

**University of Oslo  
Department of Informatics**

**Dynamic &  
XML-Based  
Contents within  
E-learning**

**Fariba  
Emadi-Dehaghi**

**Master's thesis**

**Oslo- 2005**



**"Our best weapon against unrest and [war] is education!"**

Richard Larsen  
*(Director of Learning International Networks Consortium (LINC) at MIT)*

<b>Thesis subject:</b>
<b>Dynamic &amp; XML-Based Contents within E-learning</b>
<b>Master student:</b>
Fariba Emadi-Dehaghi ( <i>faribad@ifi.uio.no</i> )
<b>Supervisors:</b>
Dr Gerhard Skagestein ( <i>gerhard@ifi.uio.no</i> ) Dr Kjell Åge Bringsrud ( <i>kjellb@ifi.uio.no</i> )

# Contents

<b>1 Introduction</b>	<b>11</b>
1.1 Use of data & databases	12
1.2 Extensible Markup Language (XML)	13
1.2.1 Why using XML?	13
1.3 E-learning	13
1.4 Distributed Learning	14
1.5 Objective and scope of Thesis	14
1.6 Additional Notes	15
<b>2 Distributed Learning</b>	<b>16</b>
2.1 Distributed system	16
2.2 Distributed learning	16
2.3 "Distributed Learning" Technologies	17
2.3.1 Learning Objects Metadata (LOM)	18
2.3.2 Students or Customers for e-learning?	18
2.4 "Smart classroom"	19
<b>3 E-learning</b>	<b>20</b>
3.1 Background	20
3.1.1 Synchronized and asynchronized learning	20
3.1.2 Advantages of e-learning	20
3.2 Advanced Distributed Learning (ADL) Technologies	21
3.2.1 Learning Objects (LO)	22
3.2.2 Learning Management System	22
3.3 IEEE Learning System Architecture	23
3.3.1 Instruction Management System (IMS)	23
3.4 What is Classfrontier?	25
3.5 A LINC for e-learning	25
<b>4 Shareable Content Object Reference Model (SCORM)</b>	<b>26</b>
4.1 Background	26
4.2 Advanced Distributed Learning (ADL) Initiative	26
4.2.1 What is SCORM?	27
4.2.2 SCORM requirements	27
4.2.3 Learning Resources & Learning Content	29
4.2.4 Asset & SCO	29



4.2.5	SCORM Run-time Environment . . . . .	29
4.2.6	SCORM Content Aggregation Model . . . . .	29
4.2.7	High-level requirements for learning objects . . . . .	30
4.2.8	Reusable Learning Contents . . . . .	31
4.3	A learning content model for my <i>course</i> example . . . . .	32
<b>5</b>	<b>Database Technology</b>	<b>34</b>
5.1	Background . . . . .	34
5.2	Database design . . . . .	34
5.2.1	Object Definition Language (ODL) . . . . .	34
5.3	Database Management System (DBMS) . . . . .	35
5.3.1	DBMS capabilities . . . . .	35
5.3.2	ACID properties . . . . .	36
5.4	Database programming . . . . .	36
5.5	Database Modeling . . . . .	37
5.5.1	Entity-relationship modeling . . . . .	37
5.5.2	Relational modeling . . . . .	37
5.5.3	Constraints in data-models . . . . .	38
5.6	Use of relational database in my 'ODC' learning system . . . . .	38
5.6.1	Problem with having Text value in MySQL . . . . .	41
<b>6</b>	<b>Mapping UML Class Diagram Into XML Document</b>	<b>43</b>
6.1	Background . . . . .	43
6.1.1	Using UML . . . . .	43
6.1.2	Three level design approach . . . . .	43
6.2	Mapping UML to XML . . . . .	44
6.2.1	Conceptual level . . . . .	44
6.2.2	Logical level . . . . .	46
6.2.3	Physical level . . . . .	47
<b>7</b>	<b>Architecture</b>	<b>49</b>
7.1	Background . . . . .	49
7.2	Client-server architecture . . . . .	49
7.3	Three-tier towards N-tier Architecture . . . . .	51
7.3.1	Evolution towards Component-based architecture . . . . .	52
7.4	Component-based architecture used for LMS developments . . . . .	54
7.5	LTSA, the heart of LTSC's elearning model . . . . .	54
7.6	The system architecture of my trial <i>ODC</i> elearning course . . . . .	55
7.7	Detailed system architectures used for my trial <i>ODC</i> system . . . . .	56
7.7.1	Usage of JSP within my <i>ODC</i> system architecture . . . . .	56
7.7.2	Usage of XML within my <i>ODC</i> system architecture . . . . .	56
7.7.3	Usage of Flash MX within my <i>ODC</i> system architecture . . . . .	57
<b>8</b>	<b>Extensible Markup Language (XML)</b>	<b>58</b>
8.1	Extensible Markup Language (XML) . . . . .	58
8.1.1	What do HTML and XML do? . . . . .	58
8.1.2	XML concepts: element, node, attribute, parent and child nodes . . . . .	58

8.1.3	XML elements . . . . .	59
8.1.4	Document Object model (DOM) . . . . .	60
8.1.5	Comments in XML . . . . .	60
8.1.6	Well-formed XML . . . . .	60
8.1.7	Valid XML . . . . .	61
8.2	Classification of XML applications . . . . .	61
8.2.1	Document Applications . . . . .	61
8.2.2	Data Applications . . . . .	62
8.3	Manipulating XML text materials . . . . .	63
<b>9</b>	<b>Document Type Definition (DTD)</b>	<b>64</b>
9.1	What is DTD? . . . . .	64
9.1.1	DTD of my <i>test</i> learning object . . . . .	64
9.1.2	Hierarchical representation of XML Schema . . . . .	65
9.1.3	Semistructure modeling of my <i>Test</i> example . . . . .	65
9.2	An DTD application example . . . . .	68
<b>10</b>	<b>Extensible Style Sheet Language (XSL)</b>	<b>70</b>
10.1	What is XSL? . . . . .	70
10.1.1	One XML file for several XSL files . . . . .	70
10.1.2	Having comment in XSL . . . . .	70
10.1.3	XSL:choose, XSL:when , XSL:otherwise, XSL:for-each . . . . .	70
10.2	Namespace in XSL . . . . .	71
10.3	An XSL application example . . . . .	72
<b>11</b>	<b>Java Server Pages (JSP)</b>	<b>76</b>
11.1	Background . . . . .	76
11.2	What is JSP? . . . . .	76
11.2.1	The advantages of using JSP . . . . .	76
11.3	JSP & database connection within my trial 'ODC' system . . . . .	76
11.4	JSP & RDB & HTML . . . . .	77
11.5	Dynamic contents . . . . .	78
11.5.1	Providing dynamic content . . . . .	78
11.5.2	Dynamic contents within my <i>ODC</i> system . . . . .	78
11.5.3	Dynamic content provided by JSP & XML . . . . .	79
11.6	User interactivity & Feedbacks . . . . .	80
<b>12</b>	<b>Flash Macromedia</b>	<b>82</b>
12.1	Flash MX . . . . .	82
12.2	Why using Flash? . . . . .	82
12.3	Graphic standards: Vector & Raster . . . . .	83
12.4	Raster images . . . . .	83
12.4.1	Disadvantages of raster images . . . . .	84
12.5	Vector graphic images . . . . .	84
12.5.1	Advantages of vector graphic . . . . .	84
12.6	ActionScript: Flash programming language . . . . .	85
12.6.1	Using Flash MX . . . . .	85

12.6.2 Text types in Flash MX . . . . .	86
12.6.3 Publishing by Flash . . . . .	86
12.6.4 Flash MX invisible buttons . . . . .	86
12.6.5 Flash MX library elements . . . . .	86
12.7 Interactivity by using Flash macromedia . . . . .	86
12.8 Static Versus Dynamic Content . . . . .	86
12.8.1 Integrating Flash with XML . . . . .	87
<b>13 Integrating XML With Flash MX (Dynamic Text)</b>	<b>88</b>
13.1 Background . . . . .	88
13.2 Integrating an XML learning object with Flash MX . . . . .	89
13.2.1 Flash movies and Preloaders . . . . .	89
13.3 Developing process of my Flash movie . . . . .	89
13.3.1 XML data file for my Flash movie courseTests.swf . . . . .	91
13.3.2 My Flash movie courseTests.swf . . . . .	92
13.3.3 My Flash movie's ActionScript . . . . .	92
13.3.4 Publishing Flash movies . . . . .	96
<b>14 Dynamic Learning Contents by JSP &amp; RDB</b>	<b>97</b>
14.1 Background . . . . .	97
14.2 My developed ' <i>Online Database Course (ODC)</i> ' example . . . . .	97
14.2.1 System architecture of my <i>ODC</i> learning system . . . . .	98
14.2.2 The main ideas for my <i>ODC</i> learning system . . . . .	98
14.3 Content and Presentation Separation . . . . .	99
14.3.1 The <i>Create Test (JSP)</i> menu option . . . . .	99
14.3.2 The <i>Test Yourself (JSP)</i> menu option . . . . .	100
14.3.3 The <i>Ask Question (JSP)</i> menu option . . . . .	103
14.4 The development of my learning objects . . . . .	103
<b>15 XML Technologies for Development of Learning Materials</b>	<b>104</b>
15.1 Background . . . . .	104
15.1.1 The <i>Show Test (XML)</i> menu option of my <i>ODC</i> system . . . . .	104
15.2 Object-oriented (OO) modeling of the Test object . . . . .	105
15.3 A trial simulation of the SCORM system architecture idea into my <i>ODC</i> system . . . . .	105
<b>16 Flash MX for Developing Tutoring Materials</b>	<b>108</b>
16.1 My developed Flash MX learning objects . . . . .	108
16.2 Feedbacks and reporting . . . . .	108
16.2.1 <i>Show Exercises</i> menu option of my <i>ODC</i> learning system . . . . .	108
16.3 ActionScript, Flash programming language . . . . .	108
16.4 Flash MX animation & tutorial learning object . . . . .	112
<b>17 Summary and Conclusion</b>	<b>116</b>
17.1 Abbreviations . . . . .	122

<b>A</b>	<b>Source files of my ODC example</b>	<b>2</b>
A.1	Developed files by using XML technologies . . . . .	2
A.1.1	File: course-tests.xml . . . . .	2
A.1.2	File: test-answers-if.xml . . . . .	5
A.1.3	File: tests_graphic.xml . . . . .	7
A.1.4	File: test-answers.xml . . . . .	9
A.1.5	File: course-tests.xml . . . . .	11
A.1.6	File: test-answers.xml . . . . .	14
A.1.7	File: test-answers-if.xml . . . . .	16
A.1.8	File: tests_graphic.xml . . . . .	18
A.1.9	File: course.dtd . . . . .	21
A.1.10	File: course.xml . . . . .	22
A.1.11	File: course.xml . . . . .	23
A.1.12	File: course-tests.html . . . . .	25
A.2	Developed files by using JSP technology . . . . .	26
A.2.1	File: mailToTeacher.jsp . . . . .	26
A.2.2	File: continueMailToTeacher.jsp . . . . .	27
A.2.3	File: testYourself.jsp . . . . .	30
A.2.4	File: continueTestYourself.jsp . . . . .	31
A.2.5	File: evaluateAnswers.jsp . . . . .	33
A.2.6	File: createTest.jsp . . . . .	35
A.2.7	File: continueCreateTest.jsp . . . . .	37
A.2.8	File: registerQuestion.jsp . . . . .	40
A.3	Developed ActionScript codes of my Flash movies . . . . .	42
A.3.1	ActionScript of UML quiz (UML_Quiz fla) . . . . .	42
A.3.2	ActionScript of NIAM quiz (niamQuiz fla) . . . . .	42
A.4	ActionScripts used for the buttons of UML & NIAM quizzes . . . . .	43
A.4.1	ActionScript: Tire-knappen in UML_Quiz fla . . . . .	43
A.4.2	ActionScript: Driver-knappen in UML_Quiz fla . . . . .	43
A.4.3	ActionScript: Car-knappen in UML_Quiz fla . . . . .	43
A.4.4	ActionScript: Person-knappen in UML_Quiz fla . . . . .	44
A.5	ActionScript of alphabet quiz (quiz-alfabeth fla) . . . . .	44
A.5.1	ActionScript codes for the used buttons A, B, C, D, E . . . . .	44
A.6	Flash integrated with XML (Sec. A.6.1, A.6.2) . . . . .	45
A.6.1	ActionScript of the Flash movie (courseTests fla) . . . . .	45
A.6.2	XML file (tests.xml) . . . . .	46
<b>B</b>	<b>Layouts of my ODC example</b>	<b>47</b>
B.1	Layout files developed by XML technologies . . . . .	47
B.2	Layouts of the <i>Course info. (XML)</i> menu option (courseInfo.eps) . . . . .	47
B.3	Layouts of the <i>Show Test (XML)</i> menu option (No. 1, 2) . . . . .	48
B.4	Layouts of the <i>Show Test (XML)</i> menu option (No. 3) . . . . .	49
B.5	Layouts of the <i>Show Test (XML)</i> menu option (No. 4) . . . . .	50
B.6	Layouts of the <i>Show Test (XML)</i> menu option (No. 5) . . . . .	51
B.7	Layout files developed by JSP technology . . . . .	52
B.8	Layouts of the <i>Main</i> page (No. 1, 2) . . . . .	52

B.9	Layout of the <i>Menu</i> page . . . . .	53
B.10	Layout of the <i>Ask Question (JSP)</i> menu option . . . . .	53
B.11	Layouts of the <i>Create Test (JSP)</i> menu option (No. 1, 2) . . . . .	54
B.12	Layouts of the <i>Create Test (JSP)</i> menu option (No. 3, 4) . . . . .	55
B.13	Layouts of the <i>Create Test (JSP)</i> menu option (No. 5) . . . . .	56
B.14	Layouts of the <i>Test Yourself (JSP)</i> menu option (No. 1) . . . . .	56
B.15	Layouts of the <i>Test Yourself (JSP)</i> menu option (No. 2) . . . . .	57
B.16	Layouts of the <i>Test Yourself (JSP)</i> menu option (No. 3) . . . . .	57
B.17	Layouts of the <i>Test Yourself (JSP)</i> menu option (No. 4) . . . . .	58
B.18	Layouts of the <i>Test Yourself (JSP)</i> menu option (No. 5) . . . . .	59
B.19	Layouts of the <i>Test Yourself (JSP)</i> menu option (No. 6) . . . . .	59
B.20	Layouts of the <i>Test Yourself (JSP)</i> menu option (No. 7) . . . . .	60
B.21	Layouts of the <i>Test Yourself (JSP)</i> menu option (No. 8) . . . . .	60
<b>C</b>	<b>Flash MX learning examples</b>	<b>61</b>
C.1	My developed Flash MX learning objects . . . . .	61
C.1.1	The sub-menu of the <i>Show Exercise (Flash)</i> menu option . . . . .	61
C.1.2	Menu option: UML Quiz (.swf) . . . . .	62
C.1.3	Menu option: UML Quiz (.swf) . . . . .	63
C.1.4	Menu option: NIAM Quiz (.swf) . . . . .	64
C.1.5	Menu option: NNIAM Quiz (.swf) . . . . .	65
C.1.6	Menu option: Alphabet Quiz (.swf) . . . . .	67
C.1.7	Menu option: Question & Answers (XML by Flash) . . . . .	70

## Preface

This is my master's degree thesis that I have done at the Department of Informatics in the University of Oslo in Norway.

The subject of my thesis is "Dynamic & XML-Based Contents within E-learning". The recent extensive work on semistructured database e.g. XML was the main motivation for me for working on this subject. Use of semistructured database XML in an educational application is an interesting subject.

In general, we are still using the same old traditional educational system that has been used for many generations. In many modern societies, offering available and efficient services has become very essential and used by many businesses, as the only technique for surviving in the market. Educational services are of those kind of matters that should come more in focus and by using modern technologies they should be brought up to date.

A well designed educational system must be easy to use, available, efficient and effective. Therefore evaluating the E-learning system as a distributed system was another issue that I have tried to focus on in my work.

During my thesis research, I got the chance of receiving a precious guidance from my supervisors, Dr Gerhard Skagestein and Dr Kjell Age Bringsrud, professors at the Department of Informatics. I would like to express my appreciation for their excellent guidance during my work on this thesis.

I hope my experience presented in my thesis will be useful for the readers too and I hope in the future, I get the chance to continue my works that started like a little journey inside the huge science world.

Finally, I would like to thank my family for their support and encouragement that always made me more motivated to work harder to achieve my goals.

Fariba Emadi-Dehaghi  
Oslo- 2005

# Chapter 1

## Introduction

The development of computer-dependend technologies has recently changed our lives dramatically. In many modern societies, lots of traditional jobs have been replaced by various mechanical systems and computerized systems. Developing effective electronic systems for offering more efficient services, has become a crucial goal for many businesses. For instance, the traditional process of buying flight and train tickets, which usually demands its customers to contact directly their travel agencies, has been replaced by modern computerized systems and softwares. These tools are able to provide suitable on-line services that let people order and buy their tickets from anywhere in the world as long as a computer with Internet connection is available.

Computerized systems, specially the ones that provide on-line services, have become very popular because they make life easier and save considerable time, which according to the following saying "Time is Money and Money is Power", may be considered a valuable part of daily life.

24-hours on-line services such as Internet banking, which provide the possibility of doing financial transactions and depositing money for users, have become so popular that many other businesses have become encouraged to use Internet and offer on-line versions of their services, too. The growth of Internet has brought many services close to people but certain properties such as availability, fast access and user friendly interfaces have become essential for any on-line services.

Recent Web technologies and database techniques, such as Extensible Markup Language (*XML*), Semantic Web, Web and Data Mining and Data clustering are basic technologies used for developing widespread web-based systems, E-commerce, E-learning, E-government and E-science [5].

One of the main applications for XML as a new infrastructure for organizing data, is within the new type of business organizations, the so-called distributed businesses, which involves understanding other technologies such as distributed systems, semi-structured databases. These topics are discussed in more details in the next chapters.

In my thesis, I have focused on E-learning systems. E-learning provide study oppor-

tunities and lets students take on-line courses to complete a university degree. Educational on-line services provide access to learning materials and resources, anywhere Internet is available, without needing the students to be present at certain locations.

Having educational materials on-line is another advantage with such systems because then, they will be available for students at anytime, and they may be used at the same time by all the students and for any desired period.

## 1.1 Use of data & databases

The statement "Information is power" points to the value of information which is retrieved from stored data. On the issue of information, two basic concepts of *data* and *database* should be addressed.

*Data* is a concept for indicating some known facts with implicit meanings that can be stored. And *database* indicates any source of data that has a well-defined structure. However, in more formal debates, database is considered to be a collection of data which is managed by a database system ([19], P. 2).

*Database system*, or more precisely *database management system (DBMS)*, is a software program that as a tool, manages data in databases. The capabilities that DBMS provide are discussed in detail in a later chapter (Sec. 5.3.1).

Traditionally, the term database was used for well-formatted text data that was stored on paper. However, its modern definition indicates a collection of related data that is saved on a computer or a floppy disk. For instance, storing our friends' names, telephone numbers and addresses in an indexed address book or using a software such as Microsoft ACCESS or EXCEL to store our desired data on a personal computer or a floppy disk, are considered small personal databases that contain some related meaningful data.

The types of data, which can be handled by database systems, have changed dramatically. Traditional database systems were just able to work with textual and numeric contents while modern database systems are able to work with all kinds of multimedia contents such as text, images, sound and video-clips.

In modern societies, it has become a fact that business success is derived from technology, and use of these technologies is considered as an engine for driving many new businesses such as e-commerce, e-learning and NetBanking. In fact, many enterprises have become totally dependent on the database technologies<sup>1</sup> for retrieving and manipulating their businesses data.

---

<sup>1</sup>Many enterprises use database engine IBM's DB2. Access and manipulation of corporate data that resides in databases like DB2 that provides most of the added value of the connectivity between the business and its customers . . . . Businesses need to maximize the benefits realized from their database investments.(Maximizing the Business Benefits of DB2, BMC Software, Inc. @ 2001 Hurwitz Group, Inc.)



## 1.2 Extensible Markup Language (XML)

XML (eXtensible Markup Language) is a standard language that defines syntax for structuring information and provides a generic method for exchanging information between computers over the Internet.

The relational database technology is considered as the traditional database technology while XML technology provides semistructured databases. XML as a new common markup language, has been developed by World Wide Web Consortium (W3C)<sup>2</sup>.

### 1.2.1 Why using XML?

In 1996, XML was introduced as an "Internet language" that provides the possibility for structuring and storing desired data that later can be shared over the Internet [12].

XML as a tag based language looks similar to HTML. However, there are some differences between them, for example HTML is a presentation language while XML is a language used for storing content of data defined and structured by developer's own tags.

Later, other technologies such as XSLT (Sec. 10.1) and CSS can be used for extracting desired data from XML documents and formatting their layout.

## 1.3 E-learning

E-learning, considered as a component of distributed learning, is a Web-based tool that uses Internet for on-line representation of its digital content. These kinds of educational distributed systems are able to provide a nationwide or even worldwide infrastructure for electronic on-line learning called *e-universities*. On-line learning has usually a large user domain, namely the students from around the world [2].

The Web-based learning-management systems, known as *course management* or *virtual learning* are software-based systems designed to use the Internet for educational purposes and replace the traditional classroom systems. For instance, on protected university sites, professors may put their course materials on-line, post course notes and messages, create class calendars, provide links to important resources, and hold on-line discussions for students after the routine class hours. Such on-line services will be provided by having a direct connection to the Internet or to another computer, which is already connected within a network.

Use of multi-media facilities such as TV, video and e-books have recently become popular in modern societies for educational purposes. Despite many developments in this field, there are still many societies that are just using the traditional education systems, which we have inherited from our previous generations.

---

<sup>2</sup>W3C founded in 1994 by Tim Berners-Lee, is an international consortium of companies that are involved with the Web and Internet. The organization is responsible to develop and maintain Web technologies such as guidelines, standards and softwares. W3C defines standards for HTTP and HTML.

Treating education like an industry will involve some other terms and technologies such as distributed learning and e-business in addition to e-learning [23], which are discussed in more details in the following chapters.

As part of my thesis, I decided to develop an on-line course example called *ODC* (Chapters 14 & 15) as a very simple version of a Learning Management System (LMS) (Sec. 3.2.2) for learning and tutoring activities. For this purpose, I decided to deploy some new technologies such as Java Server Pages (JSP) (Chap. 11) and Extensible Markup Language (XML) technologies (Chap. 8, 9, 10). Moreover, creation of various learning objects was my other major purpose that led me to use Flash MX as a macromedia tool in creating Flash movies for educational purposes (Chap. 13, 16).

## 1.4 Distributed Learning

Internet technologies have recently provided an infrastructure for developing new systems for performing certain activities in a non-traditional way. For instance, e-learning and e-business technologies are some of the new Web-based tools that use Internet and provide distributed systems for doing both educational and economical activities on-line.

In the following chapters, I have discussed in detailed e-learning and using the critical concept **distributed learning** that provides learning possibilities from any where and at any time [42].

## 1.5 Objective and scope of Thesis

In my thesis, I have worked with the recent Web and semistructured database technologies such as XML, Extensible Style sheet Language (*XSL*), Document Type Definition (*DTD*), Semantic Web, Sharable Courseware Object Reference Model (*SCORM*), Java Server Pages (*JSP*) and Macromedia Flash MX to develop several learning and training examples. These technologies are represented in my "On-line Course" example that is my developed trial e-learning system.

In my thesis, the following technologies: extensible markup language (XML), extensible style sheet language (XSL), document type definition (DTD), java Server Pages (JSP) and Macromedia Flash MX, were deployed for developing various learning contents and functionalities. Learning and working with these technologies were considered as an additional challenge for my thesis.

Moreover, working with learning standards and learning system architecture used for developing learning management systems were studied in details. Last but not the least, integrating XML with Flash and mapping an UML class diagram into an XML document has been experimented.

As a result of my various implementations, a trial on-line course example called **On-line Database Course (ODC)** is developed. My *ODC* on-line course example considered as a mini, simple version of comprehensive learning management systems, is able to offer various learning related services.

The thesis covers the following topics:

- E-learning objects & System architecture:
  - Learning objects & Distributed learning systems (Chapters: 3, 2)
  - E-learning system architectures (Chap. 7)
  - Semistructured databases & SemanticWeb (Chapters: 6, 15, ??)
- E-learning standards & Technologies:
  - Learning objects & E-learning standards (Chap. 4)
  - Dynamic content, interactivity and learning objects (Chapters: 15, 16)
  - XML technologies, multi media technologies and learning objects (Chapters: 8, 9, 10, 15, 12, 13)
  - JSP, database systems & learning objects (Chapters: 11, 14, 5)

In addition, I have focused on deploying various technologies such as Java Server Pages (JSP), eXtensible Markup Language (XML), Document Type Definition (DTD), eXtensible Style Sheet Language (XSL) and working with relational databases (RDB) and using Structured Query Language (SQL).

This report contains also some explanations and discussions about the crucial issues and problems that I experienced during the development of my trial examples and my general understanding and evaluation of developments of Web applications and Flash MX Movies.

## 1.6 Additional Notes

As part of my thesis work, I developed an on-line course system called **Online Database Course (ODC)**, which is simple experimental learning system offering various learning, tutoring and testing learning objects.

In the writing process of my thesis, my experimental "learning system" explained in details in the chapters 14, 15 and 16, has been referred to in various names such as *Course example*, *ODC system*, *ODC "learning system"* and *ODC on-line course*.

In addition to my various developed learning objects, my *ODC* trial example offers various learning related services. The *ODC* on-line course is available at the following URL:

<http://jsp.ifi.uio.no:8080/faribad/>.

The full name of the achronyms used in the following chapters are listed in the section "Abbreviations" (Sec. 17.1), at the end of the thesis.

## Chapter 2

# Distributed Learning

### 2.1 Distributed system

Distributed systems are composed of interacting components of layered system architectures enabled to deliver various services. The dependencies and relationships existing between these services must explicitly be developed.

An original and ideal idea for developing a learning management systems (LMS) (Sec. 3.2.2) is based on modules that can easily be updated to newer versions without causing any impact on the larger system. In addition, the system must easily be able to be changed and customized to any institution. And the last but not least, the integration process of the e-learning system into the existing university infrastructure system must be inexpensive. However, making educational tools compatible with any university's computing infrastructure is a very demanding job.

### 2.2 Distributed learning

In order to develop new learning processes for teaching or training purposes, it is very important to consider the relationship between distributed learning, e-learning and e-business. In the following figure, I have used a Venn diagram to illustrate the relationship between these three concepts.

Distributed learning (Sec. 3.2) consists of both non-Internet like traditional teaching classrooms and Internet -enabled learning activities. **E-learning** as a subset of distributed learning consists of the Internet-based teaching and learning programs which are considered as non-business activities.

**E-business** offers various activities that are mostly not related to learning issues like online purchasing of goods. However, some e-business activities are considered as part of learning processes such as online registrations for study courses [42].

The recent advances in computing technologies have provided much higher-speed

processing and more attractive and user-friendly interfaces within web-based learning systems, which has made web-based learning-management systems more popular. For instance, at MIT, the number of professors who are using the e-learning system of the institute, Stellar, has increased from 50 in 2001 to 260 in 2004 [2].

## 2.3 "Distributed Learning" Technologies

By deploying asynchronous technologies (Sec. 3.1.1), learning contents and decision aiding materials can at anytime and anywhere be delivered and represented to students. These kinds of delivering and representing processes are dependent on certain computer technologies represented in the following figure, that are referred as "**Distributed Learning**" Technologies by ADL (Sec. 4.2).

ADL uses the "**Advanced Distributed Learning**" name for calling the technologies provided by combination of these technologies. As I have illustrated in figure 3.1, traditional computer-based instruction and interactive multi-media technologies combined with the new Web-based intelligent tutoring and simulation facilities are providing advanced capabilities for distributed learning systems ( [40], Sec. 1).

Using web-based learning systems make life easier for faculties and students but their maintenance are usually expensive and demands lots of IT personnel efforts. For example if a professor wants to update a particular function by writing a program or using another software, he/she has to hand-code a patch to get the new software to work with the existing system. But creating such patches are usually very expensive

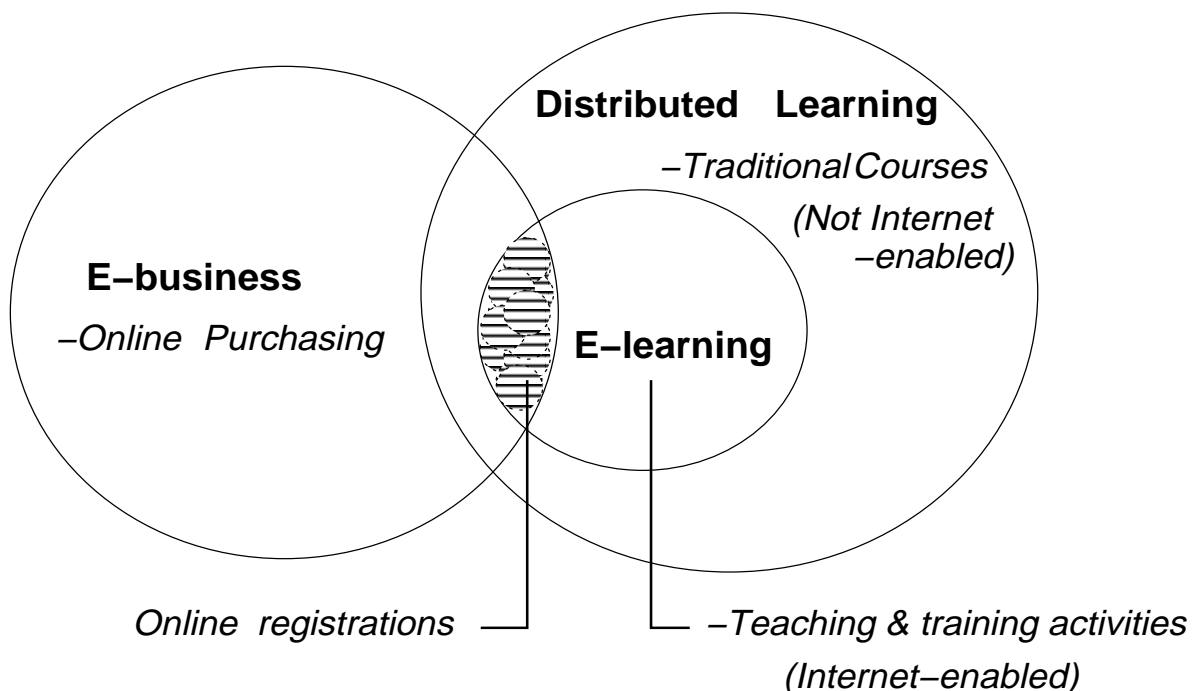


Figure 2.1: Relationship between Distributed learning, E-learning and E-business

and therefore, are not considered as a reasonable options [2].

### 2.3.1 Learning Objects Metadata (LOM)

The IEEE Learning Objects Metadata (LOM) standard is result of the cooperation between IMS Global Learning Consortium, Inc., ARIADNE and IEEE Learning Technology Standards Committee (LTSC) and is developed to provide the metadata specifications for learning objects and the relationships between them which can be used to provide exchange, reuse, and search of learning objects [40].

There are more than 70 attributes such as title, author, subject, defined by LOM draft standard that can be used for specification of learning objects. Although LOM includes an educational category, it does not contain any attributes for representing the quality and the role of learning objects within learning processes.

An approach for developing metadata standards for e-learning, suggests using instructional **role** for defining learning objects. The role concepts will provide a flexible way for specifying the various criteria prescribed by instructional theories. This *role* concept can be integrated with the role-concept of the object-oriented (OO) design for dynamic modeling of learning environment systems. Moreover, the role concept can be used for describing the instructional qualities of learning objects [1].

### 2.3.2 Students or Customers for e-learning?

Continued growth in knowledge and demand for increased productivity are encouraging people for having a lifelong learning. Therefore, many universities have become more interested to offer better learning opportunities to their students. On the

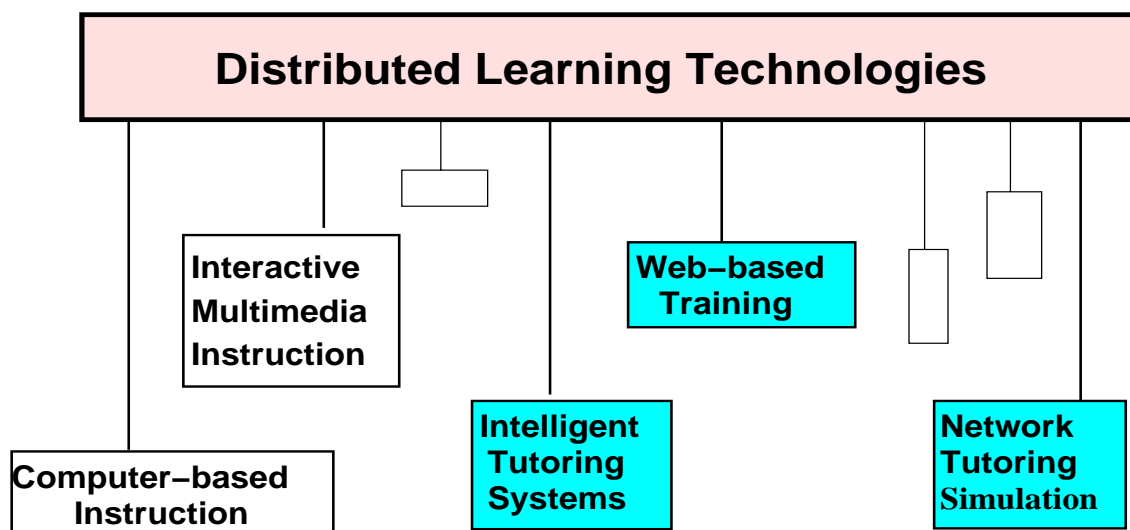


Figure 2.2: Computer-based delivery and presentation technologies

other hand, students encouraged to advance their knowledges, want to be treated as customers who want to choose higher-education institutions based on their learning possibility options and high-quality services. Recently, some institutes have also started to offer some of their administration process by on-line system. For instance, E-business is used for registering for courses or paying bills on-line.

## 2.4 "Smart classroom"

Professor Yuanchun Shi at the Tsinghua university in Beijing, China, uses an interactive distance learning tool called "Smart classroom" which displays the photos of those students who are logged in, on the wall of the classroom. The teacher uses a laser pointer to activate any of the student pictures to a live video and then they can answer any asked question.

Recently, the use and development of distance-learning classrooms has become much more in focus. By using several video cameras and a computerized camera coordinator in these kind of classrooms, the teacher's movement can automatically be followed. On the hand, every student's computer is able to exchange multimedia data by using its installed communication software, microphone and camera [23].

## Chapter 3

# E-learning

### 3.1 Background

New learning and training technologies are emerging to extend educational activities from the traditional classroom systems to the computerized systems, which can be accessed at home and workplace. Learning activities offered by computer-based and networked technologies have provided the possibility for every learner to extend his/her traditional school-ages to an entire life long learning.

There are now many academic institutions that are involved in development of the next-generation learning technologies for offering distributed learning by Web-based systems.

#### 3.1.1 Synchronized and asynchronized learning

With reference to ADL initiative, we can categorize different learning technologies into two main groups: *synchronous and asynchronous*. Synchronous learning technologies which are mostly based on video materials, are usually referred to as "**Distance learning**" that provide distance learning for students. Using synchronized learning materials requires that students be present together in one place and at a specific time. However, they don't demand any presence of educational instructors ( [40], Chap. 1). On the other hand, asynchronous learning technologies do not require bringing students in a specific place or at a specific time.

#### 3.1.2 Advantages of e-learning

E-learning systems and intelligent tutorial systems are intended to provide a tailored instruction and training for each student. Without computer-based learning systems, it will be almost impossible economically to offer individually tailored instruction to every student. Computer-based learning is supposed to provide the most individualized possible instruction at an affordable cost ( [40], Chap. 1).

Using multi-media in e-learning is another advantage that will improve the quality and effectiveness of distributed instructional systems.



## 3.2 Advanced Distributed Learning (ADL) Technologies

In synchronous technologies (Sec. 3.1.1), there are computer-based learning and training and multimedia facilities that are involved in providing "Distance learning". Figure 3.1 shows that in asynchronous systems, these technologies are combined with new Web-based technologies that are improving distributed learning into a higher level referred to as "**Advance Distributed Learning**" by Advanced Distributed Learning (ADL) Initiative (Sec. 4.2).

### *Distributed Learning (Internet-enabled)*

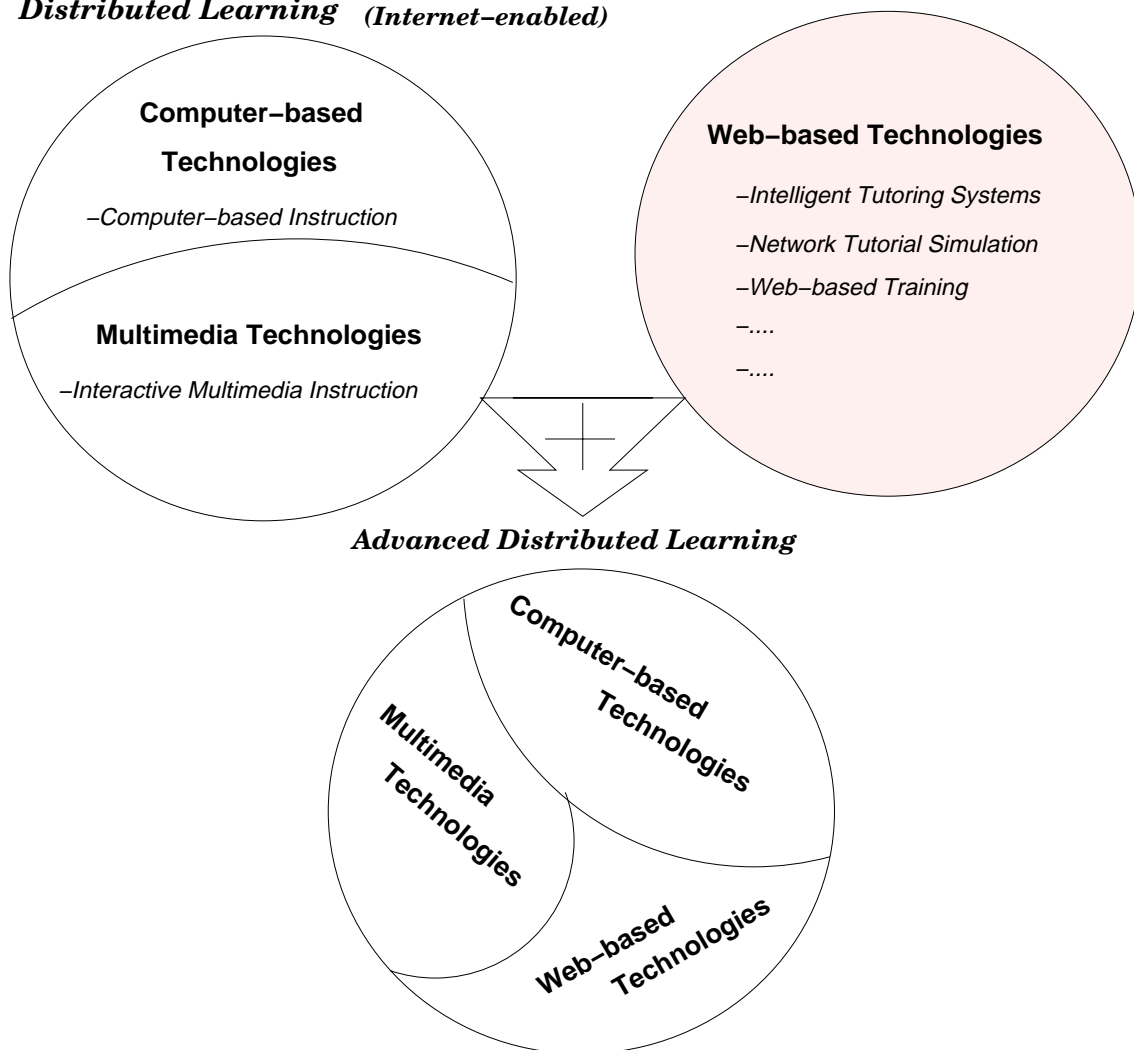


Figure 3.1: Advanced Distributed Learning Technologies

The following technologies are offering various learning activities.

- **Instructional technologies** such as computer-based learning and interactive multimedia instruction enable us to improve the efficiency of learning.

- **Intelligent tutoring systems** are enabled to provide tailored learning materials for every student.
- **Computer-based Instructions (CBI)** has the objective to automate learning & training processes into more efficient and effective ones while their costs are reduced.

ADL Initiative collaborates with many organizations particularly with IEEE, AICC, ARIADNE and IMS- Global Learning Consortium, Inc. that are working on specifications related to e-learning, standards, guidelines (Sec. 4.2.2) and common learning tools. The goal is to develop high-quality learning content "objects" for creating educational and training materials. Then the learning content objects that are searchable, reusable, shareable, durable and interoperable can be used across multiple platforms and within various learning environments (Fig. 4.2).

### 3.2.1 Learning Objects (LO)

Learning objects are the building elements used for constructing various instructional units and learning materials such as lessons, modules and courses. In order to reuse and exchange learning objects, the LOM standards (Sec. 2.3.1) provides metadata information for learning objects (Sec. 4.2.8).

In the same way that object-oriented (OO) modeling uses the *role* concept to provide more semantic information for the involved objects, a descriptive 'role' concept can be used within the LOM meta-model to provide more descriptive information for the learning objects. The 'role' term has a dynamic character that depends on relations and context. Therefore, by defining a standard definition, the term can be used to represent the instructional roles and qualities of learning objects. The roles of learning objects in various learning processes can be dynamically modeled [1].

By defining a standard attribute called *role* within LOM standard, the interactions between learning objects can be identified. Same learning objects may play different roles within different learning processes.

### 3.2.2 Learning Management System

Learning Management System (LMS) defines in a broad manner a set of server-side functionalities that are designed to control the delivery, tracking, managing learning content and reporting. In addition, LMS are made to manage learning content and student interactions and progress ([40], Book 1).

LMS is enabled to use the meta-data provided by Meta-data Assets (Sec. 4.3), to facilitate further sharing and reuse of Sharable Content Objects (SCO) (Sec. 4.2.4) utilized for creating learning contents within and across e-learning systems. Moreover, LMS uses the data provided by content packaging, to interpret the intended structure and sequences of learning resources for creating desired learning contents at the run-time. LMS is enabled to integrate with back-end systems, this indicates e-learning systems

can pass data to other applications within the same environment such as e-commerce system within an organization.

The following figure <sup>1</sup> illustrates the LMS Model defined by ADL (Sec. 4.2).

The LMS may contain other external systems such as *Human Resource* (HR), E-Commerce and *Enterprise Resource Planning* (ERP) that usually are developed based on component-based architecture (Sec. 7.3.1). The *Content Packaging* (Sec. 4.2.6) of IMS (Instruction Management System) that is called *Course Structure Format* by Sharable Content Object Reference Model (SCORM), provides the Meta-data for searching, selecting and packaging stored SCOs used for generating desired learning contents [17].

The *API Adapter* uses the Meta-data Asset of SCOs to provide the environmental requirements at run-time to generate learning resources and representing learning contents (Sec. 4.2.6). LMS is enabled to use the SCORM Run-Time Environment and track the SCOs within learning recourses ( [40], Chap. 2).

Using the component-based developing approach will be a proper architecture for developing SCORM-based learning management systems, which must be enabled to deliver various learning related services and exchange learning contents with other LMSs (Sec. 7.5).

The comprehensive business logic of LMSs can be categorized into various activities such as selecting, delivering and sharing learning contents, saving the records of students and exchanging learning materials with other LMSs. Then based on a N-tier system architecture, different system components can be developed to deliver various learning activities while they are able to communicate and cooperate with other components of the system (Sec. 7.3).

### 3.3 IEEE Learning System Architecture

The system architecture model provided as the most important model for e-Learning has been developed by IEEE learning technology systems architecture (LTSC) (Fig. 7.4). In the model, abstract system components are used to represent the system architecture for developing of learning systems (Sec. 3.2.2).

The model represent some highly abstracted system components, which during the development process of the business logic of learning systems, they will be designed into lower abstracted models (Sec. 7.5).

#### 3.3.1 Instruction Management System (IMS)

The Instructional Management System (IMS) is known as a course management system, or simply a learning server. IMS is a learning management system (Sec. 3.2.2)

---

<sup>1</sup>The figure 3.2 is fetched from this site:  
<http://aspen.ucs.indiana.edu/collabtools/imsadl1eejan01.ppt>

that is able to provide a capable and robust platform for supporting the registration, recording and reporting processes in addition to supplying and managing electronic commerce aspects for courses offered by e-learning systems [27].

