between books and reviews are seen.


Relationship cardinality

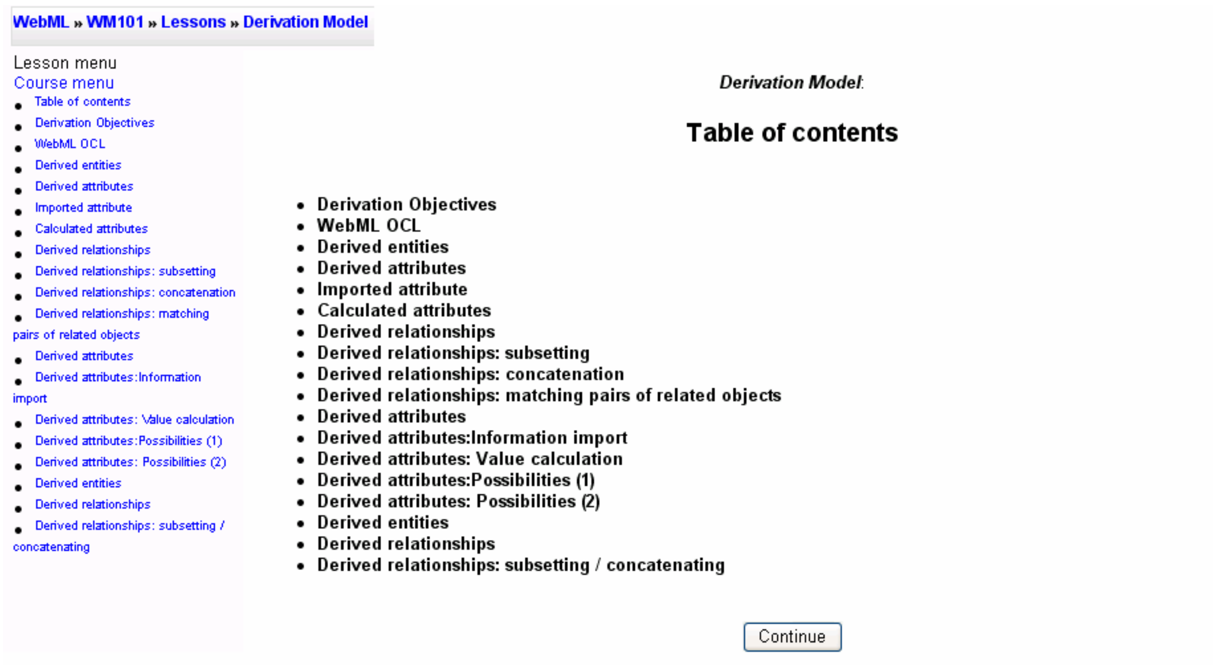
- **For each direction of the relationship, maximum and minimum cardinality contraints can be specified**
    - E.g. relationship Author_Book
        - author2book minCard: 0 maxCard: N
        - book2author minCard: 1 maxCard: N

      It reads: an author may produce zero or more books, a book may have one or more authors
    - Relationship Book_Review
        - Book2Review minCard: 0 maxCard: N
        - Review2Book minCard: 1 maxCard: 1

      It reads: a book may have zero or more reviews,
      a review deals with one and only one book

Figure 7.10 Definition of “Relationship cardinality”concept and given examples.

The lesson “Derivation Model” gives information about how to derive some information using Data Model concepts. Methods developed for calculating derived information were explained by giving examples. Figure 7.11 shows the table of contents of the lesson.


WebML » WM101 » Lessons » Derivation Model

Lesson menu
Course menu
- Table of contents
- Derivation Objectives
- WebML OCL
- Derived entities
- Derived attributes
- Imported attribute
- Calculated attributes
- Derived relationships
- Derived relationships: subsetting
- Derived relationships: concatenation
- Derived relationships: matching pairs of related objects
- Derived attributes
- Derived attributes:Information import
- Derived attributes: Value calculation
- Derived attributes:Possibilities (1)
- Derived attributes: Possibilities (2)
- Derived entities
- Derived relationships
- Derived relationships: subsetting / concatenating

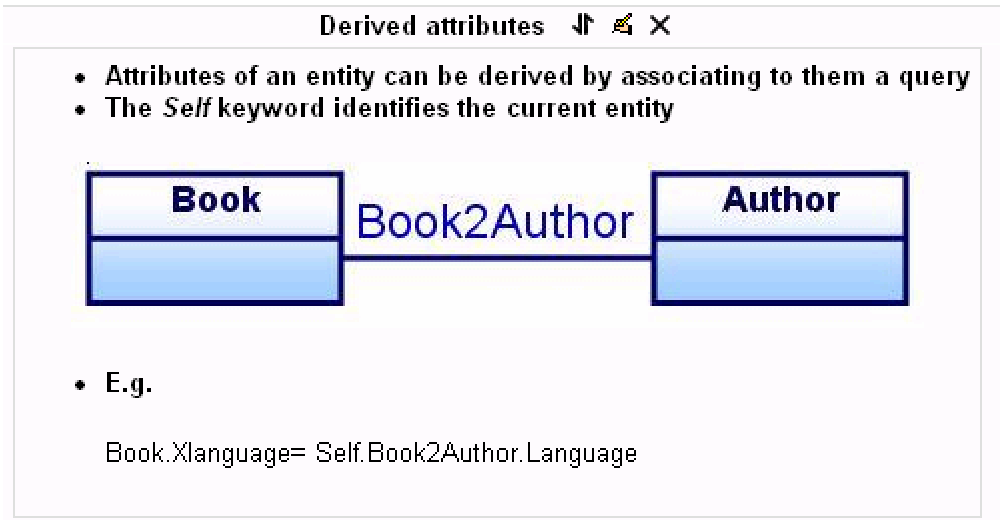
Derivation Model:

Table of contents

- Derivation Objectives
- WebML OCL
- Derived entities
- Derived attributes
- Imported attribute
- Calculated attributes
- Derived relationships
- Derived relationships: subsetting
- Derived relationships: concatenation
- Derived relationships: matching pairs of related objects
- Derived attributes
- Derived attributes:Information import
- Derived attributes: Value calculation
- Derived attributes:Possibilities (1)
- Derived attributes: Possibilities (2)
- Derived entities
- Derived relationships
- Derived relationships: subsetting / concatenating

Figure 7.11 Table of contents of lesson “Derivation Model”.

Figure 7.12 shows how these methods and examples were given in the lesson.


Derived attributes

- Attributes of an entity can be derived by associating to them a query
- The *Self* keyword identifies the current entity

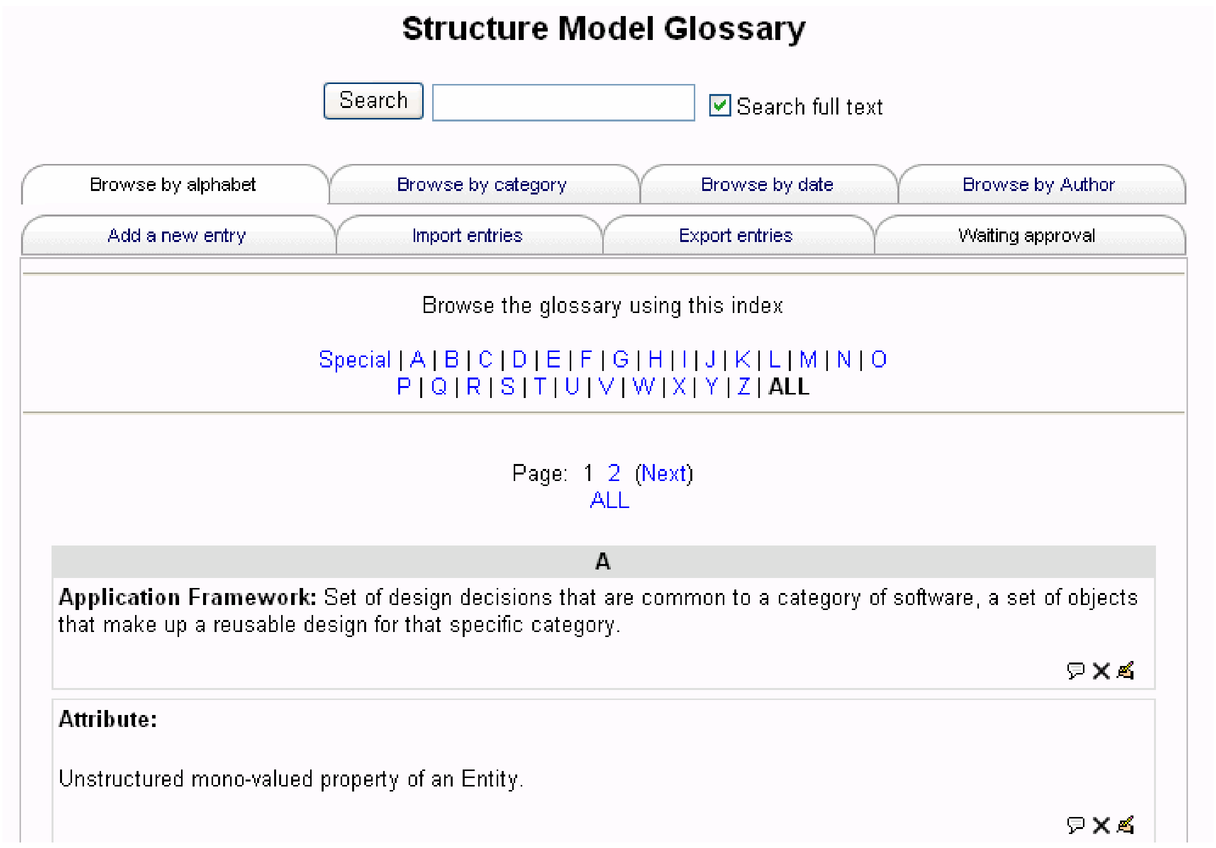
- E.g.

Book.Xlanguage= Self.Book2Author.Language

Figure 7.12 Calculating derived attributes and a given example.

Glossary

Structure Model Glossary contains terms about Data Model. It may be browsed by launching the list at one time or using the index as shown in Figure 7.13.



Application Framework: Set of design decisions that are common to a category of software, a set of objects that make up a reusable design for that specific category.

Attribute:

Unstructured mono-valued property of an Entity.

Figure 7.13 An example of glossary page.

Case Study

The site of ACME Furniture Industry, Inc is studied as a case study in the course. The site was developed by using WebML. The case study of was divided into three parts - lessons. The first part studied in Week 2 contains Structure Modeling of the site. The second part was studied at Week 3 and contains the review of Hypertext Modeling of the site. The last part is documentation about the whole WebML modeling of the site.

A brief explanation about the site process was given at the beginning of the case study. First, structure design form requirements was examined. Important entities such as product, combination, technical record and store entities and their relationships were modeled.

Appropriate mockups were modeled and then entity identification was defined. Finally, complete structure model was established. Figure 7.14 and 7.15 show some samples from the contents of the case study in Week 2.