The IMS Learning Resource Meta-data Information Model is developed based on the

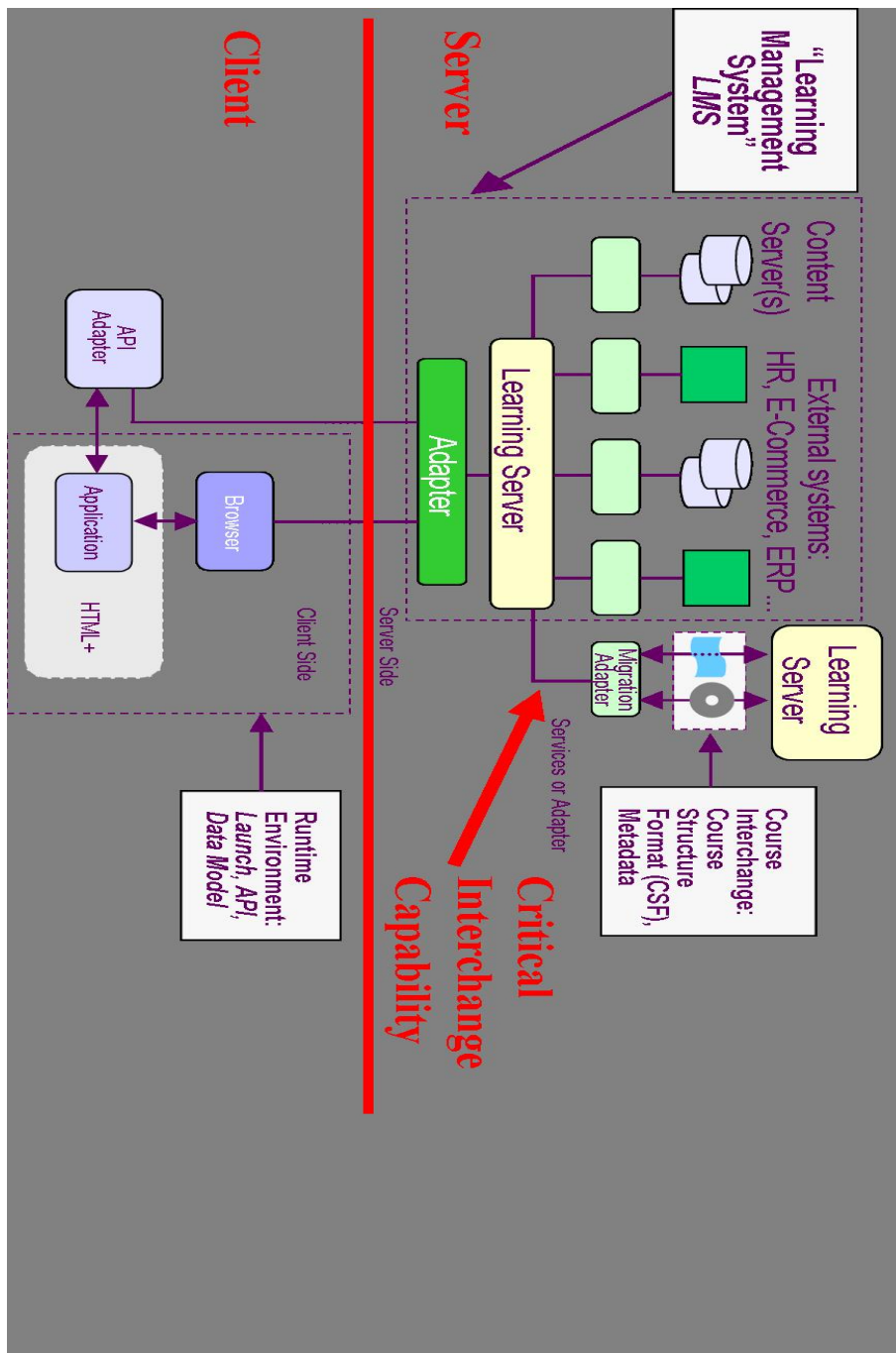


Figure 3.2: The ADL model for Learning Management System

LOM specifications (Sec. 2.3.1) developed by IEEE-LTSC [40]. The IMS specifications consisting of meta-data and content packaging data (Sec. 4.2.4), provide the possibility for describing, discovering and exchanging learning contents [25].

### 3.4 What is Classfronter?

*Classfronter* is a Norwegian Internet-based learning program used at the University of Oslo (*UiO*). This e-learning program considered as a learning management system, provides various on-line learning related services for students such as group project working, instructional supports, evaluation of learning activities and various communication facilities such as personal e-mail and calendar.

My personal experience with working a few times with Classfronter has led me to the following conclusions; the system provides various learning and communication services in one place. This is a convenient possibility for students because then they do not need to leave the system for example for checking their personal emails or calendars. In addition, the system offers are provided in both English and Norwegian languages. This advantage is considered as an important facility for international learning environments.

However, I believe by incorporating more color effects to its various user interfaces, the use of the system can become more attractive for students.

### 3.5 A LINC for e-learning

Using of e-learning technologies has recently become more popular among students because they can easily be accessed and they may be available almost anywhere at anytime. However, offering successful e-learning services by universities demands both high-speed Internet access and well-developed internal computer networks [4]. Moreover, compared to our "traditional" learning systems, e-learning systems to have much higher number of students makes it possible.

Richard Larson, the director of MIT's Learning International Networks Consortium (LINC) has recently started a project to create a flexible infrastructure system that will make e-learning become available to all Palestinian students who have much difficulty to reach their university campuses [4].

## Chapter 4

# Shareable Content Object Reference Model (SCORM)

### 4.1 Background

As a part of my thesis work, I have developed an on-line course example called *On-line Database Course (ODC)* (Chap. 14), which provides the possibility for me to work with various technologies and become familiar with some of the existing standards for developing learning contents and learning system architectures. During the development process of my trial example, I gave main attention to using and reflecting the ideas of various standards. The result of these work are presented in this chapter and the following sections 7.6, 15.3.

### 4.2 Advanced Distributed Learning (ADL) Initiative

In 1997, U.S Department of Defense (DoD) established the Advanced Distributed Learning (ADL) Initiative. ADL was supposed to promote the cooperation between government, academia and business for developing a standardization for e-learning contents and tools.

ADL has the intention to modernize education and training systems and standardize e-learning systems. To achieve this goal, ADL has developed the Sharable Content Object Reference Model (SCORM) specifications that reference a set of guidelines and interrelated technical specification for web-based learning contents. ADL has every learning contents to have certain high-level requirements (Sec. 4.2.2).

ADL Initiative collaborated with the members of Aviation Industry CBT (Computer-Based Training) Committee (AICC) to adopt the AICC Data Model, and to develop a common guidance for launching web-based data elements and developing common API specifications.

### 4.2.1 What is SCORM?

Sharable Content Object Reference Model (SCORM)<sup>1</sup> is a reference model that defines technical standards for web-based learning systems. SCORM is primarily written for e-learning systems' tool developers. Some technological groups like IEEE-LTSC<sup>2</sup>, AICC<sup>3</sup>, ARIADNE<sup>4</sup> and IMS<sup>5</sup>- Global Learning Consortium, Inc. are challenged to apply SCORM and produce a specific content model used for implementations that can be recommended to vendors [40].

Web-based E-learning systems equipped with SCORM provide better possibilities for finding, sharing, reusing, importing and exporting educational materials, and developing learning content models will become easier [40].

### 4.2.2 SCORM requirements

ADL expects that any web-based instructional material can also be delivered by any other instructional technology [40]. The idea is to create and develop some kind of libraries or learning repositories where learning objects may become accumulated, distributed and reused in different instructional purposes.

In the figure 4.1, I have tried to illustrate the long term vision of the ADL for distributed learning. The figure illustrates that the locally implemented instructional objects created based on the SCORM specifications can easily be shared and accessed across the World Wide Web By technical facilities, and based on a received request, the desired sharable content objects (SCO) (Sec. 4.2.4) are selected and assembled for creating a desired learning material for different environments and in real time<sup>6</sup>.

SCORM specifications need some high level requirements called "...lities" requirements (Sec. 4.2.7) for web-based learning objects. The high level requirements such as re-usability, accessibility, durability and interoperability, are called "...lities" requirements. Learning objects illustrated in the figure 4.1, are developed based on SCORM and contain the mentioned high level requirements that will promote use of technology-based learning systems [40]. The specifications and guidelines within SCORM are grouped into two main topics: "Content Aggregation Model" and "Run-time Environment" (Fig. 4.1).

---

<sup>1</sup>The previous SCORM was stood for "Sharable Courseware Object Reference Model". The replacement of "Courseware" with "Content" specified that the specifications can be applied to learning objects, which are smaller than an entire course.

<sup>2</sup>IEEE: Institute of Electrical and Electronics Engineers

<sup>3</sup>AICC: Aviation Industry CBT Committee

<sup>4</sup>ARIADNE: Alliance of Remote Instructional Authoring & Distribution Networks for Europe

<sup>5</sup>IMS: Instructional Management System

<sup>6</sup>Real-time normally means the maximum time which the operating system uses to respond to an external request. 'Real-time' also usually means 'fast' [6]

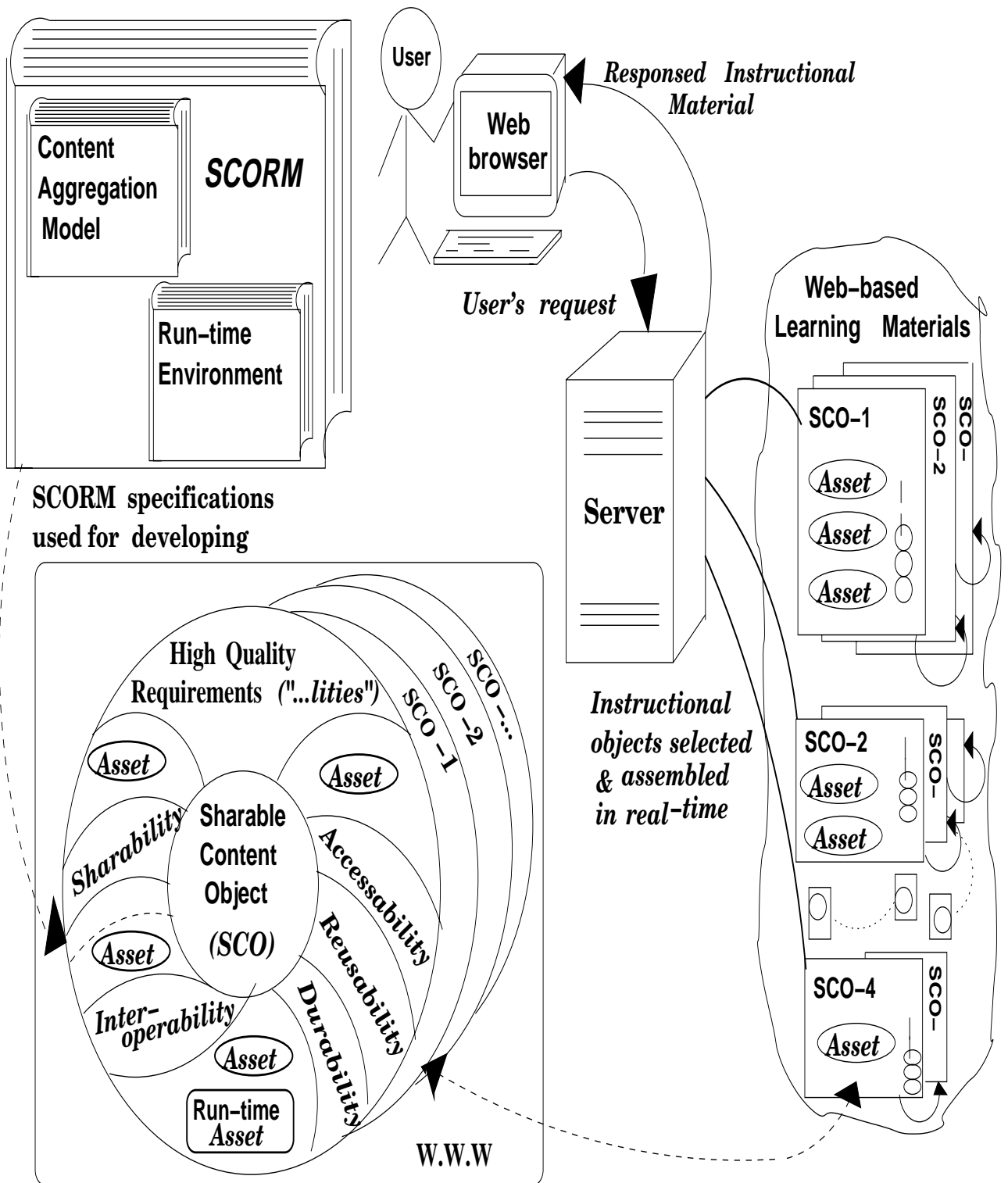


Figure 4.1: SCORM & Learning Material



### 4.2.3 Learning Resources & Learning Content

A *learning content*, also called *learning material*, is a relatively small set consisting of electronic or non-electronic learning resources that structure the learning content. Any presentation of any kind of information is considered a *learning resource* ([40], Chap. 2).

### 4.2.4 Asset & SCO

An electronic representation of any kind of data such as web page, media, sound, images and text is called an **Asset**. The term **SCO**, Sharable Content Object, refers to a collection of one or several Assets plus one launchable asset called **Asset Meta-data** which is responsible for deploying the SCORM Run-Time Environment (Sec. 4.2.5) to communicate with LMS (Sec. 3.2.2). On the other hand, LMS uses the SCORM Run-Time Environment to track the SCOs within learning recourses ([40], Chap. 2). **Meta-data** as a descriptive information provides necessary data for searching and delivering of content.

A SCO composed of Assets has the smallest logical size of content that at run-time can be tracked by LMS. All used Assets within a SCO are described within Asset Meta-data used for the searching and discovering processes within on-line repositories.

### 4.2.5 SCORM Run-time Environment

Learning environment provides the technical supports for learning systems to access and use learning objects. The specifications and guidance for launching learning objects and for tracking and communicating with learning content in a web-based environment are described in the SCORM Run-time Environment (Fig. 4.1).

The SCORM Run-time Environment contains the guidance for launching and communicating with learning data elements. In addition, it contains the specifications for tracking learning objects in a web-based environment [40].

### 4.2.6 SCORM Content Aggregation Model

*Content aggregation Model* of SCORM pedagogically provides the possibility for developers to get together/aggregate learning resources, in order to deliver a desired learning content. The SCORM Content Aggregation Model defines a content model for Web-based learning objects and contains the following specifications for identifying learning objects and aggregating learning resources into a structured learning content ([40], Chap. 1).

In figure 4.2, I illustrate the process of creating and delivering learning contents that involve the creation, selecting and assembling of simple assets into more complex learning resources that may be shared and reused (Sec. 4.2.8). Content aggregation Model consists of the following three components:

- **Content Model** that describes the SCORM components for a learning content consisting of reusable learning resources. Content Model itself consists of:
  - **Asset** that denotes every piece of data used within a learning content
  - **SCO** is a set of Assets plus one asset that is launchable and utilizes the SCORM Run-Time Environment for communicating with LMS.
  - **Content Aggregation** or Content specified by its name, represents the structure of a learning content. Content aggregation used as a map, for aggregating learning resources into a learning content such as a chapter, a course, or a module.
- **Meta-data Asset** describes the instances of learning resources belonging to a learning content. The SCORM Meta data uses IMS Learning Resource Meta-data Information Model (Sec. 2.3.1).
- **Content Packaging** is a mechanism for binding SCOs to SCO Meta-data to represent the behavior of a learning resources and represents how learning resources can be packaged for different environments (Fig. 3.2). The SCORM content packaging contains the complete requirements needed for providing SCORM-'interoperability' for learning contents.

#### 4.2.7 High-level requirements for learning objects

ADL requires that learning content objects must be designed to have certain high-level requirements such as the ones listed below. These kinds of requirements will make development and usage of e-learning contents more efficient in cost and efforts.

- **Reusability:** By this property, SCOs (Sec. 4.2.4) can be used and modified by various software development tools. A learning content may be developed by different development tools but it must have the same interface and data scheme within different environments (Sec. 4.2.8).
- **Searchability:** The structure of learning objects will provide easy and efficient data navigation within learning materials.
- **Sharable:** A SCO is the smallest and trackable unit that can be shared and reused to compose various learning recourses.
- **Accessibility:** SCOs can be indexed and become searchable and selectable. It means that they can be accessed by the request of learning softwares.
- **Durability:** SCOs can tolerate any changes and modifications that may happen during the development of new versions of a software system.
- **Interoperability:** SCOs are flexible units that can operate in computer-based learning systems that may be run by different hardwares, operating systems and web-browsers.

### 4.2.8 Reusable Learning Contents

Having reusable SCOs requires their independency from the context of learning contents. The information used for searching and discovering a SCO within an online

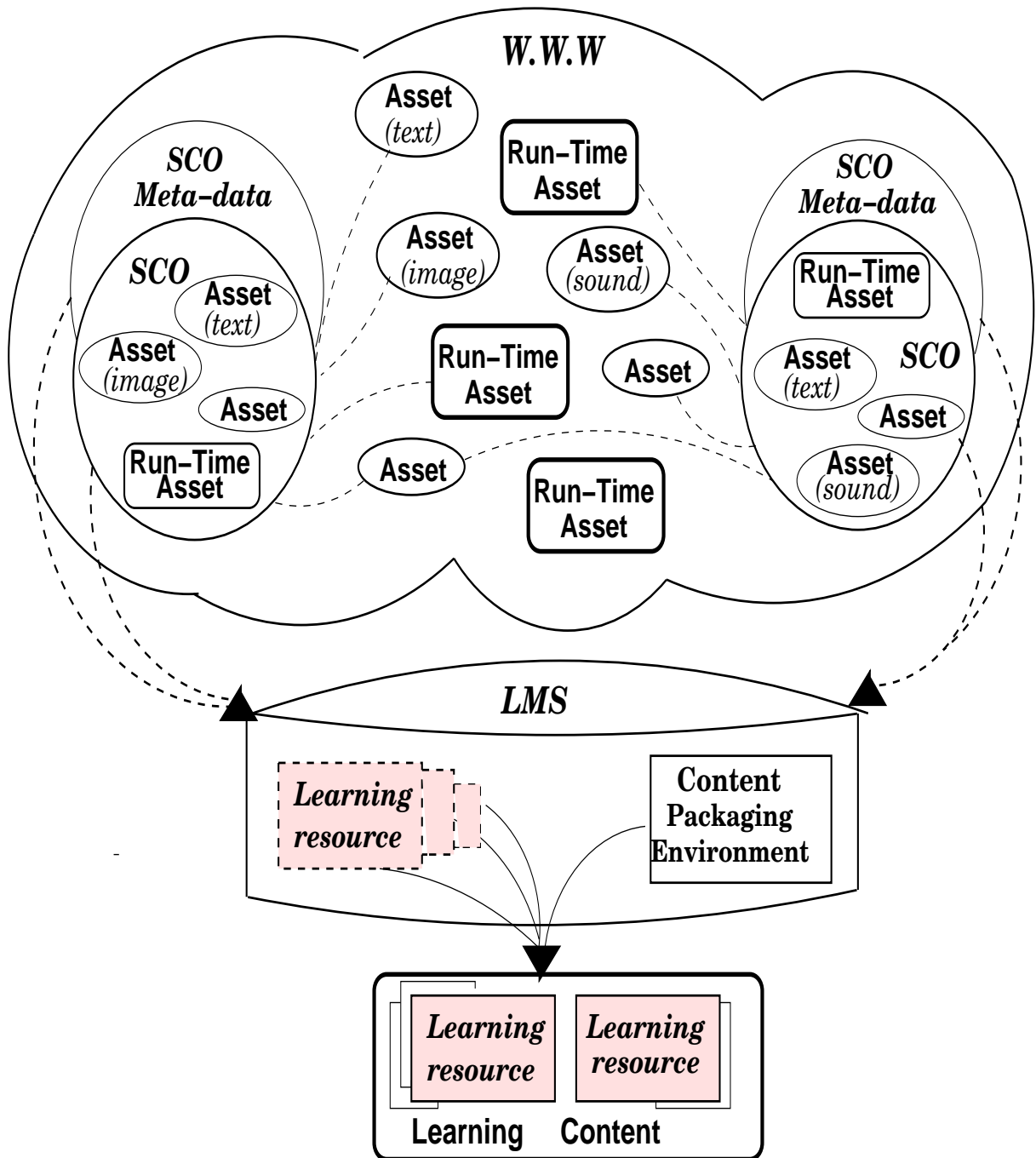


Figure 4.2: SCO & Asset within Learning Experience & LMS

repository, is provided within its Meta-data Asset that describes the content of SCO and the way it can utilize the SCORM Run-Time Environment to communicate with LMSs.

Learning resources will be organized into a predefined sequence of delivery (Fig. 4.1), and because they do not have any context within any learning resources, they do not belong to any specified learning content. The sequence of learning resources used within a learning content will be specified by the rules defined within the content package of an aggregation model that will be used and interpreted by a LMS. Content Packaging provides the information needed for packing learning resources and meta-data and interchanging them between different learning management systems. Therefore, learning resources can be reused in various aggregation contexts by the rules and attributes defined for sequencing and navigating the learning resources.

### 4.3 A learning content model for my *course* example

By the UML class diagram modeled in figure 4.3, I illustrate the SCORM concepts and specifications of learning content within a *course* example.

- **Course** : Every *Course* consists of at least one *Lecture* and contains all instructional procedures belonging to a subject offered by an e-learning system.
- **Lecture** : A *Lecture* contains learning's experiences that are supported by electronic learning resource.
- **LearningContent** : Learning content contains the teaching material for every *Lecture* object.
- **LearningResource** : Learning resources are the instruction units used for constructing learning contents such as chapters, courses, modules, etc. LMS uses the content packaging as a map that represents the sequence of learning resources for creating desired learning contents at the run-time.
- **SCO** : Sharable Content Objects consist of one or more assets. SCOs are reusable because their contents are independent of the learning contexts. The size of a SCO is equal to the smallest logical size of a content, which means the lowest granulate of teaching material within *learning resources*. Every SCO has its own Asset Meta-data.
- **Asset** : It is an electronic representation of pieces of data, for instance text, images, sound, media, web pages, assessment objects, etc. The most basic form in learning content is composed of assets. Every asset has a *type* attribute that specifies the form of the asset.
- **AssetMetaData** : Every SCO has its own Asset Meta-data that provides a descriptive information about the content of SCO and is enabled to utilize the SCORM Run-Time Environment to communicate with LMSs. The information of Asset Meta-data is used for searching and discovering a SCO within an online repository.

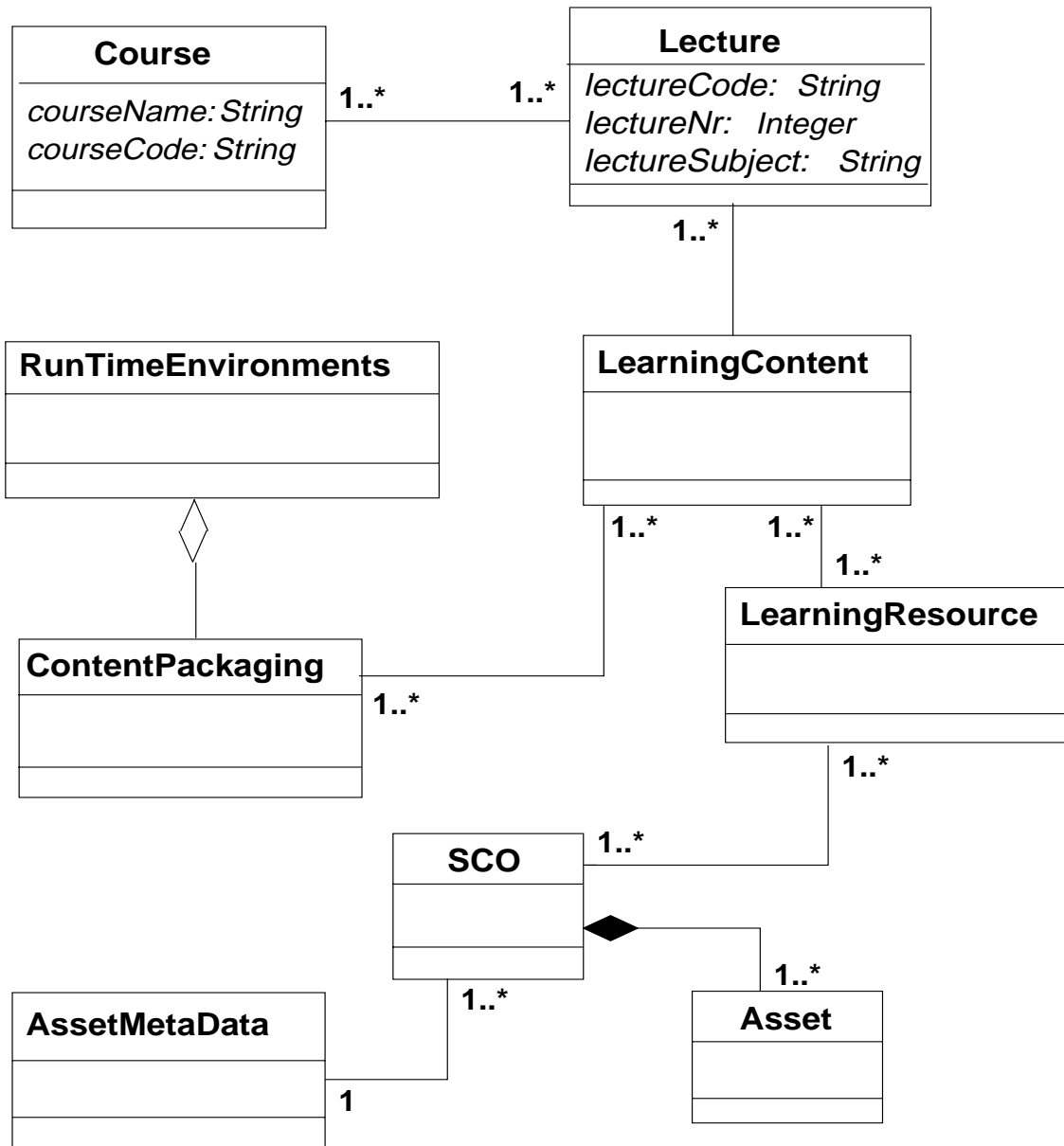


Figure 4.3: The SCORM specification within an UML class diagram

- **RunTimeEnvironment**: Run-time Environment objects provide the information about launching the *Learning Contents* offered by the *Course*. It also contains the information for tracking and reporting back every learner's progress.
- **ContentPackaging**: The content packaging objects contains the meta-data which describes the content aggregation model for a Learning Content object. Moreover, the sequences of learning resources of learning contents and the run-time requirements and properties are provided by content packaging objects.

## Chapter 5

# Database Technology

### 5.1 Background

Database technologies are used for providing collections of information from recorded data. For instance, depositing or withdrawing money at a bank, making a flight or hotel reservation and searching for a book on a computerized library are involving some sort of interacting with various database systems.

For developing the functionalities *Create Test* (Sec. 14.3.1) and *Test Yourself* (Sec. 14.3.2) of my *Online Database Course (ODC)* on-line course example (Chap. 14), the relational database technologies such as MySQL database and Structured Query Language (SQL), were used for storing various multiple choice questions used for creating learning test materials.

In this chapter, some of the crucial database concepts that are generally involved in developing distributed systems, such as e-business and e-learning systems, are briefly represented.

### 5.2 Database design

The process of designing a database starts with choosing the kind of information that is interesting to be recorded for future use. In the next step, the components of the desired information and the existing relationships between them must be discovered. Then, the chosen information that can be stored will be recorded as the *data* within a database, with a paper or electronic format.

#### 5.2.1 Object Definition Language (ODL)

In order to specifying the structure of object oriented databases, the standardized language Object Definition Language (ODL) has been developed. ODL makes it possible to have an object-oriented (OO) and high-level description for OO databases. The role that ODL plays for object-oriented database (OODB) systems is similar to the one that Document Type Definition (DTD) does for Extensible Markup Language (XML) documents (Sec. 9.1). Both of ODL and DTD, are defining the legal element structure and concepts for database systems.

## 5.3 Database Management System (DBMS)

A powerful tool called *Database Management System (DBMS)*, or just *database systems* provides the possibilities for storing, manipulating and retrieving data within a database. DBMS as an efficient system, is able to manage databases safely for a long period of time.

Traditional database management systems are able to manage data that are stored in the form of persistent data sets. In another word, database systems are typically used for storing persistent "objects", which are enabled to survive various database processes used to manipulate them [29], P. 319)

DBMS as the power of database, acts similarly to file systems, which are able to save and manage lots of files at the same time. However, the data manipulation process within database systems are usually much more advanced and complicated compare to the ordinary file reading and writing processes within file systems.

In addition, well data structured databases are providing more efficient and logical data accessing to lots of stored data. Well designed database architectures can assure having less data redundancy and more data connectivity within a database system. Moreover, it will increase the capacity of databases and provide more capabilities for doing more efficient queries within a database.

Therefore, by having efficient data structure within a database system, the system will become more flexible and having more efficient data accessing will be supported. For instance, relational database management systems (RDBMS) such as MySQL and Oracle, provide optimize controlling and manipulation of data stored within relational databases.

### 5.3.1 DBMS capabilities

Database systems support storing a large amount of data that later can be accessed and manipulated by applying a powerful query language such as SQL or Object Query Language (OQL). Moreover, DBMS uses ACID properties (Sec. 5.3.2) to support simultaneous *transactions*.

As mentioned earlier, the capabilities of database systems can be categorized into the following:

- **persistent storage:** DBMS provides recording huge amount of data that can exist independently of any processes that are using them.
- **programming interface:** DBMS provides the facilities for users and applications to access and modify the recorded data in a database.
- **transaction management:** DBMS allows same recorded data to be accessed at once by many different database processes, as long as the ACID properties are supported ( [19], P. 17).

### 5.3.2 ACID properties

The transaction manager of every DBMS controls the accessing and manipulating processes that are applied to the recorded data in a database. Same recorded data at the same time can be accessed and used by various database processes when the involved transactions are able to meet the **ACID** test, which stands respectably for the following properties, represented in the table below ([19], P. 14).

Property	Description
"A"- Atomicity	Every database transaction is executed either completely as a whole or not at all.
"C"- Consistency	All database transactions should preserve the consistency rules and constraints defined on the data and relationships of a database.
"I"- Isolation	Database transactions should be performed based on the "One-at-a-time" principle, which specifies every transaction might be executed separately from the other database processes. Therefore, the whole database will be belonged to one of the transactions, which are using the same database data.
"D"- Durability	The effect of completed transactions on a database, will last for ever as long as there are no other database processes applied in the database that may have changed the previous effected data. This ability will recover the completed transactions from any failures and errors that might happen within a database.

Table 5.1: ACID-properties applied to database transactions

## 5.4 Database programming

Most of the business applications that require data manipulation within a database are developed by a database programming that is supported by all DBMS. The recent *fourth-generation language (4GL)* programming technique (Fig. 5.1 on the facing page) consists of deploying a database programming language with its various supporting tools that are able to provide proper environment for database programming ([38], P. 183).

*Database programming* uses a specialized language for providing a database connection and performing various database operations such as retrieving and manipulating data. The supporting environment for a database programming language provides the facilities for *numerical computations (spreadsheets)*, creating *user interfaces* and generating *reports*. Most recent business applications are developed based on an *evolutionary* development process, which as a standard development technique, has been applied to a specified business domain.



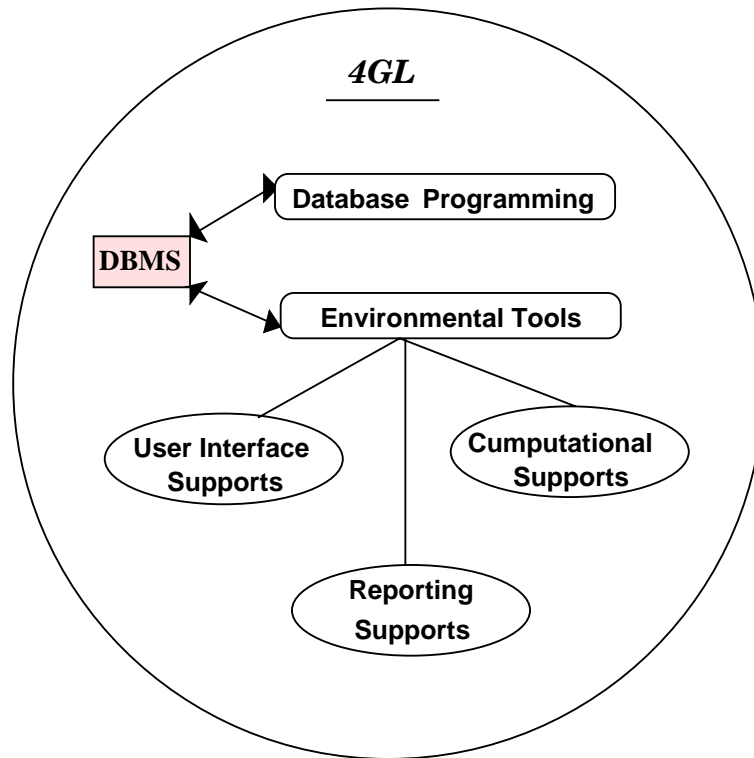


Figure 5.1: The component structure for Fourth-Generation Language applications.

## 5.5 Database Modeling

A data model should be designed carefully because well-designed data models will provide the possibilities of answering user's queries in shorter time. Data should be structured and organized based on a *data model* called *database schema*. There are different approaches that may be used for data modeling such as *Entity-Relationship* (ER) model and *relational* model.

### 5.5.1 Entity-relationship modeling

An ER model is a diagram consisting of abstract objects called entities, their associated properties called attributes and their connections called relationships, which will all together be used to present the structure of a database ([19], P. 23-25).

### 5.5.2 Relational modeling

In 1970, Ted Codd suggested relational presentations for database systems<sup>1</sup>. Today in most database implementations the relational modeling, which provides a set of two-dimensional presentations called *tables* or *relations*, is used.

<sup>1</sup>Codd wrote the following famous paper, which caused big changes within database systems. Codd, E. F., "A relational model for large shared data banks", Comm. ACM, 13:6, P. 377-387.

Data will be stored in divers relations, which later can be manipulated and retrieved. Stored data can be used to relate different relations to each other.

Relational databases used by database systems, which are able to receive queries from user side and retrieve desired answers from the databases. Then relations may be used for viewing the result data. Every relation will contain *records*, which are rows in a database, in turn are consisting of *fields* called *attributes*. Every field will have its own *name* and *value*. Moreover, all records in a table will have the same number of fields which match their field names ( [19], P. 61-63).

### 5.5.3 Constraints in data-models

Real world's "objects" may have some crucial aspects that are important to be considered although their representation in a data model might be difficult. Therefore, these kinds of extra information that talk about the type and structural aspects of the involved "objects", are defined as *constraints* on the data.

**Relational data-model** provides defining constraints and doing operations on the table attributes, tuples and on relations as a whole. Constraints, which define interrelational functionality for a database, are called "assertions" <sup>2</sup>.

*Primary key* and *foreign-key*, which are respectively integrity and referential-integrity constraints on the data stored in a relational database, are some of the SQL's techniques and expressions that will function as a part of the database schema. A primary key may consist of more than one attribute and its value(s) should be unique among the records of the table. Moreover, having a *NOT NULL* specification for an attribute, defines that the existence of that attribute value is demanded.

**Foreign key** as a constraint is defined between two separate relations to specify that the value(s) of one or more attribute(s) from each of the relations, should be exactly the same as each other ( [19], P. 47, 315). **Constraints** may be used for defining business rules and properties, which are usually very essential matters in people's lives, therefor they should be considered carefully in database programming.

## 5.6 Use of relational database in my 'ODC' learning system

By using SQL statements a 'MultiplechoiceQuestions' table is created where various multiple choice questions can be registered inside my MySQL, relational database (RDB) (Sec. 5.5.2). A 'correct answer' attribute is defined to be an enumeration defined as **enum**, which defines the four accepted attribute values which can be chosen as the correct answer of the created question.

Below, the SQL statements used for developing my 'MultiplechoiceQuestions' table

---

<sup>2</sup>More details are available on chapter 7, [19]

in addition to the SQL statements used for inserting value items into it, are presented.

```
create table MultiplechoiceQuestions(
  id      int unsigned not null AUTO_INCREMENT PRIMARY KEY,
  testNr  char(3) not null ,
  subject varchar(40) not null,
  question TEXT not null,
  answer1 TEXT not null,
  answer2 TEXT not null,
  answer3 MEDIUMTEXT,
  answer4 MEDIUMTEXT,
  correctAnswer enum ('1','2','3','4') not null
);
```

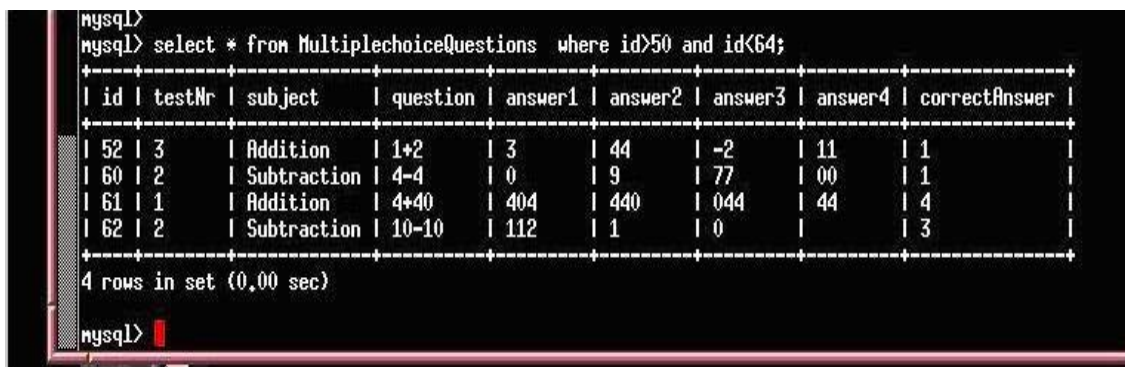
----- Inserting SQL statements: -----

```
insert into MultiplechoiceQuestions values
('','1','Addition','2+3','1','5','23','',' '2');
insert into MultiplechoiceQuestions values
('','1','Subtraction','15-13','10','5','3','2', '4');
```

Figure 5.6: The SQL table creation and inserting statements for storing multiple choice questions.

The data structure used for my 'MultiplechoiceQuestions' table lets each question have one correct answer. The following table 5.2 generated within MySQL database, represents the fetched results for the following SQL query statement:

```
SELECT * FROM MultiplechoiceQuestions
WHERE id>50 AND id<64;
```



```
mysql>
mysql> select * from MultiplechoiceQuestions where id>50 and id<64;
```

id	testNr	subject	question	answer1	answer2	answer3	answer4	correctAnswer
52	3	Addition	1+2	3	44	-2	11	1
60	2	Subtraction	4-4	0	9	77	00	1
61	1	Addition	4+40	404	440	044	44	4
62	2	Subtraction	10-10	112	1	0		3

```
4 rows in set (0.00 sec)

mysql>
```

Figure 5.2: The result table produced by running the SQL statement within my MySQL database.

The functionality process of the *Test Yourself* menu option of my trial ODC example (Sec. 14.3.2), is started by connecting to MySQL database in order to execute

the needed SQL queries (Sec. 11.3).

Further, by creating an object of the *Statement* class predefined in JSP (Sec. 11.2) various desired SQL queries can be defined for fetching desired data from the database. The following code example from my *testYourself.jsp* source file (App. A), represents the defined SQL query used for fetching saved test numbers, *testNr*, from the relational database table *MultiplechoiceQuestions* (Sec. 5.6).

```
SELECT  DISTINCT testNr
FROM    MultiplechoiceQuestions
ORDER  BY testNr;
```

The used **distinct** keyword supplies having no repetition within the fetched data values. Later, the fetched test numbers are represented within the following HTML-form used for starting the functionality of the menu option *Test Yourself* (Sec. 14.3.2).

The following figures represents some distinctly fetched *test numbers* and *subjects* fetched from my relational database used for starting the process of *Test Yourself* functionality in my trial *ODC* on-line course. Then, by using the predefined *executeQuery* method, the defined SQL query will become executed into the RDB (Sec. 11.3). In the following example, which is used within my JSP source file *testYourself.jsp* (App. A), a statement object is created to execute my desired SQL query.

The screenshot shows a web interface for an Online Database Course (ODC). The main heading is "Test yourself!". Below this, there is a prompt: "Please choose a specific test number and/or subject that you want to practice on. Multiple choice questions will be fetched after your request." There are two dropdown menus: "Tests:" and "Subjects:". The "Tests:" dropdown is open, showing a list of numbers from 1 to 9. Below the dropdowns, there is a section titled "Question" with two buttons: "Show the questions!" and "Remove the choices!".

Figure 5.3: Various *test numbers* fetched from the RDB

```
Connection con1 = DriverManager.getConnection
    (mysql_base, user_name, password);
```

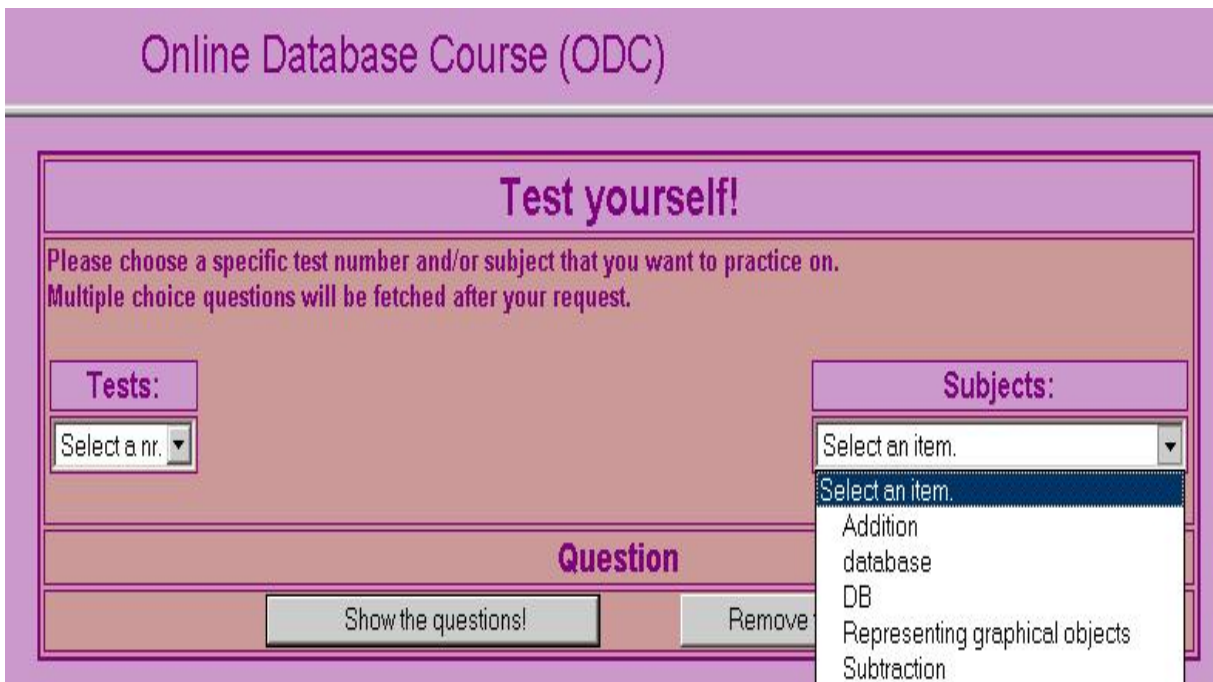


Figure 5.4: Various *test subjects* fetched from the RDB

```
Statement st1 = con1.createStatement();
String query1 = " select distinct testNr " +
               " from MultiplechoiceQuestions " +
               " order by testNr ";
ResultSet res1 = st1.executeQuery(query1);
```

### 5.6.1 Problem with having Text value in MySQL

Having long text values for certain attributes of a *question* registered into the MySQL DB, will make the presentation of its table become difficult-to-understand. And therefore, manual updating the stored values of a table, will become a difficult and complicated process. For instance, some of the attributes of my *MultiplechoiceQuestions* table created within MySQL DB (Sec. 5.6), are defined to contain *text* values as the type of their values. Text values can be consisted of several text lines and line feeds, which will make their presentation within a table difficult-to-understand. Below, a few improper overview examples of my *MultiplechoiceQuestions* table are illustrated.

The following figure 5.5 represents a result table created within MySQL database. The text values of its attributes do not contain any line feeds. Therefore, the presentation of the table is easy-to-understand.

However, the following figure 5.6 represents the same result table while the fetched tuples (Sec. 5.5.3) are containing line feeds within their large text attribute values. Having so many line feeds within the text attribute values will make their table presentation difficult-to-understand. Therefore, the manual manipulation of table data within

```
mysql>
mysql> select * from MultiplechoiceQuestions where subject="DB" and testNR=8;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | testNr | subject | question          | answer1          | answer2          | answer3          | answer4          | correctAnswer |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 74 | 8      | DB     | ODBC stand for? | Online DB Course | Open DB Connectivity | OBS! DB Continuerly | Object DB Connectivity | 2            |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0,00 sec)

mysql>
```

Figure 5.5: A RDB result table **without** the mentioned line feeds problem

relational databases (Sec. 5.5.2), may become a difficult process to manage.

```
mysql>
mysql> select * from MultiplechoiceQuestions where subject="DB" and testNR=9;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | testNr | subject | question          | answer1          | answer2          | answer3          | answer4          | correctAnswer |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 75 | 9      | DB     | ODBC stand for? | Online
DB
Course | Open DB
Connectivity | OBS! DB Continuerly | Object DB Connectivity | 2            |
| 76 | 9      | DB     | ODBC stand for? | Online
DB
Course | Open DB
Connectivity | OBS! DB Continuerly | Object DB Connectivity | 2            |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

Figure 5.6: A RDB result table **with** the mentioned line feeds problem

## Chapter 6

# Mapping UML Class Diagram Into XML Document

### 6.1 Background

In this chapter, I tried to practice the idea of developing an UML class diagram in order to move toward an XML Schema [36]. At the end, an XML document for a *Course* example has been generated to represent the result of this mapping process. For this development, the traditional design approach described in the section 6.1, was deployed.

#### 6.1.1 Using UML

Unified Modeling Language (UML) as a standard notation developed by Object Management Group (OMG) provides object modeling for developing softwares. Software developers use UML diagrams to make a blueprint of a project by specifying the objects and associations involved within its system. UML provides the possibility of using different modeling diagrams such as **class**, **state**, **use case**, **sequence** and **activity** diagrams that are used for describing respectively the objects, sequence of states, event sequences for actors to use the system's functionalities, system scenarios and the business and operational workflows for modeling a business.

UML plays an important role in modeling objects of object-oriented (OO) applications, similar to Entity Relationship Diagram (ERD) and NIAM diagrams (Sec. 16.1) where logic and natural language are used for developing the conceptual data models in relational designs (Sec. 5.5). Using UML diagrams is very popular for designing business models and they are usually used as the first step for developing object-oriented programs ([16], Chap. 1).

#### 6.1.2 Three level design approach

In the following figure, the three level designing approach consisting of three levels: **conceptual**, **logical** and **physical**, is represented [36].

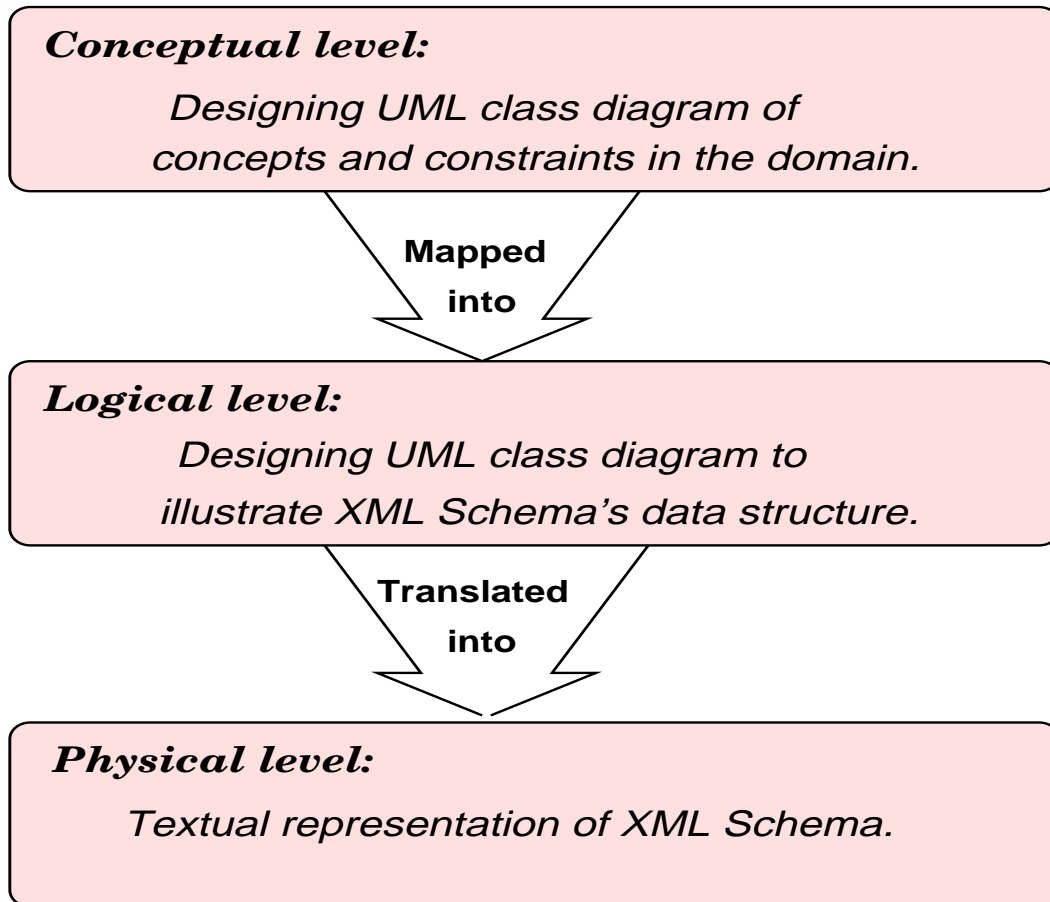


Figure 6.1: Three level design approach for mapping UML class diagram into XML Schema

## 6.2 Mapping UML to XML

In this section, I have tried to present the process of mapping the data information related to a *Course* example into XML data. First, the conceptual data of the *course* offered by a Learning Management System (LMS) (Sec. 3.2.2), is designed within an UML class diagram. Then the conceptual model will be converted into a logical model for the offered *course* on an Internet-based e-learning system. Further, the logical data model is translated into an XML Schema [36].

### 6.2.1 Conceptual level

At the conceptual level, the UML class diagram (Fig. 6.2) has been modeled to represent the concepts involved within my learning content example. The conceptual constraints of the involved objects must also be represented in the form of comments inside the model [36]<sup>1</sup>.

<sup>1</sup>The article is also available at: <http://portal.acm.org/citation.cfm?id=563924>



In the modeling process of a database, it must be considered that its designed data structure must have minimum redundancy which can be provided by eliminating unnecessary data repetition and maximizing the data connectivity between the information that will be stored in the database ([19], P. 39).

The following UML class diagram represents the involved concepts and the data model belonging to the learning materials offered at the class hours of my *Course* example. Moreover, the used **directed-association** symbols represent the **navigation** structure of the lecture materials.

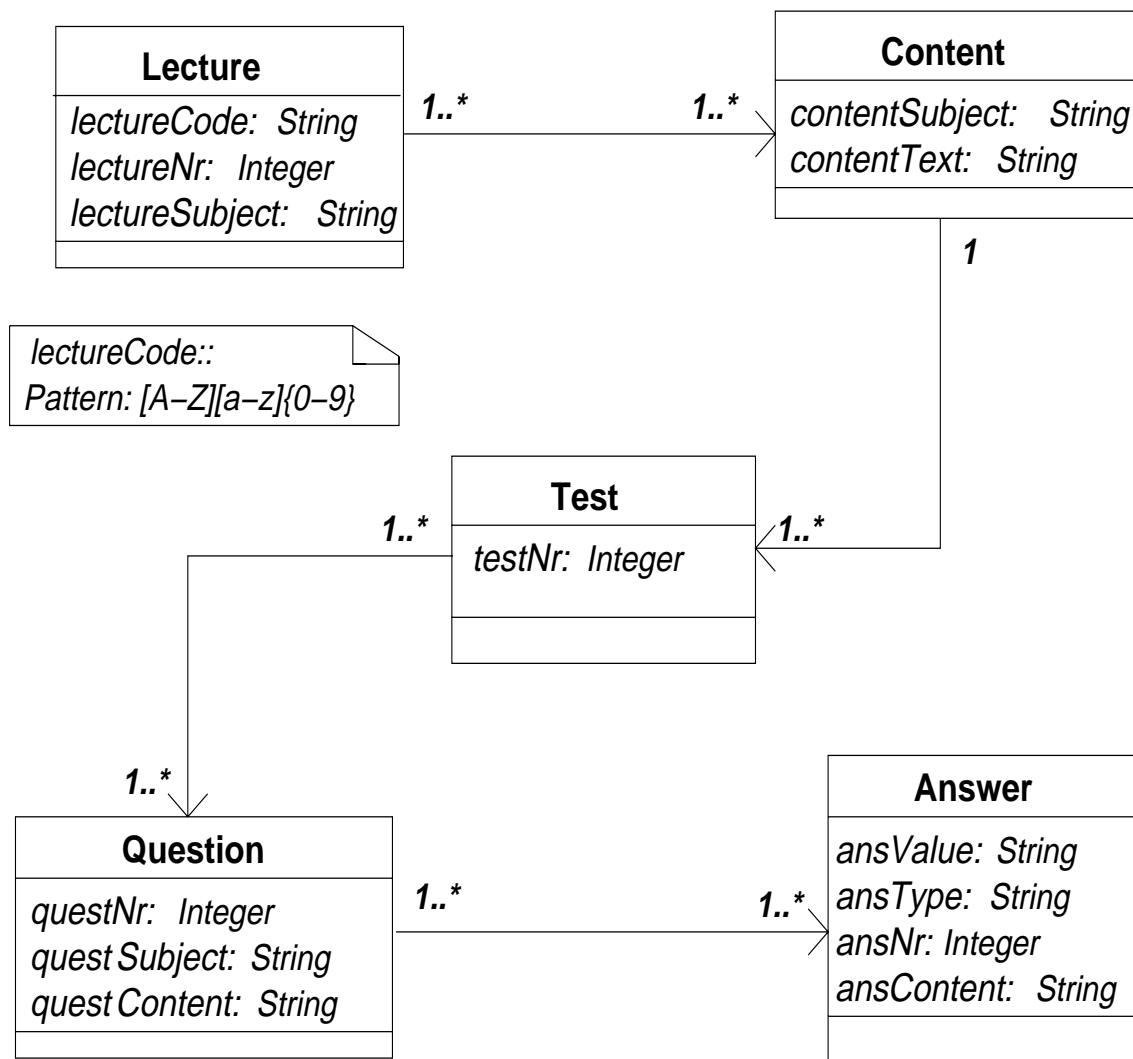


Figure 6.2: The conceptual UML class diagram for my *course* system example

The semistructured data model belonging to this conceptual data model, is provided in figure 9.2.

### 6.2.2 Logical level

In this level, another UML class diagram is developed that represents an abstract illustration of the involved concepts and constraints of my e-learning system example.

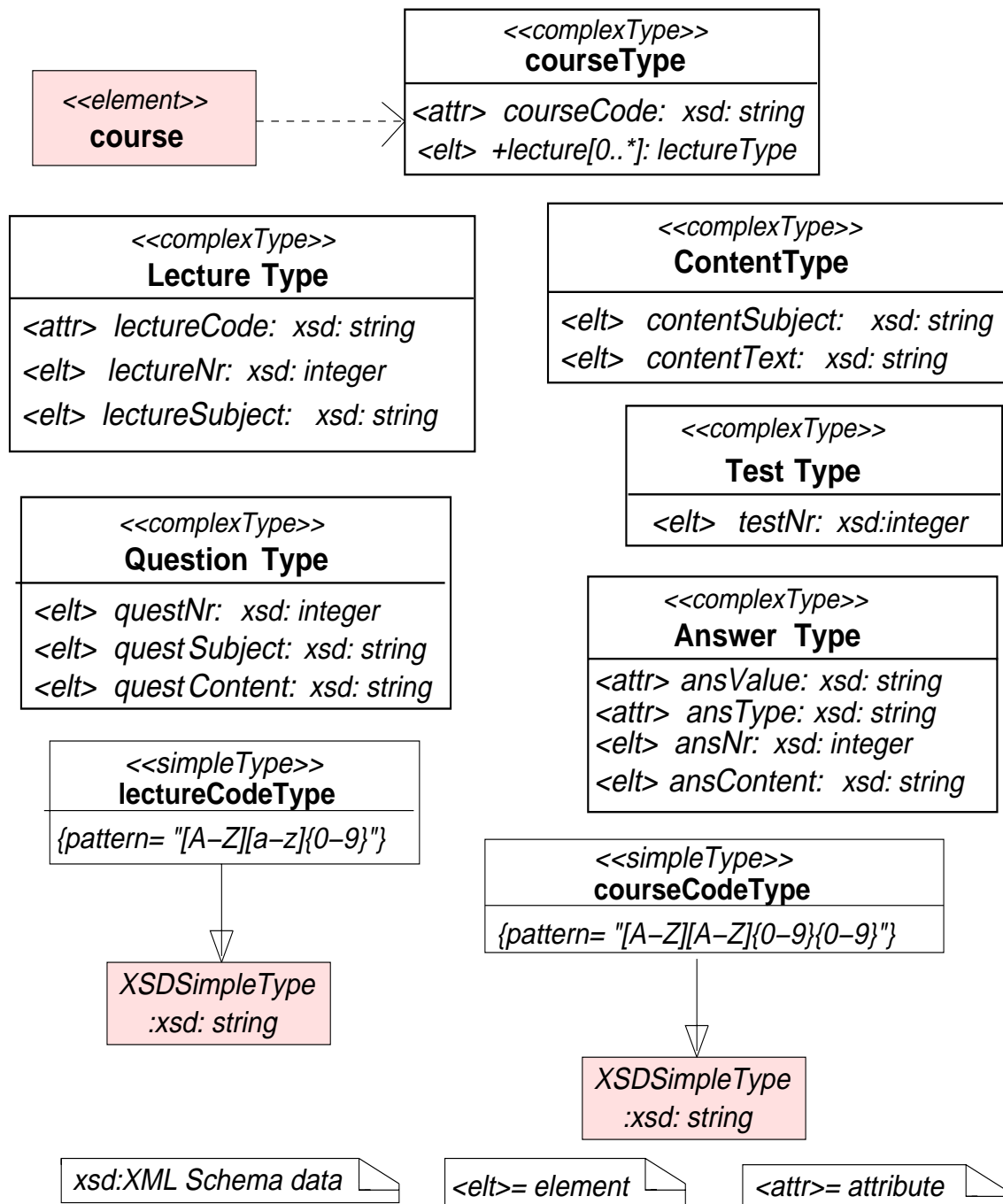


Figure 6.3: Logical level UML class Diagram for my Course system example.

In the diagram, some stereotypes such as *simpleType* and *complexType*, are used for describing the type of involved data, the data structure of system database and the type of system information ( [36], Sec. 2). This graphical representation of these logical concepts will be used as a basis for mapping the system information into an XML Schema. The used "simpleType" and "complexType" stereotypes in the logical diagram 6.3, specify that the data of these objects will be represented respectively as the child and the parent nodes (Sec. 8.1.2) within the semistructure data model of the *Test* example (Sec. 9.1.3).

### 6.2.3 Physical level

The physical level is the final level of the process of mapping the data information related to my *course* example into an XML Schema. This level contains the activities for implementing the modeled system into the physical world ( [36], Sec. 3). Therefore, the UML class diagram developed at the logical level, must be translated into a standard implementation language such as XML, which I have chosen.

Below, the physical data belonging to my *course* example is represented as an instance of my developed XML schema

```
<course courseCode="IN102">
  <lecture lectureCode="28Jan2005">
    <lectureNr>5</lectureNr>
    <lectureSubject>Distributed Systems</lectureSubject>
    <content>
      <contentSubject>Distributed computing</contentSubject>
      <contentText>
        Resource for distributed computing news....
      </contentText>
      <test>
        <testNr>1</testNr>
        <question>
          <questNr>1</questNr>
          <questSubject>Mysql</questSubject>
          <questContent>
            What kind of database is Mysql?
          </questContent>
          <answer ansValue="False" ansType="text">
            <ansNr>1</ansNr>
            <ansContent>It is a relational DB.</ansContent>
          </answer>
          <answer ansValue="True" ansType="graphic">
            <ansContent>computer.jpg</ansContent>
          </answer>
          <answer ansValue="False" ansType="sound">
            ...
          </answer>
        </question>
        <question>
          ...
        </question>
      </test>
    </test>
    ...
  </lecture>
</course>
```

```

        </test>
      </content>
    <content>
      ...
    </content>
  </lecture>
<lecture lectureCode="3Feb2005">
  ...
</lecture>
</course>

```

---

Figure 6.2.3: The physical representation of the XML Schema developed for the *Course* example

---

The above developed XML document represents an instance for an XML data structure developed based on the designed UML class diagram (Sec. 6.2.1) and the defined logical model (Sec. 6.2.2) to represent the data registered for my *Course* example. The semistructure database model of the Course "object" is represented in the section ??.

As the XML instance (Fig. 6.2.3) illustrates my Course object has been identified an attribute called *courseCode*. The Course can have many lectures that each of them are identified by its *lectureCode* attribute and the `<lectureNr>` and `<lectureSubject>` elements that respectively represent a number and a subject for each `<lecture>` element.

Further, each `<lecture>` element can contain different `<content>` elements that has its own subject and text represented by `<contentSubject>` and `<contentText>` elements. In addition, each Content can contain several Test objects represented by `<test>` elements. Each `<test>` element can consist of multiple choice questions with the following data structure. Therefore, a `<test>` element may contain many `<question>` elements while itself is represented by a number. Then each `<question>` element contains information about its number, subject and content, which are respectively represented by `<questNr>`, `<questSubject>` and `<questContent>`.

Moreover, several `<answer>` elements can be defined for each `<question>` element. The two *ansValue* and *ansType* attributes are defined to provide more information about the defined answers. For instance, the correctness value of a `<answer>` element, which can be *True* or *False* and the *ansType* attribute indicates the type of answer material, which for example can be a text, a sound or a graphical object. The `<ansNr>` and `<ansContent>` elements are used for representing the number and the content of each `<answer>` element.

During the development of my *ODC* learning system, which is explained in details in chapter 15, a similar XML document was created and used for storing various Test objects. The deployment of the XML file combined with DTD declarations and XSL formatting, has been used for developing the menu option *Show Test (XML)* (Sec. 15.1.1) of my developed trial *ODC* learning system (Sec. 14.2).

## Chapter 7

# Architecture

### 7.1 Background

As part of my thesis work, I worked with my *Online Database Course (ODC)* on-line learning application (Chap. 14, 15), which provided the facility for me to work with various technologies and become familiar with some of the existing standards for developing learning contents and learning system architectures. During the development of my *ODC* trial example, using and reflecting the ideas of various standards was in my focus (Sec. 7.6, 15.3).

In this chapter, I have briefly described different system development architectures and tried to illustrate the move from three-tier development architecture into the component-based architecture (Fig. 7.3), which is considered a capable system architecture for developing comprehensive systems such as learning management systems (Sec. 3.2.2).

At the end of this chapter, I have represented the most important model for e-learning called Learning Technology Systems Architecture (LTSA), developed by IEEE learning technology standards committee (Sec. 7.5). In addition I have presented the high-level system architecture that I had developed for constructing my *ODC* on-line learning system (Sec. 7.6).

### 7.2 Client-server architecture

The basis of client-server architecture called **Two-tier architecture** (Fig. 7.1) indicated that a central repository of information or other installed programs should be run on another computer system called a *server*. Server is responsible to store files used by all application programs and to contact database systems for processing of information.

The applications and database facilities provided by server can be used by another computer system(s) called *client(s)*. A client is a networked computer that uses a server for doing certain operations by using some applications that are installed on

the server. Processing and responding to the requests received from clients will be performed by servers that can be shared between several clients. For instance, World Wide Web is a client-server architecture used by many client computers.

The following figure shows the first tier that contain presentation and business logic parts of applications and is separated from the second tier, which make the data repository available for the applications run on the client computer(s).

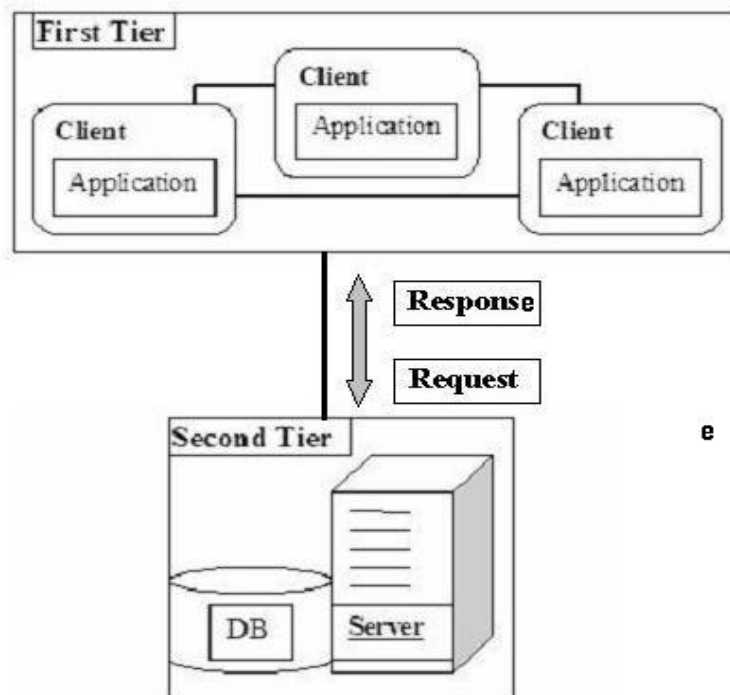


Figure 7.1: Client-server or Two-tier architecture.

Although the client/server approach had solved many problems there were still a few disadvantages with using this kind of architecture such as ([18]).

- *Overloading of client:* It could happen because of performing and processing of all business logics and functions existing on the client side while the server computer is just responsible to control the traffic of data between the client and the database on the server.
- *Degrading of server's performance:* In the Two-tier architecture the server could be responsible to load all requested applications for its client(s) and provide access to database for retrieving and manipulating its data. When the number of clients increased, there was a big chance that the server was overloaded with different clients' requests; this would cause a reduction in server performance. The bottleneck situation was another problem that could happen within a crowded

network when lots of requested data were sent to the database.

- *Maintenance difficulties:* Sometimes changing an application requires changing the whole structure of the application, which make the maintenance process of applications more difficult. After doing any changes to an application, it should be changed and recompiled or reinstalled at all client computers. Therefore, synchronizing was also a big issue that should be considered within installation of client applications, which could ensure that the latest version of the application was in use ([18], p. 208).

### 7.3 Three-tier towards N-tier Architecture

Three-tier architecture was the next approach that was developed in order to solve the previous problems. The separation of the presentation part from the business logic was the main distinguished change that was added to the two-tier architecture. Therefore, the first layer is called *presentation layer* that is responsible to handle all user interfaces and taking care of all presentational issues such as receiving, accepting information entered by users, forwarding them to the second layer, representing the feedbacks and fetched data to users on the computer screen. The following figure illustrates the three-tier architecture.

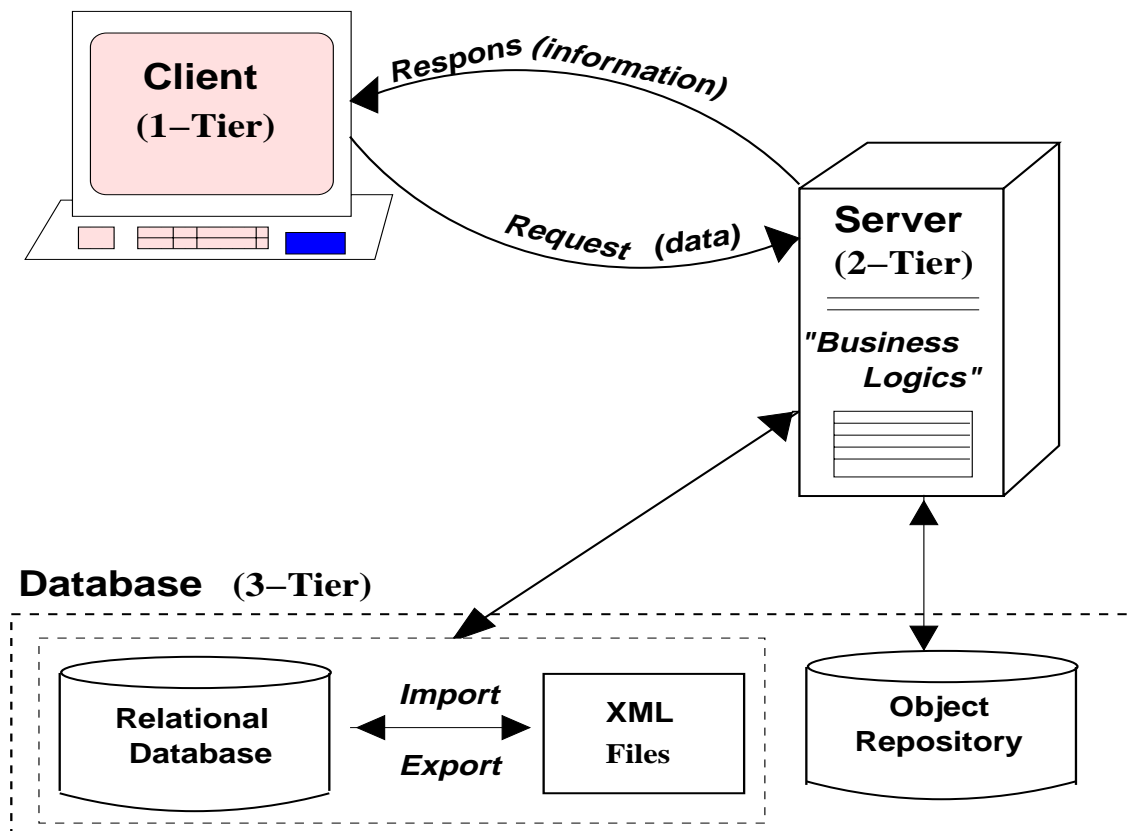


Figure 7.2: Three-tier architecture.

The second layer called *business layer* or *application layer* was placed in the middle of the three-tier architecture. The business layer contains all functions related to a developed system and the processing of data based on the business logic. The third tier called *database* layer was the data repository that contained all stored information for the developed system ([18], p. 5).

Later, *n-tier* architecture was developed that divides the middle layer of the three-tier architecture into several layers in order to separate the presentation logic from the business logic. The additional tiers were added to provide more flexibility and scalability. For example, the middle tier of the three-tier architecture could be split into two tiers, one for the Web server and the other one for the application server ([11], p. 957).

Some of the advantages provided by *n-tier* architecture applications were as follows: low costs for changing and updating business logic, reusing and pooling resources and low cost for switching database. On the other hand, there were also some disadvantages with the *n-tier* approach such as communication degradation, which happened when there were too many layers involved and needed to communicate with each other. Moreover, such applications demanded installation of other softwares, which would increase the system's costs for administration, maintenance and upgrading ([13], Chap. 2).

### 7.3.1 Evolution towards Component-based architecture

Figure 7.3 illustrates the movement from a two-tier architecture system to a *n-tier* architecture and then to a component-based system. As illustrated, the role of the first and third layers are unchanged while the architecture of the second layer has dramatically changed.

A **component** is a total behavior while a **service** is a partial behavior. A **layer** in a layered architecture is a service with two service interfaces, an import and an export interface ([38], Chap. 11). The functionality of a component can be structured into a family of services. A layered architecture is considered as a stack of several layers.

The *component-based* architecture, which has overcome the drawbacks of the *n-tier* architecture, is the latest approach used for developing large systems. This approach is also called *component-based enterprise architecture* because the common business functions for an entire organization is used to enhance its business practices.

The **presentation layer** as the first tier, is responsible for handling interactions with the end-users. This tier is divided into both client and server sides. The elements belonging to client-side of the presentation layer are responsible to deliver user interfaces and provide interaction possibilities for users such as presentation of HTML and Extensible Markup Language (XML) pages. The elements of the presentation layer belonging to server-side such as Servlets and Java Server Pages (JSP) files, are used to process the client-side requests and provide appropriate responses for them ([13],



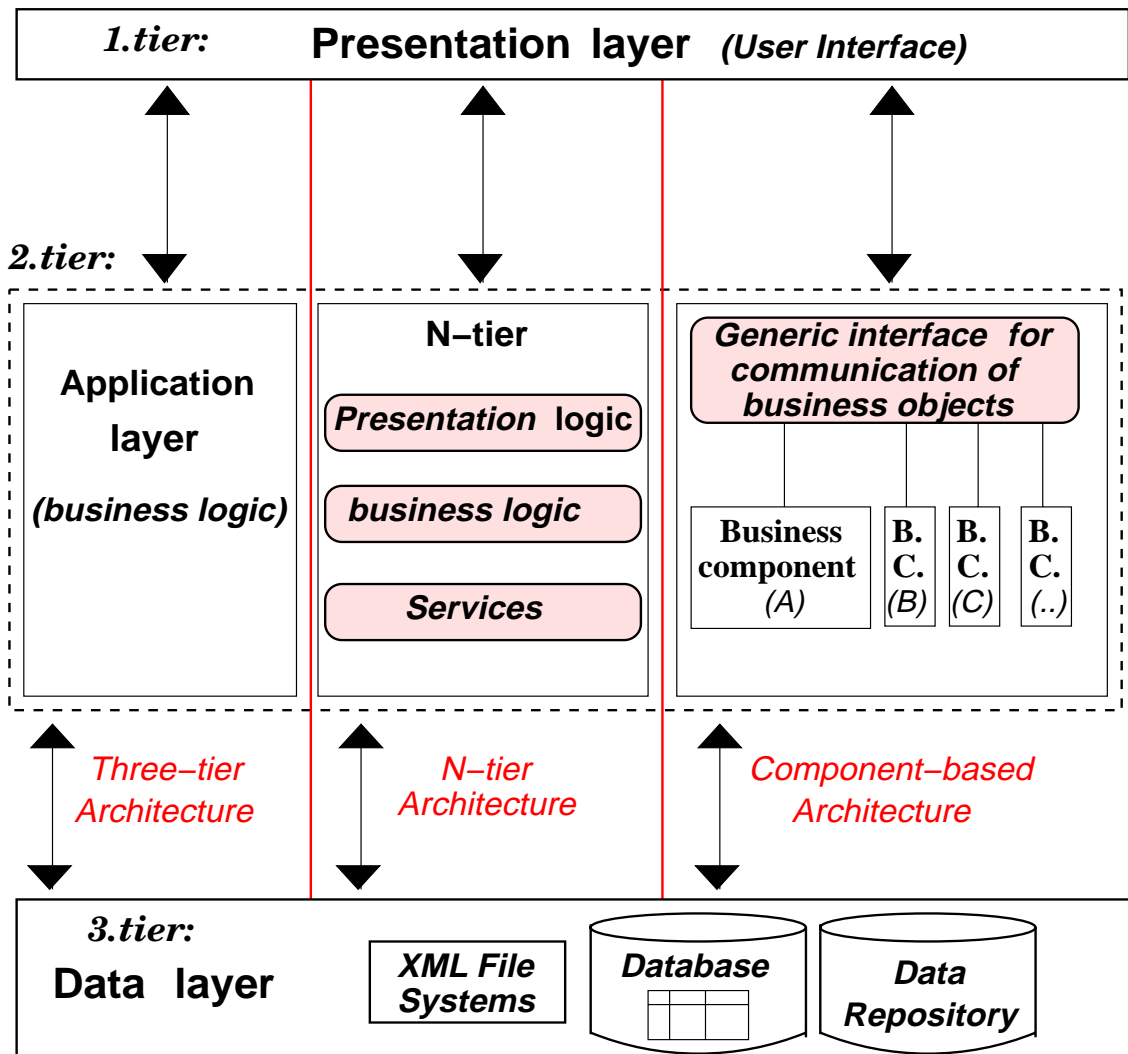


Figure 7.3: Three various approaches to application development.

Chap. 2).

The **business layer** as the second tier is responsible for performing any business processing. The third tier called **integration layer** is responsible to provide access to back-end resources such as databases and external systems.

Component-based architecture contains business objects and generic interfaces in its middle layer where an application object written into an interface can communicate with other business objects through their generic interfaces.

The big advantage of component-based architecture is that changing the rules of a business should be just applied into the application level where reusable business

components, namely business objects and their interfaces, exist ([18], p. 7).

## 7.4 Component-based architecture used for LMS developments

Deploying component-based system architecture (Sec. 7.3.1) for developing Learning Management Systems (LMS) provides the possibility for dividing the whole business logic of LMS required by SCORM (Sec. 4.2.2) into several major business activities. Then, activities belonging to the business logic of the whole system must be categorized and their responsibilities must be divided into several components. Later, different system components can separately be developed to build the desired functionalities within the business of LMS and performing any business processing for the LMS.

Business components can be developed to be capable of communicating with the other business components of the LMS. Moreover, business components can be developed to connect with the LMS database and interface layers. In addition, communicating with other LMSs will be another major responsibility for the components of a LMS business layer [24].

## 7.5 LTSA, the heart of LTSC's elearning model

The following figure represents the most important model developed by IEEE LTSC, for e-Learning . The model is called Learning Technology Systems Architecture (LTSA) which uses abstract components to represent the system architecture for developing learning systems (Sec. 3.2.2).

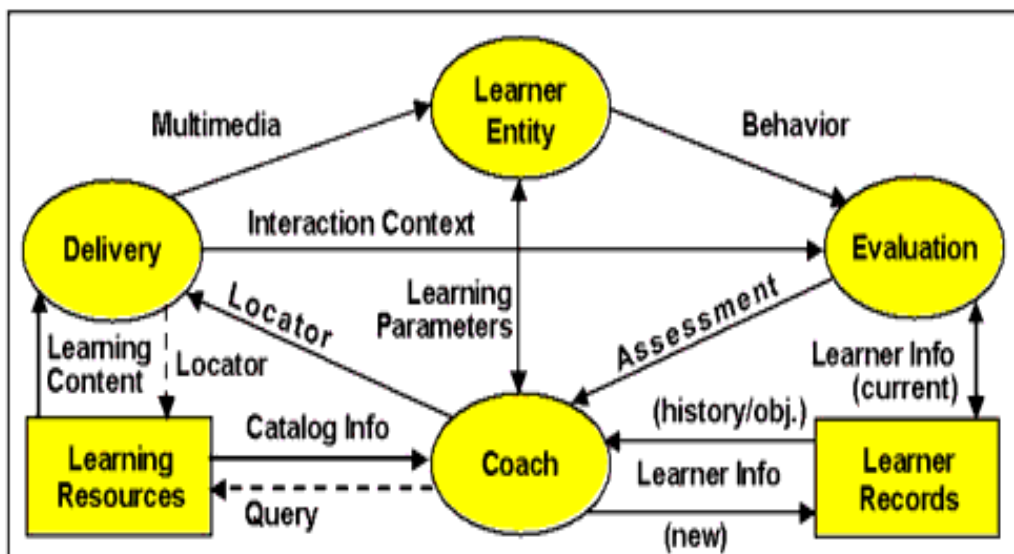


Figure 7.4: The Learning Technology Systems Architecture modeled by LTSC

The used components of IEEE LTSA architecture model are highly abstracted. For developing such learning systems in practice, the components must be modeled in lower abstract models. The system components needed for representing the business logic of a LMS, will be designed in the models (Sec. 7.3). In further development phases, the actual version of these components must be implemented (Sec. 7.3.1).

In the model (Fig. 7.4), the *Learner Entity* represents the learner(s) within the learning system and the *Learning Parameters* is used to represent the natural communication used within a learning experience. The model presents some **processes** within a learning system that involve the role of learners and course leaders (coach) in addition to some system functionalities such as evaluation and delivering of learning contents. The model demands that learning systems be able to **store** learner records and learning materials. In addition, the model indicates that performing different learning activities within a learning system demands **flowing** of various data materials, such as student information, learning contents and multimedia facilities, be accessed, reused and shared between the system components [15].

## 7.6 The system architecture of my trial ODC elearning course

The following figure represents the system architecture that was used for developing my trial ODC on-line course as a part of my thesis work. Based on this system architecture, different learning functionalities and instructional materials have been developed for my trial example (Chap. 16, 15, 14, 6).

The model presents the system of my ODC learning system example where teachers are able to create various learning objects. As an example, the functionality of the *Create Test* menu option (Sec. 14.3.1) is shown. By the *Ask Question (JSP)* menu option (Sec. 14.3.3), students are able to send their question to the course leader. The various menu options provided in my ODC on-line course are considered as a **delivery system** for my simple learning system, which provides the possibility for students to access all learning materials such as the one provided under the *Test Yourself* menu option (Sec. 14.3.2).

On the other hand, various learning resources are provided under the menu options *Test Yourself* and *Show Test (XML)* (Sec. 15.1.1) where different technologies such as JSP, XML, Document Type Definition (DTD), Extensible Style sheet Language (XSL), Macromedia Flash MX and HTML are used for generating various learning objects.

The idea in developing the architecture used for creating the system of my trial ODC on-line course example (Fig. 7.6) has some similarity with the LTSA architecture model represented by IEEE (Fig. 7.4). The IEEE-LTSA demands a two-direction communication facility between learner and teacher, whereas in my trial learning example, only a one-direction communication from students to teacher(s) has been provided under the *Ask Question (JSP)* menu option. Of course, such extra facilities can be provided in further development of my trial ODC example.

## 7.7 Detailed system architectures used for my trial *ODC* system

In this part, I have developed various figures that illustrate how the system architecture is developed and used to offer the different functionalities of the system represented under the various menu option of my *ODC* learning management system.

### 7.7.1 Usage of JSP within my *ODC* system architecture

Figure 7.6 presents the system architecture of my *ODC* learning system when my JSP applications are used to cooperate with the MySQL relational database to provide the desired system functionalities of the two menu options *Create Test* (Sec. 14.3.1) and *Test Yourself* (Sec. 14.3.2) on the client side.

### 7.7.2 Usage of XML within my *ODC* system architecture

Figure 7.7 shows the system architecture of my *ODC* on-line course when XML and XSL technologies are used to deliver the learning materials into the browser screen on the client side. The architecture is used for offering the system functionality provided by the *Show Test (XML)* menu option (Sec. 15.1.1).

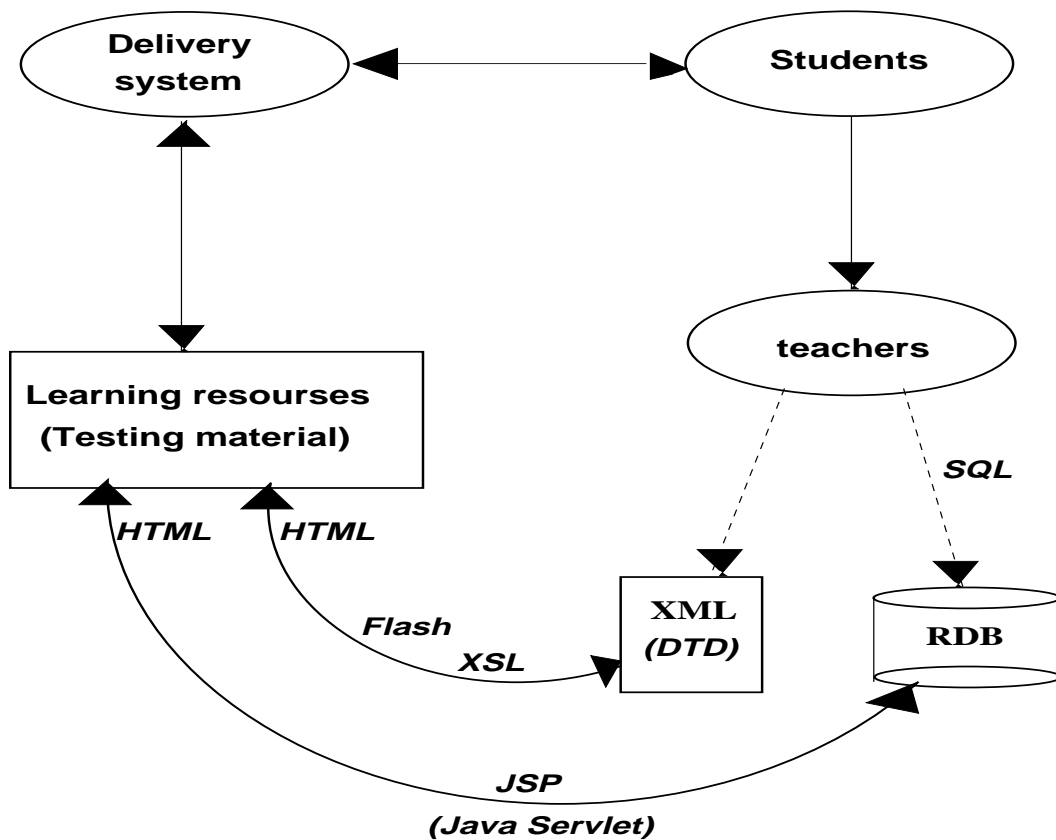


Figure 7.5: The system architecture used for my trial *ODC* on-line course

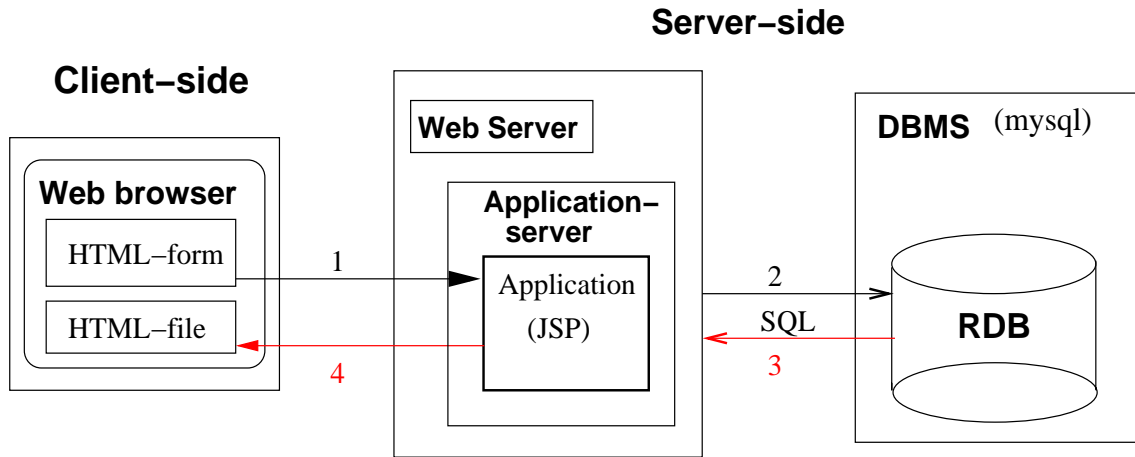


Figure 7.6: The system architecture by using JSP

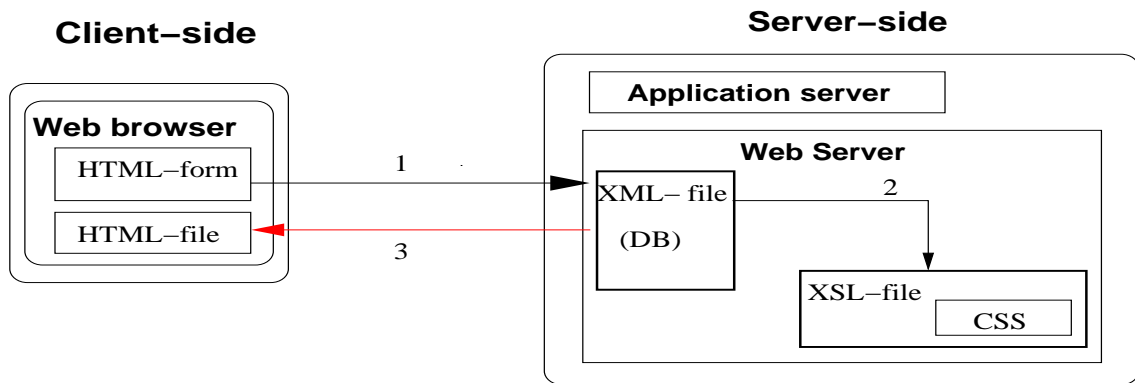


Figure 7.7: The system architecture by using XML

### 7.7.3 Usage of Flash MX within my ODC system architecture

Figure 7.8 illustrates the system architecture used for delivering the various Flash learning materials by my trial ODC learning system (Chap. 16).

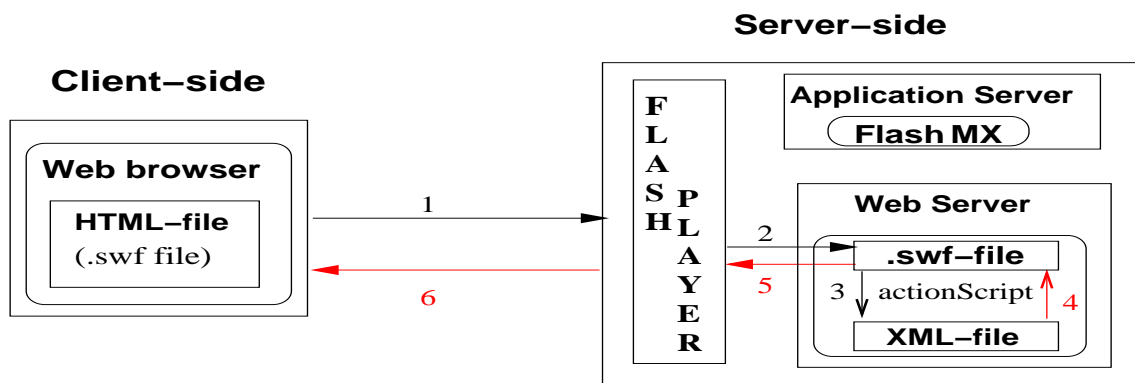


Figure 7.8: The system architecture for using Flash movies

## Chapter 8

# Extensible Markup Language (XML)

### 8.1 Extensible Markup Language (XML)

The structured markup language XML, EXtensible Markup Language, has been developed based on the Standard Generalized Markup Language (SGML), which offers the recent approach for a flexible storing of data. XML is a tag-based notation language used for "marking" text documents. XML provides flexible storing of text materials that will be represented as substrings of a text document. XML tags applied in a document should be defined based on a Document Type Definition (DTD) (Sec. 9.1), which as a flexible schema specifies the hierarchical structure of an XML document.

One of the main applications of the XML Internet technologies is that it provides an infrastructure for a new type of learning organization, the so-called **distributed learning** provided by learning management systems (LMS) (Sec. 3.2.2).

#### 8.1.1 What do HTML and XML do?

By the traditional markup language HTML, the presentation of a content is defined to a Web browser. For instance, HTML provides different fonts in a text document, puts images next to texts and creates hypertexts by linking text document to each other. XML provides the possibility for structuring the content of text documents. A text content can be structured based on the meaning of its substrings that will be defined inside various tags as the content *elements* of the text document. The tag elements of a document will be specified by the developer him/herself [10].

#### 8.1.2 XML concepts: element, node, attribute, parent and child nodes

Each data element defined as an `<element>` XML tag, is considered as a *node* within a tree-structured XML document. For storing extra information for an element, there are two alternatives to define the actual element, either as an **attribute** or as a *child* node. Usually attributes are used for the system's internal usage. However, the information that is supposed to be represented on the screen is normally defined by child nodes belonging to a *parent* node within the same XML document.

An XML attribute is always included inside an XML element to provide more information for its tagged element. The attribute value is considered as a text string, which is defined inside two quotation marks (Fig. 8.1).

### 8.1.3 XML elements

HTML tags are used for *presentation* of elements within an electronic document while XML is used for electronic storage of the document's *content*. For instance, the `<i></i>` HTML tag will be used to identify the part of a text that must be displayed in italics. As an example, the `<i>text in italics</i>` will be represented as: *text in italics*.

An XML element similar to HTML, begins with a start tag and ends with an end-tag while its relevant information will be contained inside the element.

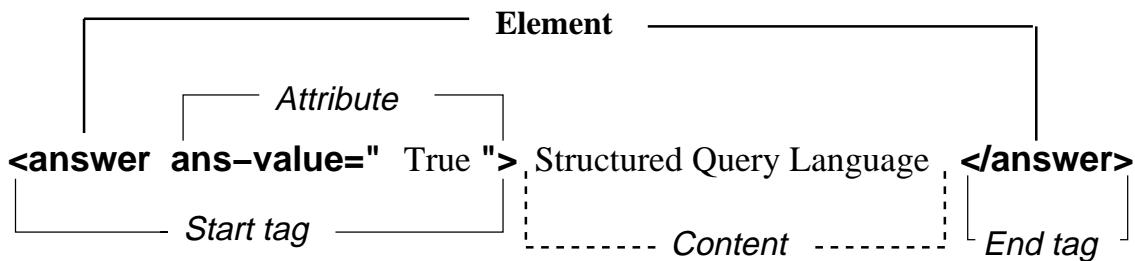


Figure 8.1: The structure of XML element

XML tags are the elements of a hierarchical structured document that contain the substrings of its content. Each XML element can be defined to have its own attributes that can be used for defining certain constraints for the element. An attribute used for defining a *constraint*, requires the element data to have certain properties. For instance, under the XML developments of my *Online Database Course (ODC)* on-line course example, I have defined a value attribute inside my `<answer>` elements called *ansValue*, which is used for identifying the correctness of each answer element (Sec. 6.2.3).

Defining stronger constraints for an XML document must be done by using any XML Schema language such as document type definition (DTD) that defines the legal elements for the document (Sec. 9.1). For developing the *Shoe Test (XML)* menu option of my *ODC* on-line course, my *course-tests.xml* XML file (Append. A.1.1) has an internal DTD declaration that contains the following attribute declaration.