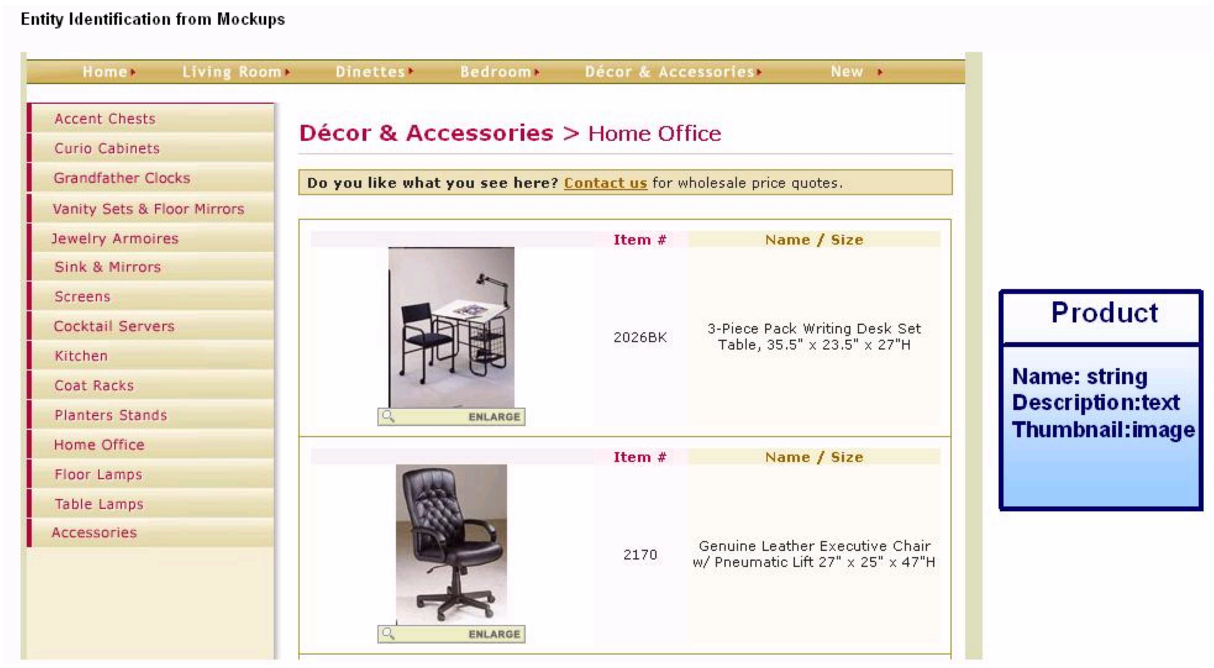


Figure 7.14 Entity Identification from Mockups

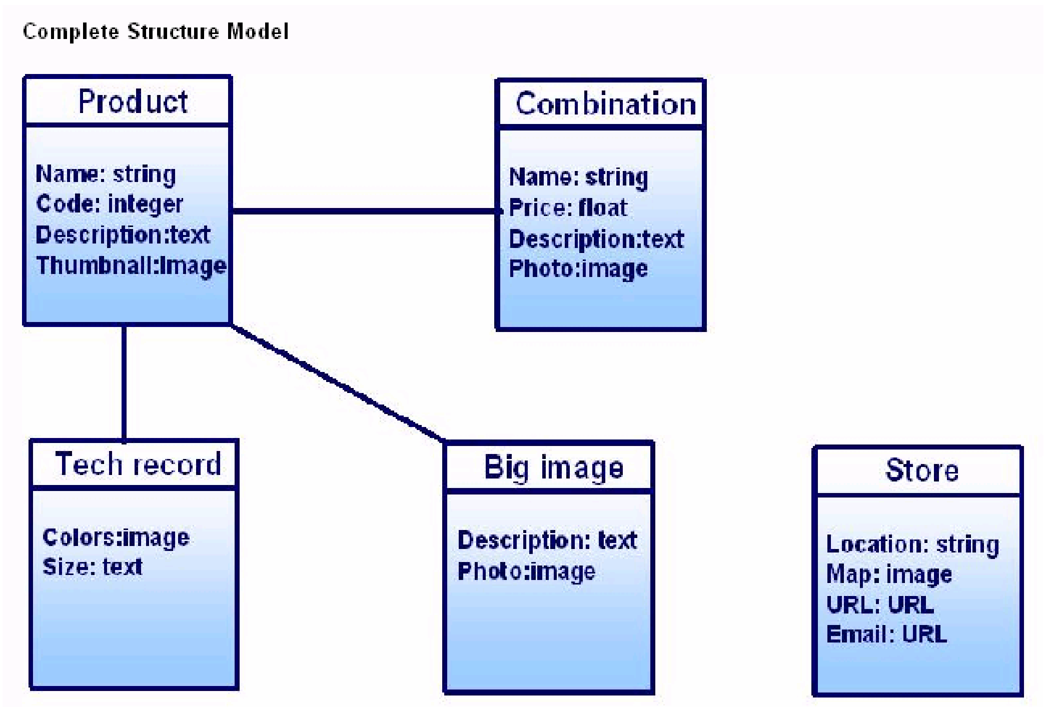


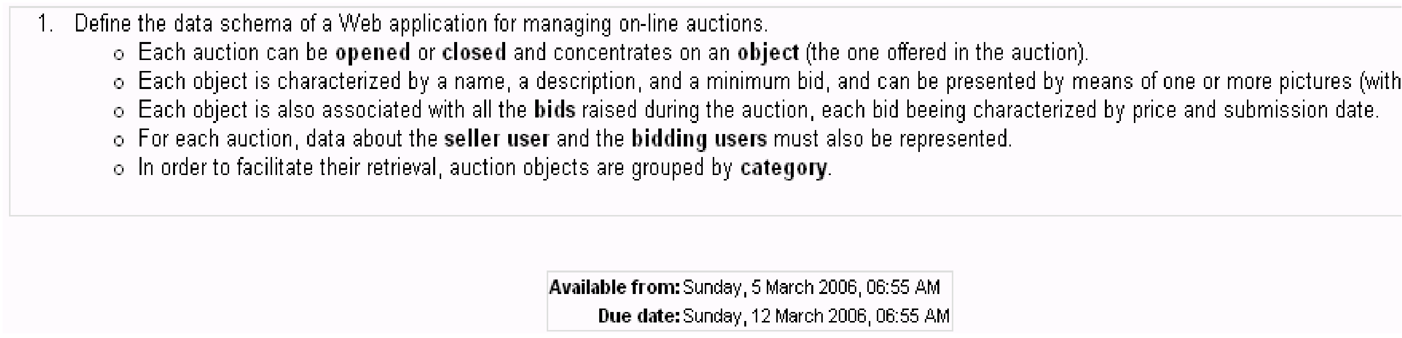
Figure 7.15 Entity Identification from Mockups

Assignment

An assignment about the lesson was added containing one question. Assignments may have two dates:

- Available from
- Due date

Students only submit assignments between this time interval. Figure 7.16 shows the assignment page of Week 2.



1. Define the data schema of a Web application for managing on-line auctions.
   - Each auction can be **opened** or **closed** and concentrates on an **object** (the one offered in the auction).
   - Each object is characterized by a name, a description, and a minimum bid, and can be presented by means of one or more pictures (with
   - Each object is also associated with all the **bids** raised during the auction, each bid beeing characterized by price and submission date.
   - For each auction, data about the **seller user** and the **bidding users** must also be represented.
   - In order to facilitate their retrieval, auction objects are grouped by **category**.

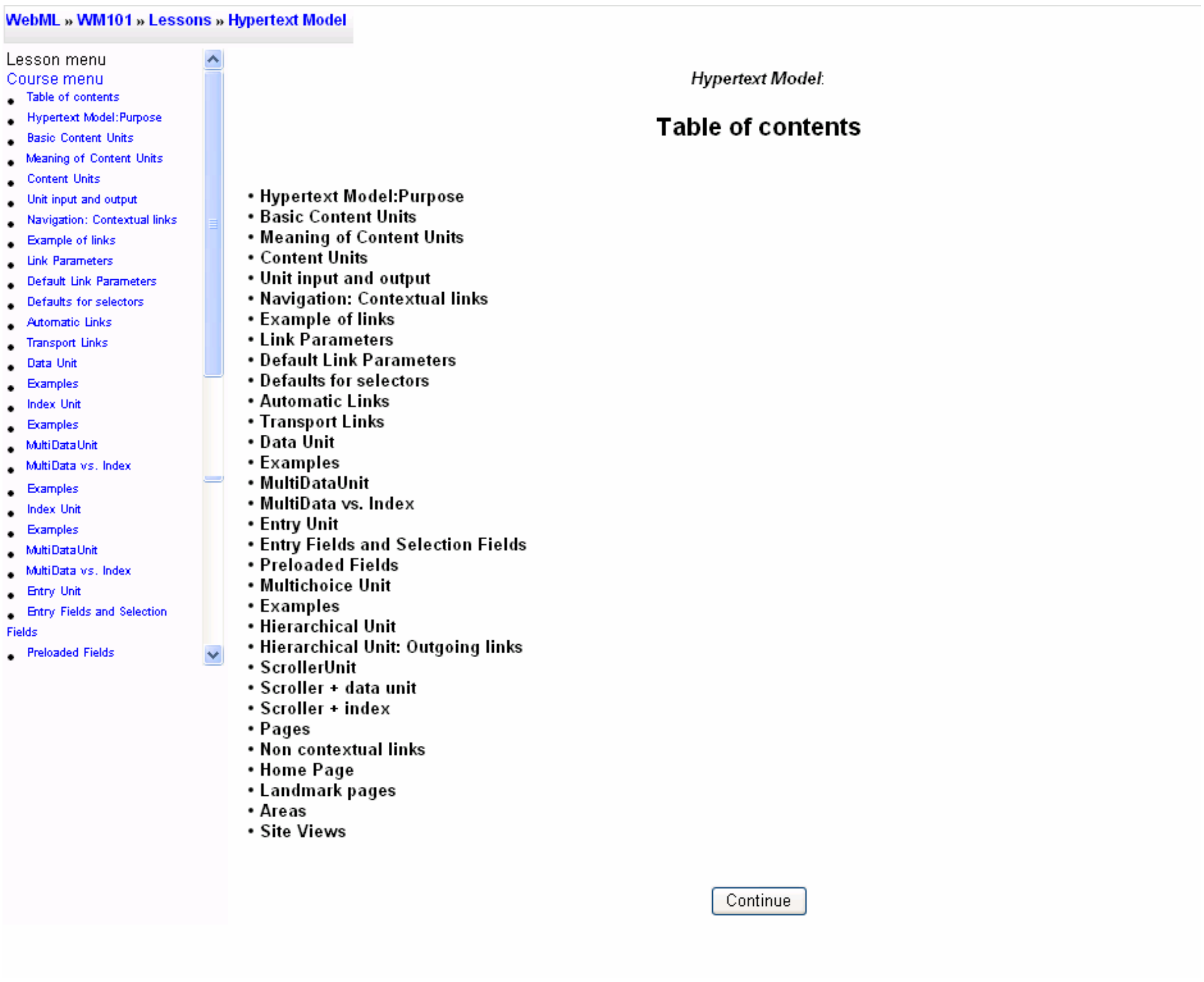
Available from: Sunday, 5 March 2006, 06:55 AM
Due date: Sunday, 12 March 2006, 06:55 AM

Figure 7.16 An example of assignment page.

7.3.3 Week 3: Hypertext Model

Third week is about Hypertext Models. A lesson was added named “Hypertext Model” which gives information about the purpose of the model, content units, links, link parameters, pages and areas. Figure 7.17 shows the table of contents of the lesson.

A resource named “Summary of WebML Hypertext Elements” which contains brief description and properties of hypertext elements was added to Week 3 in order to help students for easy understanding of the lesson.