```
<!ATTLIST answer ans-value ( True false ) #REQUIRED>
```

Later, the attribute is used within my Extensible Style sheet Language (XSL) formatting codes illustrated by the following XSL formatting fetched from my *test-answers-if.xsl* (Append. A.1.2); it then uses the value of *ans-value* attribute as a correctness constraint for performing an XSL *choose* template (Sec. 10.1.3).

```
<xsl:choose><xsl:when test="./@ans-value='True'">  
<font size="3"><b><xsl:value-of select="ans_content"/>  
</b></font></xsl:when>
```

#### 8.1.4 Document Object model (DOM)

The hierarchical data structure of an XML document is expressed by the Document Object model (DOM) for the data file. DOM specifies that the data nodes of an XML file are defined based on a tree-structured syntax. It means each data element can be the parent node for the other node(s) while, at the same time, it can be the child node for another XML element that lies one level higher in the hierarchical data structure of the XML file ([8], p.325).

#### 8.1.5 Comments in XML

Comments in an XML document must be placed between two brackets (<-) and (->) as shown in the following example: <-- An XML comment -->

An XML file can not be started by a comment tag and it is not permitted to use any extra "-" characters inside a comment tag. A comment element can not include any other comment tag. It means comments can not be nested [34].

#### 8.1.6 Well-formed XML

Although XML is very flexible in definition of elements, it demands strict constraints for other aspects defined as a grammar for XML documents. XML grammar specifies the rules for an XML document. For instance, the grammar specifies how XML tags should be represented, where they can be placed, ways of attaching attributes to XML elements, and allowable names for elements.

XML parsers are developed based on the XML grammar and are able to read any XML documents in order to specify whether documents satisfy XML grammar or not. XML documents, which satisfy XML grammar, are said to be **well-formed** [22]. Below are some of the constraints used for parsing XML-documents.

- **Root element:** Just one root element contains the whole document.
- **Start & end tags:** Each element has a *start* and an *end* tag.
- **Element name:** The name used in an element start and end tags, must be matched.
- **Attribute name:** An attribute name used in an element should be unique.
- **Attribute value:** The value of attributes must be defined inside quotation marks.



- **Proper nesting:** Tags can be nested but they can not be overlapped; this means an element can be placed in another one only if both its *start* and *end* tags appear inside the same element.
- **No "<":** The character must not be used in any attribute value.

### 8.1.7 Valid XML

The markup permitted to be used in an XML application, can be documented in a *Schema* that will later be used by document instances to compare with. When an XML-document is matched with its schema, it is said to be **valid**.

XML 1.0 is supported by DTD (Sec. 9.1) as its schema language. A validating parser will check the defined constraints of the DTD or the XML Schema within an XML document to specify whether the document is valid or not [22].

Some of the validity constraints used by a validating parser, are listed below:

- **Root element:** The name of root element is declared in the *document type declaration* such as: `<!DOCTYPE tests [ .....]>`
- **Valid elements:** The elements of the document are used based on the DTD.
- **ID:** There is just one unique ID attribute to be defined for each element.
- **Element name:** The declaration of element names can not be repeated.
- **Attribute value:** Each attribute should be declared and its value must be from the same type of value that is defined for it.

## 8.2 Classification of XML applications

XML documents are not just used for Web site publishing. In general, information stored on XML documents can be manipulated for later usages by either humans or softwares. Those documents that are developed for human usages are considered as *document applications* and the ones that manipulate information for software consumptions are called *data applications*. The content of an XML document can be edited and maintained as long as its structure is known, and then it can automatically be published on different media [28].

### 8.2.1 Document Applications

Document applications concentrate on the structure of documents and they are independent of the delivery mediums. Therefore, such a document can automatically be converted to either a postscript document and printed on a printer or it can be converted to a HTML document for any on-line representation on computer screens and handheld devices with displays such as personal digital assistants like Palmpilot and cellular phones.

Doing automatically is the keyword here, because in general, manual manipulations are considered costly procedures. Therefore, it is essential to maintain documents in a common format such as XML that is independent of the delivery medium and can automatically be converted into many publishing formats [28].

### 8.2.2 Data Applications

The development of SGML, Standard Generalized Markup Language, enabled document management systems to access application tools such as document databases and search engines, for accessing and managing data. Document management systems could then more easily store, retrieve and use characterized file documents. By the following figure, I try to illustrate how the content of my Test example stored as an ordinary document, can be saved into a relational database or be structured as an XML document (Sec. 5.5.2).

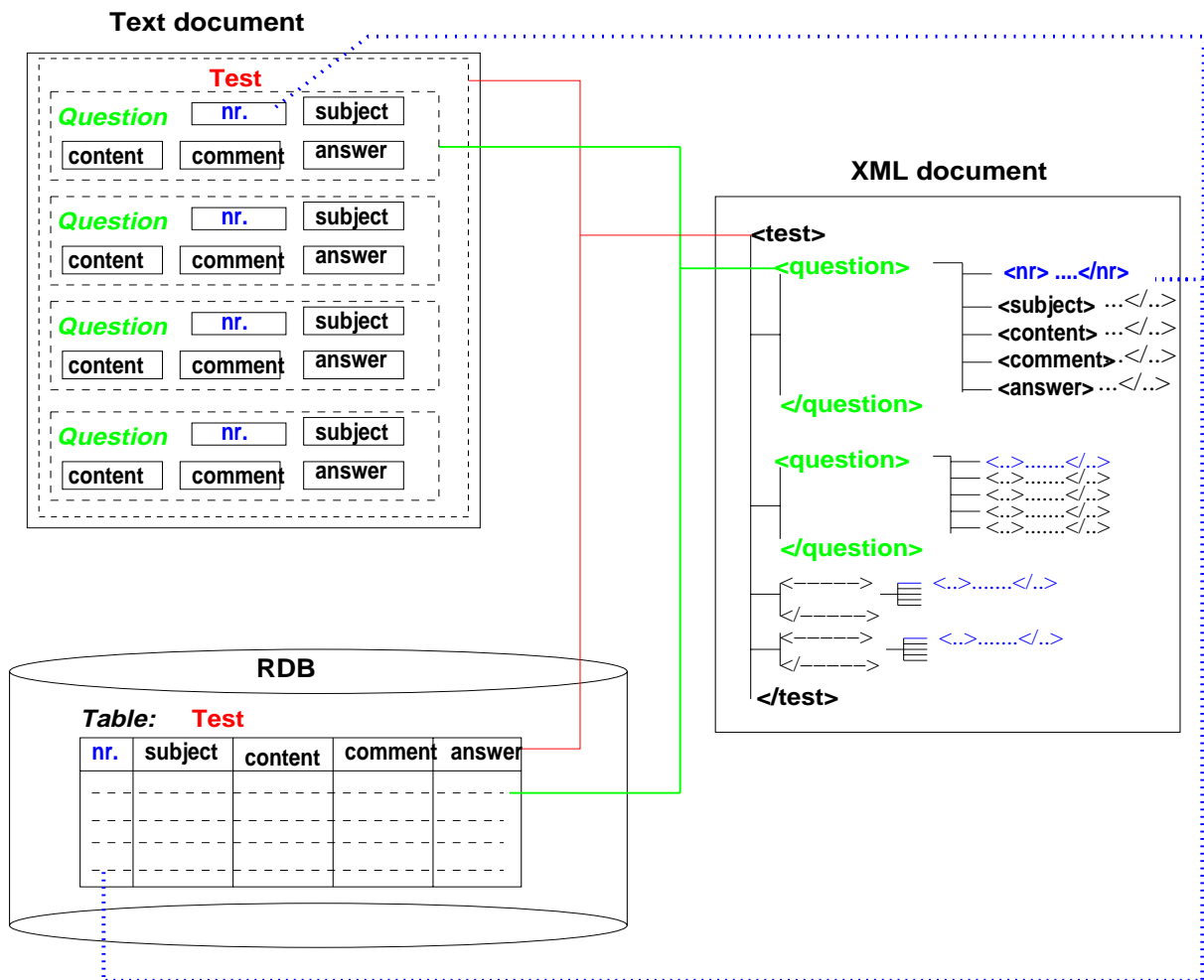


Figure 8.2: Mapping the content of my Test example into RDB & XML

A list of questions and answers of a test, which could be stored in a relational database, is given in my *tests.xml* XML document, which is read by my *tests.swf* file developed by Flash MX.

Before development of XML, document management systems were developed to create libraries of documents. The document management technology had increased the capabilities of computerized file systems and networks for providing better control and access to documents.

A document library as a database was used to store and organize the user's information, relationships between different documents, and the document information that was defined in a file which contained the document.

### **8.3 Manipulating XML text materials**

Based on my experimental development done for my thesis, I found it much easier to work with text material stored as a XML document rather than stored in relational database (RDB) data.

XML documents can more easily be manipulated. They also provide better possibilities for formatting text materials. For instance, XML documents provide more flexibility for doing text editing or using line feeds within the text material. While in RDB, storing text materials with several line feeds will cause an unorganized appearance, which will make the updating process of RDB much more complicated (Fig. 5.5).

## Chapter 9

# Document Type Definition (DTD)

### 9.1 What is DTD?

In order to process XML documents automatically, we need something like a schema to structure XML documents. Therefore, in order to define the legal element structure and "building blocks" for XML documents, the Document type definition (DTD) has been developed. DTD provides a grammar-like set of rules for representing the tag elements that can be appeared in an XML collection document and the way that they can be nested.

DTD in XML does the same role that object definition language (ODL) does for object-oriented (OO), in order to define the involved concepts within a database (Sec. 5.2.1).

#### 9.1.1 DTD of my *test* learning object

An XML file considered as a database was developed to contain all questions and answers belonging to various test materials. These XML data have been structured based on an **internal** DTD specification where each **test** element is identified by a number and the name of the course it belongs to. Based on the data structure defined for the *test* XML document, each test is recognized with its own No. and the course name that it belongs to. Each test may contains many questions which each will have its own No. and subject.

Below, the DTD specifications of the *Test* XML document is represented that implies the same data structure represented in figures 9.1 and 9.2. These DTD declarations are internally defined in the *Test* XML file (App. A).

```
<!DOCTYPE tests [  
  
<!ELEMENT tests (test)*>  
<!ELEMENT test (test_nr, questions, test_subject?, test_comment?)>  
<!ELEMENT test_nr (#PCDATA)>  
<!ELEMENT questions (question)*>  
<!ELEMENT test_subject (#PCDATA)>  
<!ELEMENT test_comment (#PCDATA)>
```

```

<!ELEMENT question (quest_nr, quest_subject?, quest_content,
                    quest_comment?, answers)>
<!ELEMENT quest_nr (#PCDATA) >
<!ELEMENT quest_subject (#PCDATA) >
<!ELEMENT quest_content (#PCDATA) >
<!ELEMENT quest_comment (#PCDATA) >
<!ELEMENT answers (answer)* >

<!ELEMENT answer (ans_nr, ans_content, answer_comment?) >
<!ELEMENT ans_nr (#PCDATA) >
<!ELEMENT ans_content (#PCDATA) >
<!ELEMENT ans_comment (#PCDATA) >

<!ATTLIST answer ans-value ( True | false ) #REQUIRED >
]>

```

---

Figure 9.1.1: The DTD specifications of my *Test* example.

---

### 9.1.2 Hierarchical representation of XML Schema

My *Test* example has been developed related to offering various learning and training services by the menu option *Show Test (XML)* on my *ODC* course example. For developing the various functionalities of this menu option, XML technologies has been applied (Sec. 15.1.1).

At the right side of figure 9.1, the data structure of my *course-tests.xml* XML file is represented (App. A). The XML document is developed to contain some learning and tutoring materials that will be delivered by various learning services developed by XML technologies that are available under the *Show Test (XML)* menu option (Sec. 15.1.1). The content of XML document contains a list of multiple choice questions gathered with their alternative answers, which their correctness can be identified by their *ans-value* attribute. Moreover, a tree-structured presentation of the XML Schema used for developing the XML document, is represented on the right side of the figure.

A similar development process of structuring such XML document considered as a database for testing materials, is represented in details in chapter 6. The semistructure model belonged to this XML Schema is represented in section 9.1.3.

### 9.1.3 Semistructure modeling of my *Test* example

In the following model, the semistructure model of the XML Schema used for developing the XML document of my *Test* example is represented. The hierarchical data structure of the XML file shows that the *test* root element contains other nodes as its child nodes within a hierarchical system structure (Sec. 8.1.6).

This figure represents the semistructure data model used for developing my *Test* XML

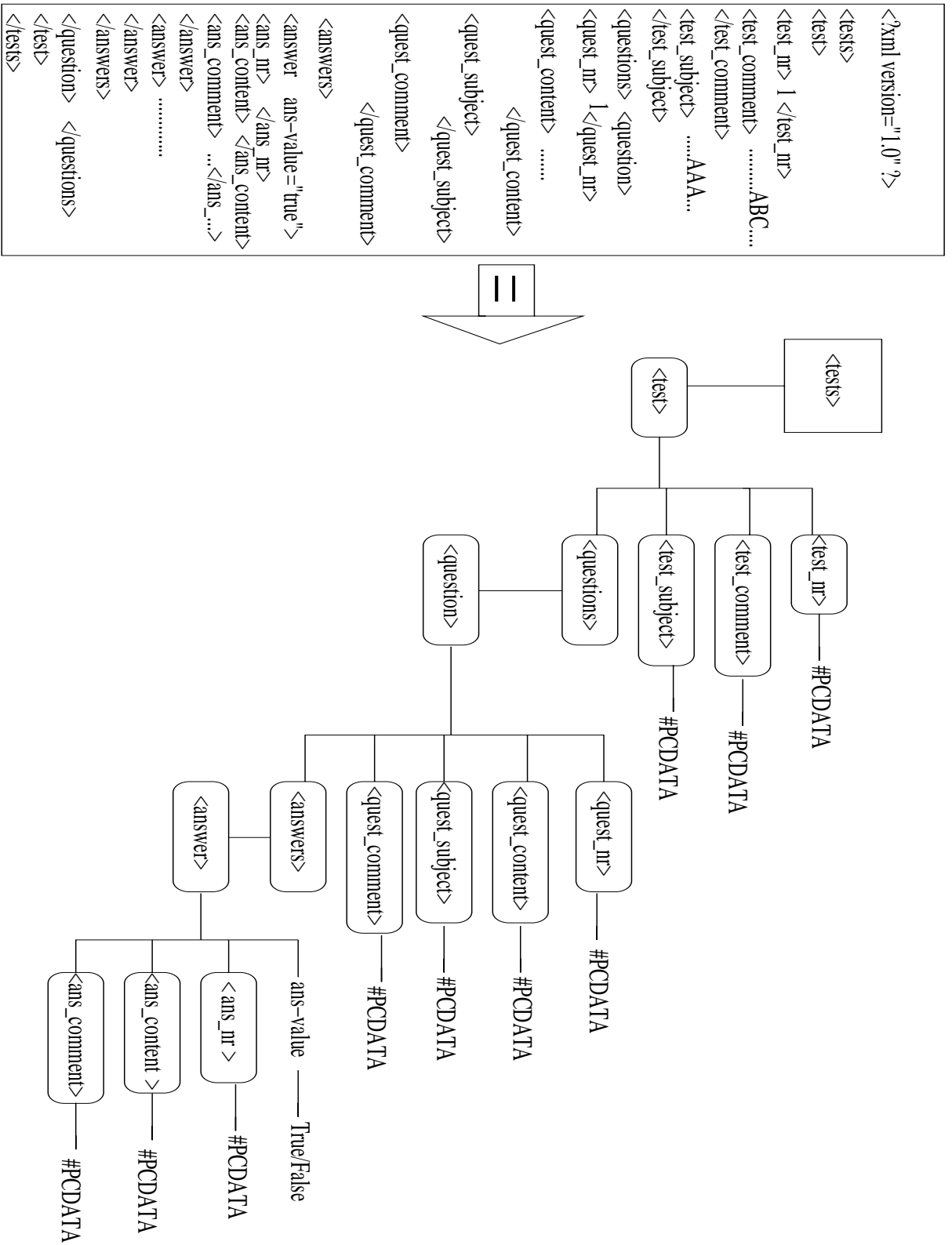


Figure 9.1: XML Schema presentation of my *Test* XML document.

document (App. A), which is considered as the basis model for developing the XML Schema represented in figure 9.1 that has been mapped into the DTD specifications of the *Test* XML document (Fig. 9.1.1).

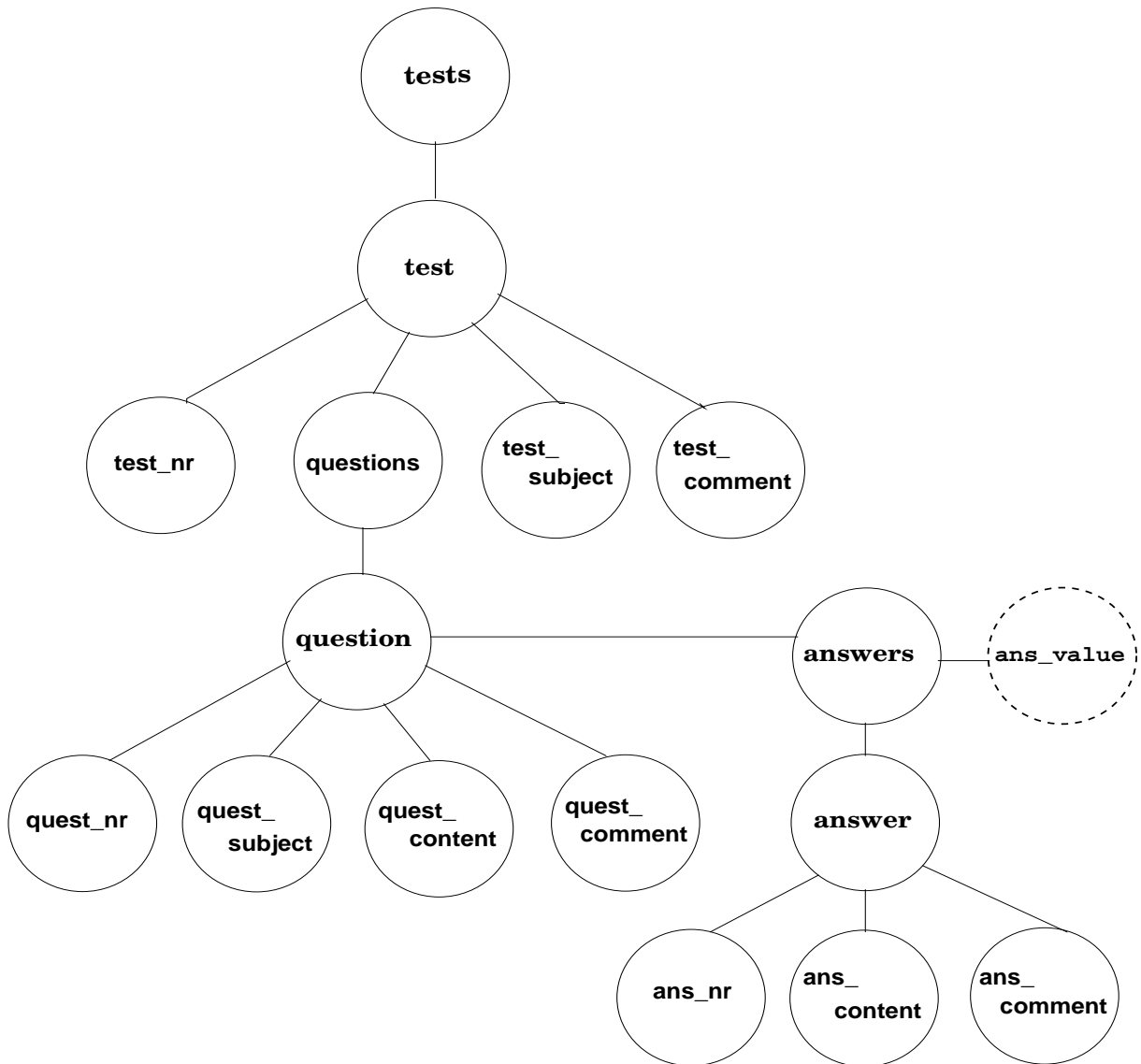


Figure 9.2: Semistructure modeling of the XML Schema of the testing material belonged to my *Test* example

The conceptual UML class diagram belonging to this semistructured data model, has been presented in figure 6.2. In this diagram, the solid circles are representing the XML elements while dashed circles are used to represent the attribute of node elements (Sec. 8.1.2). For instance, the two *ans-value* and *ans-type* nodes are used to respectively represent a correctness constraint and specify the type of answer materials for every *<answer>* element belonged to a *<question>* element (Sec. 8.1.7).

## 9.2 An DTD application example

Below, a semistructure data model belonging to a *Course* example is presented.

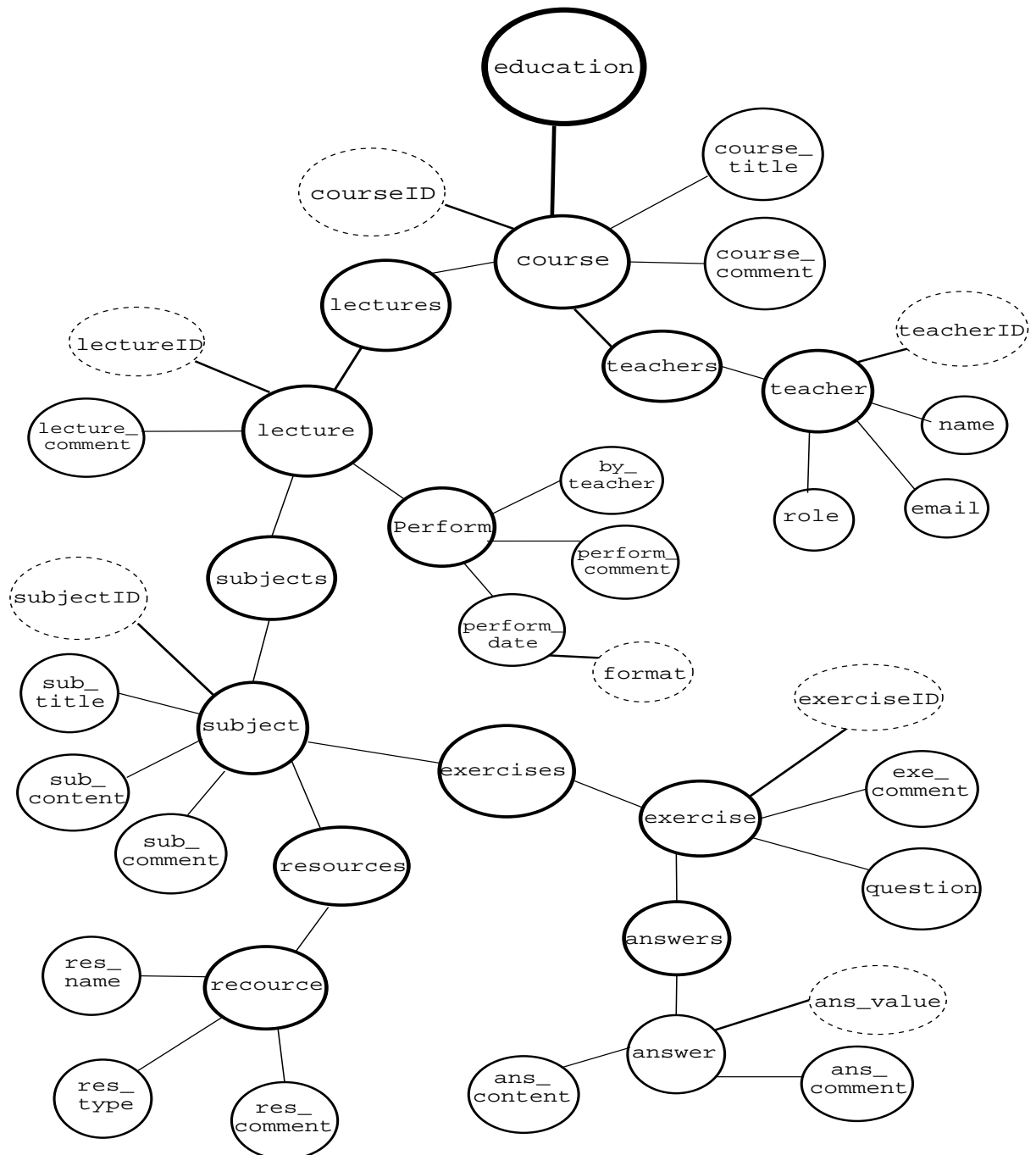


Figure 9.3: The semistructure data model used for *Course info.* menu option



The following DTD declaration has been developed as a simple version of the illustrated semistructure model presented in figure 9.3. The DTD example is the same used for representing the information belonging to my on-line course example that has been deployed for developing the menu option *Course info.*

```

<!ELEMENT education (course)+ >
<!ELEMENT course (course-title, course-comment, teachers, lectures) >
<!ATTLIST course course-code ID #REQUIRED >
<!ELEMENT course-title (#PCDATA) >
<!ELEMENT course-comment (#PCDATA) >
<!ELEMENT teachers (teacher)+ >
<!ELEMENT teacher (name, role, email, homepage) >
<!ELEMENT name (#PCDATA) >
<!ELEMENT role (#PCDATA) >
<!ELEMENT email (#PCDATA) >
<!ELEMENT homepage (#PCDATA) >
<!ELEMENT lectures (lecture*) >
<!ELEMENT lecture (lectureNr, subjects , perform, lecture-comment?) >
<!ELEMENT lectureNr (#PCDATA) >
<!ELEMENT subjects (subject+) >
<!ELEMENT subject (sub-title, sub-content, sub-comment?, resources?, exercises?) >
<!ATTLIST subject subjectID ID #IMPLIED >
<!ELEMENT sub-title (#PCDATA) >
<!ELEMENT sub-comment (#PCDATA) >
<!ELEMENT resources (resource*) >
<!ELEMENT resource (res-name, res-type, res-comment) >
<!ELEMENT res-name (#PCDATA) >
<!ELEMENT res-type (#PCDATA) >
<!ELEMENT res-comment (#PCDATA) >
<!ELEMENT sub-content (paragraph)+ >
<!ATTLIST paragraph about (NMTOKENS) #IMPLIED>
<!ELEMENT paragraph (#PCDATA) >
<!ELEMENT exercises (exercise*) >
<!ELEMENT exercise (exerciseNr, question, answers, exe-comment?) >
<!ELEMENT exerciseNr (#PCDATA) >
<!ELEMENT question (#PCDATA) >
<!ELEMENT answers (answer+) >
<!ELEMENT answer ( ans-number, ans-content, ans-comment?) >
<!ELEMENT ans-number (#PCDATA) >
<!ELEMENT ans-content (#PCDATA) >
<!ELEMENT ans-comment (#PCDATA) >
<!ELEMENT exe-comment (#PCDATA) >
<!ATTLIST answer ans-value ( true | false ) #REQUIRED >
<!ELEMENT perform (by-teacher, perform-date, perform-comment?) >
<!ELEMENT by-teacher (#PCDATA) >
<!ELEMENT perform-date (#PCDATA) >
<!ATTLIST perform-date format (NMTOKEN) #REQUIRED >
<!ELEMENT perform-comment (#PCDATA) >
<!ELEMENT lecture-comment (#PCDATA) >

```

---

The DTD of *Course info.* menu option

---

Based on this DTD specification, an XML document is created, which is explained in the section 10.3.

## Chapter 10

# Extensible Style Sheet Language (XSL)

### 10.1 What is XSL?

Extensible Style Sheet Language (XSL) is the advanced language for expressing style sheets for XML documents. XSL Transformations (XSLT) provide an essential programming skill for formatting the output of XML source files to HTML. The Cascading Style Sheet language (CSS), is applicable to XML as it is to HTML.

#### 10.1.1 One XML file for several XSL files

One of the advantages of storing information in the form of an XML document is that we can apply different XSL formatting to the same XML document. The following **processing-instruction** is defined in one of my XML documents, to specify the *'test-answers-if.xml'* XSL file to be used for data presentation of the XML material.

```
<?xml-stylesheet type="text/xsl" href="course-tests.xml"?>
```

These files are used for the development of the *Show Test (XML)* menu option of my (ODC) example (Sec. 15.1.1).

#### 10.1.2 Having comment in XSL

The XSL comment element must always be used within a template body. The comments must be placed between a start and an end XSL comment tag, for example:

```
<xsl:comment> These are comments!</xsl:comment>
```

#### 10.1.3 XSL:choose, XSL:when , XSL:otherwise, XSL:for-each

The *choose* template given in XSL, provides executing some sort of *if-else* statement used for formatting XML documents. The template is used as a part of condition testing of XML elements. Usually, the **<xsl:choose>** element is used with **<xsl:when>** and **<xsl:otherwise>** elements. The **<xsl:when>** contains the condition constraint while the **<xsl:otherwise>** element is used to contain the default case.

The **<xsl:for-each>** element provides defining a loop in the same way as for loops

in a programming language.

The `<xsl:for-each>` element provides the possibility for selecting every child node of a parent node within an XML tree-structured document. The following XSL template is picked up from my *test-answer-if.xsl* file (Append. A), which represents the way of using these XSL elements.

```
<xsl:template match="answers">
<font size="3"> Correct answer:</font><xsl:text> </xsl:text>
<xsl:for-each select="answer">
<xsl:choose>
<xsl:when test="./@ans-value='True'" >
<font size="3"><b>
<xsl:value-of select="ans_content" /></b></font>
</xsl:when>
<xsl:otherwise><xsl:text></xsl:text>
</xsl:otherwise>
</xsl:choose>
</xsl:for-each>
```

## 10.2 Namespace in XSL

In working with XSLT, we add the *namespace* declarations in the stylesheet tag for its **xmlns** attribute and then it will be used inside the query elements. Here is an XML file example that shows how another XML-namespace may be defined and a css will be deployed to make the output of this XML file.

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css" href="course.css" ?>
<COURSE xmlns:HTML="http://www.w3.org/TR/REC-html40">
  <TITLE>Putting it All Together</TITLE>
  <HTML:UL>
    <HTML:LI>Line one</HTML:LI>
    <HTML:LI>Line two</HTML:LI>
  </HTML:UL>
  <HTML:BR />
  <HTML:IMG src="female_model.jpg" />
</COURSE>
```

The textual output of this xml file will be:  
 Putting it All Together  
 Line one  
 Line two

### 10.3 An XSL application example

The following XML document has been developed based on the DTD specification given in figure 9.2. This file contains all the information stored for the administration part of my *ODC* course example.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<?xml-stylesheet type="text/xsl" href="course.xsl" ?>

<!DOCTYPE education SYSTEM "course.dtd" >
<education>
<course course-code="_102" >
<course-title>System Development</course-title>
<course-comment>The course is virtual as a trial version!
</course-comment>
<teachers>
<teacher>
<name>Teacher 1</name>
<role>Main teacher</role>
<email>teacher1@ifi.uio.no</email>
<homepage>http://heim.ifi.uio.no/~teacher1</homepage>
</teacher>
<teacher>
<name>Teacher 2</name>
<role>Guest teacher</role>
<email>teacher2@ifi.uio.no</email>
<homepage>http://heim.ifi.uio.no/~teacher2</homepage>
</teacher>
</teachers>
<lectures>
<lecture>
<lectureNr>5</lectureNr>
<subjects>
<subject subjectID="_1">
<sub-title>L-Structured Query Language</sub-title>
<sub-content>
<paragraph about="sql">Structured Query Language - SQL is a standard
interactive and programming language for getting information from and
updating a database.
</paragraph>
</sub-content>
<sub-comment>SQL is both an ANSI and an ISO standard, many
database products support SQL with proprietary extensions
to the standard language.</sub-comment>
<resources>
<resource>
<res-name>SQL quiz</res-name>
<res-type>Web site</res-type>
<res-comment>
Test your SQL knowledge on :
http://www.w3schools.com/quiztest/quiztest.asp?qtest=SQL
</res-comment>
</resource>
</resources>
<exercises>
<exercise>
<exerciseNr>1</exerciseNr>
```

```

<question>What does SQL stand for?
</question>
<answers>
<answer ans-value="false" >
<ans-number>1</ans-number>
<ans-content>Structured Question Line</ans-content>
</answer>
<answer ans-value="true" >
<ans-number>2</ans-number>
<ans-content>Structured Query Language </ans-content>
<ans-comment>Sql is presented in this lecture.</ans-comment>
</answer>
</answers>
<exe-comment>Hyper Dictionary:
http://www.hyperdictionary.com/dictionary/SQL
</exe-comment>
</exercise>
</exercises>
</subject>
</subjects>
<perform>
<by-teacher>Kjell Age Bringsrud</by-teacher>
<perform-date format="dd.mm.yy">12.05.2004</perform-date>
<perform-comment>Guest-teacher</perform-comment>
</perform>
<lecture-comment>The lecture is very important for the next exam!
</lecture-comment>
</lecture>
</lectures>
</course>
</education>

```

---

### The XML of *Course info.* menu option

---

The information of the XML document represented in figure 10.3, has been retrieved by the following XSL formatting. This XSL file has been defined within the XML file by the following processing-instruction (Sec. 10.1.1)

```
<?xml-stylesheet type="text/xsl" href="course.xsl" ?>
```

The layout result will be provided for the menu option *Course info.*

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
<xsl:template match="/education">

<html>
<head>
<style>
BODY
{ background: #bb88dd;
font-size : 10pt;
font-family : Arial, Helvetica, sans-serif;
text-align : center;
color : purple;

```

```

}
table{
width: "80%";
background-color: #bb88bb;
border: 2px solid purple;
color: purple;
}
th {
border: 1px solid purple;
background-color: #DDA0DD;
}
tr,td {
border: 1px solid purple;
}
</style>
</head>

<body>
<center>
<h3><xsl:text>Course Presentation (by XML)</xsl:text></h3>
<xsl:for-each select="course" >
<xsl:apply-templates select="."/>
</xsl:for-each>
</center>
</body>
</html>
</xsl:template>

<xsl:template match="course" >
<table>
<tr>
<th>Code</th><th>Name</th><th>Relevant information:</th>
</tr><tr>
<td><center>
<b><xsl:text> INF </xsl:text>
<xsl:value-of select="substring-after(@course-code,'_')" />
</b></center></td>
<td><center><xsl:value-of select="course-title" />
</center>
</td>
<td><xsl:value-of select="course-comment" /></td>
</tr>
</table>

<p><br /> </p>
<xsl:apply-templates select="./teachers"/>
</xsl:template>
<xsl:template match="teachers" >
<table>
<tr>
<th colspan="4">Responsible Teachers</th>
</tr><tr>
<th>Name</th><th>Role</th><th>E-mail</th><th>Homepage</th>
</tr>
<xsl:for-each select="teacher" >
<tr>
<td><xsl:value-of select="name" /></td>

```

```

<td><xsl:value-of select="role" /></td>
<td><xsl:value-of select="email" /></td>
<td><xsl:value-of select="homepage" /></td>
</tr>
</xsl:for-each>
</table>
</xsl:template>
</xsl:stylesheet>

```

---

The XSL of *Course info.* menu option

---

The layout result produced by deploying the XSL formatting 10.3 to the XML document 10.3 is presented below.

Online Database Course (ODC)

---

Course Presentation (by XML)

Code	Name	Relevant information:
INF 102	System Development	The course is virtual as a trial version!

Responsible Teachers			
Name	Role	E-mail	Homepage
Teacher 1	Main teacher	teacher1@ifi.uio.no	<a href="http://heim.ifi.uio.no/~teacher1">http://heim.ifi.uio.no/~teacher1</a>
Teacher 2	Guest teacher	teacher2@ifi.uio.no	<a href="http://heim.ifi.uio.no/~teacher2">http://heim.ifi.uio.no/~teacher2</a>

Figure 10.1: The layout of the *Course info.* menu option

## Chapter 11

# Java Server Pages (JSP)

### 11.1 Background

For developing the functionalities *Create Test* and *Test Yourself* of my *ODC* on-line course example (Chap. 14), the Java Server Pages (JSP) technology (Chap. 11) was used to communicate with MySQL relational database (RDB) (Sec. 5.5.2). This provided different testing functionalities by various developed application program interfaces (API).

### 11.2 What is JSP?

Java Server Pages (JSP) as a Java-based technology, provides an easy and secure way for creating **dynamic** Internet-based materials. JSP provides the possibility for creating and managing user **sessions** and **transactions**, which are supported by a set of built-in objects and methods used for developing Web applications.

#### 11.2.1 The advantages of using JSP

JSP technology used for developing Web applications provides the possibility for creating dynamic Web-based material. In addition, its applications are independent of both platform and server types.

JSP makes it possible to separate the **developing** process of a Web page content from its **presentation** process. In addition, JSP provides the advantage of reusing tags and objects, which simplifies the developing and maintenance processes of Web applications. Moreover, JSP contains a set of built-in objects that can be used to create and manage user sessions and transactions [18].