WebML » WM101 » Lessons » Hypertext Model

Lesson menu
Course menu

- Table of contents
- Hypertext Model:Purpose
- Basic Content Units
- Meaning of Content Units
- Content Units
- Unit input and output
- Navigation: Contextual links
- Example of links
- Link Parameters
- Default Link Parameters
- Defaults for selectors
- Automatic Links
- Transport Links
- Data Unit
- Examples
- Index Unit
- Examples
- MultiDataUnit
- MultiData vs. Index
- Examples
- Index Unit
- Examples
- MultiDataUnit
- MultiData vs. Index
- Entry Unit
- Entry Fields and Selection Fields
- Preloaded Fields

Hypertext Model:

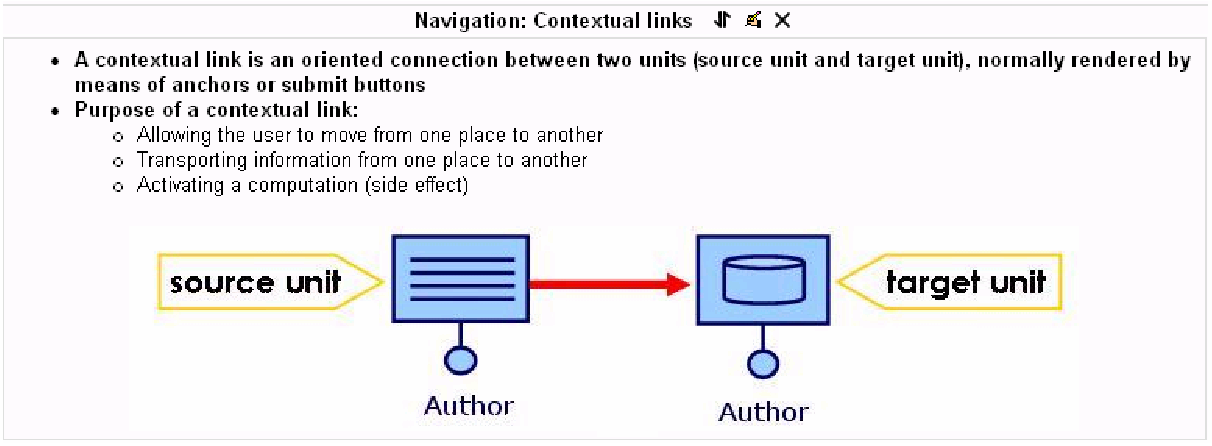
Table of contents

- Hypertext Model:Purpose
- Basic Content Units
- Meaning of Content Units
- Content Units
- Unit input and output
- Navigation: Contextual links
- Example of links
- Link Parameters
- Default Link Parameters
- Defaults for selectors
- Automatic Links
- Transport Links
- Data Unit
- Examples
- MultiDataUnit
- MultiData vs. Index
- Entry Unit
- Entry Fields and Selection Fields
- Preloaded Fields
- Multichoice Unit
- Examples
- Hierarchical Unit
- Hierarchical Unit: Outgoing links
- ScrollerUnit
- Scroller + data unit
- Scroller + index
- Pages
- Non contextual links
- Home Page
- Landmark pages
- Areas
- Site Views

Continue

Figure 7.17 Table of contents of lesson "Hypertext Model".

All concepts are supported by giving examples like shown in Figure 7.18.

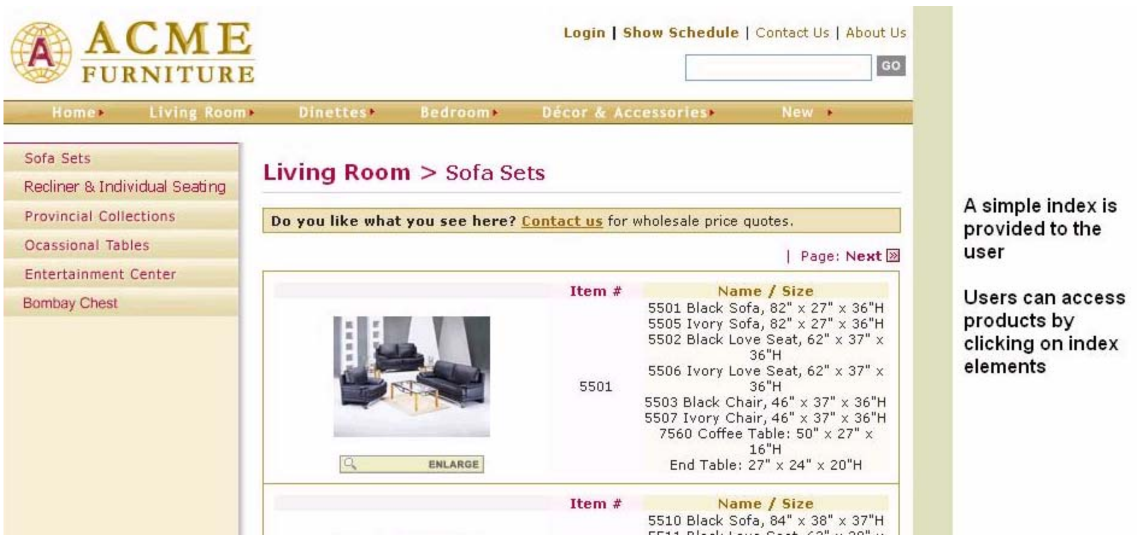

Navigation: Contextual links

- **A contextual link is an oriented connection between two units (source unit and target unit), normally rendered by means of anchors or submit buttons**
- **Purpose of a contextual link:**
    - Allowing the user to move from one place to another
    - Transporting information from one place to another
    - Activating a computation (side effect)

Figure 7.18 "Contextual links" concept and an example.

Case Study

The second part of the case study contains product index, product page and site view modeling of the Acme site. First a simple index was provided to the user. Users can access products by clicking on index elements. Products can be seen from the product page. Finally Acme home page, items, combinations and stores pages were modeled. Figure 7.19 and 7.20 show some samples from the contents of the case study in Week 3.



Living Room > Sofa Sets

Do you like what you see here? Contact us for wholesale price quotes.

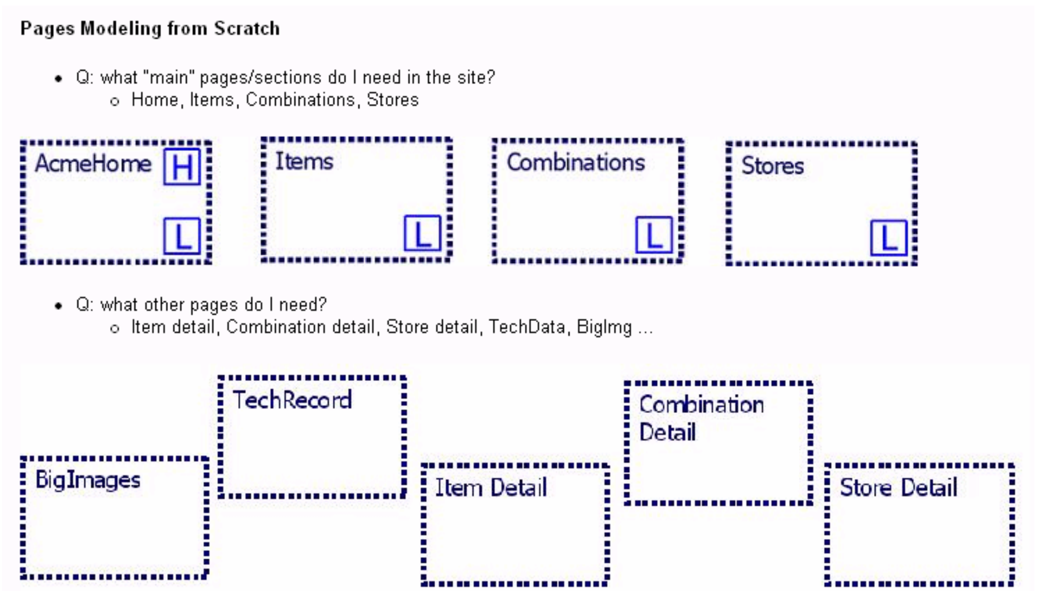
Item #	Name / Size
5501	5501 Black Sofa, 82" x 27" x 36"H 5505 Ivory Sofa, 82" x 27" x 36"H 5502 Black Love Seat, 62" x 37" x 36"H 5506 Ivory Love Seat, 62" x 37" x 36"H 5503 Black Chair, 46" x 37" x 36"H 5507 Ivory Chair, 46" x 37" x 36"H 7560 Coffee Table: 50" x 27" x 16"H End Table: 27" x 24" x 20"H

Item #	Name / Size
	5510 Black Sofa, 84" x 38" x 37"H

A simple index is provided to the user

Users can access products by clicking on index elements

Figure 7.19 “Product index” page of Acme site.



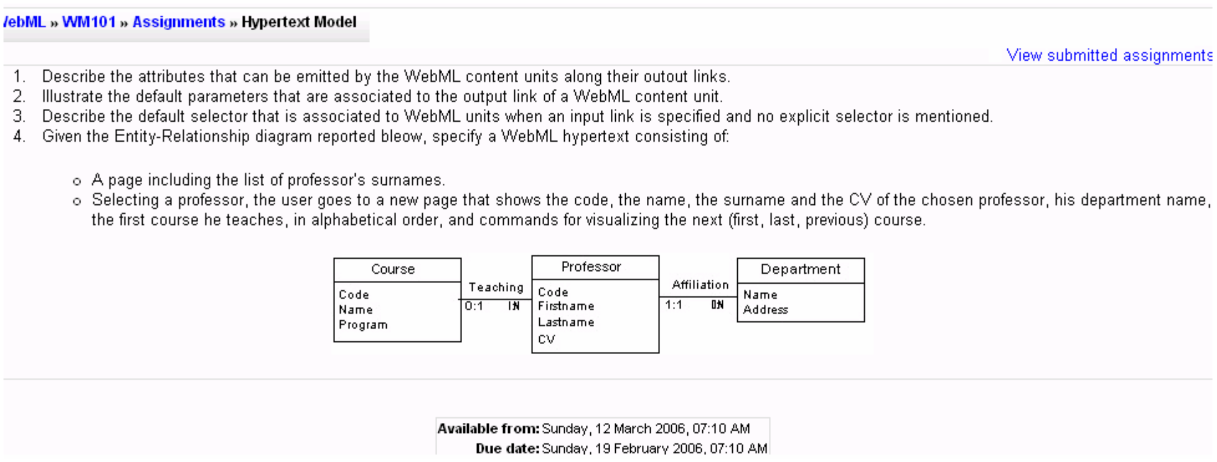
Pages Modeling from Scratch

- Q: what "main" pages/sections do I need in the site?
  - Home, Items, Combinations, Stores
- Q: what other pages do I need?
  - Item detail, Combination detail, Store detail, TechData, BigImg ...

Figure 7.20 Page modeling of Acme site.

Assignment

At the end of the lesson, an assignment was added in order to help students practice what they had learned. The assignment consists of four questions about the concepts taught in the lesson as shown in Figure 7.21.



/ebML » WM101 » Assignments » Hypertext Model

View submitted assignments

1. Describe the attributes that can be emitted by the WebML content units along their outout links.
2. Illustrate the default parameters that are associated to the output link of a WebML content unit.
3. Describe the default selector that is associated to WebML units when an input link is specified and no explicit selector is mentioned.
4. Given the Entity-Relationship diagram reported bleow, specify a WebML hypertext consisting of:
   - A page including the list of professor's surnames.
   - Selecting a professor, the user goes to a new page that shows the code, the name, the surname and the CV of the chosen professor, his department name, the first course he teaches, in alphabetical order, and commands for visualizing the next (first, last, previous) course.

Available from: Sunday, 12 March 2006, 07:10 AM
Due date: Sunday, 19 February 2006, 07:10 AM

Figure 7.21 Hypertext Model assignment page.

7.3.4 Week 4: Presentation Model

Fourth week contains a lesson named "Presentation Model". The lesson gives information about purpose and process of Presentation Model. At the end of the lesson, the topic is strengthen by using an example which explains layout and graphic appearance of pages of the site global.acer.com which is developed by using WebML as shown in Figure 7.22.

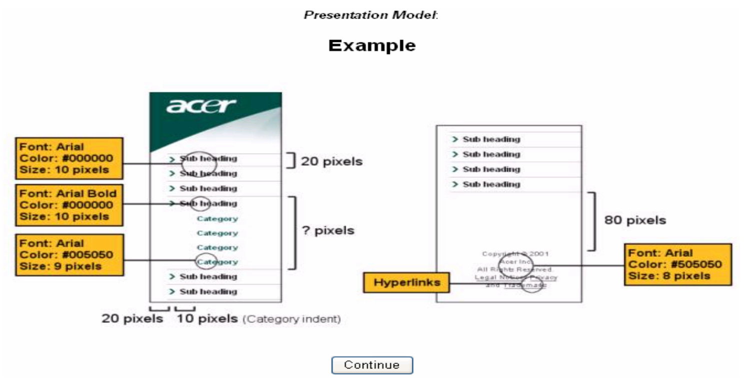
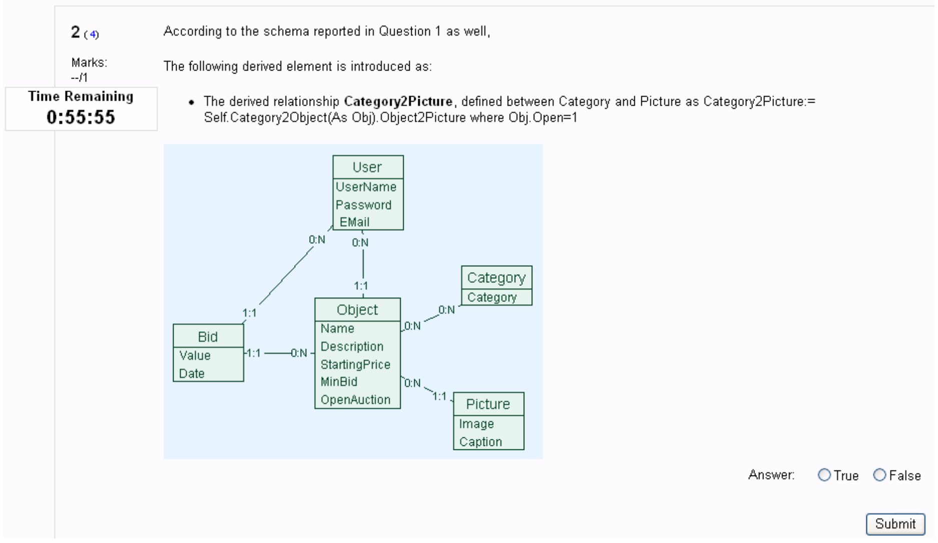


Figure 7.22 An example explaining the layout and graphic appearance of pages of the site global.acer.com

7.3.5 Week 5: Term Examination

Week 5 includes a term examination. The term examination contains four questions which are about the topics submitted so far. The first two questions are true-false. Third one is a short answer and last one is a matching question. Time lmit of the examination is one hour. There is a counter while the examination which students can control the time left. When the examination is finished, the score is explained to the student. Figure 7.23 and 7.24 show the second and fourth question pages of the term examination .


2 (4)

Marks:
--/1

Time Remaining
0:55:55

According to the schema reported in Question 1 as well,

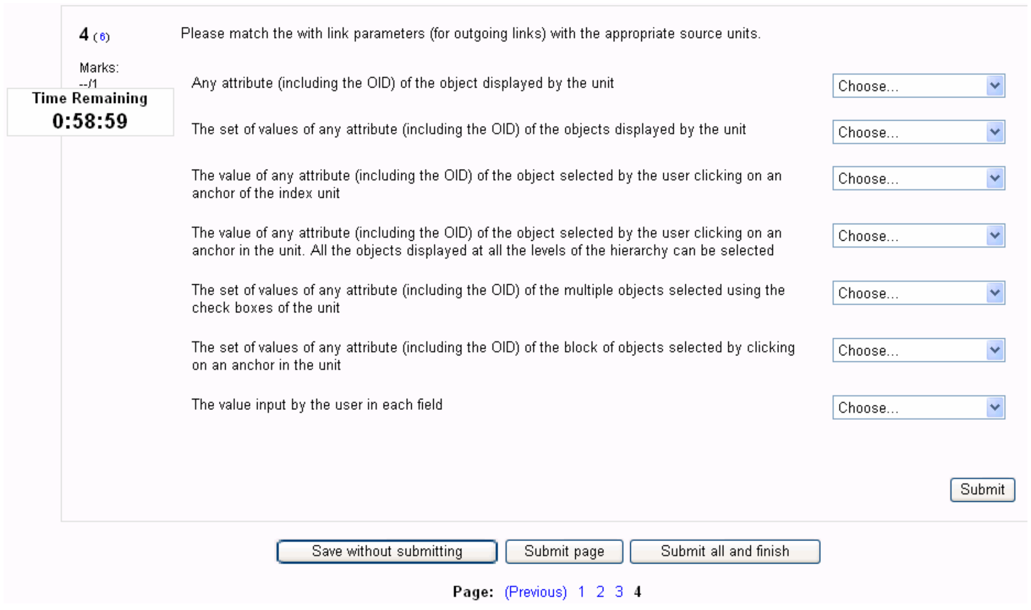
The following derived element is introduced as:

- The derived relationship **Category2Picture**, defined between Category and Picture as Category2Picture:= Self.Category2Object(As Obj).Object2Picture where Obj.Open=1

Answer: True False

Submit

Figure 7.23 Second question of the term examination.


4 (6)

Marks:
--/1

Time Remaining
0:58:59

Please match the with link parameters (for outgoing links) with the appropriate source units.

Any attribute (including the OID) of the object displayed by the unit	Choose...
The set of values of any attribute (including the OID) of the objects displayed by the unit	Choose...
The value of any attribute (including the OID) of the object selected by the user clicking on an anchor of the index unit	Choose...
The value of any attribute (including the OID) of the object selected by the user clicking on an anchor in the unit. All the objects displayed at all the levels of the hierarchy can be selected	Choose...
The set of values of any attribute (including the OID) of the multiple objects selected using the check boxes of the unit	Choose...
The set of values of any attribute (including the OID) of the block of objects selected by clicking on an anchor in the unit	Choose...
The value input by the user in each field	Choose...

Submit

Save without submitting | Submit page | Submit all and finish

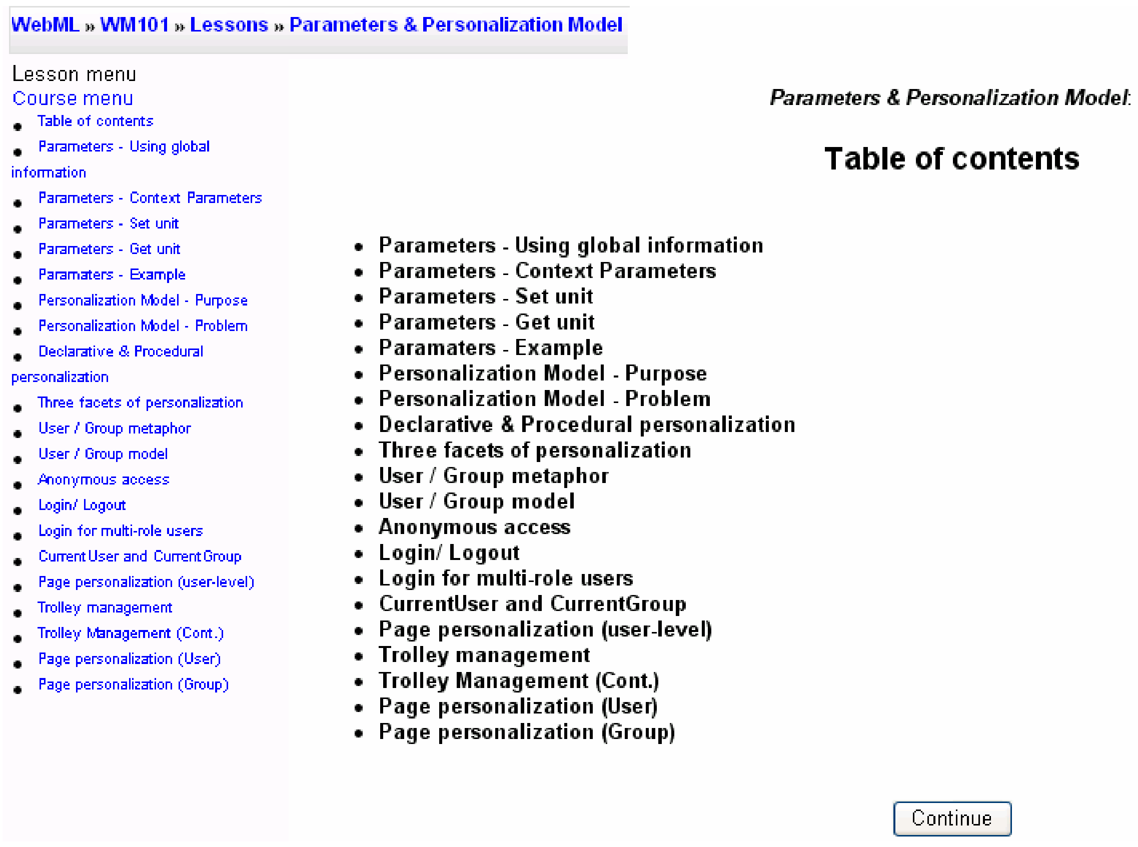
Page: (Previous) 1 2 3 **4**

Figure 7.24 Fourth question of the term examination.

7.3.6 Week 6: Parameters & Personalization Model

Week 6 contains a lesson named “Parameters & Personalization Model”. First, global information, set and get units for parameteres were explained.

Then purpose and personlization types, user/group assigning, login/logout and access operations were explained. The lesson’s table of content is shown in Figure 7.25.


WebML » WM101 » Lessons » Parameters & Personalization Model

Lesson menu
Course menu
- Table of contents
- Parameters - Using global information
- Parameters - Context Parameters
- Parameters - Set unit
- Parameters - Get unit
- Paramaters - Example
- Personalization Model - Purpose
- Personalization Model - Problem
- Declarative & Procedural personalization
- Three facets of personalization
- User / Group metaphor
- User / Group model
- Anonymous access
- Login/ Logout
- Login for multi-role users
- CurrentUser and CurrentGroup
- Page personalization (user-level)
- Trolley management
- Trolley Management (Cont.)
- Page personalization (User)
- Page personalization (Group)

Parameters & Personalization Model:

Table of contents

- Parameters - Using global information
- Parameters - Context Parameters
- Parameters - Set unit
- Parameters - Get unit
- Paramaters - Example
- Personalization Model - Purpose
- Personalization Model - Problem
- Declarative & Procedural personalization
- Three facets of personalization
- User / Group metaphor
- User / Group model
- Anonymous access
- Login/ Logout
- Login for multi-role users
- CurrentUser and CurrentGroup
- Page personalization (user-level)
- Trolley management
- Trolley Management (Cont.)
- Page personalization (User)
- Page personalization (Group)

Figure 7.25 Table of contents of lesson “Hypertext Model”.

Examples were added to the lesson to strengthen the concepts of the lesson. Figure 7.26 gives an example about page personalization for a group.

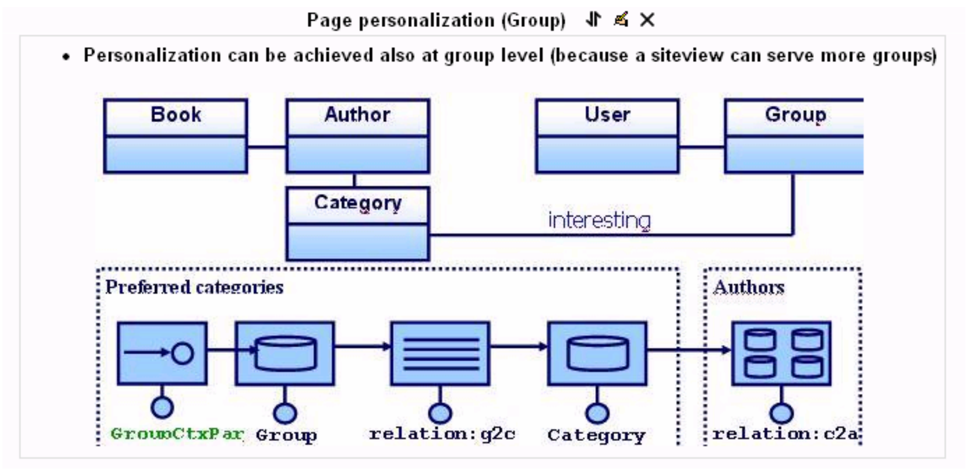
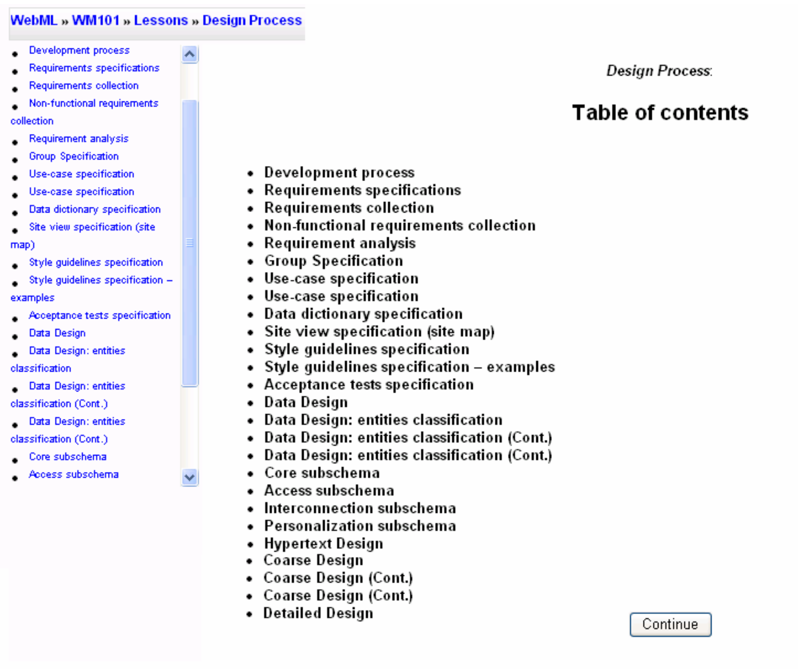

Page personalization (Group)
• Personalization can be achieved also at group level (because a siteview can serve more groups)
Book
Author
User
Group
Category
interesting
Preferred categories
Authors
GroupCtxPar
Group
relation:g2c
Category
relation:c2a

Figure 7.26 Example about page personalization for a group.