### 11.3 JSP & database connection within my trial 'ODC' system

The following JSP coding as part of the *testYourself.jsp* source file (App. A) is developed for the menu option *Test Yourself* (Sec. 14.3.2), which establishes connection to MySQL database for running a SQL statement. The JSP coding indicates that first



of all, the specific MySQL driver must be downloaded. Then the connection will be established when valid user-name and password are provided.

```
<%
String driver= "org.gjt.mm.mysql.Driver";
Class.forName(driver);
String mysql_base="jdbc:mysql://tafla.ifi.uio.no/MyUsername";
String user_name="MyUsername";
String password="MyPassword";
try{
Connection con1=DriverManager.getConnection
                (mysql_base, user_name, password);
...
}%>
```

## 11.4 JSP & RDB & HTML

The following JSP coding in the *testYourself.jsp* source file (App. A) is developed for the menu option *Test Yourself* (Sec. 14.3.2), which makes connection to MySQL database. This generates the following HTML-form (Chap. B) to receive user's data, which is used for delivering the desired test materials on the clients browser screen.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<link rel="STYLESHEET" type="text/css" href="../style.css" >
</head>
...
<form action="continueTestYourself.jsp" metode="post">
</p>
</font>
<%@page import="java.sql.*" %>
<%
String driver= "org.gjt.mm.mysql.Driver";
Class.forName(driver); //downloading mysql driver
String mysql_base="jdbc:mysql://tafla.ifi.uio.no/MyUsername";
...
buf1.append("<table align=\"left\"><th>Tests:</th><tr>"+
            "<td align=\"center\"><select name=\"testNr\">"+
            "<option selected value=\"\"> Select a nr.</option>");
...
buf2.append("<table align=\"right\"><th>Subjects:</th><tr>"+
            "<td align=\"center\"><select name=\"questionSubject\">"+
            "<option selected value=\"\"> Select an item.</option>");
while (res2.next()){
...
}%>
...
</form>
</table>
</body>
</html>
```

The JSP coding defined between the two signs `<% JSP codes %>` are generating the content materials for a *Test* object while HTML tags are taking care of the presentation parts such as the following HTML-form delivered on the browser screen. All layouts generated for the menu option *Test Yourself (JSP)* are given in Appendix B.

## 11.5 Dynamic contents

Most part of today's Web sites are delivering **static** contents, which implies having content presentations totally independent of any kind of user information. In contrast, **dynamic** presentation delivers various content materials that will be differently generated based on the received user-specific information. For example, Net-Banking provides on-line presentation of customer personal accounts, which usually are generated based on some confidential account information such as user-name, account numbers, password, etc.

JSP as a Java based programming language, makes it possible to offer dynamic contents by Web applications. The following sections gives more discussion on developing dynamic contents for my *ODC* on-line course.

### 11.5.1 Providing dynamic content

To provide dynamic data within my *ODC* learning system, I used various HTML-forms to receive desired information from users. Then, the received data was used within Java Server Pages (JSP) codings enabled to make contact with my relational database (RDB) for providing the additional information for *generating the content*. For *content presentation*, again JSP can be used either with HTML as the publishing language of the World Wide Web, or with eXtensible Markup Language (XML) and eXtensible Style Sheet Language (XSL).

### 11.5.2 Dynamic contents within my *ODC* system

The *Test Yourself* menu option (Sec. 14.3.2) of my trial *ODC* learning system is able to generate learning test materials based on the received information from students. The functionality of this menu option starts by the user interface illustrated in the Appendix B. There, by indicating either both *test number* and *subject* or just each of them alone a *Test* object material will be generated. The source files used for developing the functionality of *Test Yourself* menu option is provided under appendix A and their layouts are available under Appendix B.

Figure 11.3 represents a dynamically generated testing material developed based on the specific *test number* and *subject* values. For generating the dynamic test materials for the *Test Yourself* menu option, the JSP, RDB and HTML technologies (Sec. 11.4) are used.

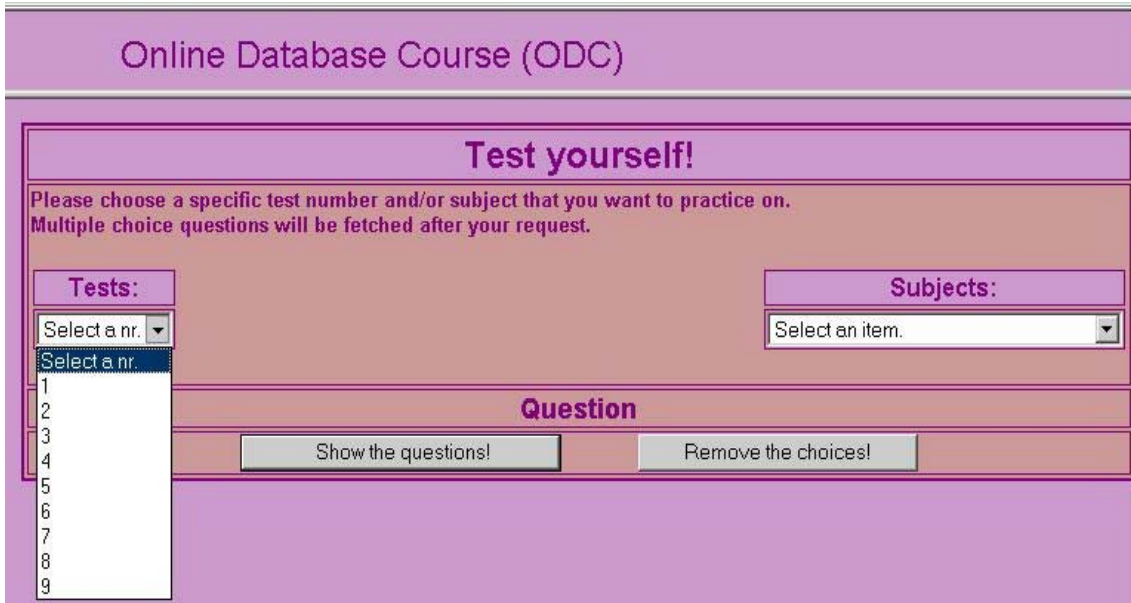


Figure 11.1: *Test Yourself* menu option: dynamic presentaion of retrieved test numbers

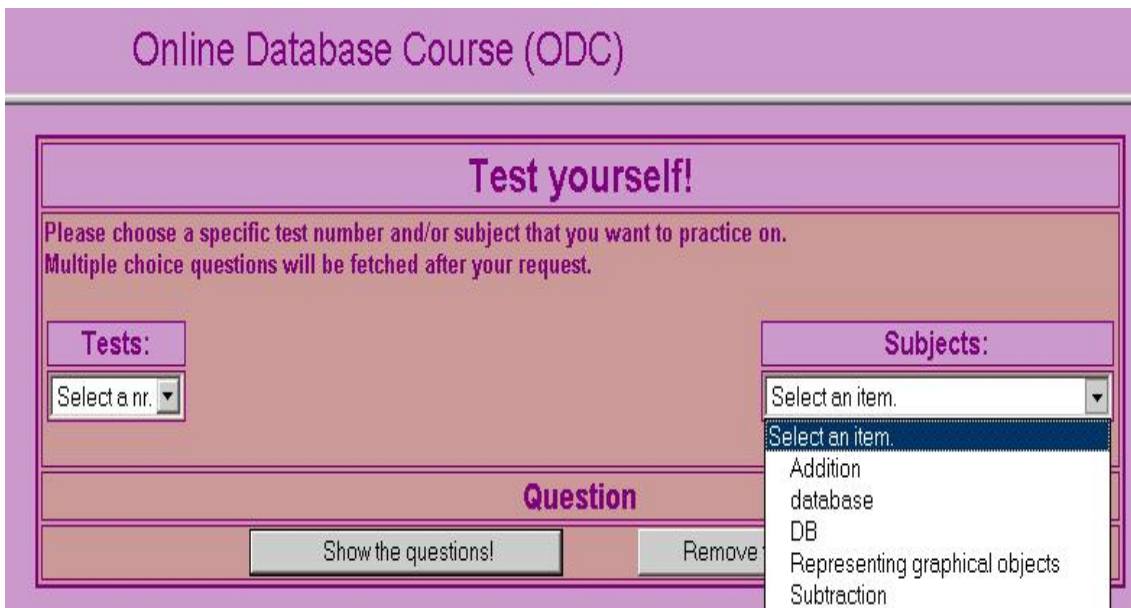


Figure 11.2: *Test Yourself* menu option: dynamic presentaion of retrieved subjects

### 11.5.3 Dynamic content provided by JSP & XML

The combination of JSP, RDB and HTML technologies was used in the development of the functionalities of the two menu options *Create Test* and *Test Yourself* (Sec. 14.3.1, 14.3.2) in my *ODC* trial e-learning system. However, due to limited time and similarity in developing process, a new presentation of the same generated content by XML and XSL, has not been fully completed. In fact, the process of structuring information in

Online Database Course (ODBC)	
Subject: Addition Test number: 1 Number of fetched questions: 2	
Question nr. 1: 4+40	
answer 1: 404 answer 2: 440 answer 3: 044 answer 4: 44	
The correct answer is :	
Nr. 1 <input type="checkbox"/> Nr. 2 <input type="checkbox"/> Nr. 3 <input type="checkbox"/> Nr. 4 <input type="checkbox"/>	
Question nr. 2: 6+7	
answer 1: 11 answer 2: 23 answer 3: 33 answer 4: 13	
The correct answer is :	
Nr. 1 <input type="checkbox"/> Nr. 2 <input type="checkbox"/> Nr. 3 <input type="checkbox"/> Nr. 4 <input type="checkbox"/>	
<input type="button" value="Check my answers"/> <input type="button" value="Let me try again"/>	

Figure 11.3: A dynamic test material generated by my *ODC* learning system

the form of an XML document and using XSL, Cascading Style Sheet (CSS) and HTML for presenting the content on the Web browser screen have already been developed for my other menu option called *Show Test (XML)* (Sec. 15.1.1). Therefore, for presenting the content by using JSP, XML and XSL, it only suffices to replace the HTML tagged content presentation with the content presentation used for *Show Test (XML)* menu option.

## 11.6 User interactivity & Feedbacks

The main idea behind developing my *ODC* on-line course example was to provide an educational environment which is able to provide the most important communication possibilities and interactions between the students and the system which has the teacher role in any instructional activities. Generating proper feedbacks from the system to students, is made to be continual for informing the students when they are using the functionality of the *Test Yourself* option and the teacher(s) when they are using the *Create Test* functionality.

The following figures 11.4 and 11.5 represent some of the system feedbacks created to inform the users about the result of their interactions with the system (App. B).

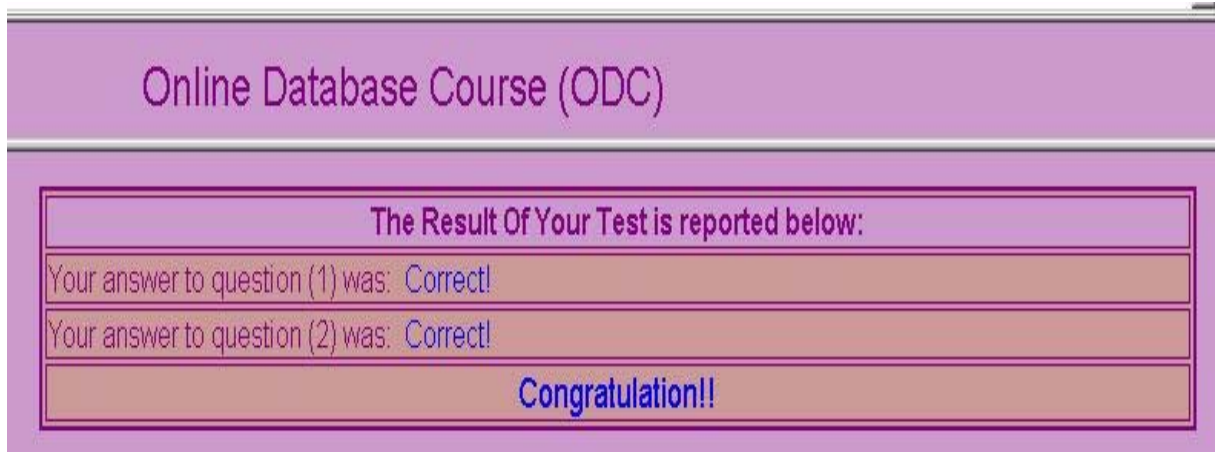


Figure 11.4: System generated feedback for answering correctly to all questions

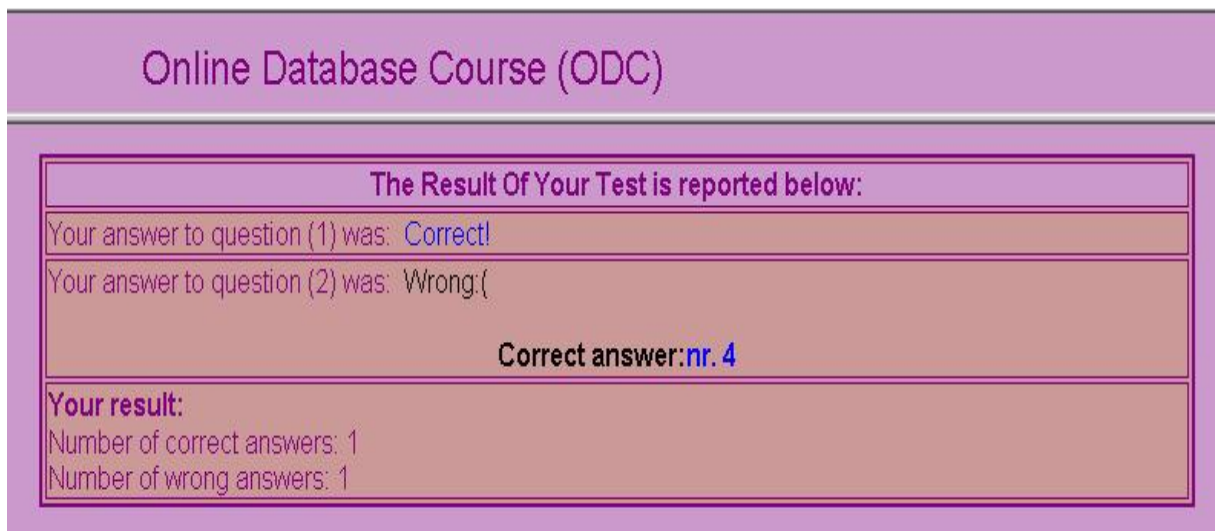


Figure 11.5: System generated feedback for reporting a result answer

## Chapter 12

# Flash Macromedia

### 12.1 Flash MX

Macromedia Flash 6, now called Flash MX, is a professional standard tool used for producing functional web applications. Flash provides the opportunity of using multi media such as text, image, sound and animation in developing movies used within web sites. Most Flash movies are developed for entertainment and e-commerce purposes.

A Flash application file is postfixed by ".fla" that later can be converted into a playable version with the same name but postfixed by .swf<sup>1</sup>. Then the published Flash movie (.swf) can be embedded in a HTML file (Sec. 13.3.4) stored on the server. Flash movies can be visited on Web browsers that are equipped with Flash player plug-in.

### 12.2 Why using Flash?

Flash is asynchronous which means it begins to load the XML file but doesn't pause the movie to wait for a response. Instead, it begins loading the Extensible Markup Language (XML) file and continues playing the SWF file. This is a very important concept. If one immediately tried to access the nodes and attributes from within the XML packet before it had finished loading in, Flash would return the value undefined for many of the values which could lead to unexpected results within the movie. By using the onLoad method to detect when the file has finished loading, Flash automatically executes the declared function. The function takes one optional parameter, which tells whether the file was able to be successfully loaded and parsed or not. If one didn't check whether the XML file was able to be loaded before trying to access the XML packet the SWF file is likely to experience unexpected errors or possibly fail to work at all.

---

<sup>1</sup>SWF is pronounced swiff, which stands for Shock Wave Flash Object

## 12.3 Graphic standards: Vector & Raster

There are two main standards, called **vector** and **raster**, for representing graphical information on the Internet. Flash has the advantage to combine both vector and raster standards. In Flash, vectors are used for presentation of the drawn parts of images. And vector representation combined with bite mapping, is used to import the additional information that can only be represented by raster techniques.

Raster images such as satellite photos, which are usually used for representation of the real world objects, can be mapped into vector graphic features by using coordinate systems (points and lines) and mathematical transformations. The used features are labeled by various attributes such as color, size and rotation stored in an attribute database/attribute table. Later, based on the stored attributes, much of the analysis in vector systems are performed to map the resulted image.

By raster technique, static images are constructed by bite mapping where the position and color of each single pixel of image are described. Most images available on the Web are of the raster type. On the other hand, vector graphic images consist of lines with defined coordinates and the surface in between. Therefore, the size of vector files become much less than the size of their raster versions. However, raster technique is more appropriate for representing rich colored images. Because representing the same rich colored image by vector graphic will become very complicated and the size of the file will become much larger than its raster version [35].

In general, downloading time is an important issue that must be considered when working with Web development. Users can easily lose their interests when loading a Web site takes a long time. Because of that, using vector graphics is more suitable for Web graphical developments. For this reason, Macromedia Flash enabled for delivering vector graphics, has become a popular tool for developing Web applications.

## 12.4 Raster images

By raster technique an image will be simply represented in a 2-dimensional matrix where each of its squares represents a pixel of the image with a color code. Raster files are postfixed by one of the following .jpeg, .gif, .bmp, .png, .tif, ...

Raster technique is appropriate for representing rich colored pictures when thousands of different colors are involved, because every pixel of the picture can be defined by its own color. The figure below, shows how raster graphics are represented by pixels containing color codes<sup>2</sup>.

---

<sup>2</sup>The image is fetched from: <http://www.answers.com/topic/raster-graphics>

### 12.4.1 Disadvantages of raster images

There are some disadvantages with raster technique. One example is that by zooming in on a raster image the pixels of the image will become bigger and bigger. After a while, its pixels become so big that the image will be presented as unrecognizable squares.

The large size of big raster images has the additional problem of long loading times. Yet another problem occurs when a raster image is displayed on other screens with different resolution than the one used in its creation.

## 12.5 Vector graphic images

The vector graphic technique uses coordinates and mathematical transformations to define every line between two specific points within an image. Therefore, size of vector graphic images become comparatively smaller than the corresponding raster images. Macromedia Flash uses the vector graphic technique for creation of movies. Then **file streaming** technique, which lets the content of a file become gradually loaded, is used for loading a Flash movie into the user's computer and displaying it on the screen.

### 12.5.1 Advantages of vector graphic

Flash movies are generated by vector graphic images with gradually loading process. That is, the first loaded part of an animation is started to play while the rest of the movie is being loaded. The same mixed process of *loading & playing* continues until the whole animation file is loaded and played [35]. Contrary to raster images, displaying of vector graphics images is independent of the resolution of the screen (Sec. 12.4.1). Therefore, the quality of the resulting movies will always be kept under zooming.

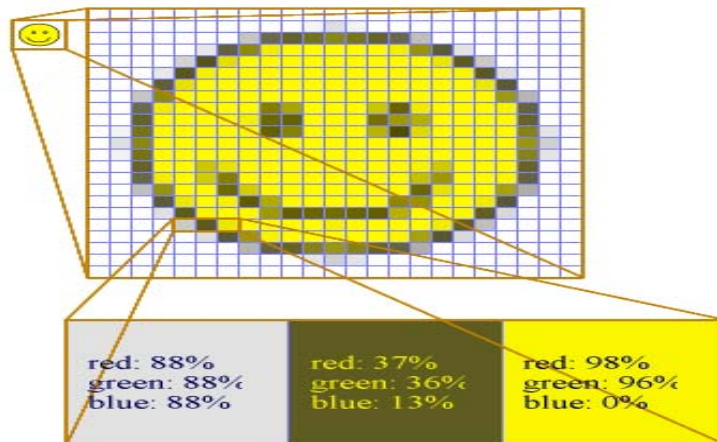


Figure 12.1: Raster images are built up by pixels.



## 12.6 ActionScript: Flash programming language

Macromedia Flash has its own programming language called **ActionScript** which has been developed based on the same specification language used for JavaScript<sup>3</sup>. This similarity<sup>4</sup> makes learning ActionScript easy for web developers and has increased Flash popularity into a level that about 94% of all web browsers on the Internet are equipped with various version of Macromedia Flash players<sup>3</sup>. ActionScript language enables Flash movies to work with HTML and XML and therefore as a scripting language is able to communicate with other parts of the web. Flash is a robust solution for delivery of dynamic web content from back-end recourses like databases<sup>5</sup>.

ActionScript has some built-in commands called *functions*<sup>6</sup> that are able to perform certain jobs. For instance, the **goAndStop** function will move the playhead from one frame to another frame specified as a parameter and stop there ( [41], P. 297). In addition, ActionScript lets developers define their own functions such as the function developed in this thesis and used inside 'courseTests.swf' Flash movie (Sec. 13.3.3):

```
function ShowData(MyNode) {...}
```

*Method* is another concept used in ActionScript that is very similar to *function*. However, a *method* must be used by an object to perform certain jobs. A method will be called by adding its name to the name of an object instance ( [41], P. 306). The two 'load' and 'onLoad' methods are called by my 'testsData' XML object used in the scripts of my 'courseTests.swf' Flash movie (Sec. 13.3.3):

```
testsData.load("tests.xml");
testsData.onLoad = fileLoading;
```

### 12.6.1 Using Flash MX

The content of a Flash movie is built in **layers**. To develop a Flash movie, one must create the desired layers and list them in a column on the left side of the Timeline that contains the sequence of layouts for the movie. Then for adding multi media objects into the movie, **keyframe(s)** must be added into the actual timeline(s).

The **toolbar** window is divided into *Tools*, *View*, *Colors* and *Options* parts, and contains the main functionalities for creating and modifying objects used in Flash MX movies. In addition, Flash has a **library** that can be used for storing created items as symbols. Later, more instances of the stored symbols can be created.

Flash offers two methods for creating animations: **frame-by-frame** and **tweeing**. On a frame-by-frame animation, there are the contents of frames which are changed. While in tweeing, Flash will automatically construct the necessary changes needed for illustrating the movements between two keyframes ( [35], P. 42).

<sup>3</sup><http://www.devnewz.com/devnewz-3-20030519XMLforUIDescriptioninFlashMX.html>

<sup>4</sup>ActionScript has based in the JavaScript standards but it isn't 100% completed with JavaScript.

<sup>5</sup>ColdFusion is a program used for creating dynamic datadriven Flash interfaces developed based on CFML (ColdFusion Markup Language) or JSP (Java Server Pages) [35], P. 39.

<sup>6</sup>In earlier version of Flash MX 2004, *functions* were called *actions*.

For storing information, ActionScript, like other programming languages, provides the possibility for developers to define their own **variables** without specifying any variable type. ([41], P. 307). The 'testsData' variable defined by the following ActionScript code, instantiates an XML document object (Sec. 13.3.3) for my 'courseTests.swf' Flash movie : `testsData = new XML();`

### 12.6.2 Text types in Flash MX

The possibility of having the following text types for a text element, is provided in the properties window of Flash MX: **Dynamic**, **Static** and **Input**. Any created text field can contain these text types in addition to its own name, which can be used inside ActionScript codes.

### 12.6.3 Publishing by Flash

When flash application is finished, one should make a published version of it which may be done in Flash. The result will be a .swf file which may later be used inside a HTML file (Sec. 13.3.4).

### 12.6.4 Flash MX invisible buttons

By moving the mouse over the defined areas that are covered by the invisible buttons, some relevant information about that object are displayed on the screen.

### 12.6.5 Flash MX library elements

The library system in Flash makes it possible to store various macromedia objects in its system. So a developer may prepare all of the necessary elements at once and then store them as independent symbols in the library for later use. The Flash library system provides the possibility for **reusing** and **manipulating** the stored objects as many times as desired.

## 12.7 Interactivity by using Flash macromedia

Flash is a technology that has become a full web application development tool. It has multimedia capabilities that let one create attractive features from simple animations to high-quality cartoons with sound and interactivity in graphical web sites. Flash can be used to develop any kind of home pages from simple sites to cooperative e-commerce or e-learning sites.

## 12.8 Static Versus Dynamic Content

**Static content** refers to a document whose content is always the same material saved on a web page. The material can be represented on the screen at any time when a

request for its site is received by a Web browser.

A **dynamic document**, on the other hand, is the content of a web page that can be changed by a program at any time when the page is requested. The data entered by a user is often used by the program to identify which material must be displayed as the content of the page.

Dynamic documents are usually generated from other sources like databases when a script receives a user's personal data and then based on them, the content of the requested Web page will be constructed. For instance, Web sites used for electronic mail, bank and commerce systems are generated for each individual user based on her/his entered personal data. Because of the popularity for having dynamic contents within Web applications, the issue has come more in focus.

Macromedia Flash as a presentation tool was also challenged to conjunct its script language, ActionScript (Sec. 12.6), with some other server scripting languages such as Perl, PHP, ASP<sup>7</sup> and JSP to represent dynamic contents within Flash movies ( [35], P. 588).

### 12.8.1 Integrating Flash with XML

Flash MX has been integrated with XML technology and therefore, XML files can now be applied as external data sources into Flash MX movies (Sec. 13.2). Flash MX is equipped with a fast XML parser. Integrating Macromedia Flash with XML has provided the opportunity to use XML technology for taking care of content materials and use Flash as a multi-media tool for presentational issues. XML data files can be used to contain output materials as the end results for Flash movies that can easily be changed and updated even without a need to open Flash applications.

---

<sup>7</sup>Microsoft Active Server Pages (ASP) pages use Microsoft VBScript or JScript.

## Chapter 13

# Integrating XML With Flash MX (Dynamic Text)

### 13.1 Background

As part of my Thesis, I decided to develop various learning objects by using various visual and audio effects. Therefore, I chose to work with Macromedia Flash MX which facilitates working with XML and multi-media presentations.

In this chapter, I have presented my various developed learning materials that were created as experimental examples for my Thesis. One of these examples is a Flash movie that represents a Flash tutorial developed as the result of my group project *Click & Learn Flash* developed for the *Information design* course (Sec. 16.4).

My developed learning and training examples, are presented in the various sections of this chapter. Some of the development processes of my Flash movies are explained in details while some of them are briefly discussed.

In this work the following features were in focus: being **creative**, providing interactivity, representing dynamic results and integrating XML with multi-media issues. Some of my Flash movie exercises are able to give feedback to students and provide **interactivity** and communication with students by responding to their interactions and delivering **dynamic** feedbacks. In one of my Flash movies, I have concentrated on working with **XML** and tried to integrate my XML learning material within my Flash MX movie.

My intention of creating such trial learning objects has been to experience how difficult and time demanding it is to produce various learning materials by using and integrating various technologies.

## 13.2 Integrating an XML learning object with Flash MX

In this section, I have explained one of my experiences during working with Macromedia Flash for developing learning materials within different Flash movies. In this section, my courseTests.swf movie developed by Flash MX is explained. The Flash movie is able to read XML data file (Sec. 13.3.1) and then represent its content into a text box.

The trail example focused on working with XML data integrated with Flash. Therefore, the presentation part is kept simple and no fancy graphics objects are used for this part. The document management system (Sec. 8.2.2) will manage to find the relevant XML file for the courseTests.swf application file (Sec. 12.8.1). The text data from XML file will be read into the Flash movie and the result will be displayed on Web browsers equipped with Macromedia Flash Player (Sec. 13.3.4).

The Macromedia Flash MX is also able to receive XML data from a server-side scripting and present them as simple text or use them within other animated objects. By using server-side scripting we can pull out dynamic data from a database and then generate an XML output file consisting of proper data elements for Flash movies (Sec. 12.8.1). The retrieved information may be represented, for example, as ordinary text or as data for rendering chart diagrams [30].

Separating XML data from the presentation part of Flash movies makes the movies independent of the changing process of content data. This facility gives more flexibility to produce movies with dynamic contents that can easily be updated without any effect on their presentation parts.

### 13.2.1 Flash movies and Preloaders

Flash movies consist of **loading** and **presenting** scenes. The loading part of a movie is used for its *loading process*, which is the time needed for downloading the entire Flash movie, .swf file. In general, **Preloader** refers to any sequence of multimedia materials to be presented on the screen during the time a .swf file needs to be loaded [30]. Developing preloaders from simple texts to advanced animations used for downloading Flash movies that are time consuming, is very popular. The loading time used for all my Flash movies and examples were quite short, so it was not necessary to develop any preloaders for them.

## 13.3 Developing process of my Flash movie

My Flash movie, courseTests fla, developed by Flash MX is able to display the stored information of my external XML file, tests.xml. The published version of my Flash movie file, courseTests.swf, must be populated with my XML file in the same directory. The movie is represented as the menu option *Show Tests (XML)* of my *Online Course* example developed as a trial learning system for my thesis.

For creating the movie, first the two layers Content and Action were created. The Action layer, which must be placed on top of the other layers, was created to contain all scripting codes for my movie. Then, for each layer, one keyframe was defined. In the first keyframe of Content layer, a text box with the name *output\_data* was created. This text field must be defined to contain dynamic text (Sec. 12.6.2) and its *Render Text as HTML* attribute must be chosen. Later, this text box is referred in my ActionScript codes to represent the XML text content (Sec. 13.3.1) into my movie (Sec. 13.3.3).

For developing my Flash movie, I followed the following stages:

- Creating an XML data file for working with Flash (Sec. 13.3.1).
- Creating the visual elements in my Flash movie for representing output data.
- Developing ActionScript codes used for reading my XML data and displaying them into my movie.

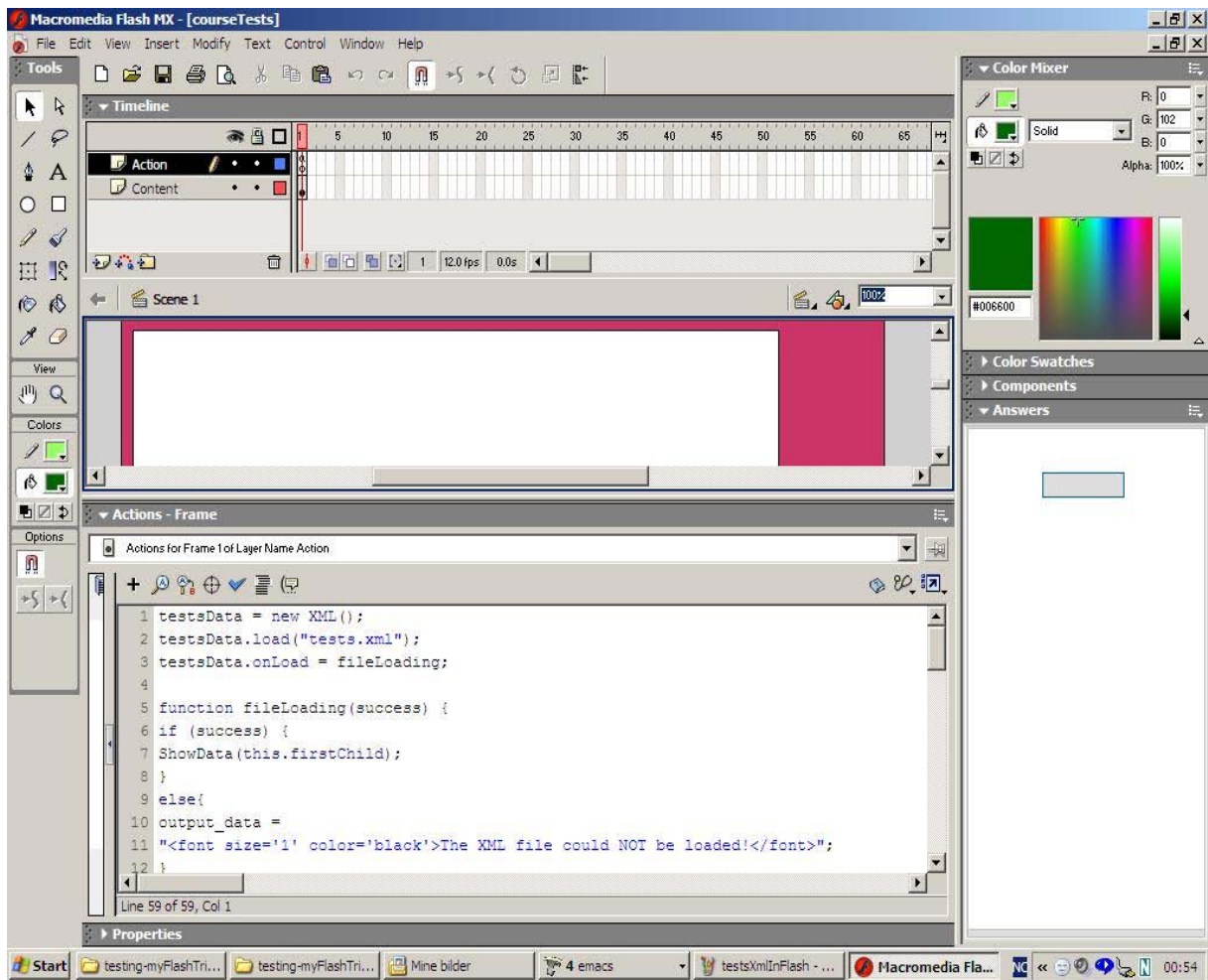


Figure 13.1: The courseTests.fla file developed by Flash MX

### 13.3.1 XML data file for my Flash movie courseTests.swf

The data structure used for my XML data within tests.xml is explained below. The file contains a list of questions and answers belonging to a *test* element.

It is important to note that XML files developed for Flash movies must only contain XML data elements used within the DOM of the XML file (Sec. 8.1.4). Therefore, the regular XML processing instructions (PI) must not be included in my XML file. In addition, the XML file can not contain any DTD (Sec. 9.1) declarations or XSL formatting (Sec. 10.1). The XML data elements stored inside my tests.xml file used for my movie, are discussed below.

```
<test>
<question>
<nr>1</nr>
<subject>Semistructured DB</subject>
<content>What does XML stand for?</content>
<comment>XML needs XSL technology!</comment>
<answer>Extensible Markup Language</answer>
</question>
<question>
<nr>2</nr>
<subject>Database</subject>
<content>What kind of database is Mysq1?</content>
<comment>The question was discussed in the class!</comment>
<answer>Mysq1 is a relational database</answer>
</question>
<question>
<nr>3</nr>
<subject>Extracting data</subject>
<content>What does SQL stand for?</content>
<comment>It is a query language.</comment>
<answer>Structured Query Language</answer>
</question>
</test>
```

---

Figure 13.3.1: The XML file tests.xml contains a list of questions and answers

---

The following tags are used within the data structure of my XML data file and therefore are applied in my ActionScript codes (Sec. 12.6) in order to pull out their data into my Flash movie. The root element of the document called **<test>** element, may contain so many *<question>* elements as its child nodes. Further more, each **<question>** data element acts as the parent node for its own child nodes that are from the following types: *<nr>*<sup>1</sup>, *<subject>*, *<content>*, *<comment>* and *<answer>*.

---

<sup>1</sup> *nr* stands for number.

### 13.3.2 My Flash movie courseTests.swf

The presentation display of my Flash movie consists of Content and Action layers. The *Action* layer contains all ActionScript codings defined for the movie (Sec. 13.3.3), while the Content layer contains just a simple text box used for showing XML data on the screen.

In my courseTests.swf movie, the use of XML by Flash was in focus and therefore, its introduction display is a simple text box that contains texts and no multi-media components. Several multi-media facilities were experimented during development of my other Flash movie that I have developed during the course *Information Design* at IFI<sup>2</sup> (Sec. 16.4). Below, the content of XML data file is retrieved and represented by different font sizes and colors.

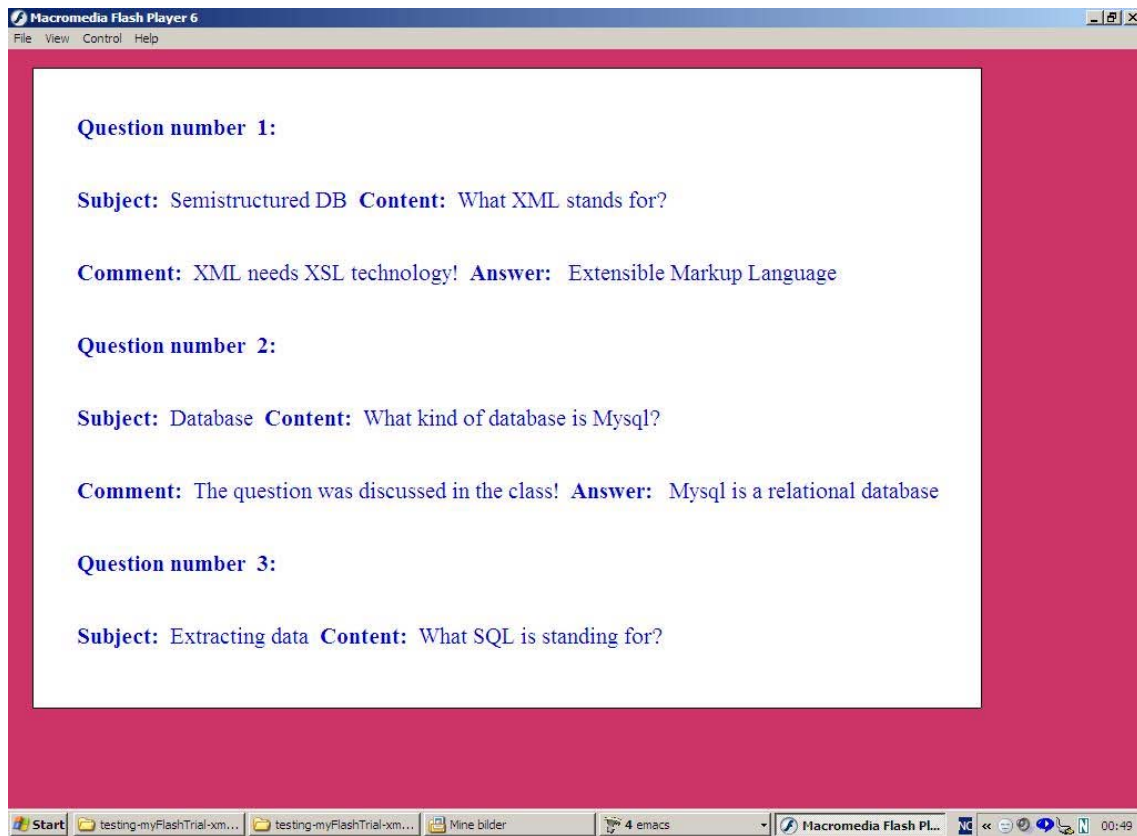


Figure 13.2: The published version (.swf) of file courseTests fla

### 13.3.3 My Flash movie's ActionScript

The Action codes defined for this movie, is started by loading my XML file. In case the XML document is not available to be loaded, an error message is sent to the screen. Otherwise, the XML data will be parsed and the desired information will be retrieved.

<sup>2</sup>The Department of Informatics in the University of Oslo in Norway



By defining local data variables, the XML data elements are fetched and put in my defined local variables that are named the same as their relevant XML elements such as nr<sup>1</sup>, subject, content, comment and answer. The values of variables will be shown within the display of my Flash movie [30].

The following ActionScript codings are defined inside the first frame of my *Action* layer defined for my movie application, courseTests.fla. The script codes will iterate through the tree structure of my XML data file to handle all <question> elements belonging to a <test> element.

```
testsData = new XML();
testsData.load("tests.xml");
testsData.onLoad = fileLoading;

function fileLoading(success) {
    if (success) {
        ShowData(this.firstChild);
    }
    else {
        output_data = "<font size='1' color='black'>" +
            "The XML file could NOT be loaded!</font>";
    }
}

function ShowData(MyNode) {
    if (MyNode.nodeName.toUpperCase() == "TEST") {
        output_data = "";
        question= MyNode.firstChild;

        while(question != null) {
            if (question.nodeName.toUpperCase()== "QUESTION") {
                nr="";
                subject="";
                content="";
                comment="";
                answer="";

                theNode = question.firstChild;

                while (theNode != null) {
                    if (theNode.nodeName.toUpperCase()== "NR"){
                        nr= theNode.firstChild.nodeValue;
                    }
                    if (theNode.nodeName.toUpperCase() == "SUBJECT") {
                        subject = theNode.firstChild.nodeValue;
                    }
                    if (theNode.nodeName.toUpperCase() == "CONTENT") {
                        content = theNode.firstChild.nodeValue;
                    }
                    if (theNode.nodeName.toUpperCase() == "COMMENT") {
                        comment = theNode.firstChild.nodeValue;
                    }
                    if (theNode.nodeName.toUpperCase()== "ANSWER") {
                        answer = theNode.firstChild.nodeValue;
                    }
                    theNode = theNode.nextSibling;
                }
            }
        }
    }
}
```

```

    }
    output_data += "<p><font size='1' color='red'><b>Question" +
        " number &nbsp;" + nr + " :</b><br><b>Subject:</b>" +
        "&nbsp;&nbsp;" + subject + "&nbsp;&nbsp;<b>Content:</b>" +
        "&nbsp;&nbsp;" + content + "<br><b>Comment:</b>&nbsp;" +
        "&nbsp;&nbsp;" + comment + "&nbsp;&nbsp;<b>Answer:</b>&nbsp;" +
        "&nbsp;&nbsp;" + answer + "</font></p>";
    }
    question = question.nextSibling;
}
}
}

```

---

Figure 13.3.3: ActionScript coding defined for courseTests.fla application

---

The following instruction from my tests.xml file, which contains descriptive data elements belonging to a test object (Sec. 13.3.1), specifies that this movie application is working with an XML document (Sec. 12.6.1). Therefore, the XML object belonging to Flash ActionScript language is used in the following code to initialize an XML variable for containing the text content of my XML document:

```
testsData = new XML();
```

The XML variable of Flash Actionscript has properties for initiating a new XML document and then loading my XML file defined by calling *load* method in the following script (Sec. 12.6):

```
testsData.load("tests.xml");
```

The XML variable *testsData*, associates the rest of the action codes with my tests.xml file. When the XML file is correctly loaded, extracting data from the XML document into the Flash movie will also be started. So the following *onLoad* method is used to check the file's loading process and call the *fileLoading* function to be executed (Sec. 12.6). The following method is defined to declare whether the XML file is successfully loaded or not:

```
testsData.onLoad = fileLoading;
```

Later, all XML data fetched from my XML document will be presented on the created *output\_data* window. The text box can contain several text lines because I defined the value *Multi-line* for its attribute (Sec. 12.6.2). When the answer to the Boolean *success* is true, the *ShowData* function will parse the XML document.

Now, if the XML file is successfully loaded, the whole content of XML file will be represented as a long text into the output window. However, the following ActionScript codes will fetch desired data and present them in a formatted layout. The *ShowData* function extracts the desired data from the tests.xml file (Sec. 12.6). Then the fetched information will be presented on my *output\_data* window on the screen.

In general, XML elements of a document contain a parent/child relationship and they



```
"&nbsp;&nbsp;&nbsp;"+content + " <br><b>Comment:</b>&nbsp;&nbsp;&nbsp;" +
"&nbsp;&nbsp;&nbsp;"+ comment+"&nbsp;&nbsp;&nbsp;<b>Answer:</b>&nbsp;&nbsp;&nbsp;" +
"&nbsp;&nbsp;&nbsp;"+ answer+ " </font></p>";
```

### 13.3.4 Publishing Flash movies

Later the compiled version of my Flash movie, `courseTests.swf`, is put inside a HTML page to be correctly displayed in Web browsers. My Flash movie is defined to be available under the menu option *Show Exercises (Flash)* on my *Online Database Course (ODC)* example.

The following code is placed into the HTML page used for publishing the Flash movie. The code shows that the used `<EMBED>` tag, which will be ignored by IE, must be placed within the `<OBJECT>` tag. Similarly, the `<OBJECT>` tag won't be recognized by Netscape Navigator and old Microsoft browsers and only the `<EMBED>` tag will be used by them to load Macromedia Flash Players and Flash movies (Sec. 12.1).

---

```
<OBJECT classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
        codebase="http://active.macromedia.com/flash2/cabs/
                swflash.cab#version=4,0,0,0" ID=flashbanger
                WIDTH="85%" HEIGHT="80%">
  <PARAM NAME=movie VALUE="courseTests.swf">
  <PARAM NAME=quality VALUE=high>
  <PARAM NAME=bgcolor VALUE=#000000>

  <EMBED src="courseTests.swf" quality=high bgcolor=#FFFFFF
        WIDTH="%85" HEIGHT="%80"
        TYPE="application/x-shockwave-flash"
        PLUGINSOURCE="http://www.macromedia.com/shockwave/
        download/index.cgi?P1_Prod_Version=ShockwaveFlash">
  </EMBED>
</OBJECT>
```

---

Figure 13.3.4: HTML embedding codes for putting Flash movies into Web sites

---

The HTML page contains two specific tags: `<OBJECT>` and `<EMBED>` that respectively are used by Internet Explorer (IE) and Netscape Navigator. Using only one of them, may cause browser incompatibility. Therefore, both of the tags should always be defined [41].

## Chapter 14

# Dynamic Learning Contents by JSP & RDB

### 14.1 Background

In this chapter, I have explained the development process of my experimental infrastructure e-learning system, which offers an on-line course example equipped with various learning functionalities. The trial learning system, which is called "**Online Database Course (ODC)**", has been developed as part of my thesis work.

On the left side of my *ODC* on-line course, a menu consisting of various menu options is provided. In front of the menu options' names, there are some technology names, which are indicating the main technology used for developing the functionality offered by that specific menu option.

My *ODC* learning system was prepared as a prototype module for experimenting the development process of creating various learning functionalities for representing a simple example of a learning management system (Sec. 3.2.2). By further extension in the future, the *ODC* on-line course system can be developed into more mature format and with more capabilities.

For developing the two functionalities *Create Test* (Sec. 14.3.1) and *Test Yourself* (Sec. 14.3.2) of my *ODC* on-line course example, the JSP technology (Chap. 11) combined with MySQL relational database (Sec. 5.5.2) and HTML coding, was utilized for offering various educational facilities by the different developed application program interfaces (API).

### 14.2 My developed '*Online Database Course (ODC)*' example

In general, learning management systems as explained in chapter 3 are considered as complex systems that are enabled to describe, discover and deliver learning contents and offer various learning related services within learning centers. In addition, learning management systems such as IMS are enabled to exchange learning materials

between different learning management systems.

The SCORM specifications are developed based on other standard specifications such as IEEE LOM or IMS Meta-data, IMS Content Packaging, IMS Simple Sequencing, AICC CMI<sup>1</sup>. Therefore, LMS based on SCORM will become a large and complex system that requires a quite complex and demanding development process.

The implementation of my *ODC* learning system as a mini trial example, was made to represent a test prototype for a learning management system. The system architecture used for developing my *ODC* LMS is represented in the section 14.2.1. Different technologies such as XML, SCORM, JSP, RDB and Flash were used for developing several services for my trial learning system, *ODC*.

During the development of *ODC* on-line course example, I tried the Classfronter<sup>2</sup>, a Norwegian learning management system for web-based learning facilities, which is used in some faculties and departments of the University of Oslo.

Since it is important to have the communication facilities between the students and a learning system, various learning objects provided by the *ODC* on-line course system, are created to follow users' activities in order to give proper feedbacks and processing reports to them. Providing some kind of continuous "dialog" between the system and students are developed. For example when:

- my various Flash movie as learning objects are answered by students (Sec. 16.2.1)
- different test objects are created by teachers (Sec. 14.3.1)
- student messages are sent to course leader(s) (Sec. 14.3.3)
- various test objects or HTML forms are answered by students (Sec. 14.3.2, 15.1.1).

### 14.2.1 System architecture of my *ODC* learning system

The system architecture used for developing my trial *ODC* learning system has been illustrated in figure 7.5. The system was developed by considering the facilities provided within IEEE-LSTA (Sec. 7.5).

### 14.2.2 The main ideas for my *ODC* learning system

One of the ideas behind my *ODC* learning system was to provide a pleasant instructional environment for students to work with the course materials. Therefore, **color**

---

<sup>1</sup>The full names of these abbreviations are represented at the end of the thesis under the "Abbreviations" section. The exact used versions of these standards are available at: <http://www.imslobal.org/af/afv1p0/imsafwhitepaperv1p0.html>

<sup>2</sup>More information about Classfronter is available at <http://www.hivolda.no/index.php?ID=12677&lang=nyn> and <https://tavle.uio.no/> More comprehensive explanations about its services are represented in: [http://www.usit.uio.no/it/dlo/cf/info/veiledninger/cf\\_veiledning.html](http://www.usit.uio.no/it/dlo/cf/info/veiledninger/cf_veiledning.html)

**effects** for developing various interfaces are considered. Producing **user-friendly interfaces** was another objective of the work. Therefore, different sites of the *ODC* system are developed to be **easy-to-understand** and simple-to-learn while continuous system reporting is supplied.

Understandable sites with user-friendly functionalities and interfaces provided by e-learning systems will make learning processes more enjoyable and productive. In addition, a learning system must be able to produce various feedbacks where the result of any active procedure are reported back to its students. My trial learning system's ability in generating various reports and feedbacks, was implemented to represent the same **reporting** capability that must be provided within any LMSs.

## 14.3 Content and Presentation Separation

To separate the content of a learning object from its presentation, the following trial experiences represented as the menu options *Create Test (JSP)* and *Test Yourself (JSP)* of my *ODC* system, were developed. The *Create Test (JSP)* provides the facility for the course leader to either define new *test* objects or complete the existing *test* objects (Sec. 14.3.1).

### 14.3.1 The *Create Test (JSP)* menu option

Under this menu option the course leader has the possibility to define new *test* material for the students. Later, an defined *test* can be completed by more questions. The process of editing and manipulating an existing *test* object, must be done directly within the MySQL database.

By using JSP and RDB the content of every *test* as a learning object is dealt with. While by using HTML, the stored *test* objects are represented on the screen.

The *Create Test* menu option calls the *createTest.jsp* file (App. A) that provides an HTML form for Teacher to create or extend a *test* by filling out a HTML form (App. B) and creating new multiple choice questions for a *test*. Further, the entered data will be submitted to the *continueCreateTest.jsp* file (App. A) which is responsible to check the entered data. After the data are accepted, they are registered as a new question belonging to a *test* into the *MultiplechoiceQuestions* table of MySQL database. During the process of creating a *test object*, and depending on the occurred situations, different feedbacks will be sent into the screen to inform the user.

The following figures illustrate some of the main processes, such as creating new questions into RDB, and then reporting the entered data back to user. Further processes will not continue until the feedback sent to the user is answered and confirmed in order to activate the process of registering data into the database.

Figure 14.1: *Create Test* menu option: existed test numbers and subjects are presented

Figure 14.2: *Create Test* menu option: new question is given

### 14.3.2 The *Test Yourself (JSP)* menu option

For developing the menu option *Test Yourself (JSP)*, the JSP and RDB technologies have been used. By this menu option the created *test* objects (Sec. 14.3.1) can be fetched



**Online Database Course (ODC)**

**The following question is entered:**  
(Please confirm to register the question into DB)

Test nr:	3
Subject:	Mathematics
Question:	55-23
<b>Answer's alternatives</b>	
Answer1:	34
Answer2:	32
Answer3:	12
Answer4:	123
Correct answer:	Nr. 2

Confirm:

[<< Back](#)

Figure 14.3: *Create Test* menu option: feedback and request for confirming

**Online Database Course (ODC)**

**The question is successfully registered into the database.**

[<< Back to the main page](#)      [>> Create another question](#)

Figure 14.4: *Create Test* menu option: confirmation of database registration

from the MySQL database based on a desired test number or a specific subject or a combination of both of them,.

Creating an interactive learning system, which is able to deliver dynamic learning contents back to students, will be the most encouraging facility that can motivate students to continue using a LMS. Therefore, by developing the *Test Yourself (JSP)* menu option of my *ODC* learning system, I focused on creating the possibility for students to be able to choose any specific test number or any of the test subjects used for categorizing the various stored questions within my relational database (Sec. 5.6).

The following figure displays the first HTML-form to start the functionality of the *Test Yourself (JSP)* menu option. All of the layouts belonging to this functionality are provided in the Appendix B.

Online Database Course (ODC)

**Test yourself!**

Please choose a specific test number and/or subject that you want to practice on.  
Multiple choice questions will be fetched after your request.

**Tests:**  **Subjects:**

**Question**

Figure 14.5: The first HTML-form used for starting the testing process of *Test Yourself (JSP)* menu option.

Every question belonging to a created *test* object (Sec. 14.3.1), will have its own *subject*, which will later be used by this menu functionality to aggregate all stored questions that are registered under the same desired subject received from a student.

The *Test Yourself (JSP)* menu option developed by JSP, RDB technologies combined with HTML, makes it possible to create tailored *test* objects based on the desired subject received by every student. The system will generate a *test* object full of questions related to the chosen subject and display it back on the screen. Then an answered *test* object can again be sent into the system where the users' results will be evaluated and based on that a final report will be generated and sent back on the screen in real-time<sup>6</sup>.

Learning systems which generate dynamic contents and reports tailored for each student, can be quite attractive and encouraging for students to continue using a LMS in order to improve their knowledge and working with learning materials.

The JSP source files developed for the functionality of the *Test Yourself (JSP)* menu option are presented in the Appendix A and the layouts generated for this functionality are available in Appendix B.

### 14.3.3 The Ask Question (JSP) menu option

Providing an easy dialog and communication between the students and the teacher of a course is one of the major activities that must be provided in any learning process. Learning management systems (LMS) must be able to provide communication facilities for the learners of their systems. In fact, I believe a well designed LMS can in many aspects fill the teacher role and in fact act as a "e-teacher" itself. Therefore, the *Ask Question (JSP)* menu option was developed within my trial LMS which provides the possibility for students to send their questions directly to the course leader without any need to open another e-mail system. In further developments, my trial *ODC* system can be extended to provide e-mail accounts for students for receiving letters in order to create complete communication between both sides of an educational process consisting of a student and a teacher whom I believe can be replaced by a well designed *e-teacher*.

## 14.4 The development of my learning objects

The various facilities provided on my trial *ODC* learning system, make it to look like a simple version of learning management systems (Sec. 3.2.2). This trial learning system is able to **create** and **deliver** test materials created by teacher. My Flash movies provide various training materials for students. Moreover, **searching**, **selecting** and **assembling** are deployed for creating the learning objects delivered by menu option *Test Yourself (JSP)* (Sec. 14.3.2, 14.3.1). The following figure shows the technologies used for creation of my learning objects. As it shows, my learning system example is equipped by Flash MX, JSP, XML, XSL, DTD combined with the SCORM standards for developing and deploying various learning objects (Sec. 13.3, 16.4, 16.1).

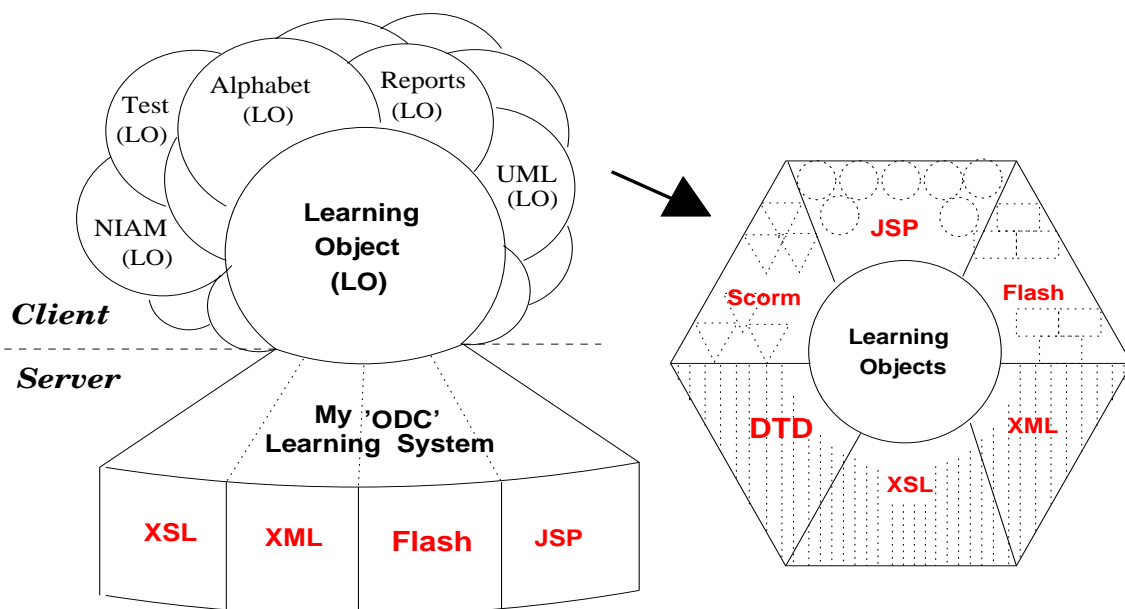


Figure 14.6: Technologies used for creation of my learning objects

## Chapter 15

# XML Technologies for Development of Learning Materials

### 15.1 Background

In this chapter, the development process of some of the functionalities of my developed *ODC* prototype system that were designed to offer a simple version of some of the LMS functionalities (Sec. 3.2.2) is represented.

The main developments presented in this chapter belong to my *ODC* on-line course menu options *Show Test (XML)* and *Show Exercises (Flash)* where the offered learning materials have mainly been developed by eXtensible Markup Language (XML), Document Type Definition (DTD) and eXtensible Style Sheet Language (XSL) technologies. By further extension, my *ODC* e-learning system can be developed into more mature format with more features.

#### 15.1.1 The *Show Test (XML)* menu option of my *ODC* system

Under this menu option, XML and DTD are used to structure various *Test* elements as XML documents. Later, the contents of them are fetched by using XSL instructions and then HTML codings are used to represent the content of *test* elements into screen. The data structures defined for my *test* XML documents as the produced learning objects of my *ODC* learning system, contain the two attributes *ans-value* and *ans-type* for every <answer> element <sup>1</sup>. These attributes, illustrated in the following examples, are defined to provide some kind of meta-data for the elements and specify a semantic role for each of the <answer> elements (Sec. 2.3.1).