7.3.7 Week 7: Design Process

At Week 7, the lesson "Design Process" was submitted for teaching the main concepts of Design Process. Development process, requirements collection and analysis, use-case specification, data design, hypertext design and coarse design concepts were explained. Figure 7.27 gives a list of the lesson contents.


WebML » WM101 » Lessons » Design Process

- Development process
- Requirements specifications
- Requirements collection
- Non-functional requirements collection
- Requirement analysis
- Group Specification
- Use-case specification
- Use-case specification
- Data dictionary specification
- Site view specification (site map)
- Style guidelines specification
- Style guidelines specification – examples
- Acceptance tests specification
- Data Design
- Data Design: entities classification
- Data Design: entities classification (Cont.)
- Data Design: entities classification (Cont.)
- Core subschema
- Access subschema

Design Process

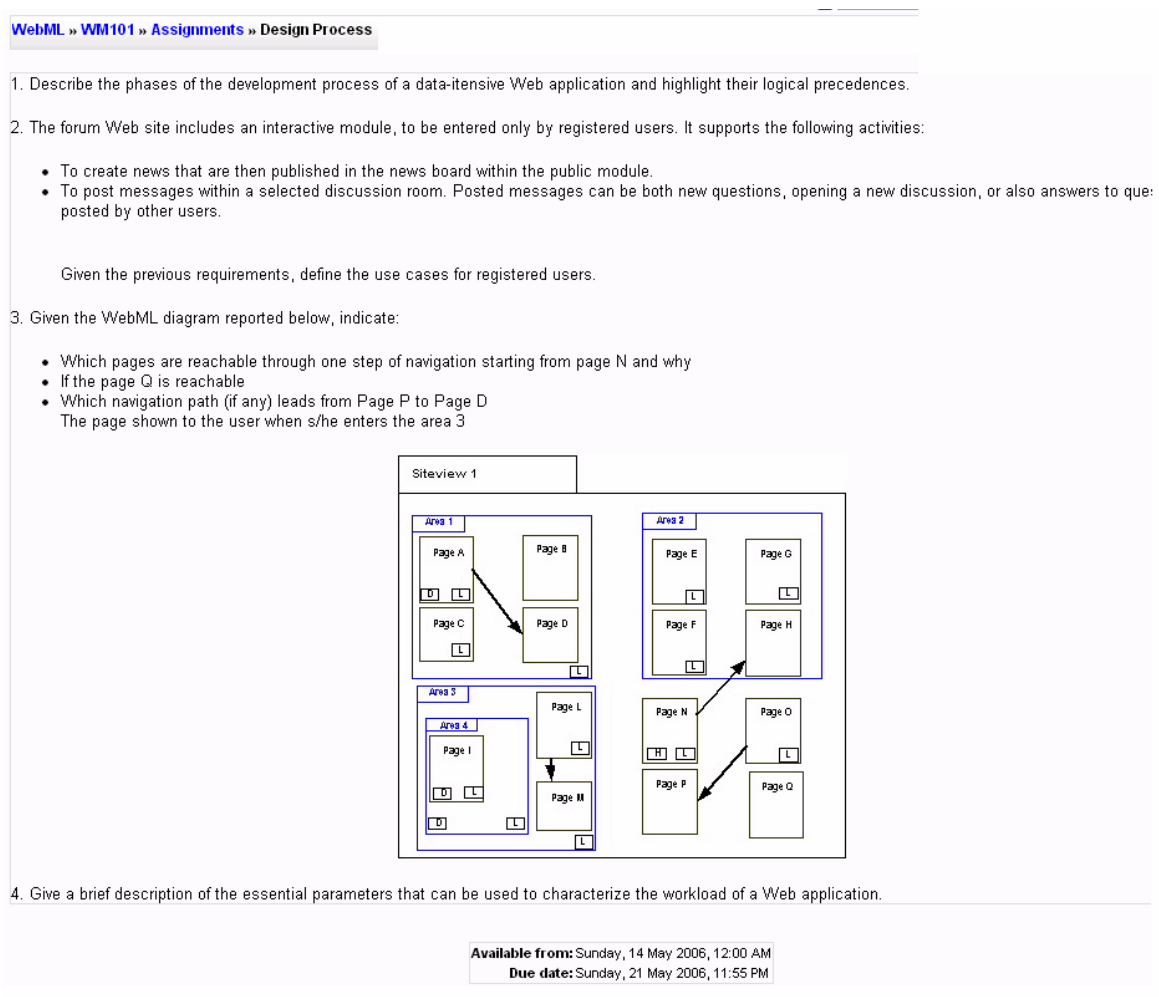
Table of contents

- Development process
- Requirements specifications
- Requirements collection
- Non-functional requirements collection
- Requirement analysis
- Group Specification
- Use-case specification
- Use-case specification
- Data dictionary specification
- Site view specification (site map)
- Style guidelines specification
- Style guidelines specification – examples
- Acceptance tests specification
- Data Design
- Data Design: entities classification
- Data Design: entities classification (Cont.)
- Data Design: entities classification (Cont.)
- Core subschema
- Access subschema
- Interconnection subschema
- Personalization subschema
- Hypertext Design
- Coarse Design
- Coarse Design (Cont.)
- Coarse Design (Cont.)
- Detailed Design

Figure 7.27 Contents of "Design Process" lesson.

At the end of the lesson, an assignment about the lesson was added containing four questions which is shown in Figure 7.28

.



WebML » WM101 » Assignments » Design Process

1. Describe the phases of the development process of a data-itensive Web application and highlight their logical precedences.

2. The forum Web site includes an interactive module, to be entered only by registered users. It supports the following activities:

- To create news that are then published in the news board within the public module.
- To post messages within a selected discussion room. Posted messages can be both new questions, opening a new discussion, or also answers to que
posted by other users.

Given the previous requirements, define the use cases for registered users.

3. Given the WebML diagram reported below, indicate:

- Which pages are reachable through one step of navigation starting from page N and why
- If the page Q is reachable
- Which navigation path (if any) leads from Page P to Page D
The page shown to the user when s/he enters the area 3

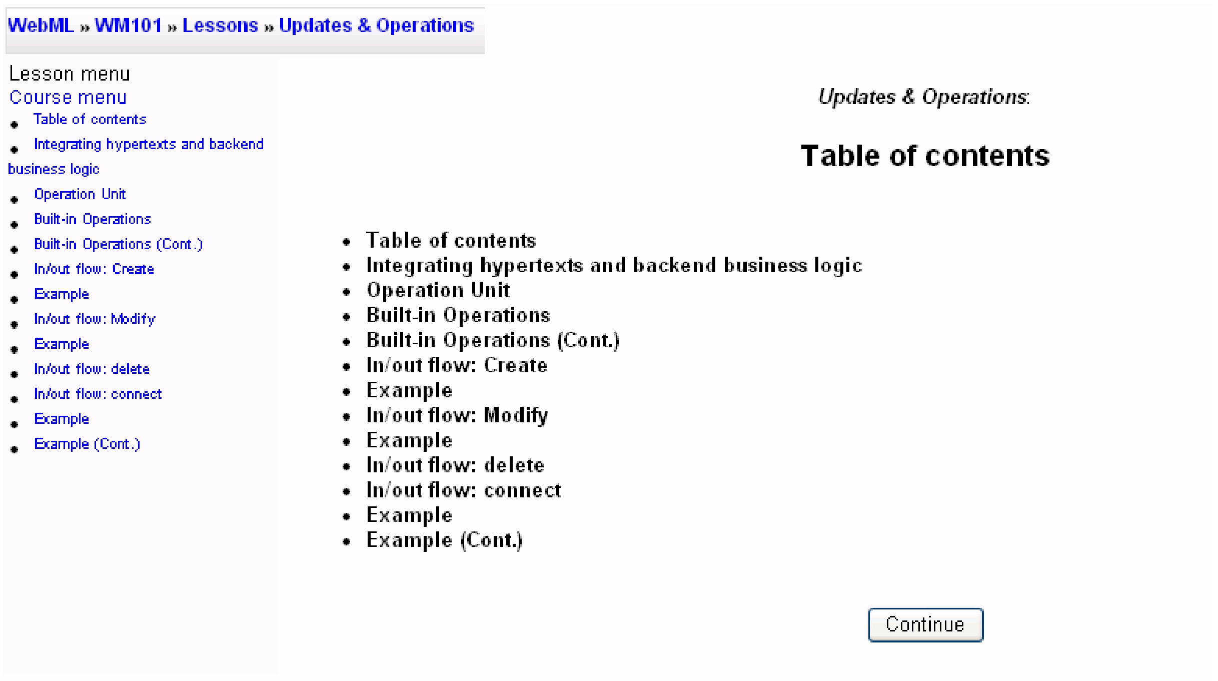
4. Give a brief description of the essential parameters that can be used to characterize the workload of a Web application.

Available from: Sunday, 14 May 2006, 12:00 AM
Due date: Sunday, 21 May 2006, 11:55 PM

Figure 7.28 Assignment.page of Week 7.

7.3.8 Week 8: Updates and Operations

Week 8 is aimed to give information about some basic updates and operations of WebML. First, a lesson “Updates and Operations” was submitted which contains built-in operations, create, modify, delete, connect and disconnect unit concepts. Figure 7.29 gives the content of the lesson.


WebML » WM101 » Lessons » Updates & Operations

Lesson menu
Course menu
- Table of contents
- Integrating hypertexts and backend business logic
- Operation Unit
- Built-in Operations
- Built-in Operations (Cont.)
- In/out flow: Create
- Example
- In/out flow: Modify
- Example
- In/out flow: delete
- In/out flow: connect
- Example
- Example (Cont.)

Updates & Operations:

Table of contents

- Table of contents
- Integrating hypertexts and backend business logic
- Operation Unit
- Built-in Operations
- Built-in Operations (Cont.)
- In/out flow: Create
- Example
- In/out flow: Modify
- Example
- In/out flow: delete
- In/out flow: connect
- Example
- Example (Cont.)

Figure 7.29 Assignment.page of Week 7.

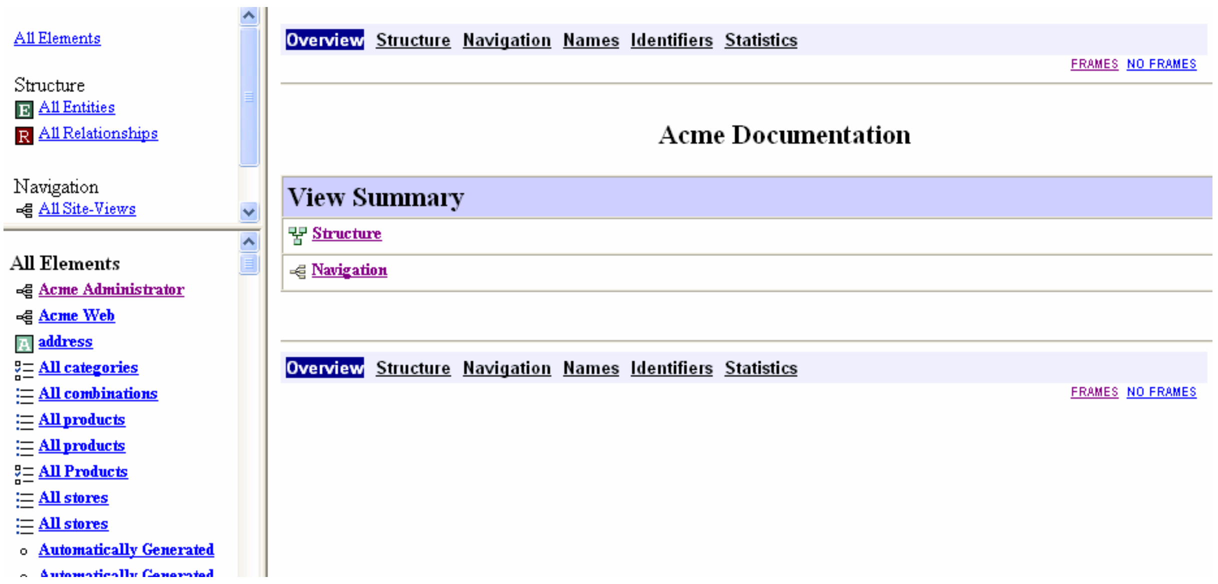
Case Study

The last part of the case study is a documentation about the modeling of the whole Acme site. The case study was given as a resource in a directory “Case Study”.

AcmeCaseStudy folder explains the sructure and navigation study of Acme Site. Diagrams are shown of Entity Hierarchy, Entity Summary, Relationship Summary, Site-View Summary, Global Parameter Summary, Personalization Settings, Users, Groups and Business rules in detail.

Students can extract the file AcmeCaseStudy.rar, copy the directory to their localhost and view as a site. They can find all details about WebML structure of Acme Site.

Figure 7.30 shows the home page of the documentation of Acme site modeling. In the left frame, all elements are listed for easy access. The right frame shows an overview of six links which are Structure, Navigation, Names, Identifiers and Statistics.


All Elements

Structure
E All Entities
R All Relationships

Navigation
All Site-Views

All Elements
- Acme Administrator
- Acme Web
- address
- All categories
- All combinations
- All products
- All products
- All Products
- All stores
- All stores
- Automatically Generated

Overview Structure Navigation Names Identifiers Statistics
FRAMES NO FRAMES

Acme Documentation

View Summary

Structure

Navigation

Overview Structure Navigation Names Identifiers Statistics
FRAMES NO FRAMES

Figure 7.30 Home page of the documentation of Acme site modeling.

Structure model page as shown in 7.31, gives the structure diagram of the Acme site. Entity hierarcy, entity summary and relationship summary can be accessed by clicking the appropriate links.

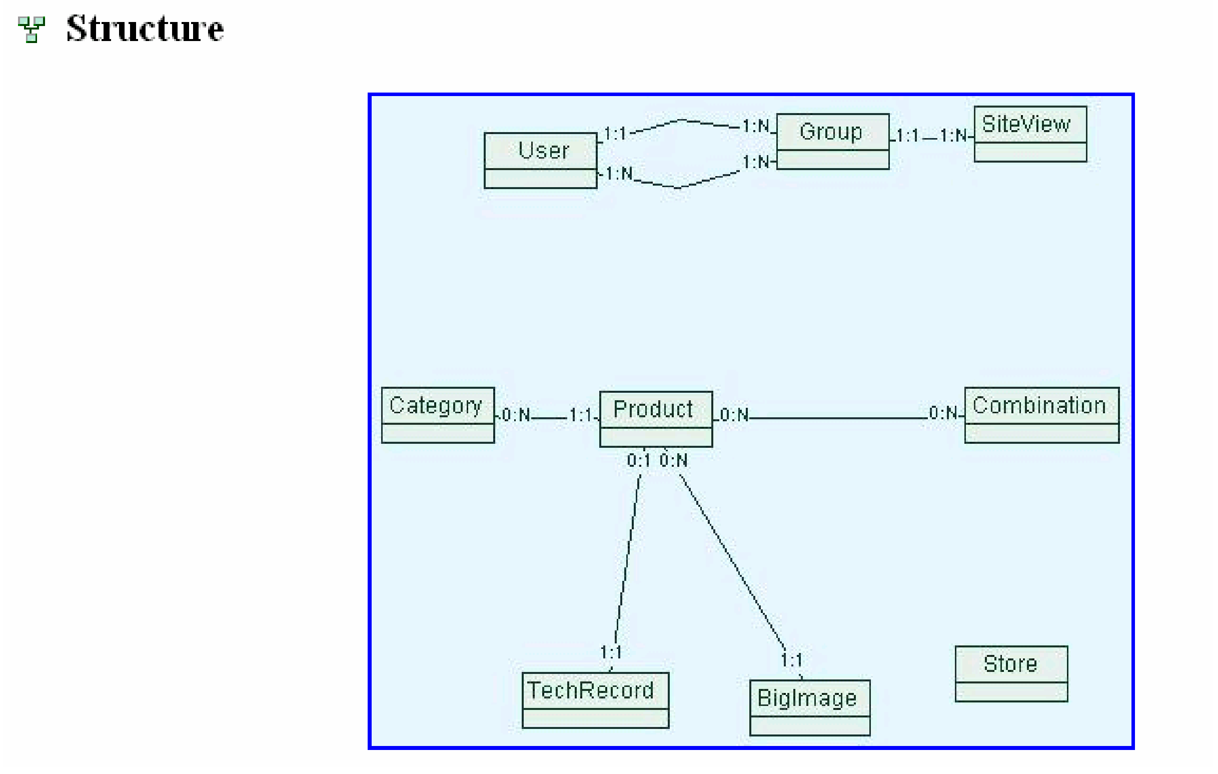
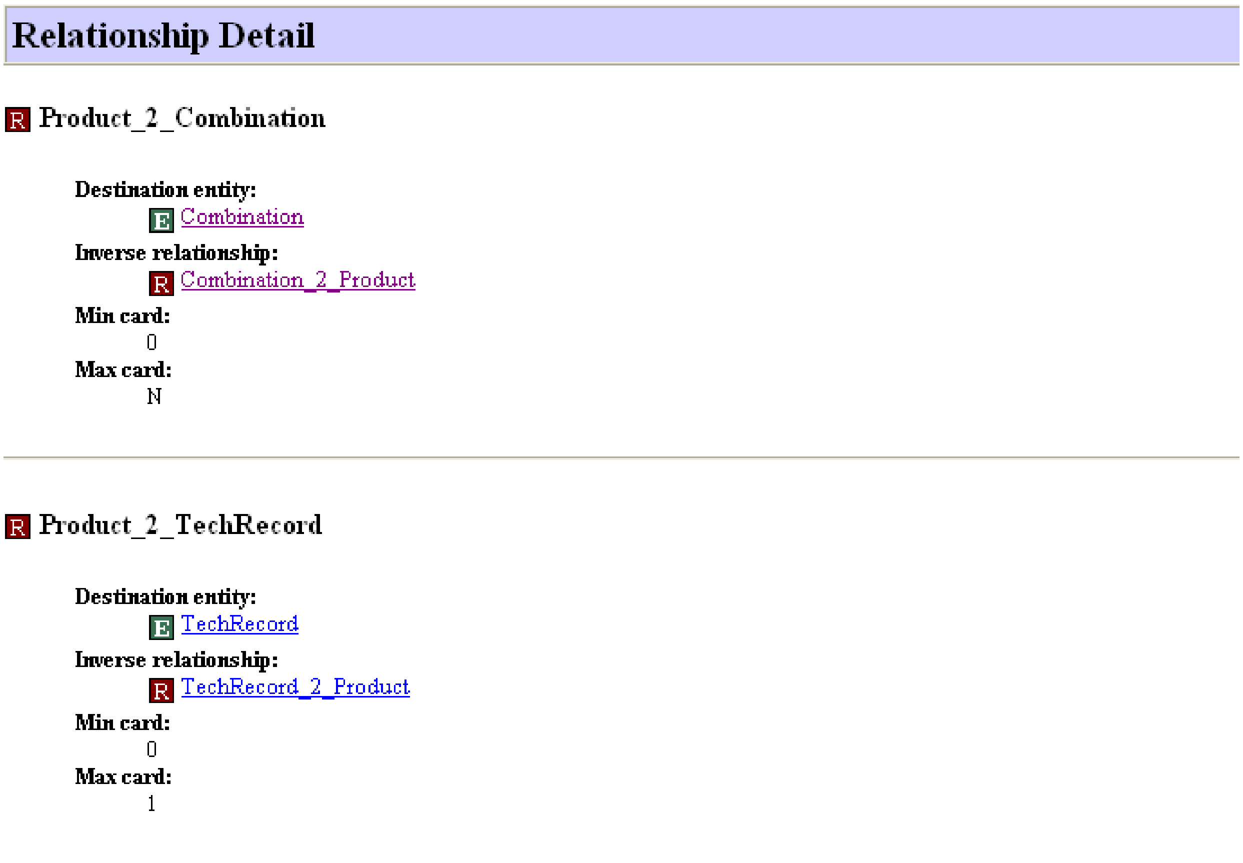


Figure 7.31 The page of structure diagram of the Acme site.

Figure 7.32 shows some samples from the relationship details of the site.


Relationship Detail

R Product_2_Combination

Destination entity:
E Combination
Inverse relationship:
R Combination_2_Product
Min card:
0
Max card:
N

R Product_2_TechRecord

Destination entity:
E TechRecord
Inverse relationship:
R TechRecord_2_Product
Min card:
0
Max card:
1

Figure 7.32 Relationship details of Product_2_Combination and Product_2_TechRecord.

Navigation page contains site view and global parameters information. Figure 7.33 shows the diagram of site view of Acme Web. Home page, product page, store page and offers page are connected by linking the content units to eachother. All content units which exist in pages can be accessed by clicking on them.

Figure 7.34 shows a sample of content units accessed by clicking from the site view diagram. The links which can be seen as a hierarchicy are linked to the appropriate site views an pages.