```
<answer ans-value="False"  ans_type="graphic">  
<answer ans-value="true">
```

---

<sup>1</sup>The source files *course-tests.xml* and *tests\_graphic.xml* are attached in the appendix A

## 15.2 Object-oriented (OO) modeling of the Test object

For an OO presentation of the *Test* object used within my *ODC* course example, the following diagram has been modeled to present the involved conceptual objects and logical relationships used for creating test materials (Sec. 6.2.1). The model illustrates the similarity between the data structure of my XML Schema presented in figure 9.1 and an OO perspective for presenting the learning objects developed by my trial *ODC* learning system and delivered in the form of generated *Test* materials.

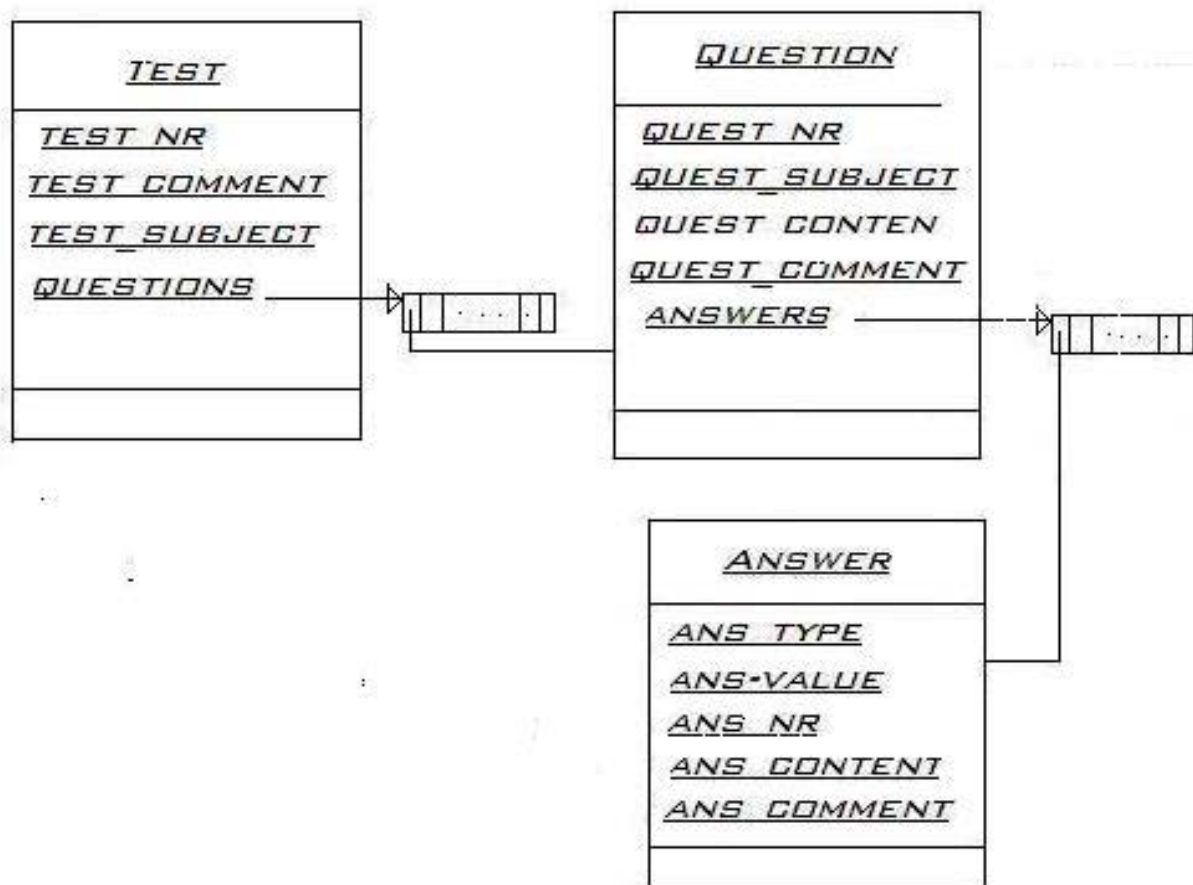


Figure 15.1: A UML class diagram for *Test* example

## 15.3 A trial simulation of the SCORM system architecture idea into my *ODC* system

LMS developed based on the SCORM specifications, is considered a complex and comprehensive system that can manage various learning activities such as delivering learning and training materials, generating reports, storing students' archives, course registration in addition to cooperating with other LMSs. The process of developing LMSs is quite time and resource demanding. Therefore, I decided to create a trial

learning system for experiencing some of the involved technologies and standards and related to e-learning. My developed *ODC* learning system as a prototyping model, is the result of this experiment (Sec. 14.2).

### The SCORM Learning Content Architecture

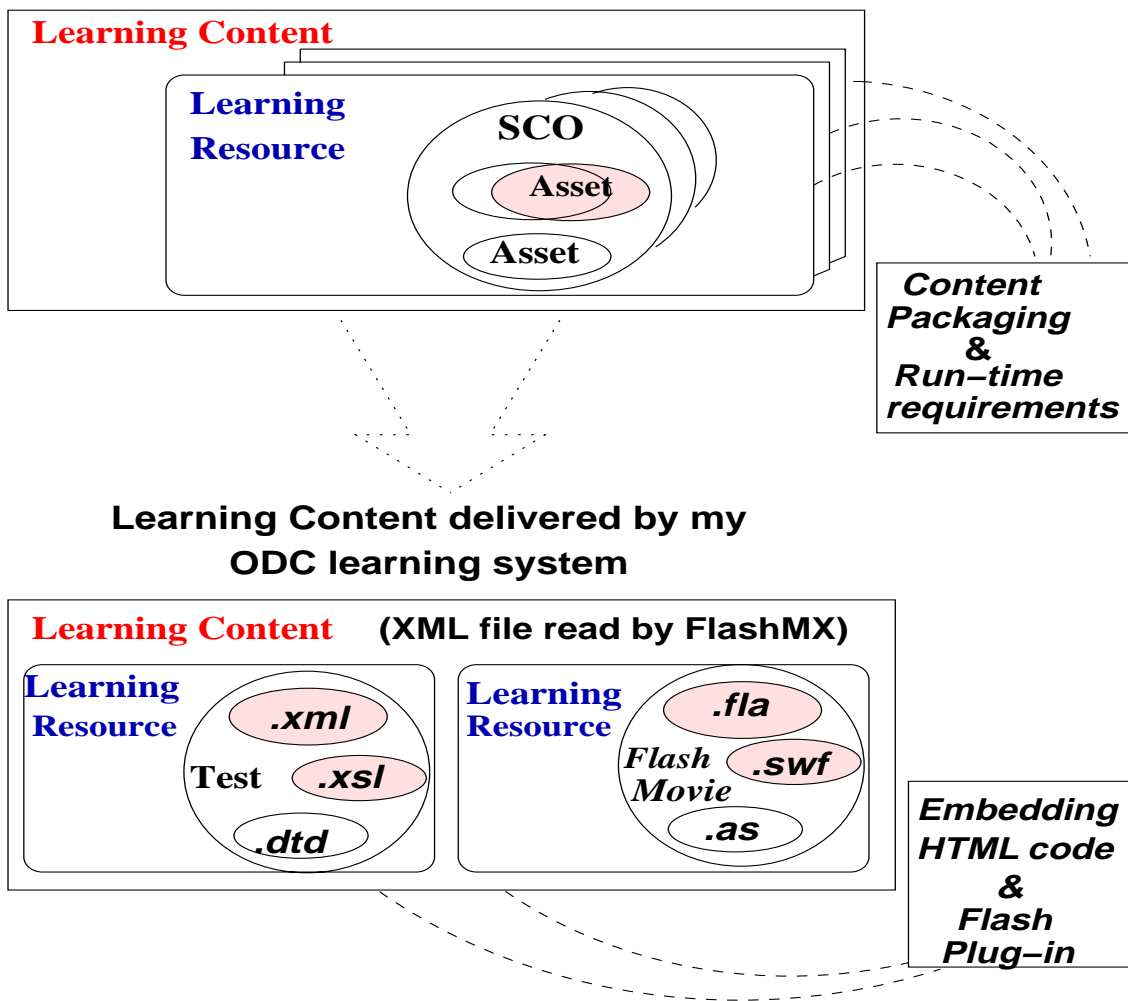


Figure 15.2: A simple reflection of SCORM within my *ODC* learning system.

In figure 15.2, I have illustrated a simple simulation of the SCORM architecture suggested for developing learning objects with the one I have used for delivering my developed learning objects within my *ODC* learning system<sup>2</sup>. The upper part of the figure 15.2 reflects the SCORM architecture idea for developing learning contents (Sec. 4.2) while the lower part illustrates the data architecture used for delivering my Flash movie as a learning object, which can represent the content of an XML document (Chap. 13). This Flash movie is available on my *ODC* learning system under the menu

<sup>2</sup>My *ODC* learning system is available by URL: [http://jsp.ifi.uio.no:8080/faribad/Course\\_coding/](http://jsp.ifi.uio.no:8080/faribad/Course_coding/)

option *Show Exercises (Flash)* and then by choosing the *Question & Answers (XML by Flash)* sub-menu. In figure 15.2, the XML document (.xml) contains the data belonging to a Test object that in other situation can be represented as a learning content by using a DTD declaration file (.dtd) and XSL formatting (.xsl). However, in my Flash movie (Sec. 13.2), just the .xml file is used as a learning resource to cooperate with the one which contains my Flash movie. The Flash movie itself, is a result of having a .fla file (Sec. 12.1), and the ActionScript codes defined for reading my XML data, is represented in the form of an (.as) file (Sec. 12.6). For delivering the movie, a published version of .fla file called .swf file (Sec. 12.6.3), must be created.

My learning content simulation in figure 15.2, represents some environmental requirements for delivering my developed learning objects by the *ODC* learning system example. For instance, these environmental requirements demand using the proper HTML embedding codes defined based on type of browser, which are required for the presentation of Flash movies (.swf files) within Web sites (Sec. 13.3.4).

Another environmental requirement demands accessing right version of Flash Plugin for presentation of my Flash movies on the Web browsers. These environmental requirements, similar to the environmental requirements introduced within SCORM architecture, must be provided for delivering some of my learning contents by my trial *ODC* learning management system.

Under the simulation process, the produced .fla Flash movie is considered as a SCO, which consists of the presentation material of the movie represented as an asset (.fla) and the published version of the movie represented as another asset (.swf). The movie's ActionScript codes (.as) has been simulated as the meta-data asset, which must be provided within every SCO (Sec. 4.2.6). My XML file, which implicitly contain some DTD declarations, is also considered as an asset within the Test learning object considered as a SCO.

## Chapter 16

# Flash MX for Developing Tutoring Materials

### 16.1 My developed Flash MX learning objects

Flash MX allows delivering interactive on-line learning materials consisting of various texts, graphics, sounds, and videos materials. Flash provides an excellent compression system for compressing speech sounds to 2% of the original size. Therefore, Flash MX is considered as an ideal tool for creating on-line presentations and interactive exercises [37].

### 16.2 Feedbacks and reporting

My trial learning system is developed to have **reporting** capability used for delivering continual feedbacks to students while various HTML forms are delivered or different exercises and testing materials are answered. Feedbacks produced during working with various learning materials, are developed to provide more convenient learning activities and better tutoring guidance (App. B).

#### 16.2.1 *Show Exercises* menu option of my ODC learning system

Below, a few of my Flash movies created as instructional tutoring materials are represented. All of my Flash MX movies developed as learning objects related to my thesis, are completely presented (App. C). My developed Flash MX movie exercises provide various learning activities for students where they must put the right elements in their right places. Based on every movement, a proper **feedback** will be presented on the screen. At the end, a short **reporting** message will also be displayed on the screen.

### 16.3 ActionScript, Flash programming language

ActionScript is the scripting language of Macromedia Flash. It controls the objects used within Flash movies and provides navigation facilities within the movies, using hyperlinks for connecting to the Web.



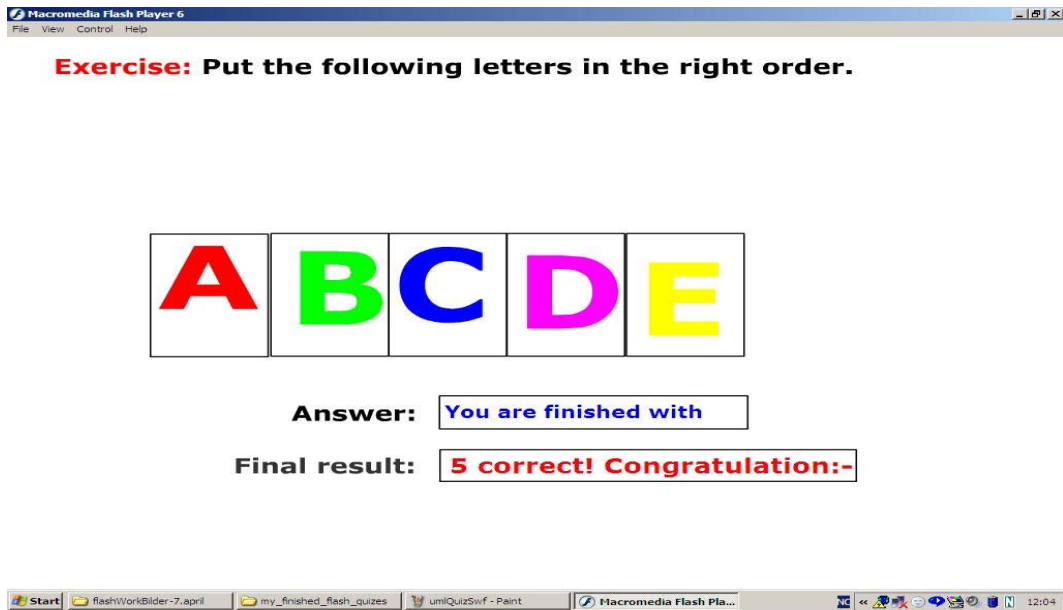


Figure 16.1: A Flash MX movie provided to deliver an alphabet tutoring exercise

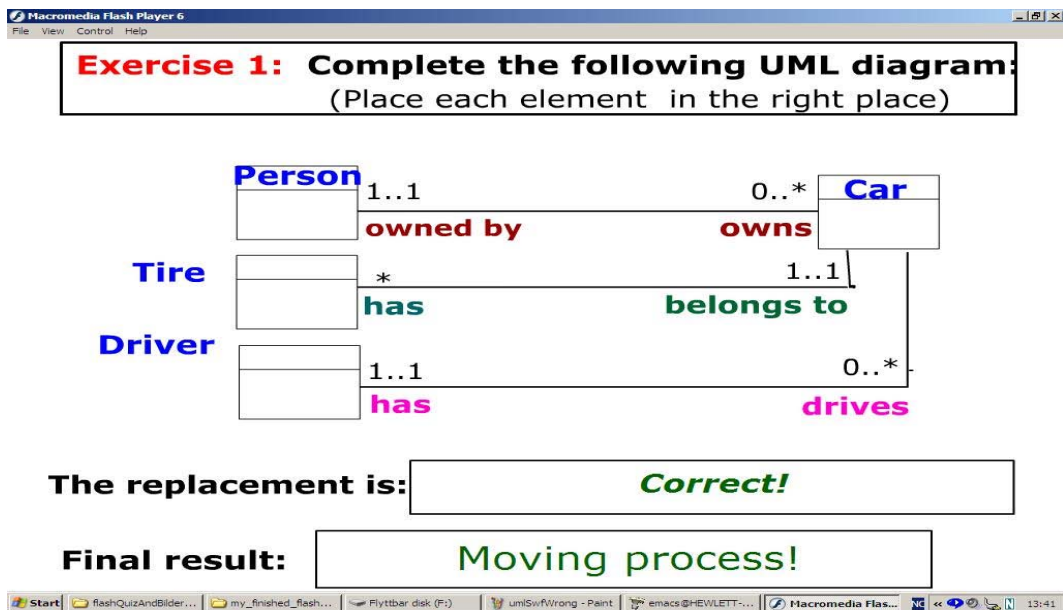


Figure 16.2: An UML exercise with desired feedbacks

The following ActionScript codes are defined within my various learning objects enabled to deliver *dynamic* feedbacks. A comprehensive presentation of my developed ActionScripts belonging to my various Flash movie exercises (Sec. 16.2.1) is incorporated in Appendix C.

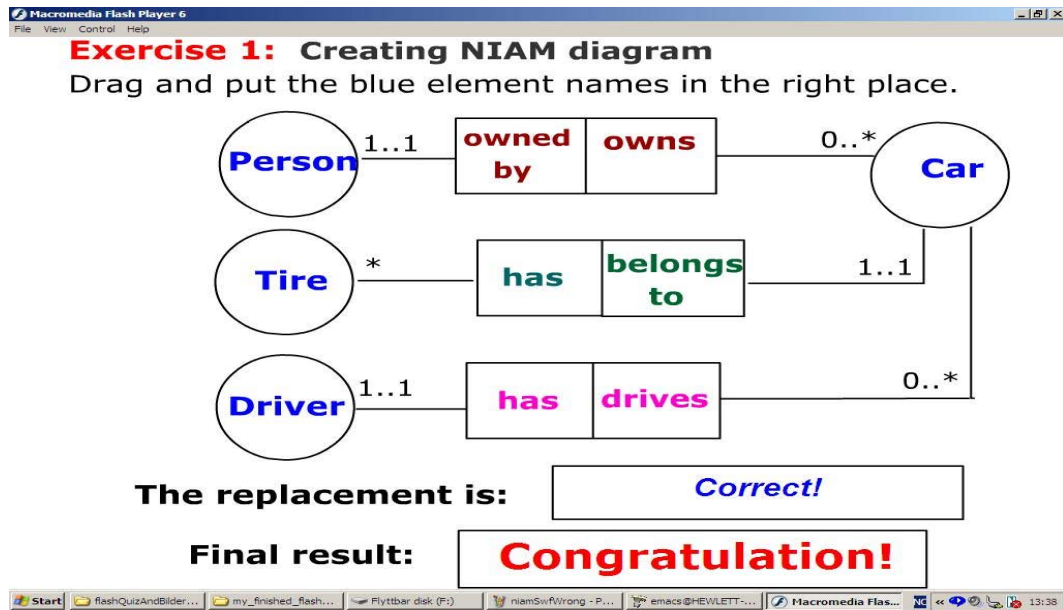


Figure 16.3: A NIAM exercise Flash MX movie with desired feedbacks

```

on(press) {
    startDrag(this);
    _root.answer1="The movement is";
}

on(release) {
    stopDrag();
    if (this._droptarget == "/tire") {
        _root.answer1="Correct!";
        _root.allCorrect +=1;
    }
    else{
        _root.answer1 = "wrong! Try again.";
    }
}

```

---

Example 1: The ActionScript codings defined for the invisible *Tire* button of the UML exercise 16.2.1.

---

```

allCorrect = 0;

_root.onEnterFrame= function(){
    if(_root.allCorrect == 0) {
        _root.endComment = "Start to solve the problem!";
    }
    else{
        if(_root.allCorrect >= 1){
            _root.endComment = "Please continue!";
            if(_root.allCorrect == 5){
                _root.answer1="You are finished with";
            }
        }
    }
}

```

```

        }
        _root.endComment = "5 correct! Congratulation:-)";
    }
    else{
        _root.endComment = "Please start again!";
    }
}
}

```

Example 2: The ActionScript codings provided for an alphabet quiz 16.2.1.

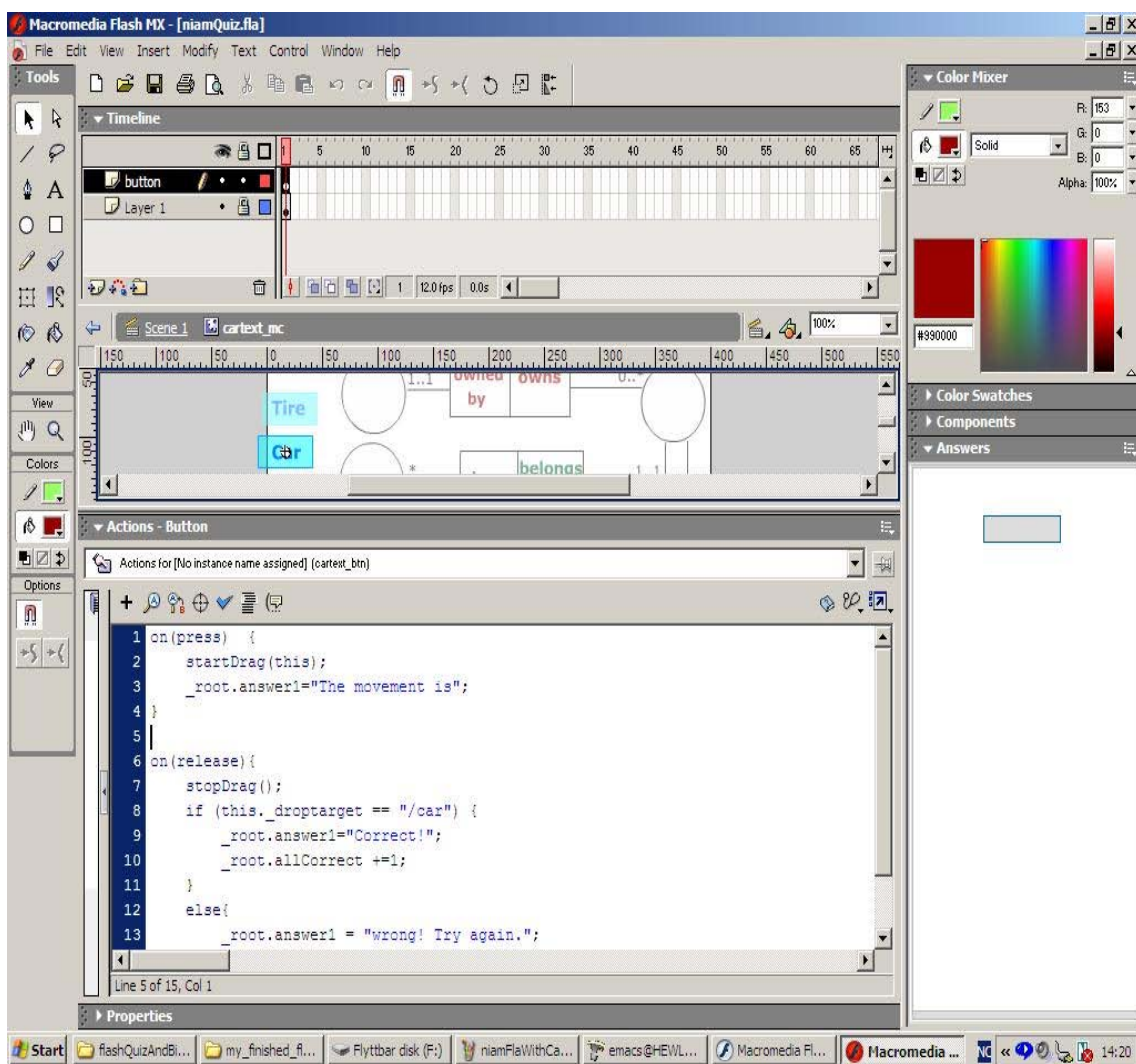


Figure 16.4: The ActionScript codings defined within Flash MX for the invisible *Car* button of my NIAM exercise 16.2.1.

## 16.4 Flash MX animation & tutorial learning object

As an experiment with developing learning materials by macromedia Flash MX enabled to provide multi media effects such as using texts, sounds, animations and hypertext links, the following Flash animation called **Flash-Finder**, was developed. The Flash was created as my group project for the **Information design** course, which I took during my master's studies<sup>1</sup>.

The following figures illustrate some of the main scenes of my developed Flash tutorial example belonging to the *Flash-Finder* group of my *Information design* course. For developing various menu options of the trial Flash-tutorial, different keyframes are created. This Flash tutorial provides an easy navigation system by allowing pauses and using reverse and forward buttons. The developed animation and the short tutorial Flash MX is available under the menu option *Show Exercises (Flash)* of my trial ODC on-line course<sup>2</sup>.

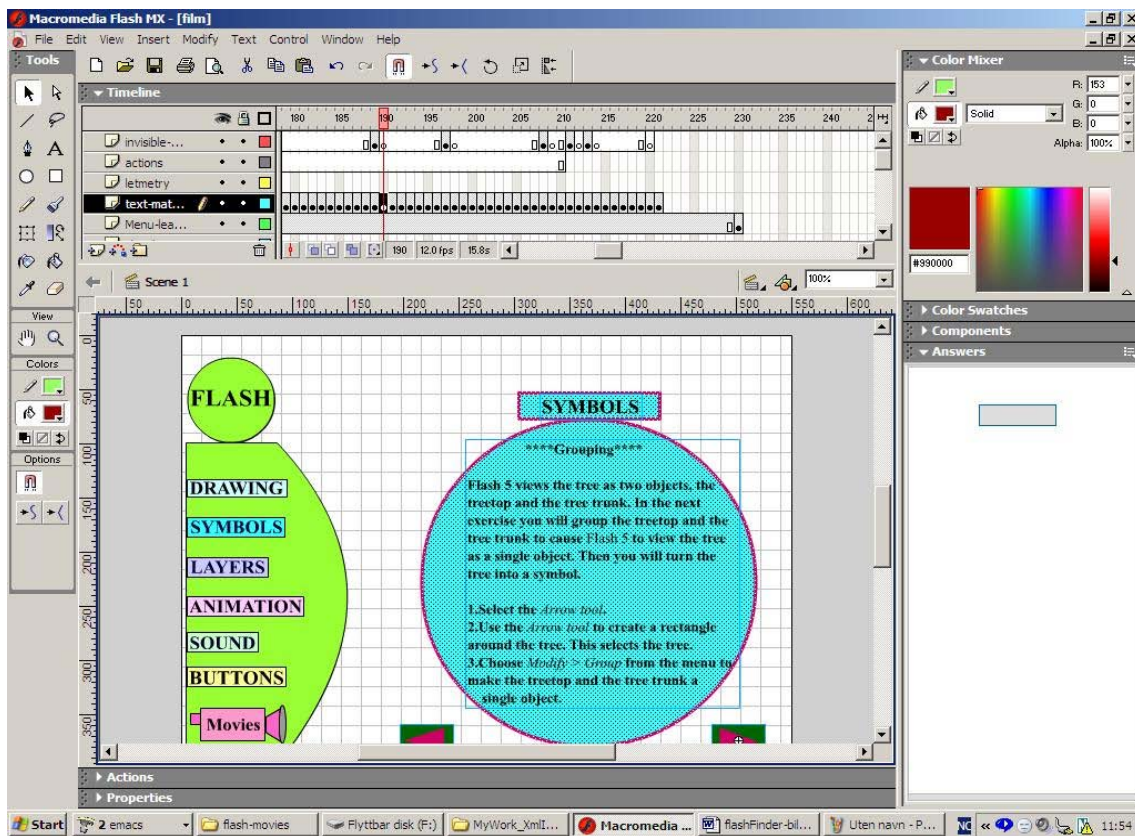


Figure 16.5: The menu option of the Flash MX tutorial example (.fla file)

<sup>1</sup>The Information design course (INF4210) by Dr Dino Karabeg,  
<http://www.uio.no/studier/emner/matnat/ifi/INF4210/index-eng.html>

<sup>2</sup>My trial ODC on-line course is available under the following URL,  
[http://jsp.ifi.uio.no:8080/faribad/Course\\_coding](http://jsp.ifi.uio.no:8080/faribad/Course_coding)

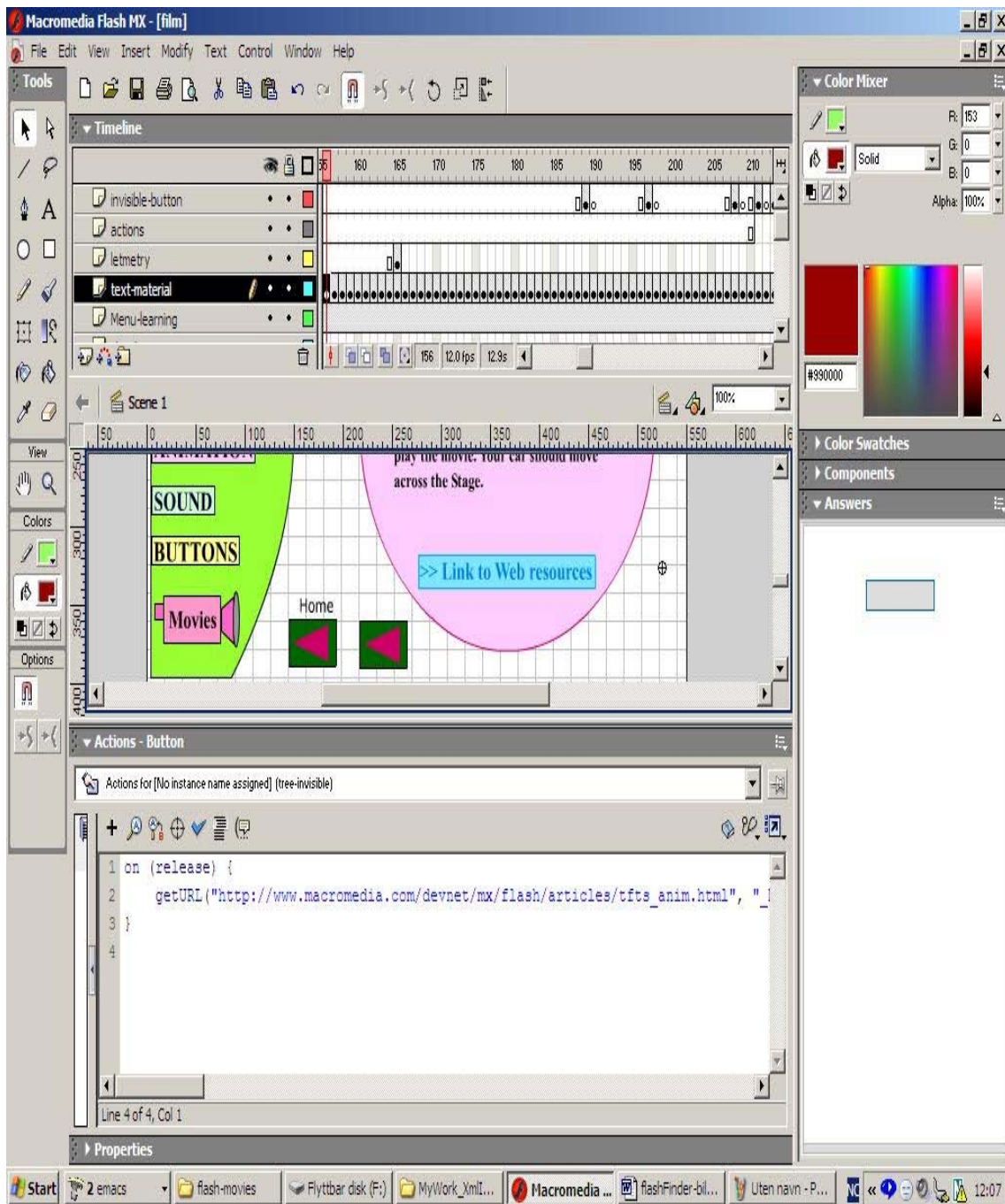


Figure 16.6: The ActionScript for a hyperlink within the Flash MX tutorial example (.fla file)



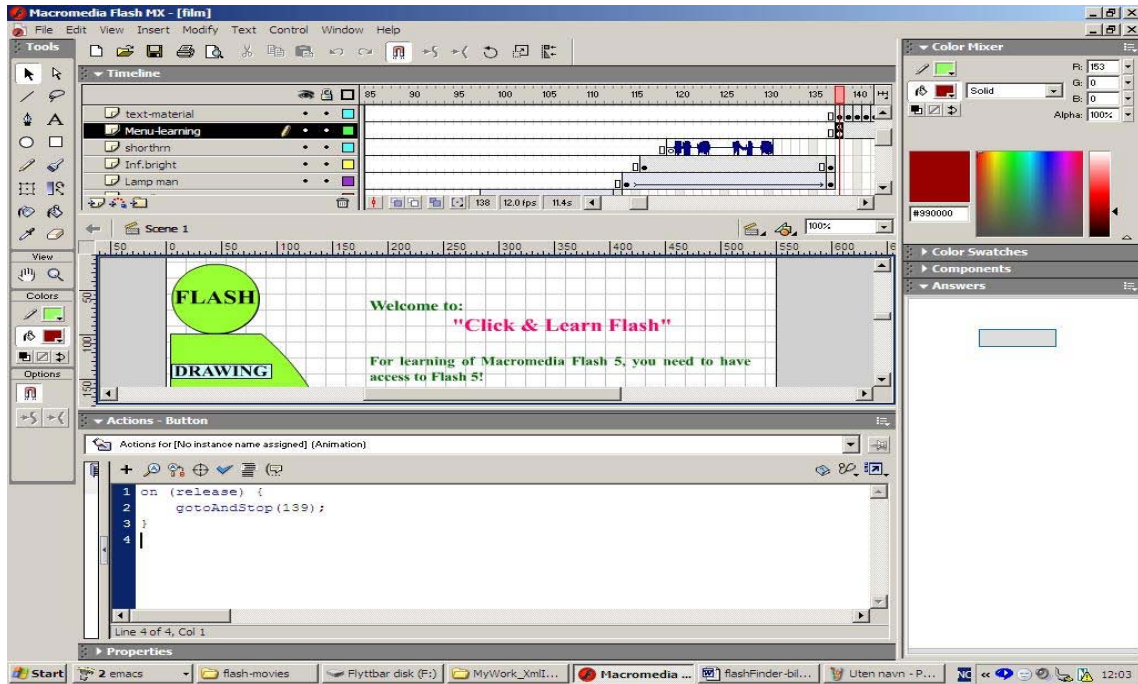


Figure 16.7: The ActionScript for a menu option of the Flash MX tutorial example (.fla file)

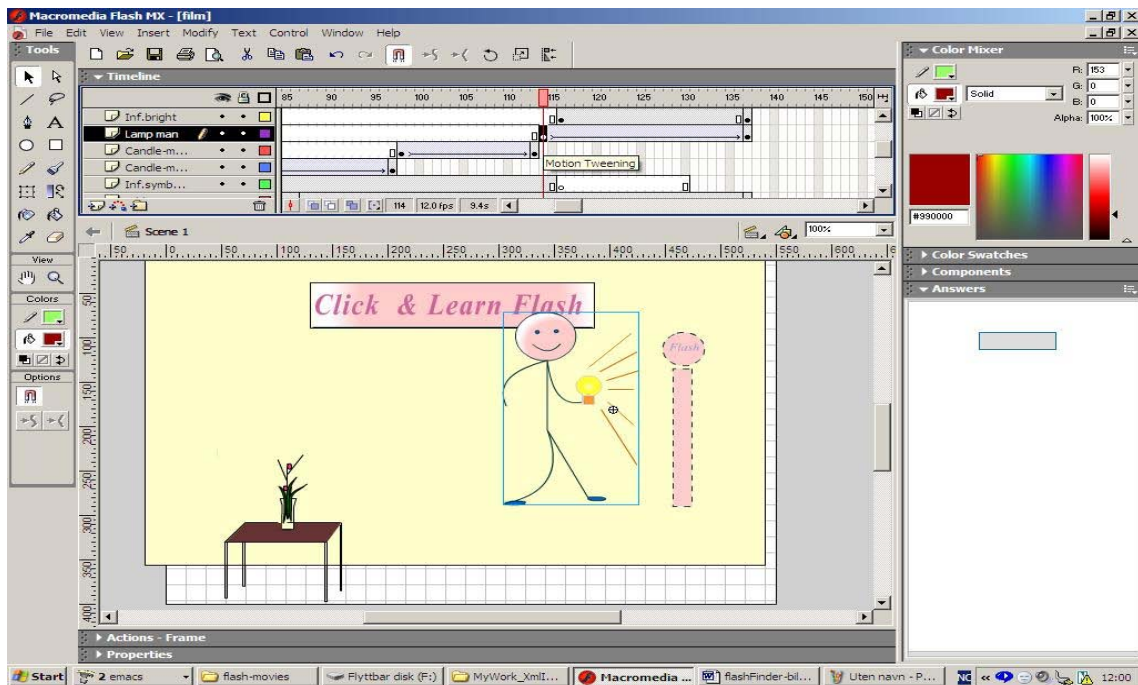


Figure 16.8: The intro animation of the Flash MX tutorial example (.fla file)

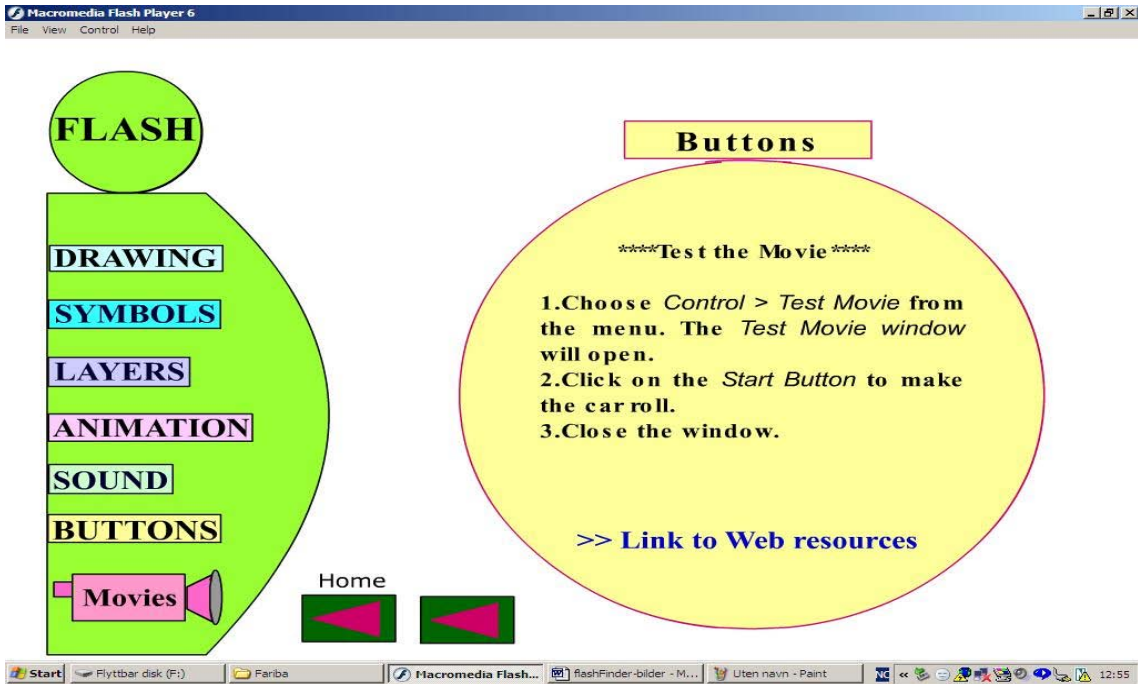


Figure 16.9: The published version of the hyperlinked page of the Flash MX tutorial example (.swf file)

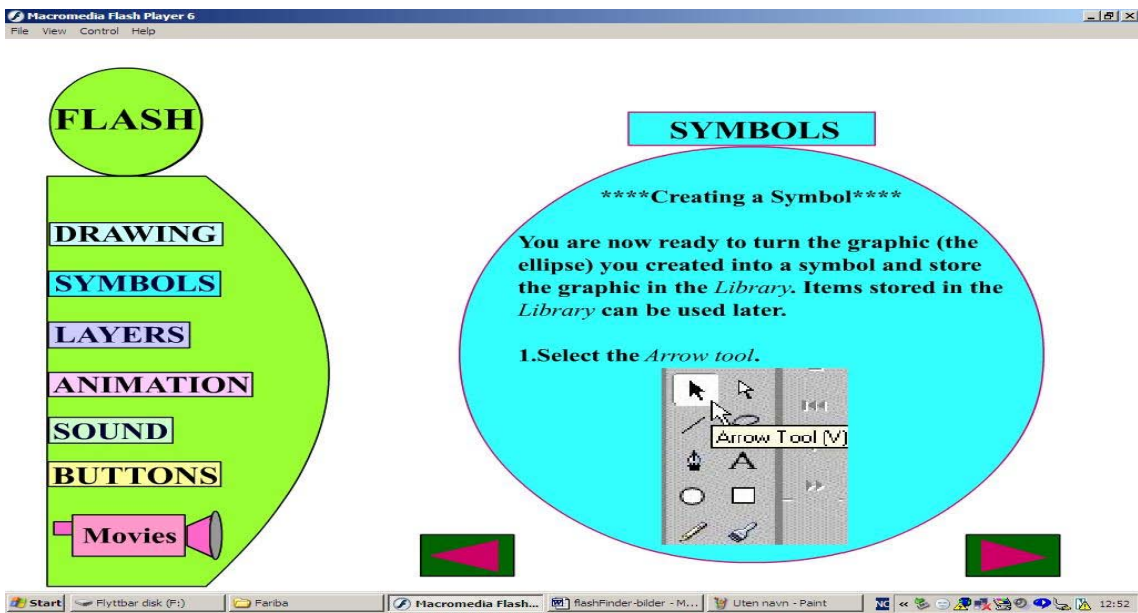


Figure 16.10: The published version of the Flash MX tutorial example (.swf file)

## Chapter 17

# Summary and Conclusion

The subject of my thesis was **Dynamic & XML-Based Contents within E-learning** to research about the subject, I used different technologies such as Java Server Pages, eXtensible Markup Language, eXtensible Style Sheet Language, Document Type Definition (DTD) and Macromedia Flash MX, to develop various learning objects and instructional facilities delivered by my trial *ODC* on-line course. I decided to develop the course as a simple example for modeling of the functionalities of Learning Management Systems. Using e-learning facilities will dramatically change the learning cultures.

By using new instructional technologies such as SCORM (ADL), more effective teaching and life-long learning will universally be provided and accessible.

For developing various learning materials and functionalities, I deployed JSP technology for separating the generating process of my learning contents from their presentations. Because of the advantage of using JSP my dynamic Internet-based learning materials are both platform and server independent. In addition, for generating dynamic contents used by some of the developed functionalities of my on-line course, JSP and MySQL relational database technologies were used.

Later, being provided with interaction possibilities, learners will be able to enter some information such as subject and test numbers, used for generating tailored testing materials for them.

Some of the learning materials were structured by DTD to be stored as XML documents. Later by using XSL formatting, the desired information were retrieved from XML files. In general, HTML as the world wide web presentation language, was used for presentation of the static parts of contents on the Web. While JSP technology was used for generating the dynamic parts of the contents.

One of the features of developing my learning materials into XML documents, was that I could use their element structures to retrieve the desired information for creating various test materials.



The advantage of using XML structured databases is that developers do not need to think about the content and just focus on developing their Web applications.

Having standard structures for creating learning materials can provide the possibility for exchanging and sharing learning objects among learning management systems over the Internet.

Flash as a macromedia tool, was used for creating various learning materials consisting of several multimedia effects used for supplying more effective and attractive learning materials.

XML technologies can be integrated with other technologies such as JSP and Macromedia Flash MX equipped with vector graphic presentational facilities that are proper for Web-based developments. One advantage of using vector graphic techniques is that the presentation quality can be maintained without any loss of resolution of layouts. This is an important consideration for Web application developments especially for learning purposes in which the accept of the system is very dependent on the display quality and communication speed. Another advantage of Flash MX which motivated me to use it for developing some of my learning object examples, is that the loading process of the movies is taking place while the presentation of the loaded parts can already be started.

Based on my experience for integrating XML with Flash, I can say that the process of defining and mapping the Schema of an XML document into ActionScript is quite time demanding. However, by having standard defined XML Schema for structuring various learning materials, the same developed ActionScript coding can be used for representing different learning materials, although the presentational part of Flash movies can be developed for various learning situations. Then, the created learning materials can be used by web-based learning management systems.

Offering learning materials consisting of various multi-media effects can make learning processes become more attractive and interesting for students although development of such materials can be quite time consuming for course leaders and costly for their organizations.

Learning contents can be granulated to smaller units called learning resources that can be created totally independent from any specific learning contents. Then, by developing international standards such as the SCORM, the possibility for creating learning resources that can easily be reused, shared and accessed by various learning management systems, across the WWW, can be provided.

The SCORM specifications requires that learning materials created in the form of small objects called SCOs, must not be dependent on the context of learning contents. So that they can be reused in various learning resources.

While the environmental requirements and content packaging for various environ-

ments must be provided in the form of meta-data for generating learning contents.

By using technical facilities and based on received user requests, shareable content objects can be selected and assembled into learning resources. The meta-data provided by content packaging, a predefined sequence for delivering of learning resources, can be used to aggregate the desired learning recourses into learning contents in real time and provided for different environments.

To allow the reuse of learning objects (LO), various standards such as IEEE LOM and SCORM, have been developed to describe learning objects, their relationships, etc.

The current LOM standard lacks instructional information for representing the quality and the instructional role of learning objects used within a learning process. A solution has been suggested for using standard definitions of *roles* to be added as attributes to the LOM standards for representing learning objects used in different learning processes.

Based on my experience of developing my trial *ODC* learning system, I believe deploying component-based system architecture will be the proper approach for developing LMSs where each of the SCORM high-level requirement can be considered as a business logic provided by one or more business components.

The following issues represent some of the ideas for further work related to the subject of the thesis.

- Evaluating the problems of having data redundancy within XML-based databases.
- How data connectivity can be increased within XML databases while redundancy of data is reduced.
- Evaluating the effect of data redundancy related to effective data accessing and disk storage capabilities.

In the future, I would like to work with a further developing process for my *ODC* online course system in order to develop more mature format and readiness. Because a real learning management system should be equipped with more learning and tutoring functionalities. Based on The SCORM specifications learning management systems must be able to connect with other LMSs.

# Bibliography

- [1] **Allert, H. & Dhraief, H. & Nejd, W.**, "Meta-Level Category, Role' in Metadata Standards for Learning: Instructional Roles and Instructional Qualities of Learning Objects", *Published at COSIGN-2002, 02- 04 September 2002, University of Augsburg, Lehrstuhl für Multimedia-Konzepte und Anwendungen, Germany.*, P. 14-22
- [2] **Atwood, S.**, *One system fits all* MIT Magazine of Innovation Technology Review, Vol. 106/N0.10, P. 8, Des.2003/Jan.2004
- [3] **Bendik Bygstad Den Polytekniske Høgskolen**, "Implementing CRM in a knowledge based organization", A longitudinal case study of six years use of Customer Relationship Management.
- [4] **Bender, E.**, *A LINC for E-learning*, MIT Magazine of Innovation Technology Review, October 2004, Vol. 107/N0.8
- [5] **Berendt, B. & Hotho, A. & Mladenic, D. & Someren, M. v. & Spiliopoulou, M. & Stumme, G.**, "A Roadmap for Mining From Web to Semantic Web", <http://eprints.pascal-network.org/archive/00000841/01/roadmap.pdf> *Reading date: 12.09.2004*
- [6] **Berkeley Design Technology, Inc.**, <http://www.bdti.com/> *Reading date: 12.07.2004*
- [7] **Bhangal, S., Farr, A., Rey, P.**, *Foundation Flash 5*, Friends of ED, 2000, ISBN 1-903450-31-4
- [8] **Bleyle, J., Burdekin, J., Burger, M., Crawford, D. and Taylor M.**, *Using Flash MX*, Macromedia Inc. San Francisco, 2002, <http://alf3.urz.unibas.ch/doku/getfile.cfm/FlashMXTutorials.pdf?did=63.pdf?did=63> *Reading date: 11.05.2004*
- [9] **Bush, V.** (1945), "As We May Think", *The Atlantic Monthly*, July 1945, Volume 176, No. 1, pp. 101-108, <http://www.theatlantic.com/unbound/flashbks/computer/bushf.htm1> *Reading date: 12.09.2004*
- [10] **Chun, R.**, *Macromedia flash MX Advanced*, Peachpit Press, 2002, ISBN 0-201-75846-6

- [11] **Connolly, T. & Begg, C.**, *Database Systems*, Addison Wesley, third addition, 2002, ISBN 0-201-70857-4
- [12] **David, Matthew A.** (2001), "Integrating XML with Macromedia Flash", [http://www.flashgimp.com/articles/flash\\_articles/4\\_matthewdavid\\_flash\\_article.index.php](http://www.flashgimp.com/articles/flash_articles/4_matthewdavid_flash_article.index.php), <http://www.matthewdavid.ws> *Reading date: 15.09.2004*
- [13] **Eeles, P. & Houston, K. & Kozaczynski, W.**, *Building J2EE Applications*, Wesley, 2003, ISBN 0-201-79166-8
- [14] **Elmasri and Navathe**, *Fundamentals of database systems*, Third Edition, Addison-Wesley 2000, ISBN 0-201-54263-3
- [15] **Feasey, D.**, Learning Technology Systems Architecture (LTSA) <http://eyepopping.manilasites.com/stories/storyReader\188> *Reading date: 20.02.2005*
- [16] **Fowler, M. & Scott, K.**, *UML Distilled*, Second Edition, Wesley, 2000, ISBN 020165783X
- [17] **Fox, G.**, **IMS ADL IEEE LTSA Overview and Critique of Standards**, <http://aspn.ucs.indiana.edu/collabtools/imsadlieeejan01.ppt> *Reading date: 01.03.2005*
- [18] **Ganguli, M.**, *Making use of JSP*, Wiley, 2002, ISBN 0-471-21974-6
- [19] **Garcia-Molina, H. & Ullman, J. D. & Widom, J.**, *Database Systems. The complete book*, Prentice Hall (New Jersey), 2002, ISBN 0-13-031995-3
- [20] **Hannemyr, G.**, "Begynnelse på en historie om Internet", *Nettsamfunn, Tano-Aschehoug*, 1999, Oslo, pp. 11-27
- [21] **Harold, E. R.**, *XML Bible*, Second Edition, 2001
- [22] **Harold, E. R. & Means, S.**, *XML IN A NUTSHELL*, O'Reilly, 2002, ISBN 0-596-00292-0
- [23] **Huang, G. T.**, *China's Clever Classroom* MIT Magazine of Innovation Technology Review, June 2004, Vol. 107, N0.5
- [24] **IMS Abstract Framework: White Paper (Version 1.0)**, IMS Global Learning Consortium, Inc. <http://www.imsglobal.org/af/afv1p0/imsafwhitepaper1p0.html> *Reading date: 20.02.2005*
- [25] **IMS Global Learning Consortium, Inc.**, <http://www.imsglobal.org/background.cfm> *Reading date: 23.03.2005*
- [26] **INF312** (Lecture's notes of course INF312, 2002), Institute for informatics, University in Oslo, INF312-beyondrel-99, Gerhard Skagestein, Aug. 2001, [http://www.w3.org/TR/2001/REC-xml\\\_schema-0-20010502/primer.html](http://www.w3.org/TR/2001/REC-xml\_schema-0-20010502/primer.html) *Reading date: 22.10.2003*

- [27] **Learning Management System**, <http://www.imsinc.com/Standard.asp?PageID=13>  
*Reading date: 11.09.2004*
- [28] **Marchal, B.**, "XML By Example", John Pierce, 2000, ISBN 0-7897-2242-9
- [29] **Mathiassen, L. & Munk-Madsen, A. & Nilsen, P. & Stage, J.**, *Object Oriented Analysis & Design*, Marko Publishing ApS, 2000, ISBN 87-7751-150-6
- [30] **Nadhani, P.**, "Build a Better Charting Engine Using Flash and XML", *DevX, Jupiter-media Corporation*, 2004
- [31] **Nelson, T. H.** (1982), *Literary Machines*, Mindful Press, 1993, Sausalito, CA, ISBN 089347052X
- [32] **NTNU**, *Innføring i Informasjons Teknologi*, Trondheim, 2002, ISBN 82-519-1800-6
- [33] **Patterson, D. & Hennessy, J. L.**, *Copmputer Organization & Design*, Morgan Kaufmann Publisher, Inc., 1998, ISBN 1-55860-491-X
- [34] **Quin, L.**, *Open Source XML Database Toolkit*, Wiley, 2000
- [35] **Reinhardt, R. & Lentz, J. W.**, *Flash 5 Bible*, Hungry Minds Inc., 2001, ISBN 0-7645-3515-5
- [36] **Routledge, N. & Bird, L. & Goodchild, A.**, "UML and XML Schema", *Thirteenth Australasian Database Conference, Malbourne, Australia*, Proc. Conferences in Research and Practice in Information Technology, Vol. 5, 157-166, 2002, Australian Computer Society, ISBN 0-909925-83-6
- [37] **Soemarmo, M.**, *Creating interactive online language lessons with Macromedia Flash MX*, Ohio University, [http://www.geocities.com/tesol\\_evonline/2004sessions/description/lessons\\_withFlashMX/index.html](http://www.geocities.com/tesol_evonline/2004sessions/description/lessons_withFlashMX/index.html), *Readingdate : 22.01.2005*
- [38] **Sommerville, I.**, *Software Engineering*, 6th Edition, Pearson Education, 2001, ISBN 0-201-39815-X
- [39] **Tamino**, Software AG the xml company, "Tamino xml server"
- [40] **The SCORM Overview**, "Shareable Content Object Reference Model (SCORM) Version 1.2", *Advanced Distributed Learning*, 2001
- [41] **Underdahl, B.**, *Macromedia Flash MX 2004, A Beginner's Guide*, Corel VENTURA, 2004, ISBN 0-07-222982-9
- [42] **Zastrocky, M.**, "Distributed Learning, e-learning and e-business: What do they mean and where's the content?", *Academic Strategies with the Gartner Group*, 2000

## 17.1 Abbreviations

The used abbreviations in this thesis are alphabetically represented as follow:

- ADL: Advanced Distributed Learning (ADL)
- AICC: Aviation Industry CBT Committee
- ARIADNE: Alliance of Remote Instructional Authoring & Distribution Networks for Europe
- ASP: Microsoft Active Server Pages
- CBT: Computer-Based Training
- CMI: Computer Managed Instruction
- CRM: Customer Relationship Management
- DB: Database
- DBMS: Database Management System
- DOM: Document Object model
- DTD: Document Type Definition
- ERD: Entity Relationship Diagram
- 4GL: Fourth-generation language
- GUI: Graphical User Interface
- IE: Internet Explorer
- IEEE: Institute of Electrical and Electronics Engineers
- IMS: Instructional Management System
- JSP: Java Server Pages
- LINC: Learning International Networks Consortium at MIT
- LMS: Learning Management System
- LOM: Learning Objects Metadata
- LTSC: Learning Technology Standards Committee
- MIT: Massachusetts Institute of Technology
- NIAM: Nijssen's Information Analysis Methodology
- ODC: Online Database Course (my trial developed learning system)
- ODL: Object Definition Language (in OO databases)
- OMG: Object Management Group
- OO: Object-Oriented
- OQL: Object Query Language
- PI: Processing Instructions
- RDB: Relational Database
- SCO: Sharable Content Object
- SCORM: Sharable Content Object Reference Model
- SGML: Standard Generalized Markup Language
- SQL: Structured Query Language
- UiO: University of Oslo
- UML: Unified Modeling Language
- W3C: World Wide Web Consortium
- WWW: World Wide Web
- XML: Extensible Markup Language
- XSL: Extensible Style sheet Language

# Contents

## **Appendix A: Source files**

My various source files developed for my *ODC* on-line course example, are presented.

## **Appendix B: Layouts**

Some of the result outputs and layouts provided by my *ODC* on-line course example, are presented.

## **Appendix C: Flash MX learning examples**

My various learning objects developed by Flash MX, are presented.

# Appendix A

## Source files of my *ODC* example

In this section, my various source files developed for my *ODC* on-line course example, are presented.

### A.1 Developed files by using XML technologies

---

---

#### A.1.1 File: course-tests.xml

---

---

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="course-tests.xsl"?>

<!DOCTYPE tests [
<!ELEMENT tests (test)*>
<!ELEMENT test (test_nr, questions, test_subject?, test_comment?)>
<!ELEMENT test_nr (#PCDATA)>
<!ELEMENT questions (question)*>
<!ELEMENT test_subject (#PCDATA)>
<!ELEMENT test_comment (#PCDATA)>
<!ELEMENT question (quest_nr,quest_subject?,quest_content,
                                quest_comment?,answers)>
<!ELEMENT quest_nr (#PCDATA) >
<!ELEMENT quest_subject (#PCDATA) >
<!ELEMENT quest_content (#PCDATA) >
<!ELEMENT quest_comment (#PCDATA) >
<!ELEMENT answers (answer)* >
<!ELEMENT answer (ans_nr, ans_content, answer_comment?) >
<!ELEMENT ans_nr (#PCDATA) >
<!ELEMENT ans_content (#PCDATA) >
<!ELEMENT ans_comment (#PCDATA) >

<!ATTLIST answer ans-value ( True | false ) #REQUIRED >
]>

<tests>
<test>
<test_nr>1</test_nr>
<test_subject>Database</test_subject>
<test_comment>Multiple choice questions</test_comment>
```



```

<questions>
<question>
<quest_nr>1</quest_nr>
<quest_subject>Mysql</quest_subject>
<quest_content>What kind of database is Mysql?</quest_content>
<quest_comment>The question was discussed in the class!
</quest_comment>
<answers>
<answer ans-value="false">
<ans_nr>1</ans_nr>
<ans_content>It is an XML DB.</ans_content>
<ans_comment>(To fetch the data, XML tags should be used!)
</ans_comment>
</answer>
<answer ans-value="false">
<ans_nr>2</ans_nr>
<ans_content>It is an object oriented DB.</ans_content>
<ans_comment>(To fetch the data, Java programming should be used.)
</ans_comment>
</answer>
<answer ans-value="true">
<ans_nr>3</ans_nr>
<ans_content>It is a relational DB.</ans_content>
<ans_comment>To fetch the data, SQL should be used.)</ans_comment>
</answer>
</answers>
</question>
<question>
<quest_nr>2</quest_nr>
<quest_subject>SQL</quest_subject>
<quest_content>What does SQL stand for?</quest_content>
<quest_comment>It is represented in the manual!</quest_comment>
<answers>
<answer ans-value="false">
<ans_nr>1</ans_nr>
<ans_content>Structured Question Line</ans_content>
<ans_comment>
Sql is a language to retrieve data from a relation database.
</ans_comment>
</answer>
<answer ans-value="true">
<ans_nr>2</ans_nr>
<ans_content>Structured Query Language</ans_content>
<ans_comment>(We use sql to fetch data from mysql database.)
</ans_comment>
</answer>
<answer ans-value="false">
<ans_nr>3</ans_nr>
<ans_content>Structional Querring Language</ans_content>
<ans_comment>No comments!</ans_comment>
</answer>
</answers>
</question>
</questions>
</test>
<test>
<test_nr>2</test_nr>