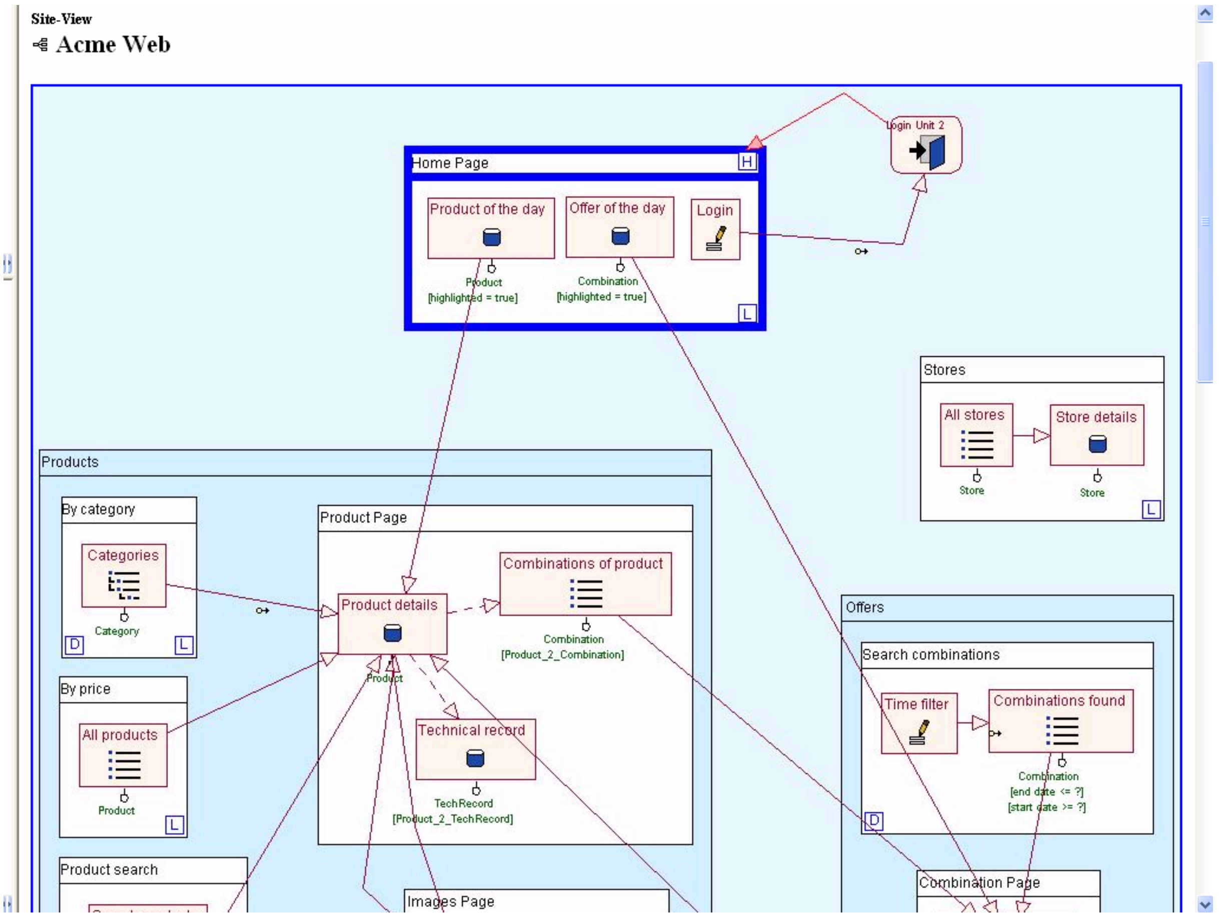
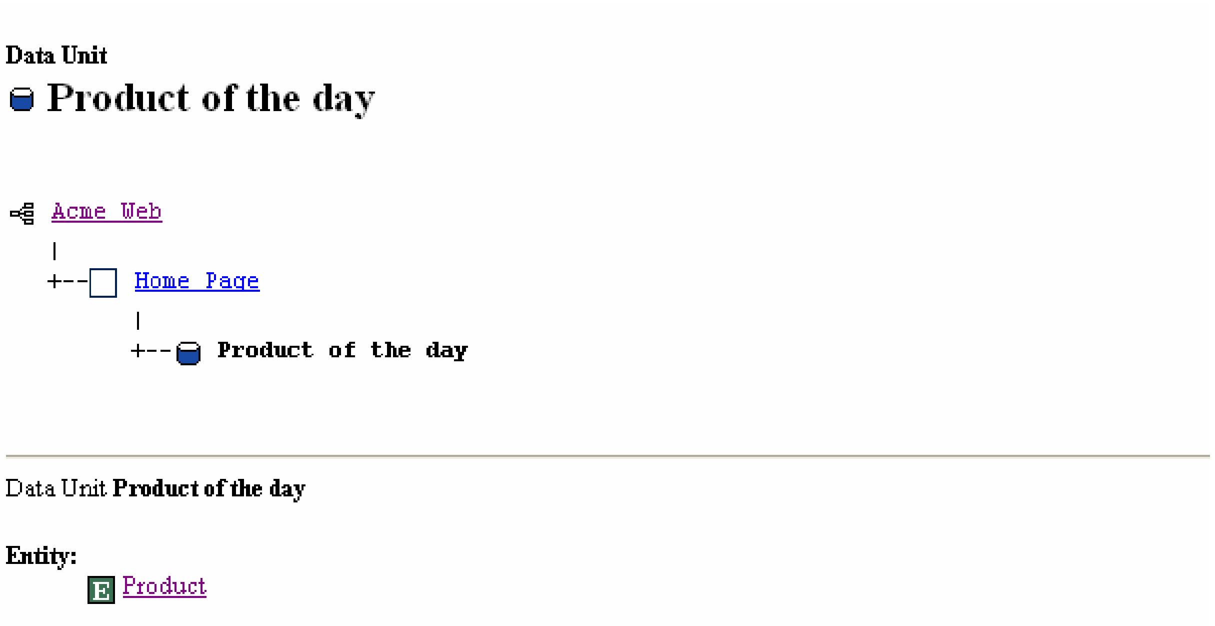


Figure 7.33 Site view diagram of Acme Web.



Data Unit

Product of the day

Acme Web
|
+--Home Page
|
+--**Product of the day**

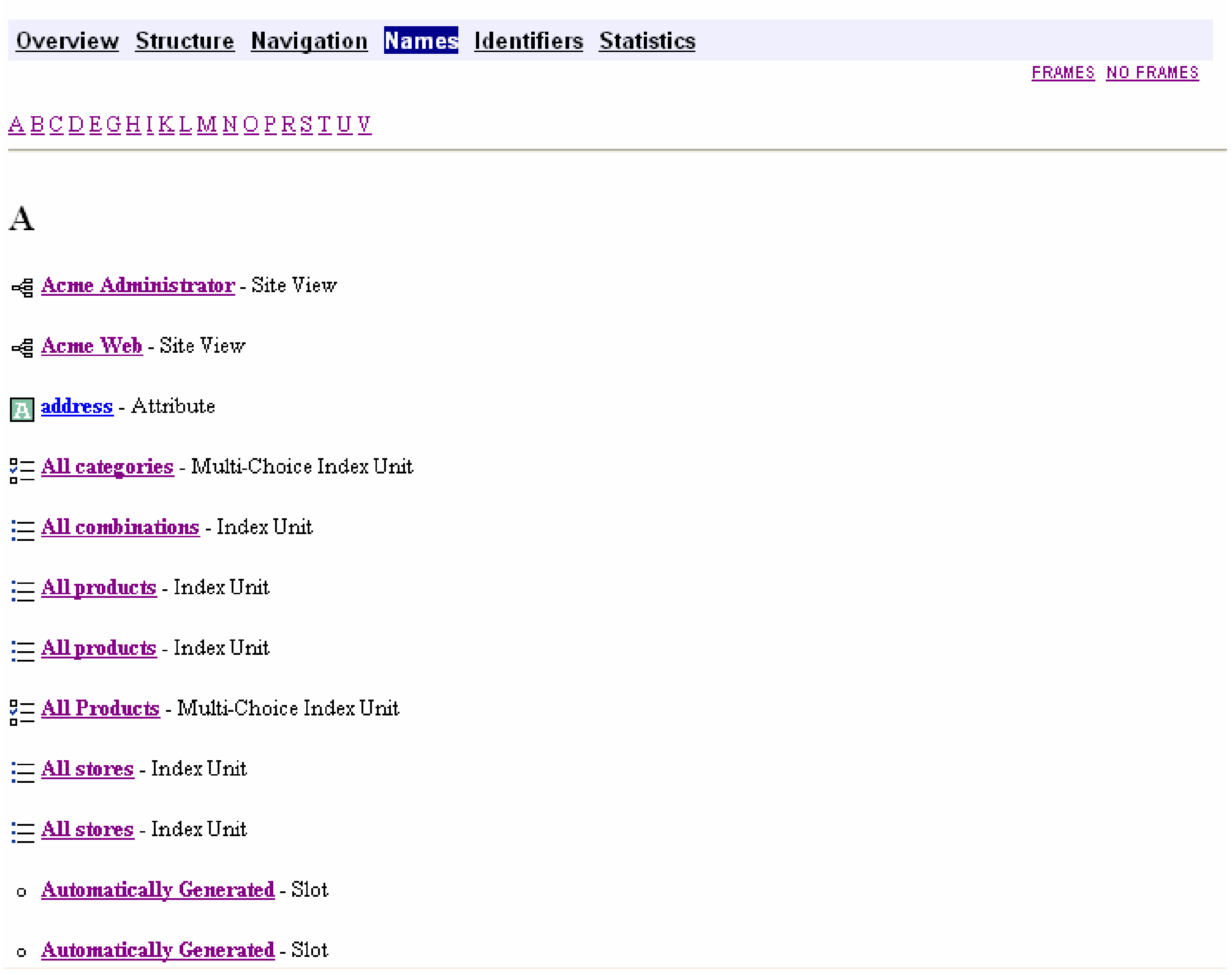
Data Unit **Product of the day**

Entity:

E Product

Figure 7.34 "Product of the day" data unit.

The page “Names” contains all the modeling concepts of the Acme site listed alphabetically as shown in Figure 7.35.


Overview Structure Navigation Names Identifiers Statistics

FRAMES NO FRAMES

A B C D E G H I K L M N O P R S T U V

A

Acme Administrator - Site View

Acme Web - Site View

address - Attribute

All categories - Multi-Choice Index Unit

All combinations - Index Unit

All products - Index Unit

All products - Index Unit

All Products - Multi-Choice Index Unit

All stores - Index Unit

All stores - Index Unit

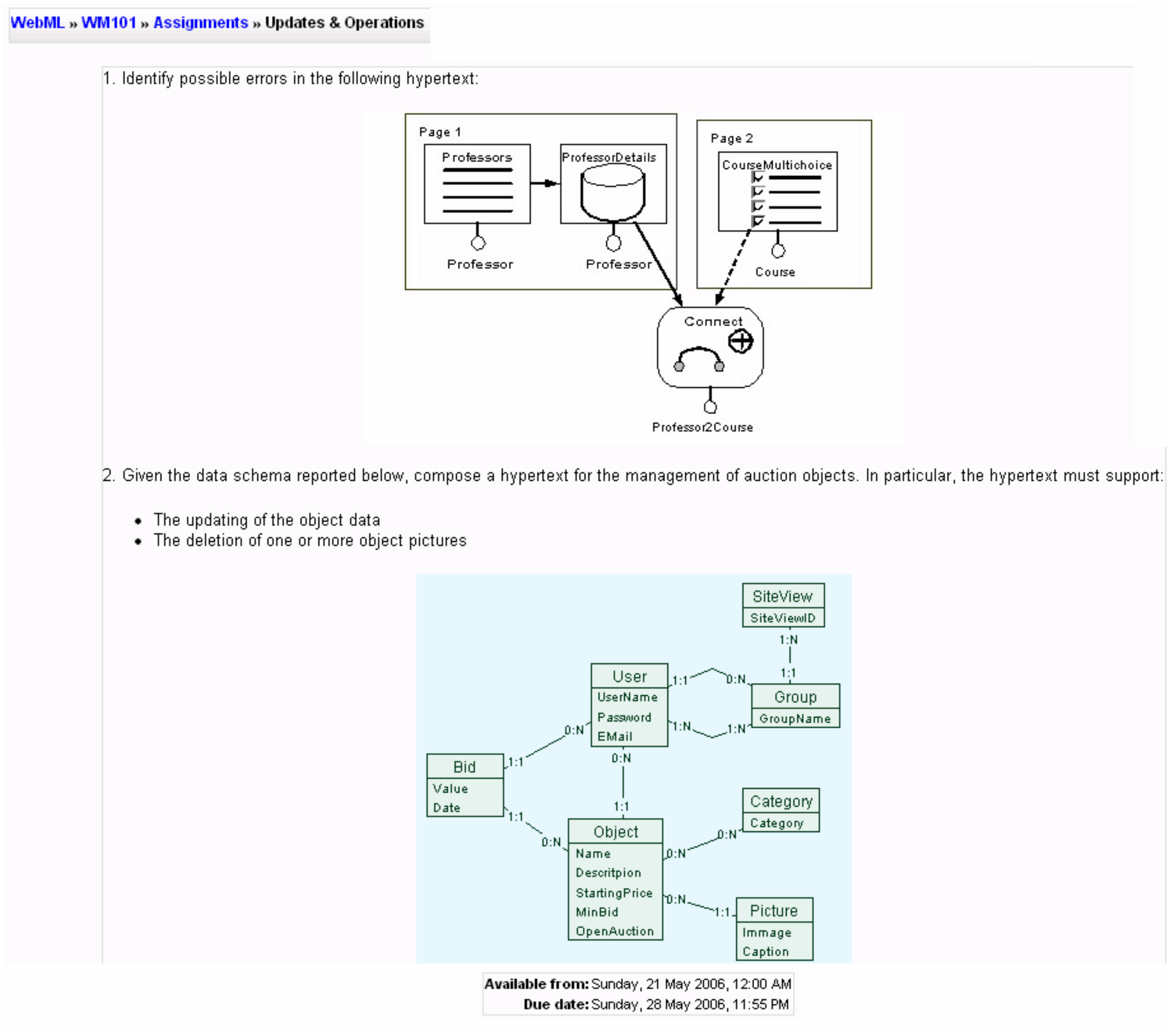
Automatically Generated - Slot

Automatically Generated - Slot

Figure 7.35 “Names” page of Acme site modeling concepts.

Assignment

An assignment was added containing two questions which is shown in Figure 7.36.



WebML » WM101 » Assignments » Updates & Operations

1. Identify possible errors in the following hypertext:

2. Given the data schema reported below, compose a hypertext for the management of auction objects. In particular, the hypertext must support:

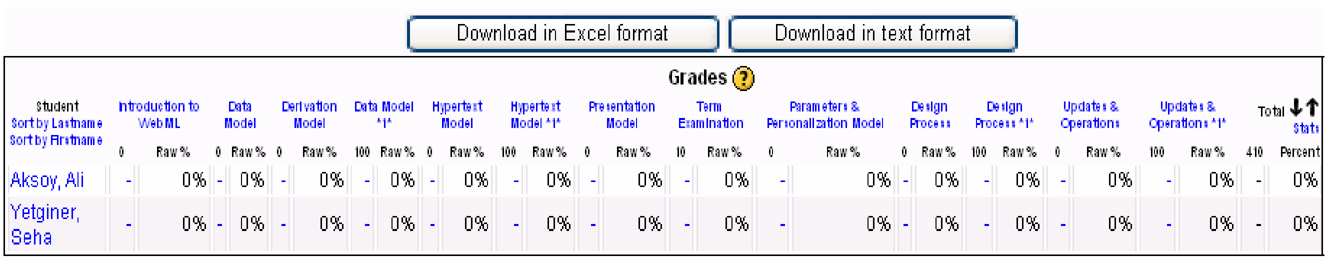
- The updating of the object data
- The deletion of one or more object pictures

Available from: Sunday, 21 May 2006, 12:00 AM
Due date: Sunday, 28 May 2006, 11:55 PM

Figure 7.36 Assignment.page of Week 8.

7.3.9 Week 9: Final Examination

Week 9 contains a final examination about what have been taught so far. After students took the final examination, grades were computed according to the scores of assignments and examinations. Grades can be reported in Excel or Text format. In the “grades page” of the course, each score for each students are listed as in Figure 7.37.


Download in Excel format | Download in text format

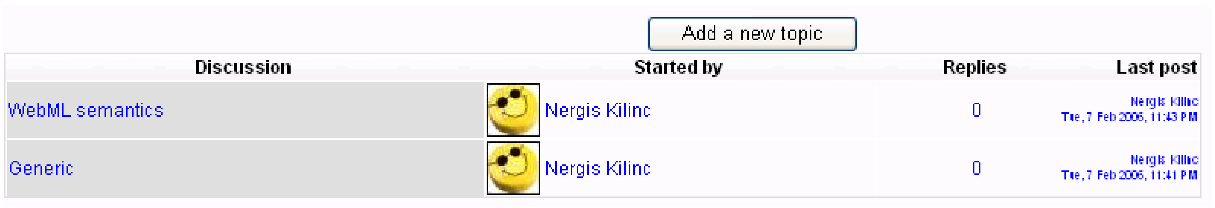
Grades ?

Student Sort by Lastname Sort by Firstname	Introduction to WebML		Data Model		Derivation Model		Data Model *1*		Hypertext Model		Hypertext Model *1*		Presentation Model		Term Examination		Parameters & Personalization Model		Design Process		Design Process *1*		Updates & Operations		Updates & Operations *1*		Total ↓↑ Stats	
	0	Raw %	0	Raw %	0	Raw %	100	Raw %	0	Raw %	100	Raw %	0	Raw %	10	Raw %	0	Raw %	0	Raw %	100	Raw %	0	Raw %	100	Raw %	410	Percent
Aksoy, Ali	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%
Yetginer, Seha	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%	-	0%

Figure 7.37 An example of a grade page showing a detailed score information about each student.

Forums

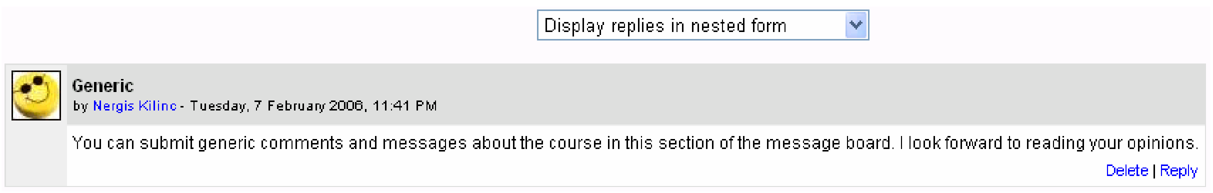
Two forums were added to the course as shown in Figure 7.38.


Add a new topic

Discussion	Started by	Replies	Last post
WebML semantics	Nergis Kilinc	0	Nergis Kilinc Tue, 7 Feb 2006, 11:43 PM
Generic	Nergis Kilinc	0	Nergis Kilinc Tue, 7 Feb 2006, 11:41 PM

Figure 7.38 Forums page of the course

First one is a generic forum which contains comments and messages about the course (Figure 7.39). Teacher and students may post their opinions and exchange views, discuss course events.


Display replies in nested form

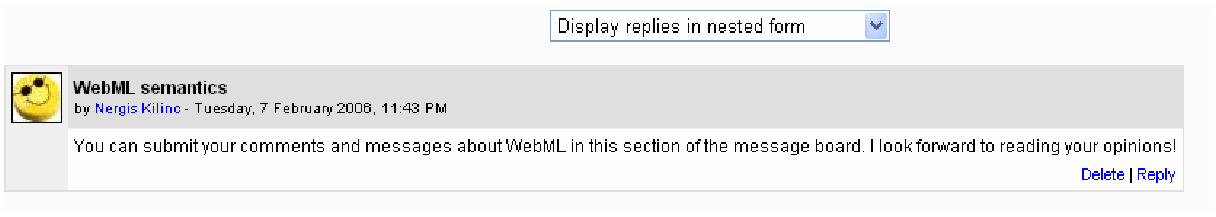
Generic
by Nergis Kilinc - Tuesday, 7 February 2006, 11:41 PM

You can submit generic comments and messages about the course in this section of the message board. I look forward to reading your opinions.

Delete | Reply

Figure 7.39 Forum page of the course named “Generic”

Second one has more specific content: WebML semantics (Figure 7.40). Teacher and students may post their opinions and exchange views about today and future of WebML as well as brand new ideas.


Display replies in nested form

WebML semantics
by Nergis Kilinc - Tuesday, 7 February 2006, 11:43 PM

You can submit your comments and messages about WebML in this section of the message board. I look forward to reading your opinions!

Delete | Reply

Figure 7.40 Forum page of the course named “WebML semantics”

8 CONCLUSION AND FUTURE WORK

In this thesis, two new concepts WebML and Moodle are discussed by teaching WebML and using features of Moodle.

WebML is a new alternative method for designing data-intensive complex Web sites. Learning Management System is also a new concept which changes the concept and habits of learning.

WebML gives a new point of view for people who wish to design data-intensive complex Web sites. It supplies an effective work style for technical personnel. Cost decrease, less effort satisfies both investor and technical personnel. Adaptation process of technical personnel is very quick.

Problems in the management and update of Web sites, as these grow in size and complexity are solved making a few operations using WebML.

Current education concepts do not restrict a specific location and time anymore. Because teacher-centred learning which restricts a specific location and time, student-centred learning is preffered which supplies a model for any-time and anywhere concept.

Just-enough learning and granular learning concepts of student-centred learning makes education life of students easier.

These two innovative concepts provide for introducing new ideas and generate new opinions about designing data-intensive Web site as well as awareness of Learning Management Systems which are going to be used more frequently in the future.

As a future work, the translation of WebML specifications into WML for multi-device Web sites can be done.

Unfortunately, Learning Management Systems are not used widely in Turkey. For this reason experimental study for this technology is needed to be given weight. An idea of extending

Learning Management Systems to mobile-devices would be a very practical way of e-learning.

REFERENCES

[1] Glossary of Terms: Student-centred Learning", (12.02.2006), http://www.mba.hobsons.com/glossary.jsp

[2] E-learning", (12.02.2006), http://en.wikipedia.org/wiki/E-learning

[3] Student-centred Learning", (12.02.2006), http://www.bath.ac.uk/e-learning/student_centredness.htm

[4] Brook, Chris and P. Bowen, "Economics, Politics and Learning; negotiating distributed learning in the print based learning environment", (12.02.2006), http://www.aare.edu.au/01pap/bro01653.htm

[5] "SCORM", (12.02.2006), http://en.wikipedia.org/wiki/SCORM

[6] "Learning management system", (12.02.2006), http://en.wikipedia.org/wiki/Learning_management_system

[7] Hall, J., "Assessing Learning Management Systems", (12.02.2006), http://www.clomedia.com/content/templates/clo_feature.asp?articleid=91&zoneid=29

[8] "AICC FAQ (Frequently Asked Questions)", (12.02.2006), http://www.aicc.org/pages/aicc_faq.htm

[9] "IMS Global", (12.02.2006), http://en.wikipedia.org/wiki/IMS_Global

[10] "Online Education and Learning Management Systems", (12.02.2006), http://www.studymentor.com/studymentor/

[11] "Open Source Course Management Systems", (12.02.2006), http://www.edtechpost.ca/pmwiki/pmwiki.php/EdTechPost/OpenSourceCourseManagementSystems

[12] "About Moodle", (12.02.2006), http://docs.moodle.org/en/About_Moodle

[13] "Philosophy", (12.02.2006), http://docs.moodle.org/en/Philosophy

[14] "Features", (12.02.2006), http://docs.moodle.org/en/Features

[15] "Case for Moodle", (12.02.2006), http://docs.moodle.org/en/Case_for_Moodle

[16] "Moodle", (12.02.2006), http://en.wikipedia.org/wiki/Moodle

[17] Ceri,Stefano, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera, "Designing Data-Intensive Web Applications", Morgan Kauffman Publishers, USA 2003

[18] Ceri,Stefano, P. Fraternali and A. Bongio, "Web Modeling Language (WebML): a modeling language for designing Web sites", (12.02.2006), http://www1.www9.org/w9cdrom/177/177.html

[19] Offutt, Jeff, Y. Wu, X. Du and H. Huang, "Bypass testing of Web Applications", IEEE International Symposium on Software Reliability Engineering (ISSRE 2004), November 2-5, 2004, Bretagne, France, IEEE Computer Society 2004

[20] Book, M. and V. Gruhn, "Modeling Web-Based Dialog Flows for Automatic Dialog Control", 19th IEEE International Conference on Automated Software Engineering (ASE 2004), September 20-24, 2004, Linz, Austria, IEEE Computer Society 2004

[21] Hassan, Ahmed E. and R. Holt, "Architecture Recovery of Web Applications", International Conference on Software Engineering (ICSE 2002), May 19-25, 2002, Orlando, Florida, USA, IEEE Computer Society 2002

[22] "The Web Modeling Language", (12.02.2006), http://www.webml.org/webml/page1.do

[23] "WebML in a Nutshell", (12.02.2006), http://www.webml.org/webml/page3.do?UserCtxParam=0&GroupCtxParam=0&ctx1=EN

[24] Brambilla,Marco, S. Ceri, S. Comai, M. Dario, P. Fraternali and I. Manolescu, "Declarative Specification of Web Applications exploiting Web Services and Workflows", ACM SIGMOD/PODS 2004 Conference, June 13-18, 2004, Maison de la Chimie, Paris, France, Association for Computing Machinery and Special Interest Group on Management of Data 2004

[25] Woukeu, A., L. Carr, G. Wills and W. Hall, "Rethinking Web Design Models: Requirements for Addressing the Content", Technical Report ECSTR-IAM03 -002, Department of Electronics and Computer Science, University of Southampton, Southampton, UK, 2001

APPENDIX: CD containing Thesis text, EasyPHP Set up, Backup of Moodle Course Files, PHP installer and Moodle Windows Installer