```

```

<test_subject>Algebra</test_subject>
<test_comment>You should answer to both questions questions
</test_comment>
<questions>
<question>
<quest_nr>1</quest_nr>
<quest_subject>Addition</quest_subject>
<quest_content>2+3</quest_content>
<quest_comment>Easy exercise!</quest_comment>
<answers>
<answer ans-value="true">
<ans_nr>1</ans_nr>
<ans_content>5</ans_content>
<ans_comment>Both numbers are less than 10.</ans_comment>
</answer>
<answer ans-value="false">
<ans_nr>2</ans_nr>
<ans_content>24</ans_content>
<ans_comment>The answer is larger than 20. </ans_comment>
</answer>
<answer ans-value="false">
<ans_nr>3</ans_nr>
<ans_content>33</ans_content>
<ans_comment>This answer is also larger than 30.</ans_comment>
</answer>
</answers>
</question>
<question>
<quest_nr>2</quest_nr>
<quest_subject>Subtraction</quest_subject>
<quest_content>20-30</quest_content>
<quest_comment>Middle difficult exercise!</quest_comment>
<answers>
<answer ans-value="false">
<ans_nr>1</ans_nr>
<ans_content>45</ans_content>
<ans_comment>Both numbers are larger than 40.</ans_comment>
</answer>
<answer ans-value="true">
<ans_nr>2</ans_nr>
<ans_content>-10</ans_content>
<ans_comment>The answer is less than 0. </ans_comment>
</answer>
<answer ans-value="false">
<ans_nr>3</ans_nr>
<ans_content>11</ans_content>
<ans_comment>This answer is also larger than 10.</ans_comment>
</answer>
</answers>
</question>
</questions>
</test>
</tests>

```

---

### A.1.2 File: test-answers-if.xsl

---

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">

<xsl:template match="/tests">
<html>
<head>
<style>

BODY
{ background: #bb88dd;
  font-size : 10pt;
  font-family : Arial, Helvetica, sans-serif;
  text-align : center;
  color : purple;
}
table{
width: "80%";
background-color: #bb88bb;
border: 2px solid purple;
color: purple;
}
  th {
    border: 1px solid purple;
background-color: #DDA0DD;
  }
  tr,td {
    border: 1px solid purple;
  }
</style>
</head>

<body>
<center>
<h4><xsl:text>
Here is the answer list for all questions on
different registered tests (represented by XML)
</xsl:text></h4>
<xsl:for-each select="test" >
<hr color="purple"/>
<br />
<xsl:apply-templates select="."/>
</xsl:for-each>
</center>
</body>
</html>
</xsl:template>
<xsl:template match="test" >
<table>
<tr>
<th colspan="3"><center><font size="4">Test nr.
<xsl:text> </xsl:text>
<xsl:value-of select="test_nr" />

```

```

</font></center>
</th>
</tr>
<tr>
<td><b>Subject:</b></td>
<td><b><xsl:value-of select="test_subject" /></b> </td>
</tr>
<tr>
<td><b>Number of questions:</b></td>
<td colspan="2"><b><xsl:text>"</xsl:text>
<xsl:value-of select="count(./questions/question)" />
<xsl:text>"</xsl:text>
</b>
</td>
</tr>
</table>
<br />
<xsl:apply-templates select="./questions" />
</xsl:template>

<xsl:template match="questions" >
<table>
<tr>
<th colspan="3"><center><font size="4">Questions:</font>
</center></th>
</tr>
<xsl:for-each select="question">
<tr>
<td colspan="3" bgcolor="gray">
<font size="4" color="yellow">Question nr.
<xsl:text> </xsl:text><xsl:value-of select="quest_nr" />
<xsl:text>: </xsl:text>
</font>
<xsl:value-of select="quest_content" />
</td>
</tr>
<tr>
<td colspan="3">
<font size="3">
<xsl:text> Number of alternative answers: "</xsl:text>
<b><xsl:value-of select="count(./answers/answer)" /></b>
<xsl:text>" </xsl:text></font>
<br />
<xsl:apply-templates select="./answers" />
</td>
</tr>
</xsl:for-each>
</table>
<p><br /> </p>
</xsl:template>
<xsl:template match="answers">
<font size="3"> Correct answer:</font>
<xsl:text> </xsl:text>
<xsl:for-each select="answer">
<xsl:choose>
<xsl:when test="./@ans-value='True'" >
<font size="3"> <b>

```

```

<xsl:value-of select="ans_content" /></b></font>
</xsl:when>
<xsl:otherwise><xsl:text></xsl:text>
</xsl:otherwise>
</xsl:choose>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>

```

---



---

### A.1.3 File: tests\_graphic.xml

---



---

```

<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="tests_graphic.xsl"?>

<!DOCTYPE tests [
<!ELEMENT tests (test)*>
<!ELEMENT test (test_nr, questions, test_subject?, test_comment?)>
<!ELEMENT test_nr (#PCDATA)>
<!ELEMENT questions (question)*>
<!ELEMENT test_subject (#PCDATA)>
<!ELEMENT test_comment (#PCDATA)>
<!ELEMENT question (quest_nr,quest_subject?,quest_content,quest_comment?,answers)>
<!ELEMENT quest_nr (#PCDATA) >
<!ELEMENT quest_subject (#PCDATA) >
<!ELEMENT quest_content (#PCDATA) >
<!ELEMENT quest_comment (#PCDATA) >
<!ELEMENT answers (answer)* >
<!ELEMENT answer (ans_nr, ans_content, answer_comment?, ans_type?) >
<!ELEMENT ans_nr (#PCDATA) >
<!ELEMENT ans_content (#PCDATA) >
<!ELEMENT ans_comment (#PCDATA) >
<!ATTLIST answer ans_type (text | graphic | sound) #REQUIRED >
<!ATTLIST answer ans-value ( True | False ) #REQUIRED >

]>

<tests>
<test>
<test_nr>1</test_nr>
<test_subject>Representing graphical objects by XML!</test_subject>
<test_comment>Multiple choice questions</test_comment>
<questions>
<question>
<quest_nr>1</quest_nr>
<quest_subject>Comparison</quest_subject>
<quest_content>Who is the youngest one?</quest_content>
<quest_comment>
The graphical objects are fetched by XSLT coding from an XML db.
</quest_comment>
<answers>
<answer ans-value="True" ans_type="graphic">
<ans_nr>1</ans_nr>
<ans_content>funny_man.jpg</ans_content>
<ans_comment>Happy boy!</ans_comment>

```

```

</answer>
<answer ans-value="False"  ans_type="graphic">
<ans_nr>2</ans_nr>
<ans_content>biz_man.jpg </ans_content>
<ans_comment>A rich business man!</ans_comment>
</answer>
<answer ans-value="False"  ans_type="graphic">
<ans_nr>3</ans_nr>
<ans_content>female_model.jpg</ans_content>
<ans_comment>A nice young lady!</ans_comment>
</answer>
<answer ans-value="False"  ans_type="text">
<ans_nr>4</ans_nr>
<ans_content>They all are as old as each other!</ans_content>
<ans_comment>It means none of the options are correct!</ans_comment>
</answer>
</answers>
</question>
<question>
<quest_nr>2</quest_nr>
<quest_subject>HTML</quest_subject>
<quest_content>What does HTML stand for?</quest_content>
<answers>
<answer ans-value="True"  ans_type="text">
<ans_nr>1</ans_nr>
<ans_content>Hyper Text Markup Language</ans_content>
</answer>
<answer ans-value="False"  ans_type="text">
<ans_nr>2</ans_nr>
<ans_content>High The Markup Language</ans_content>
</answer>
<answer ans-value="False"  ans_type="text">
<ans_nr>3</ans_nr>
<ans_content>Hyper The Markup Language</ans_content>
</answer>
<answer ans-value="False"  ans_type="text">
<ans_nr>4</ans_nr>
<ans_content>Hyper Transfer Markup Language</ans_content>
<ans_comment>It talks about text materials!</ans_comment>
</answer>
</answers>
</question>
<question>
<quest_nr>3</quest_nr>
<quest_subject>Sound</quest_subject>
<quest_content>Which of these recorded sounds is not related to a
human?
</quest_content>
<answers>
<answer ans-value="False"  ans_type="sound">
<ans_nr>1</ans_nr>
<ans_content>sound_ahum.wav</ans_content>
<ans_comment>Ahum, ahum!</ans_comment>
</answer>
<answer ans-value="True"  ans_type="sound">
<ans_nr>2</ans_nr>
<ans_content>sound_horn.wav</ans_content>

```

```

<ans_comment>A horn sound!</ans_comment>
</answer>
<answer ans-value="False" ans_type="sound">
<ans_nr>3</ans_nr>
<ans_content>sound_different.wav</ans_content>
<ans_comment>What is the difference?</ans_comment>
</answer>
<answer ans-value="False" ans_type="sound">
<ans_nr>4</ans_nr>
<ans_content>sound_relax.wav</ans_content>
<ans_comment>It's good to be relax!</ans_comment>
</answer>
</answers>
</question>
</questions>
</test>
</tests>

```

---

#### A.1.4 File: test-answers.xml

---

```

<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="test-answers.xsl"?>
<!DOCTYPE tests [

<!ELEMENT tests (test)*>
<!ELEMENT test (test_nr, questions, test_subject?, test_comment?)>
<!ELEMENT test_nr (#PCDATA)>
<!ELEMENT questions (question)*>
<!ELEMENT test_subject (#PCDATA)>
<!ELEMENT test_comment (#PCDATA)>
<!ELEMENT question (quest_nr, quest_subject?, quest_content, quest_comment?, answers)>
<!ELEMENT quest_nr (#PCDATA) >
<!ELEMENT quest_subject (#PCDATA) >
<!ELEMENT quest_content (#PCDATA) >
<!ELEMENT quest_comment (#PCDATA) >
<!ELEMENT answers (answer)* >
<!ELEMENT answer (ans_nr, ans_content, answer_comment?) >
<!ELEMENT ans_nr (#PCDATA) >
<!ELEMENT ans_content (#PCDATA) >
<!ELEMENT ans_comment (#PCDATA) >
<!ATTLIST answer ans-value ( True | False ) #REQUIRED >

]>

<tests>
<test>
<test_nr>1</test_nr>
<test_subject>Database</test_subject>
<test_comment>Multiple choice questions</test_comment>
<questions>
<question>
<quest_nr>1</quest_nr>
<quest_subject>Mysql</quest_subject>
<quest_content>What kind of database is Mysql?</quest_content>

```

```

<quest_comment>The question was discussed in the class!</quest_comment>
<answers>
<answer ans-value="False">
<ans_nr>1</ans_nr>
<ans_content>It is an XML DB.</ans_content>
<ans_comment>(To fetch the data, XML tags should be used!)</ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>2</ans_nr>
<ans_content>It is an object oriented DB.</ans_content>
<ans_comment>(To fetch the data, Java programming should be used.)</ans_comment>
</answer>
<answer ans-value="True">
<ans_nr>3</ans_nr>
<ans_content>It is a relational DB.</ans_content>
<ans_comment>(To fetch the data, SQL should be used.)</ans_comment>
</answer>
</answers>
</question>
<question>
<quest_nr>2</quest_nr>
<quest_subject>SQL</quest_subject>
<quest_content>What does SQL stand for?</quest_content>
<quest_comment>It is represented in the manual!</quest_comment>
<answers>
<answer ans-value="False">
<ans_nr>1</ans_nr>
<ans_content>Structured Question Line</ans_content>
<ans_comment>
Sql is a language to retrieve data from a relation database.
</ans_comment>
</answer>
<answer ans-value="True">
<ans_nr>2</ans_nr>
<ans_content>Structured Query Language</ans_content>
<ans_comment>(We use sql to fetch data from mysql database.)</ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>3</ans_nr>
<ans_content>Structional Querring Language</ans_content>
<ans_comment>No comments!</ans_comment>
</answer>
</answers>
</question>
</questions>
</test>
<test>
<test_nr>2</test_nr>
<test_subject>Algebra</test_subject>
<test_comment>You should answer to both questions questions</test_comment>
<questions>
<question>
<quest_nr>1</quest_nr>
<quest_subject>Addition</quest_subject>
<quest_content>2+3</quest_content>
<quest_comment>Easy exercise!</quest_comment>
<answers>

```



```

<answer ans-value="True">
<ans_nr>1</ans_nr>
<ans_content>5</ans_content>
<ans_comment>Both numbers are less than 10.</ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>2</ans_nr>
<ans_content>24</ans_content>
<ans_comment>The answer is larger than 20. </ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>3</ans_nr>
<ans_content>33</ans_content>
<ans_comment>This answer is also larger than 30.</ans_comment>
</answer>
</answers>
</question>
<question>
<quest_nr>2</quest_nr>
<quest_subject>Subtraction</quest_subject>
<quest_content>20-30</quest_content>
<quest_comment>Middle difficult exercise!</quest_comment>
<answers>
<answer ans-value="False">
<ans_nr>1</ans_nr>
<ans_content>45</ans_content>
<ans_comment>Both numbers are larger than 40.</ans_comment>
</answer>
<answer ans-value="True">
<ans_nr>2</ans_nr>
<ans_content>-10</ans_content>
<ans_comment>The answer is less than 0. </ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>3</ans_nr>
<ans_content>11</ans_content>
<ans_comment>This answer is also larger than 10.</ans_comment>
</answer>
</answers>
</question>
</questions>
</test>
</tests>

```

---

### A.1.5 File: course-tests.xsl

---

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="/tests">
<html>
<head>
<style>

```

```

BODY
{ background: #bb88dd;
  font-size : 10pt;
  font-family : Arial, Helvetica, sans-serif;
  text-align : center;
  color : purple;
}
table{
width: "80%";
background-color: #bb88bb;
border: 2px solid purple;
color: purple;
}
  th {
border: 1px solid purple;
background-color: #DDA0DD;
}
  tr,td {
border: 1px solid purple;
}
</style>
</head>

<body>
<center>
<h3>
<xsl:text>List of the questions of Test 1 represented by XML</xsl:text>
</h3>
<xsl:for-each select="test" >
<hr color="purple"/>
<br />
<xsl:apply-templates select="."/>
</xsl:for-each>
</center>
</body>
</html>
</xsl:template>

<xsl:template match="test" >
<table>
<tr>
<th colspan="3"><center><font size="4">Test nr. <xsl:text> </xsl:text>
<xsl:value-of select="test_nr" />
</font></center>
</th>
</tr>
<tr>
<td><b>Subject:</b></td>
<td><b><xsl:value-of select="test_subject" /></b> </td>
</tr>
<tr>
<td><b>Number of questions:</b></td>
<td colspan="2"><b><xsl:text>"</xsl:text>
<xsl:value-of select="count(/questions/question)" />
<xsl:text>"</xsl:text>
</b>

```

```

</td>
</tr>
<tr>
<td>Comment:</td>
<td colspan="2"><xsl:value-of select="test_comment" /></td>
</tr>
</table>
<br />
<xsl:apply-templates select="./questions" />
</xsl:template>
<xsl:template match="questions" >
<table>
<tr>
<th colspan="3"><center><font size="4">Questions:</font></center></th>
</tr>
<xsl:for-each select="question">
<tr>
<td colspan="3" bgcolor="gray"><font size="4" color="yellow">Question nr.:
<xsl:text> </xsl:text><xsl:value-of select="quest_nr" />
</font></td>
</tr>
<tr>
<td colspan="3"><b>Subject:</b><xsl:text>
</xsl:text><xsl:value-of select="quest_subject" />
<br /><b>Comment:</b><xsl:text> </xsl:text>
<xsl:value-of select="quest_comment" />
</td>
</tr>
<tr><td colspan="3">
<b>Question:</b>
<xsl:text> " </xsl:text>
<xsl:value-of select="quest_content" />
<xsl:text> " </xsl:text>
</td>
</tr>
<tr>
<td colspan="3">
<font size="3" color="yellow"><center>
<xsl:text>"</xsl:text>
<xsl:value-of select="count(./answers/answer)" />
<xsl:text>" </xsl:text>answer alternatives:</center></font></td>
</tr>
<xsl:apply-templates select="./answers" />
</xsl:for-each>
</table>
<p><br /> </p>
</xsl:template>
<xsl:template match="answers">
<xsl:for-each select="answer">
<tr>
<td colspan="3"><b>Nr.<xsl:text> </xsl:text>
<xsl:value-of select="ans_nr" /></b><xsl:text>: </xsl:text>
<xsl:value-of select="ans_content" />
<br />
Comment:<xsl:text> </xsl:text>
<xsl:value-of select="ans_comment" />
</td>

```

```

</tr>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>

```

---



---

### A.1.6 File: test-answers.xsl

---



---

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="/tests">
<html>
<head>
<style>

BODY
{ background: #bb88dd;
  font-size : 10pt;
  font-family : Arial, Helvetica, sans-serif;
  text-align : center;
  color : purple;
}
table{
width: "80%";
background-color: #bb88bb;
border: 2px solid purple;
color: purple;
}
  th {
    border: 1px solid purple;
background-color: #DDA0DD;
}
  tr,td {
    border: 1px solid purple;
}
</style>
</head>
<body>
<center>
<h4><xsl:text>
A list of answers belonged to the questions of
various XML-based Test materials, is presented.
</xsl:text></h4>
<xsl:for-each select="test" >
<hr color="purple"/>
<br />
<xsl:apply-templates select="."/>
</xsl:for-each>
</center>
</body>
</html>
</xsl:template>

<xsl:template match="test" >
<table>

```

```

<tr>
<th colspan="3"><center><font size="4">Test nr. <xsl:text> </xsl:text>
<xsl:value-of select="test_nr" />
</font></center>
</th>
</tr>
<tr>
<td><b>Subject:</b></td>
<td><b><xsl:value-of select="test_subject" /></b> </td>
</tr>
<tr>
<td><b>Number of questions:</b></td>
<td colspan="2"><b><xsl:text>"</xsl:text>
<xsl:value-of select="count(./questions/question)" />
<xsl:text>"</xsl:text>
</b>
</td>
</tr>
</table>
<br />
<xsl:apply-templates select="./questions" />
</xsl:template>
<xsl:template match="questions" >
<table>
<tr>
<th colspan="3"><center><font size="4">Questions:</font></center></th>
</tr>
<xsl:for-each select="question">
<tr>
<td colspan="3" bgcolor="gray"><font size="4" color="yellow">Question nr.
<xsl:text> </xsl:text><xsl:value-of select="quest_nr" />
<xsl:text>: </xsl:text>
</font>
<xsl:value-of select="quest_content" />
</td>
</tr>
<tr>
<td colspan="3">
<font size="3" color="yellow"><center>
<xsl:text>"</xsl:text>
<xsl:value-of select="count(./answers/answer)" />
<xsl:text>" </xsl:text>alternative answers:</center></font></td>
</tr>
<xsl:apply-templates select="./answers" />
</xsl:for-each>
</table>
<p><br /> </p>
</xsl:template>
<xsl:template match="answers">
<xsl:for-each select="answer">
<tr>
<td colspan="3"><b>Answer nr.<xsl:text> </xsl:text>
<xsl:value-of select="ans_nr" />
<xsl:text>: </xsl:text> </b>
<xsl:value-of select="ans_content" />
<br />
<strong>

```

```

<xsl:text>Correct value:
</xsl:text></strong>
<font size="3" color="yellow">
<xsl:value-of select="./@ans-value" />
</font>
</td>
</tr>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>

```

---

### A.1.7 File: test-answers-if.xml

---

```

<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="test-answers-if.xsl"?>

<!DOCTYPE tests [
<!ELEMENT tests (test)*>
<!ELEMENT test (test_nr, questions, test_subject?, test_comment?)>
<!ELEMENT test_nr (#PCDATA)>
<!ELEMENT questions (question)*>
<!ELEMENT test_subject (#PCDATA)>
<!ELEMENT test_comment (#PCDATA)>
<!ELEMENT question (quest_nr, quest_subject?, quest_content, quest_comment?, answers)>
<!ELEMENT quest_nr (#PCDATA) >
<!ELEMENT quest_subject (#PCDATA) >
<!ELEMENT quest_content (#PCDATA) >
<!ELEMENT quest_comment (#PCDATA) >
<!ELEMENT answers (answer)* >
<!ELEMENT answer (ans_nr, ans_content, answer_comment?) >
<!ELEMENT ans_nr (#PCDATA) >
<!ELEMENT ans_content (#PCDATA) >
<!ELEMENT ans_comment (#PCDATA) >
<!ATTLIST answer ans-value ( True | False ) #REQUIRED >
]>

<tests>
<test>
<test_nr>1</test_nr>
<test_subject>Database</test_subject>
<test_comment>Multiple choice questions</test_comment>
<questions>
<question>
<quest_nr>1</quest_nr>
<quest_subject>Mysql</quest_subject>
<quest_content>What kind of database is Mysql?</quest_content>
<quest_comment>The question was discussed in the class!</quest_comment>
<answers>
<answer ans-value="False">
<ans_nr>1</ans_nr>
<ans_content>It is an XML DB.</ans_content>
<ans_comment>(To fetch the data, XML tags should be used!)</ans_comment>
</answer>

```

```

<answer ans-value="False">
<ans_nr>2</ans_nr>
<ans_content>It is an object oriented DB.</ans_content>
<ans_comment>(To fetch the data, Java programming should be used.)</ans_comment>
</answer>
<answer ans-value="True">
<ans_nr>3</ans_nr>
<ans_content>It is a relational DB.</ans_content>
<ans_comment>To fetch the data, SQL should be used.)</ans_comment>
</answer>
</answers>
</question>
<question>
<quest_nr>2</quest_nr>
<quest_subject>SQL</quest_subject>
<quest_content>What does SQL stand for?</quest_content>
<quest_comment>It is represented in the manual!</quest_comment>
<answers>
<answer ans-value="False">
<ans_nr>1</ans_nr>
<ans_content>Structured Question Line</ans_content>
<ans_comment>
Sql is a language to retrieve data from a relation database.
</ans_comment>
</answer>
<answer ans-value="True">
<ans_nr>2</ans_nr>
<ans_content>Structured Query Language</ans_content>
<ans_comment>(We use sql to fetch data from mysql database.)</ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>3</ans_nr>
<ans_content>Structional Querring Language</ans_content>
<ans_comment>No comments!</ans_comment>
</answer>
</answers>
</question>
</questions>
</test>
<test>
<test_nr>2</test_nr>
<test_subject>Algebra</test_subject>
<test_comment>You should answer to both questions questions</test_comment>
<questions>
<question>
<quest_nr>1</quest_nr>
<quest_subject>Addition</quest_subject>
<quest_content>2+3</quest_content>
<quest_comment>Easy exercise!</quest_comment>
<answers>
<answer ans-value="True">
<ans_nr>1</ans_nr>
<ans_content>5</ans_content>
<ans_comment>Both numbers are less than 10.</ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>2</ans_nr>

```

```

<ans_content>24</ans_content>
<ans_comment>The answer is larger than 20. </ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>3</ans_nr>
<ans_content>33</ans_content>
<ans_comment>This answer is also larger than 30.</ans_comment>
</answer>
</answers>
</question>
<question>
<quest_nr>2</quest_nr>
<quest_subject>Subtraction</quest_subject>
<quest_content>20-30</quest_content>
<quest_comment>Middle difficult exercise!</quest_comment>
<answers>
<answer ans-value="False">
<ans_nr>1</ans_nr>
<ans_content>45</ans_content>
<ans_comment>Both numbers are larger than 40.</ans_comment>
</answer>
<answer ans-value="True">
<ans_nr>2</ans_nr>
<ans_content>-10</ans_content>
<ans_comment>The answer is less than 0. </ans_comment>
</answer>
<answer ans-value="False">
<ans_nr>3</ans_nr>
<ans_content>11</ans_content>
<ans_comment>This answer is also larger than 10.</ans_comment>
</answer>
</answers>
</question>
</questions>
</test>
</tests>

```

---



---

### A.1.8 File: tests\_graphic.xsl

---



---

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="tests">
<html>
<head>
<style>

BODY
{ background: #bb88dd;
font-size : 10pt;
font-family : Arial, Helvetica, sans-serif;
text-align : center;
color : purple;

```



```

}
table{
width: "80%";
background-color: #bb88bb;
border: 2px solid purple;
color: purple;
}
  th {
    border: 1px solid purple;
background-color: #DDA0DD;
}
  tr,td {
    border: 1px solid purple;
}
</style>
</head>

<body>
<center>
<h4><xsl:text>Usage of divers micromedia elements in XML</xsl:text></h4>
<xsl:for-each select="test" >
<hr color="purple"/>
<br />
<xsl:apply-templates select="."/>
</xsl:for-each>
</center>
</body>
</html>
</xsl:template>
<xsl:template match="test" >
<table>
<tr>
<th colspan="3"><center><font size="4">Test nr. <xsl:text> </xsl:text>
<xsl:value-of select="test_nr" />
</font></center>
</th>
</tr>
<tr>
<td><b>Subject:</b></td>
<td><b><xsl:value-of select="test_subject" /></b> </td>
</tr>
<tr>
<td><b>Number of questions:</b></td>
<td colspan="2"><b><xsl:text>"</xsl:text>
<xsl:value-of select="count(./questions/question)" />
<xsl:text>"</xsl:text>
</b>
</td>
</tr>
</table>
<br />
<xsl:apply-templates select="./questions" />
</xsl:template>

<xsl:template match="questions" >
<table>
<tr>

```

```

<th colspan="4"><center><font size="4">Questions:</font></center></th>
</tr>
<xsl:for-each select="question">
<tr>
<td colspan="4" bgcolor="gray"><font size="4" color="yellow">Question nr.
<xsl:text> </xsl:text><xsl:value-of select="quest_nr" />
<xsl:text>: </xsl:text>
</font>
<xsl:value-of select="quest_content" />
</td>
</tr>
<tr>
<td colspan="4">
<font size="3">
<xsl:text> Number of alternative answers: "</xsl:text>
<b><xsl:value-of select="count(./answers/answer)" /></b>
<xsl:text>" </xsl:text></font>
<br />
<xsl:apply-templates select="./answers" />
</td>
</tr>
</xsl:for-each>
</table>
<p><br /> </p>
</xsl:template>
<xsl:template match="answers">
<tr>
<xsl:for-each select="answer">
<td>
<b>Answer nr.<xsl:text> </xsl:text>
<xsl:value-of select="ans_nr" />
<xsl:text> : </xsl:text></b>
<br />
<xsl:choose>
<xsl:when test="./@ans_type='graphic'" >
<font color="pink">
<xsl:text> (Graphical object) </xsl:text>
</font>
<br/>
<img>
<xsl:attribute name="src">
<xsl:value-of select="ans_content" />
</xsl:attribute>
<xsl:attribute name="height">
<xsl:text>100</xsl:text>
</xsl:attribute>
<xsl:attribute name="width">
<xsl:text>50</xsl:text>
</xsl:attribute>
</img>
</xsl:when>
<xsl:when test="./@ans_type='sound'" >
<xsl:text> (Sound file) </xsl:text>
<br/><br/>
<a>
<xsl:attribute name="href">
<xsl:value-of select="ans_content" />

```

```

</xsl:attribute>
<xsl:text> </xsl:text>
<b><font color="yellow">
<xsl:value-of select="ans_content" />
</font></b>
</a>
</xsl:when>
<xsl:otherwise>
<font color="green">
<xsl:text> (Textual file) </xsl:text>
</font>
<p><font size="2">
<b><xsl:value-of select="ans_content" /></b>
</font></p>
</xsl:otherwise>
</xsl:choose>
</td>
</xsl:for-each>
</tr>
</xsl:template>
</xsl:stylesheet>

```

---



---

### A.1.9 File: course.dtd

---



---

```

<!ELEMENT education (course)+ >
<!ELEMENT course (course-title, course-comment, teachers, lectures) >
<!ATTLIST course course-code ID #REQUIRED >
<!ELEMENT course-title (#PCDATA) >
<!ELEMENT course-comment (#PCDATA) >
<!ELEMENT teachers (teacher)+ >
<!ELEMENT teacher (name, role, email, homepage) >
<!ELEMENT name (#PCDATA) >
<!ELEMENT role (#PCDATA) >
<!ELEMENT email (#PCDATA) >
<!ELEMENT homepage (#PCDATA) >
<!ELEMENT lectures (lecture*) >
<!ELEMENT lecture (lectureNr, subjects , perform, lecture-comment?) >
<!ELEMENT lectureNr (#PCDATA) >
<!ELEMENT subjects (subject+) >
<!ELEMENT subject (sub-title, sub-content, sub-comment?, resources?, exercises?) >
<!ATTLIST subject subjectID ID #IMPLIED >
<!ELEMENT sub-title (#PCDATA) >
<!ELEMENT sub-comment (#PCDATA) >
<!ELEMENT resources (resource*) >
<!ELEMENT resource (res-name, res-type, res-comment) >
<!ELEMENT res-name (#PCDATA) >
<!ELEMENT res-type (#PCDATA) >
<!ELEMENT res-comment (#PCDATA) >
<!ELEMENT sub-content (paragraph)+ >
<!ATTLIST paragraph about (NMTOKENS) #IMPLIED>
<!ELEMENT paragraph (#PCDATA) >
<!ELEMENT exercises (exercise*) >
<!ELEMENT exercise (exerciseNr, question, answers, exe-comment?) >
<!ELEMENT exerciseNr (#PCDATA) >

```

```

<!ELEMENT question (#PCDATA) >
<!ELEMENT answers (answer+) >
<!ELEMENT answer ( ans-number, ans-content, ans-comment?) >
<!ELEMENT ans-number (#PCDATA) >
<!ELEMENT ans-content (#PCDATA) >
<!ELEMENT ans-comment (#PCDATA) >
<!ELEMENT exe-comment (#PCDATA) >
<!ATTLIST answer ans-value ( true | false ) #REQUIRED >
<!ELEMENT perform (by-teacher, perform-date, perform-comment?) >
<!ELEMENT by-teacher (#PCDATA) >
<!ELEMENT perform-date (#PCDATA) >
<!ATTLIST perform-date format (NMTOKEN) #REQUIRED >
<!ELEMENT perform-comment (#PCDATA) >
<!ELEMENT lecture-comment (#PCDATA) >

```

### A.1.10 File: course.xml

```

<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<?xml-stylesheet type="text/xsl" href="course.xsl" ?>

<!DOCTYPE education SYSTEM "course.dtd" >

<education>
<course course-code="_102" >
<course-title>System Development</course-title>
<course-comment>The course is virtual as a trial version!</course-comment>
<teachers>
<teacher>
<name>Teacher 1</name>
<role>Main teacher</role>
<email>teacher1@ifi.uio.no</email>
<homepage>http://heim.ifi.uio.no/~teacher1</homepage>
</teacher>
<teacher>
<name>Teacher 2</name>
<role>Guest teacher</role>
<email>teacher2@ifi.uio.no</email>
<homepage>http://heim.ifi.uio.no/~teacher2</homepage>
</teacher>
</teachers>
<lectures>
<lecture>
<lectureNr>5</lectureNr>
<subjects>
<subject subjectID="_1">
<sub-title>L-Structured Query Language</sub-title>
<sub-content>
<paragraph about="sql">Structured Query Language - SQL is a standard
interactive and programming language for getting information from and
updating a database.</paragraph>
</sub-content>
<sub-comment>
SQL is both an ANSI and an ISO standard, many database products
support SQL with proprietary extensions to the standard language.

```

```

</sub-comment>
<resources>
<resource>
<res-name>SQL quiz</res-name>
<res-type>Web site</res-type>
<res-comment>
Test your SQL knowlwdge on :
http://www.w3schools.com/quiztest/quiztest.asp?qtest=SQL
</res-comment>
</resource>
</resources>
<exercises>
<exercise>
<exerciseNr>1</exerciseNr>
<question>What does SQL stand for?</question>
<answers>
<answer ans-value="false" >
<ans-number>1</ans-number>
<ans-content>Structured Question Line</ans-content>
</answer>
<answer ans-value="true" >
<ans-number>2</ans-number>
<ans-content>Structured Query Language </ans-content>
<ans-comment>Sql is presented in this lecture.</ans-comment>
</answer>
</answers>
<exe-comment>Hyper Dictionary:
http://www.hyperdictionary.com/dictionary/SQL
</exe-comment>
</exercise>
</exercises>
</subject>
</subjects>
<perform>
<by-teacher>Kjell Age Bringsrud</by-teacher>
<perform-date format="dd.mm.yy">12.05.2004</perform-date>
<perform-comment>Guest-teacher</perform-comment>
</perform>
<lecture-comment>The lecture is very important for the next exam!
</lecture-comment>
</lecture>
</lectures>
</course>
</education>

```

---



---

### A.1.11 File: course.xsl

---



---

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="/education">
<html>
<head>
<style>
BODY

```

```

{ background: #bb88dd;
  font-size : 10pt;
  font-family : Arial, Helvetica, sans-serif;
  text-align : center;
  color : purple;
}
table{
width: "80%";
background-color: #bb88bb;
border: 2px solid purple;
color: purple;
}
  th {
  border: 1px solid purple;
background-color: #DDA0DD;
}
  tr,td {
  border: 1px solid purple;
}
</style>
</head>
<body>
<center>
<h3><xsl:text>Course Presentation (by XML)</xsl:text></h3>
<xsl:for-each select="course" >
<xsl:apply-templates select="."/>
</xsl:for-each>
</center>
</body>
</html>
</xsl:template>

<xsl:template match="course" >
<table>
<tr>
<th>Code</th><th>Name</th><th>Relevant information:</th>
</tr>
<tr>
<td><center>
<b><xsl:text> INF </xsl:text>
<xsl:value-of select="substring-after(@course-code, '_')" /></b></center></td>
<td><center><xsl:value-of select="course-title" /></center> </td>
<td><xsl:value-of select="course-comment" /></td>
</tr>
</table>
<p><br /> </p>
<xsl:apply-templates select="./teachers"/>
</xsl:template>
<xsl:template match="teachers" >
<table>
<tr>
<th colspan="4">Responsible Teachers</th>
</tr>
<tr>
<th>Name</th><th>Role</th><th>E-mail</th><th>Homepage</th>
</tr>
<xsl:for-each select="teacher" >

```

```

<tr>
<td><xsl:value-of select="name" /></td>
<td><xsl:value-of select="role" /></td>
<td><xsl:value-of select="email" /></td>
<td><xsl:value-of select="homepage" /></td>
</tr>
</xsl:for-each>
</table>
</xsl:template>
</xsl:stylesheet>

```

---



---

### A.1.12 File: course-tests.html

---



---

```

<head>
<link rel="STYLESHEET" type="text/css" href="../style.css" >
</head>
<body class="p10" >
<p align="center">
<font size="4">
Different tests are registered in an XML database which may fetch here:
</p>
</font>
<ul>
<p>
<blockquote><li><font size="3" color="blue"><a href="course-tests.xml">
Represent all tests
</a>
</font>
<br />
(questions + answer alternatives)
</li>
</blockquote>
<blockquote><li><font size="3" color="blue">
<a href="test-answers.xml" target="\new">
Show all tests with the correctness value of all answer alternatives
</a>
</font>
<br /> (questions + answer alternatives and their correctness values)
</li>
</blockquote>
<blockquote>
<li><font size="3" color="blue"><a href="test-answers-if.xml">
Show all tests with the correctness value of all answer alternatives
</a>
</font>
<br /> (questions + just correct answers)
</p>
</li>
</blockquote>
<blockquote>
<li>
<font size="3" color="blue">
<a href="tests_graphic.xml">Tests with graphical elements</a>

```







```

Properties proper= new Properties();
proper.put("mail.smtp.host", mail_host);
Session sess=Session.getDefaultInstance(proper,null);
Vector mistakes= new Vector();
try{
Message your_email=new MimeMessage(sess);
if ((email_student.equals("")) || (email_student==null)){
mistakes.addElement(new String("OBS! Please enter your email address!"));
}
else{
//start:(if-2):
if (((email_student.indexOf("@")) >0) &&
((email_student.indexOf(".", email_student.indexOf("@")) !=-1)){
try{ //start:try-2
InternetAddress email_from= new InternetAddress(email_student);
InternetAddress email_to= new InternetAddress(email_teacher);
your_email.setFrom(email_from);
your_email.setSubject(email_subject);
your_email.setText(email_question);
your_email.setRecipient(Message.RecipientType.TO, email_to);
if (mistakes.size() < 1){
Transport.send(your_email);
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
<html>
<head>
<link rel="STYLESHEET" type="text/css" href="../style.css">
</head>
<body class="p10">
<p>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</p>
<table class="t2" align="center" width="70%">
<tr><td bgColor="pink">
<font class="leftPur" size="3pt">
Your email with :
</font>
<blockquote>
<p><font class="leftPur" size="2pt">Subject:</font>
<blockquote>
<%=email_subject%>
</blockquote>
</p>
<p><font class="leftPur" size="2pt">Content:</font>
<blockquote>
<%=email_question%>
</blockquote>
</p>
</blockquote>
</td></tr>
<tr><td>
<p align="center">
<font size="3pt" color="red"> <blink>Successfully sent</blink>
</font> to your teacher!
<br/>Your answer will be sent to:</font><br/>
<font align="center" size="4pt">
<b>"<%=email_student%> "</b>
</font>

```

```

</p>
</td></tr>
</table>
<p><a href="../main.jsp"><< Back to main page</a></p>
</body>
</html>
<%
}
}

    catch (AddressException e){
        mistakes.addElement(new String("Entered email is not valid!"));
    }
}
else{
    mistakes.addElement(new String("Your email address was wrong!<br/>" +
        "(A correct example: username@hotmail.com)"));
}
%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
<html>
<head>
<link rel="STYLESHEET" type="text/css" href="../style.css">
</head>
<body class="p10">
<p>&nbsp;&nbsp;&nbsp;</p>
<table class="t2" align="center" width="70%">
<tr><td bgColor="pink">
<font class="leftPur" size="3pt">
Your email is</font>
<font color="red" size="3pt">
    <blink>NOT sent! </blink></font>          <br />
<font class="leftPur" size="2pt">
    Following mistakes are observed:</font>
<blockquote>
<ul>
<%
for (int i=0; i< mistakes.size(); i++)
{
String error = (String)mistakes.elementAt(i);
out.println("<li>" + error + "</li>");
}
%>

</ul>
</blockquote>
</td></tr>
<tr><td>
<p align="center">
    Please correct your mistakes and send your email again. &nbsp;&nbsp;&nbsp;<br />
<font size="3pt" ><a href="mailtoTeacher.jsp"><< Back to "Ask question"</a>
</font>
</p>
</td></tr>
</table>
</body>
</html>

```

```

<%
}
}
}
}
catch(MessagingException e){
throw new JspException(e.getMessage());
}
}
%>

```

---

### A.2.3 File: testYourself.jsp

---

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<link rel="STYLESHEET" type="text/css" href="../style.css" >
</head>
<body class="p10">
<table align="center" class="t2">
<tr><th colspan="2"><font size="5pt">Test yourself!</font></th></tr>
<form action="continueTestYourself.jsp" metode="post">
<tr>
<td colspan="2">
<font class="leftPur" size="2">
<p>
Please choose a specific test number and/or subject that you want to
practice on.<br />Multiple choice questions will be fetched after your request.
</p></font>
<%@page import="java.sql.*" %>

<%
String driver= "org.gjt.mm.mysql.Driver";
Class.forName(driver); //downloading mysql driver
String mysql_base="jdbc:mysql://tafla.ifi.uio.no/MyUsername";
String user_name="MyUsername";
String password="MyPassword";
try{
Connection con1=DriverManager.getConnection(mysql_base, user_name, password);
Statement st1=con1.createStatement();
String query1="select distinct testNr from MultiplechoiceQuestions"+
" order by testNr";
ResultSet res1 = st1.executeQuery(query1);

StringBuffer buf1= new StringBuffer();
buf1.append("<table align=\"left\"><th>Tests:</th><tr><td align=\"center\">"+
"<select name=\"testNr\"><option selected value=\"\"> Select a nr.</option>");
while (res1.next()){
String gs=res1.getString("testNr");
buf1.append("<option value=\""+gs);
buf1.append(">");
buf1.append("<br>");
buf1.append("</option>");
}
buf1.append("</select></td></tr></table>");
out.println(buf1.toString());

```











```

buf.append(correct_count);
buf.append("<br />Number of wrong answers:  ");
buf.append(wrong_count);
buf.append("</div>");
}
}
}
}
catch (Exception e){
    out.println("The operation can not be proceeded!");
}
buf.append("</td></tr>");
buf.append("</table>");
out.println(buf.toString());
%>
</body>
</html>

```

---



---

### A.2.6 File: createTest.jsp

---



---

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<link rel="STYLESHEET" type="text/css" href="../style.css" >
</head>
<body class="p10">
<table align="center" class="t2">
<tr><th colspan="2"><font size="5pt">Test-creating</font></th></tr>
<form action="continueCreateTest.jsp" method="post">
<tr>
<td colspan="2">
<font class="leftPur" size="2pt">
Choose either a registered test number:
<div align="center">
<%@page import="java.sql.*" %>
<%
String driver= "org.gjt.mm.mysql.Driver";
Class.forName(driver);
String mysql_base="jdbc:mysql://tafla.ifi.uio.no/faribad";
String user_name="faribad";
String password="hoved";
try{
Connection con1=DriverManager.getConnection(mysql_base,user_name,password);
Statement st1=con1.createStatement();
String query1="select distinct testNr from MultiplechoiceQuestions"+
" order by testNr";

ResultSet res1 = st1.executeQuery(query1);
StringBuffer buf1= new StringBuffer();
buf1.append("<table><tr><td><select name=\"testNr\">");
buf1.append("<OPTION SELECTED value=\"\">Select A Test Nr.</OPTION>");

while (res1.next()){
String gs=res1.getString("testNr");
buf1.append("<option value=\"\"");
buf1.append(gs);

```





```

<tr>

<%@page import="java.sql.*, java.util.*" %>
<%
String testNR=request.getParameter("testNr");
String newTestNR=request.getParameter("newTestNr");
String questionSub=request.getParameter("questionSubject");
String newQuestionSub=request.getParameter("newQuestionSubject");
String quest=request.getParameter("question");
String ans_1=request.getParameter("answer1");
String ans_2 =request.getParameter("answer2");
String ans_3 =request.getParameter("answer3");
String ans_4 =request.getParameter("answer4");
String correct=request.getParameter("correctAnswer");

if((((testNR== null) || (testNR.equals("")) )
    && ((newTestNR== null) || (newTestNR.equals("")))) ) {
    out.println("<td bgColor=\"pink\"><font size=\"3pt\">" +
        "Test nr. was Not chosen!" +
        "<p align=\"center\">OBS! The required " +
        "fields are <font size=\"3\" color=\"red\">" +
        "NOT completely filled!" +
        " </font><br />Please try again." +
        " &nbsp;&nbsp;&nbsp;</p>" );
}
else if((((questionSub== null) || (questionSub.equals(""))))
    && ((newQuestionSub== null) || (newQuestionSub.equals("")))) ) {
    out.println("<td bgColor=\"pink\"><font size=\"3pt\">"+
        "Subject was Not chosen!" +
        "<p align=\"center\">OBS! The required " +
        "fields are <font size=\"3\" color=\"red\">" +
        "NOT completely filled!</font><br />" +
        " Please try again. &nbsp;&nbsp;&nbsp;</p>" );
}
else if( ((quest== null) || (quest.equals("")))) {
    out.println("<td bgColor=\"pink\"><font size=\"3pt\">"+
        "Question was empty!" +
        "<p align=\"center\">OBS! The required " +
        "fields are <font size=\"3\" color=\"red\">" +
        "NOT completely filled! </font><br />" +
        "Please try again. &nbsp;&nbsp;&nbsp;</p>" );
}
else if( ((ans_1== null) || (ans_1.equals("")))) {
    out.println("<td bgColor=\"pink\"><font size=\"3pt\">"+
        "Answer 1 was empty!<p align=\"center\">OBS! " +
        "The required fields are <font size=\"3\" " +
        "color=\"red\">NOT completely filled!" +
        " </font><br />Please try again. &nbsp;&nbsp;&nbsp;</p>" );
}
else if( ((ans_2== null) || (ans_2.equals("")))) {
    out.println("<td bgColor=\"pink\"><font size=\"3pt\"> Answer 2" +
        " was empty!<p align=\"center\">OBS! The required fields are" +
        " <font size=\"3\" color=\"red\">NOT completely filled!" +
        " </font><br />Please try again. &nbsp;&nbsp;&nbsp;</p>");
}
else if( ((correct== null) || (correct.equals("")))) ) {
    out.println("<td bgColor=\"pink\"><font size=\"3pt\">Correct answer" +

```



```

</form>

<%
session.putValue("ses_testNR",testNR);
session.putValue("ses_newTestNR",newTestNR);
session.putValue("ses_questionSub",questionSub);
session.putValue("ses_newQuestionSub",newQuestionSub);
session.putValue("ses_quest",quest);
session.putValue("ses_ans_1",ans_1);
session.putValue("ses_ans_2",ans_2);
session.putValue("ses_ans_3",ans_3);
session.putValue("ses_ans_4",ans_4);
session.putValue("ses_correct",correct);
}
%>

</font>
</td></tr>
</table>
<p><font size="3"><a href="createTest.jsp"> << Back </a></font></p>
</body>
</html>

```

### A.2.8 File: registerQuestion.jsp

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<head>
<head>
<link rel="STYLESHEET" type="text/css" href="../style.css" >
</head>
<body class="p10" >
<table align="center" class="t2">

<%@page import="java.sql.*, java.util.*" %>
<%

String tn=(String)session.getValue("ses_testNR");
String ntn=(String)session.getValue("ses_newTestNR");
String qs=(String)session.getValue("ses_questionSub");
String nqs=(String)session.getValue("ses_newQuestionSub");
String que=(String)session.getValue("ses_quest");
String an1=(String)session.getValue("ses_ans_1");
String an2=(String)session.getValue("ses_ans_2");
String an3=(String)session.getValue("ses_ans_3");
String an4=(String)session.getValue("ses_ans_4");
String cor=(String)session.getValue("ses_correct");
String nr, sub=null;

if (!(tn == null) || (tn=="")) && ((ntn == null) || (ntn==""))){
    nr=tn;
}
else {
    nr=ntn;
}

```

```

        if (!(qs == null) || (qs=="")) && ((nqs == null) || (nqs==""))){
            sub=qs;
        }
    else {
        sub=nqs;
    }

    if (!(tn == null) || (tn.equals("")) &&
        ((ntn == null) || (ntn.equals("")))){
        nr=tn;
    }
    else {
        nr=ntn;
    }
    if (!(qs == null) || (qs.equals("")) &&
        ((nqs == null) || (nqs.equals("")))){
        sub=qs;
    }
    else {
        sub=nqs;
    }

String driver= "org.gjt.mm.mysql.Driver";
Class.forName(driver);
String mysql_base="jdbc:mysql://tafla.ifi.uio.no/faribad";
String user_name="faribad";
String password="hoved";
try{

if((tn == null)|| (tn.equals("")) &&
    ((ntn == null) || (ntn.equals("")))) || (qs == null)||
    (qs.equals("")) && ((nqs == null) || (nqs.equals("")))){
    out.println("<tr><th>Can not proceed the request!!" +
        "Check the values of test nr. or subject!" +
        "<a href=\"continueCreateTest.jsp\">" +
        " << Go Back</a></th></tr>");
}
else{
Connection connect=
    DriverManager.getConnection(mysql_base, user_name, password);
    Statement state= connect.createStatement();
String quer="insert into MultiplechoiceQuestions" +
    "(id,testNr,subject,question,answer1,answer2,answer3," +
    "answer4,correctAnswer) values('','" +nr + "',''" + sub + "',''" +
    + que+"','" + an1+ "',''" + an2 + "',''" + an3+ "',''" + an4
    + "',''" + cor +"')";
state.executeUpdate(quer);
state.close();
connect.close();
out.println("<tr><th>The question is successfully registered" +
    " into the database.</th></tr>");
}
}
catch(SQLException e){
out.println("<tr><th>Sorry, the system is unable to register" +
    " the question into the database!</th></tr>");
}
}

```





---

---

## A.4 ActionScripts used for the buttons of UML & NIAM quizzes

---

---

---

### A.4.1 ActionScript: Tire-knappen in UML\_Quiz.fla

---

```
on(press) {
    startDrag(this);
    _root.answer1="The movement is";
}

on(release) {
    stopDrag();
    if (this._droptarget == "/tire") {
        _root.answer1="Correct!";
        _root.allCorrect +=1;
    }
    else{
        _root.answer1 = "wrong! Try again.";
    }
}
```

---

### A.4.2 ActionScript: Driver-knappen in UML\_Quiz.fla

---

```
on(press) {
    startDrag(this);
    _root.answer1="The movement is";
}

on(release){
    stopDrag();
    if (this._droptarget == "/driver") {
        _root.answer1="Correct!";
        _root.allCorrect +=1;
    }
    else{
        _root.answer1 = "wrong! try again.";
    }
}
```

---

### A.4.3 ActionScript: Car-knappen in UML\_Quiz.fla

---

```
on(press) {
    startDrag(this);
    _root.answer1="The movement is";
}

on(release){
    stopDrag();
    if (this._droptarget == "/car") {
        _root.answer1="Correct!";
        _root.allCorrect +=1;
    }
}
```

```

    }
    else{
        _root.answer1 = "wrong! Try again.";
    }
}

```

#### A.4.4 ActionScript: Person-knappen in UML\_Quiz.fla

---

```

on(press) {
    startDrag(this);
    _root.answer1="The movement is";
}

on(release) {
    stopDrag();
    if (this._droptarget == "/person") {
        _root.answer1="Correct!";
        _root.allCorrect +=1;
    }
    else{
        _root.answer1 = "wrong! Try again.";
    }
}

```

### A.5 ActionScript of alphabet quiz (quiz-alfabeth.fla)

---

```

allCorrect = 0;

_root.onEnterFrame= function(){
    if(_root.allCorrect == 0) {
        _root.endComment = "Start to solve the problem!";
    }
    else{
        if(_root.allCorrect >= 1){
            _root.endComment = "Please continue!";
            if(_root.allCorrect == 5){
                _root.answer1="You are finished with";
                _root.endComment = "5 correct! Congratulation:-)";
            }
        }
        else{
            _root.endComment = "Please start again!";
        }
    }
}

```

#### A.5.1 ActionScript codes for the used buttons A, B, C, D, E

---

```

on(press) {
    startDrag(this);
    _root.answer1="The movement is ";
}

```

```

    }
on(release) {
    stopDrag();
    if (this._droptarget == "/b"){
        _root.answer1 = "Correct!";
        _root.allCorrect +=1;
    }
    else{
        _root.answer1 = "Wrong! Try again.";
    }
}
}

```

## A.6 Flash integrated with XML (Sec. A.6.1, A.6.2)

### A.6.1 ActionScript of the Flash movie (courseTests.fla)

```

testsData = new XML();
testsData.load("tests.xml");
testsData.onLoad = fileLoading;
function fileLoading(success) {
    if (success) {
        ShowData(this.firstChild);
    }
    else{
        output_data = "<font size='1' color='black'>" +
            "The XML file could NOT be loaded!</font>";
    }
}
function ShowData(MyNode) {
    if (MyNode.nodeName.toUpperCase() == "TEST") {
        output_data = "";
        question= MyNode.firstChild;
        while(question != null){
            if (question.nodeName.toUpperCase()== "QUESTION"){
                nr="";
                subject="";
                content="";
                comment="";
                answer="";
            }
            theNode = question.firstChild;
            while (theNode != null) {
                if (theNode.nodeName.toUpperCase()== "NR"){
                    nr= theNode.firstChild.nodeValue;
                }
                if (theNode.nodeName.toUpperCase() == "SUBJECT") {
                    subject = theNode.firstChild.nodeValue;
                }
                if (theNode.nodeName.toUpperCase() == "CONTENT") {
                    content = theNode.firstChild.nodeValue;
                }
                if (theNode.nodeName.toUpperCase() == "COMMENT") {
                    comment = theNode.firstChild.nodeValue;
                }
                if (theNode.nodeName.toUpperCase()== "ANSWER"){

```

```

        answer = theNode.firstChild.nodeValue;
    }
    theNode = theNode.nextSibling;
}
output_data += "<p><font size='1' color='red'><b>Question number &nbsp;   ";
    + nr+ " :</b><br><b>Subject:</b>&nbsp;   &nbsp;  "+ subject +
    "&nbsp;   &nbsp;  <b>Content:</b>&nbsp;   &nbsp;  "+ content +
    "<br><b>Comment:</b>&nbsp;   &nbsp;  "+ comment +
    "&nbsp;   &nbsp;  <b>Answer:</b>&nbsp;   &nbsp;   "+ answer +
    "</font></p>";
    }   question = question.nextSibling;
}
}
}
}

```

---

## A.6.2 XML file (tests.xml)

---

```

<test>
<question>
<nr>1</nr>
<subject>Semistructured DB</subject>
<content>What XML stands for?</content>
<comment>XML needs XSL technology!</comment>
<answer>Extensible Markup Language</answer>
</question>
<question>
<nr>2</nr>
<subject>Database</subject>
<content>What kind of database is Mysql?</content>
<comment>The question was discussed in the class!</comment>
<answer>Mysql is a relational database</answer>
</question>
<question>
<nr>3</nr>
<subject>Extracting data</subject>
<content>What SQL is standing for?</content>
<comment>It is a query language.</comment>
<answer>Structured Query Language</answer>
</question>
</test>

```

---

## Appendix B

# Layouts of my *ODC* example

In this section, some of the result outputs and layouts provided by my *ODC* on-line course example, are presented.

### B.1 Layout files developed by XML technologies

The following figures represent the layout files belonged to my 'ODC' on-line course example developed by XML, XSL and DTD technologies.

### B.2 Layouts of the *Course info. (XML)* menu option (courseInfo.eps)

Online Database Course (ODC)		
Course Presentation (by XML)		
Code	Name	Relevant information:
INF 102	System Development	The course is virtual as a trial version!

Responsible Teachers			
Name	Role	E-mail	Homepage
Teacher 1	Main teacher	teacher1@ifi.uio.no	<a href="http://heim.ifi.uio.no/~teacher1">http://heim.ifi.uio.no/~teacher1</a>
Teacher 2	Guest teacher	teacher2@ifi.uio.no	<a href="http://heim.ifi.uio.no/~teacher2">http://heim.ifi.uio.no/~teacher2</a>

### B.3 Layouts of the *Show Test (XML)* menu option (No. 1, 2)

Online Database Course (ODC)

Different tests are registered in an XML database which may fetch here:

- Represent all tests  
(questions + answer alternatives)
- Show all tests with the correctness value of all answer alternatives  
(questions + answer alternatives and their correctness values)
- Show all tests with the correctness value of all answer alternatives  
(questions + just correct answers)
- Tests with graphical elements  
(Usage of Textual/Graphic/Sound objects)

Online Database Course (ODC)

List of the questions of Test 1 represented by XML

Test nr. 1	
Subject:	Database
Number of questions:	"2"
Comment:	Multiple choice questions

Questions:	
<b>Question nr.: 1</b>	
Subject:	Mysql
Comment:	The question was discussed in the class!
Question:	" What kind of database is Mysql? "
"3" answer alternatives:	
Nr. 1:	It is an XML DB. Comment: (To fetch the data, XML tags should be used!)
Nr. 2:	It is an object oriented DB. Comment: (To fetch the data, Java programming should be used.)
Nr. 3:	It is a relational DB. Comment: To fetch the data, SQL should be used.)
<b>Question nr.: 2</b>	
Subject:	SQL
Comment:	It is represented in the manual!
Question:	" What does SQL stand for? "
"3" answer alternatives:	
Nr. 1:	Structured Question Line Comment: Sql is a language to retrieve data from a relation database.
Nr. 2:	Structured Query Language Comment: (We use sql to fetch data from mysql database.)
Nr. 3:	Structional Querring Language Comment: No comments!

### B.4 Layouts of the *Show Test (XML)* menu option (No. 3)

Here is the answer list for all questions on different registered tests (represented by XML)

Test nr. 1	
Subject:	Database
Number of questions:	"2"
Questions:	
<b>Question nr. 1:</b> What kind of database is Mysql?	
"3" alternative answers:	
Answer nr. 1: It is an XML DB. Correct value: <span style="color: red;">False</span>	
Answer nr. 2: It is an object oriented DB. Correct value: <span style="color: red;">False</span>	
Answer nr. 3: It is a relational DB. Correct value: <span style="color: red;">True</span>	
<b>Question nr. 2:</b> What does SQL stand for?	
"3" alternative answers:	
Answer nr. 1: Structured Question Line Correct value: <span style="color: red;">False</span>	
Answer nr. 2: Structured Query Language Correct value: <span style="color: red;">True</span>	
Answer nr. 3: Structional Querring Language Correct value: <span style="color: red;">False</span>	

Test nr. 2	
Subject:	Algebra
Number of questions:	"2"
Questions:	
<b>Question nr. 1:</b> 2+3	
"3" alternative answers:	
Answer nr. 1: 5 Correct value: <span style="color: red;">True</span>	

## B.5 Layouts of the *Show Test (XML)* menu option (No. 4)

### Online Database Course (ODC)

Here is the answer list for all questions on different registered tests (represented by XML)

#### Test nr. 1

<b>Subject:</b>	Database
<b>Number of questions:</b>	"2"

#### Questions:

**Question nr. 1:** What kind of database is Mysql?

Number of alternative answers: "3"  
Correct answer: **It is a relational DB.**

**Question nr. 2:** What does SQL stand for?

Number of alternative answers: "3"  
Correct answer: **Structured Query Language**

#### Test nr. 2

<b>Subject:</b>	Algebra
<b>Number of questions:</b>	"2"

#### Questions:

**Question nr. 1:** 2+3

Number of alternative answers: "3"  
Correct answer: **5**

**Question nr. 2:** 20-30

Number of alternative answers: "3"  
Correct answer: **-10**



**B.6 Layouts of the *Show Test (XML)* menu option (No. 5)**

Online Database Course (ODC)


---

Usage of divers micromedia elements in XML

---

Test nr. 1			
Subject:	Representing graphical objects by XML!		
Number of questions:	"3"		

Questions:			
<b>Question nr. 1:</b> Who is the youngest one?			
Number of alternative answers: "4"			
<b>Answer nr. 1 :</b> (Graphical object) 	<b>Answer nr. 2 :</b> (Graphical object) 	<b>Answer nr. 3 :</b> (Graphical object) 	<b>Answer nr. 4 :</b> (Textual file)  They all are as old as each other!
<b>Question nr. 2:</b> What does HTML stand for?			
Number of alternative answers: "4"			
<b>Answer nr. 1 :</b> (Textual file)  Hyper Text Markup Language	<b>Answer nr. 2 :</b> (Textual file)  High The Markup Language	<b>Answer nr. 3 :</b> (Textual file)  Hyper The Markup Language	<b>Answer nr. 4 :</b> (Textual file)  Hyper Transfer Markup Language
<b>Question nr. 3:</b> Which of these recorded sounds is not related to a human?			
Number of alternative answers: "4"			
<b>Answer nr. 1 :</b> (Sound file)  <a href="#">sound_ahum.wav</a>	<b>Answer nr. 2 :</b> (Sound file)  <a href="#">sound_horn.wav</a>	<b>Answer nr. 3 :</b> (Sound file)  <a href="#">sound_different.wav</a>	<b>Answer nr. 4 :</b> (Sound file)  <a href="#">sound_relax.wav</a>

## B.7 Layout files developed by JSP technology

The following figures represent the layouts belonged to JSP files of my 'ODC' online course example.

## B.8 Layouts of the *Main* page (No. 1, 2)

Online Database Course (ODC)

**Welcome to ODC**

Date: 24.May.2005

**Weekly messages for the course:**

Today is **Tuesday**;  
Teaching-class cl. 14:15-16:00!

This trial Online Database Course is offering an interactive learning process to those students who want to learn the fundamental concepts of database systems.

Divers exercises and tests are developed by the following facilities:

- Flash macromedia,
- JSP, Java Servlet
- mysql and XML databases

Online Database Course (ODC)

**Welcome to ODC**

Date: 24.May.2005

**Weekly messages for the course:**

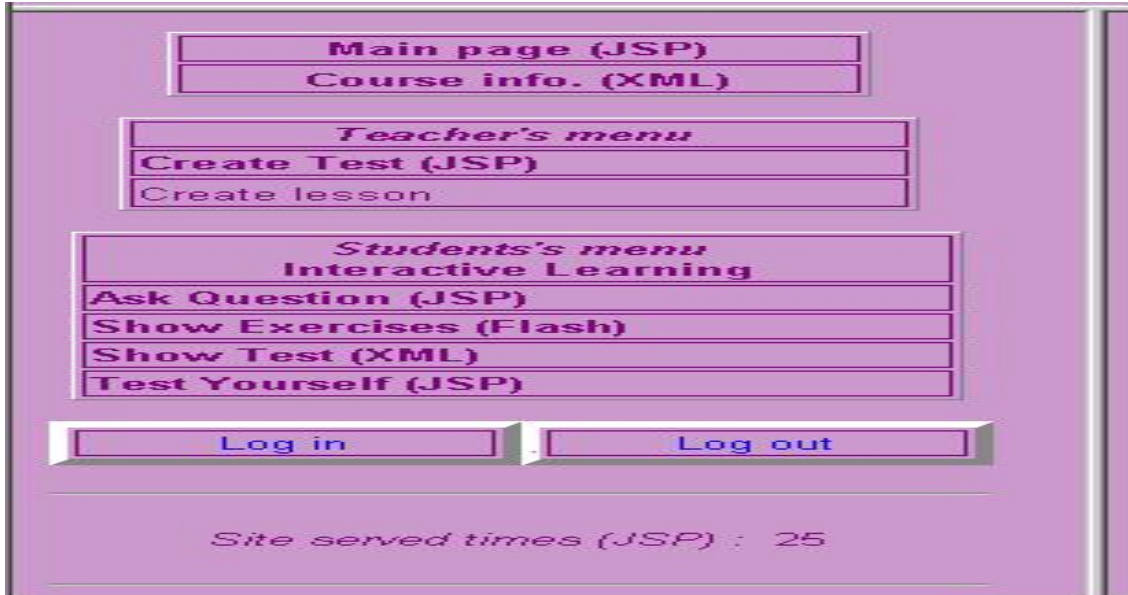
Today is **Tuesday**;  
Teaching-class cl. 14:15-16:00!

This trial Online Database Course is offering an interactive learning process to those students who want to learn the fundamental concepts of database systems.

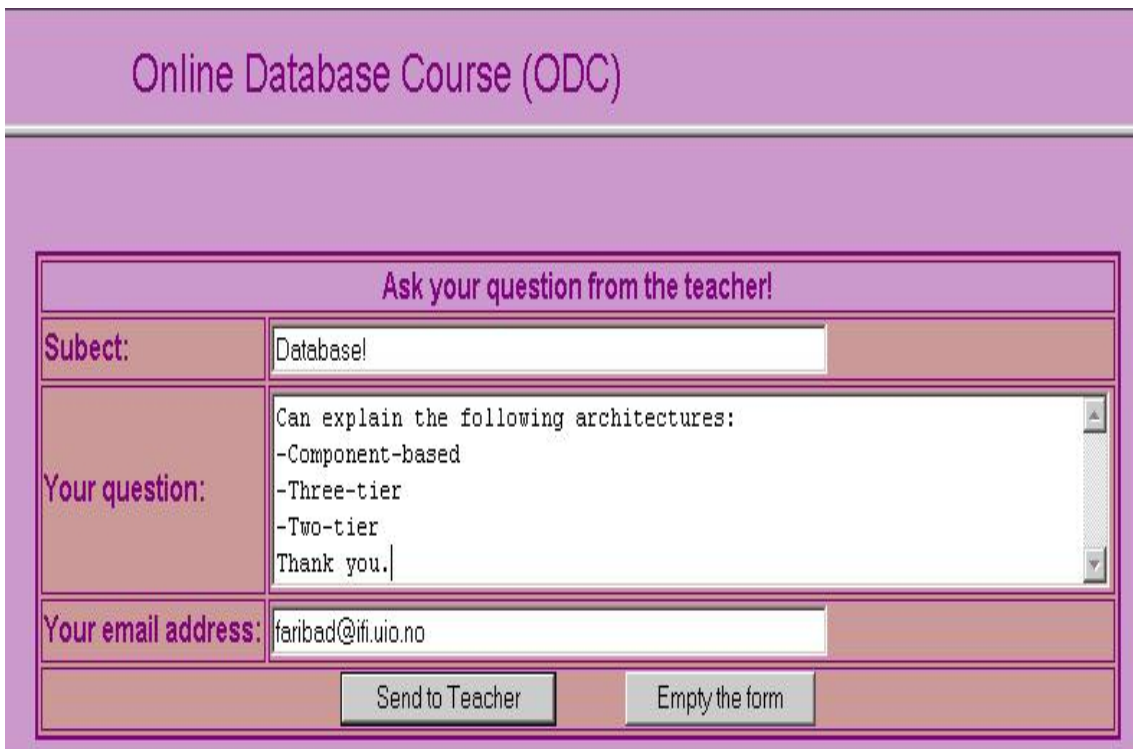
Divers exercises and tests are developed by the following facilities:

- Flash macromedia,
- JSP, Java Servlet
- mysql and XML databases

### B.9 Layout of the Menu page



### B.10 Layout of the Ask Question (JSP) menu option



## B.11 Layouts of the *Create Test (JSP)* menu option (No. 1, 2)

Online Database Course (ODC)

**Test-creating**

Choose either a registered test number:

Select A Test Nr.  or enter a new test number:

Choose either a registered subject:

Please Select A Subject  enter a new subject:

**Answers** (At least two answer-alternatives.)

Alternative 1:

Alternative 2:

Alternative 3:

Alternative 4:

**Correct answer:**

1.  2.  3.  4.

Online Database Course (ODC)

**Test-creating**

Choose either a registered test number:

Select A Test Nr.  or enter a new test number:

Choose either a registered subject:

Please Select A Subject  enter a new subject:

**Answers** (At least two answer-alternatives.)

Alternative 1:

Alternative 2:

Alternative 3:

Alternative 4:

**Correct answer:**

1.  2.  3.  4.

**B.12 Layouts of the *Create Test (JSP)* menu option (No. 3, 4)**

Online Database Course (ODC)

**Test-creating**

Choose either a registered test number:  or enter a new test number:

Choose either a registered subject:  or enter a new subject:

**Question**

**Answers** (At least two answer-alternatives.)

Alternative 1:

Alternative 2:

Alternative 3:

Alternative 4:

**Correct answer:**  1  2  3  4

Online Database Course (ODC)

**The following question is entered:**  
(Please confirm to register the question into DB)

Test nr:	3
Subject:	Mathematics
Question:	55-23
<b>Answer's alternatives</b>	
Answer1:	34
Answer2:	32
Answer 3:	12
Answer 4:	123
Correct answer:	Nr. 2

Confirm:

<< Back





**B.15 Layouts of the *Test Yourself (JSP)* menu option (No. 2)**

Online Database Course (ODC)

**Test yourself!**

Please choose a specific test number and/or subject that you want to practice on.  
Multiple choice questions will be fetched after your request.

**Tests:** Select a nr. [dropdown menu with options 1, 2, 3, 4, 5, 6, 7, 8, 9]

**Subjects:** Select an item. [dropdown menu]

**Question**

Show the questions! Remove the choices!

**B.16 Layouts of the *Test Yourself (JSP)* menu option (No. 3)**

Online Database Course (ODC)

**Test yourself!**

Please choose a specific test number and/or subject that you want to practice on.  
Multiple choice questions will be fetched after your request.

**Tests:** Select a nr. [dropdown menu]

**Subjects:** Select an item. [dropdown menu with options: Addition, database, DB, Representing graphical objects, Subtraction]

**Question**

Show the questions! Remove

**B.17 Layouts of the *Test Yourself (JSP)* menu option (No. 4)**

Online Database Course (ODBC)	
Subject: Addition Test number: 1 Number of fetched questions: 2	
Question nr. 1: 4+40	
answer 1: 404 answer 2: 440 answer 3: 044 answer 4: 44	
The correct answer is :	
Nr. 1 <input type="checkbox"/> Nr. 2 <input type="checkbox"/> Nr. 3 <input type="checkbox"/> Nr. 4 <input type="checkbox"/>	
Question nr. 2: 6+7	
answer 1: 11 answer 2: 23 answer 3: 33 answer 4: 13	
The correct answer is :	
Nr. 1 <input type="checkbox"/> Nr. 2 <input type="checkbox"/> Nr. 3 <input type="checkbox"/> Nr. 4 <input type="checkbox"/>	
<input type="button" value="Check my answers"/> <input type="button" value="Let me try again"/>	



**B.18 Layouts of the *Test Yourself (JSP)* menu option (No. 5)**

Online Database Course (ODC)

The Result Of Your Test is reported below:

Your answer to question (1) was: Wrong:(  
Correct answer:nr. 4

Your answer to question (2) was: Wrong:(  
Correct answer:nr. 4

Your result:  
Number of correct answers: 0  
Number of wrong answers: 2

**B.19 Layouts of the *Test Yourself (JSP)* menu option (No. 6)**

Online Database Course (ODC)

The Result Of Your Test is reported below:

Your answer to question (1) was: Correct!

Your answer to question (2) was: Correct!

Congratulation!!

## B.20 Layouts of the *Test Yourself (JSP)* menu option (No. 7)

Online Database Course (ODC)	
Subject: Addition Test number: 1 Number of fetched questions: 2	
Question nr. 1: 4+40 answer 1: 404 answer 2: 440 answer 3: 044 answer 4: 44 The correct answer is : Nr. 1 <input type="checkbox"/> Nr. 2 <input type="checkbox"/> Nr. 3 <input type="checkbox"/> Nr. 4 <input checked="" type="checkbox"/>	
Question nr. 2: 6+7 answer 1: 11 answer 2: 23 answer 3: 33 answer 4: 13 The correct answer is : Nr. 1 <input type="checkbox"/> Nr. 2 <input type="checkbox"/> Nr. 3 <input type="checkbox"/> Nr. 4 <input checked="" type="checkbox"/>	
<input type="button" value="Check my answers"/> <input type="button" value="Let me try again"/>	

## B.21 Layouts of the *Test Yourself (JSP)* menu option (No. 8)

Online Database Course (ODC)	
The Result Of Your Test is reported below:	
Your answer to question (1) was: Correct!	
Your answer to question (2) was: Wrong:(	
Correct answer: nr. 4	
<b>Your result:</b> Number of correct answers: 1 Number of wrong answers: 1	

## Appendix C

# Flash MX learning examples

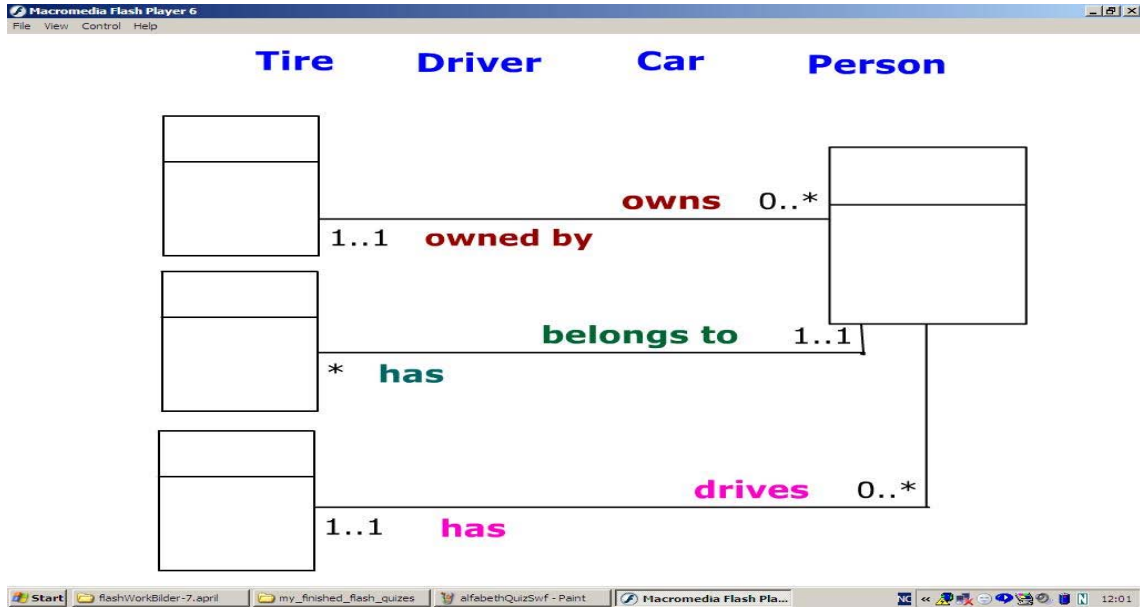
In this section, my various learning objects developed by Flash MX, are presented.

### C.1 My developed Flash MX learning objects

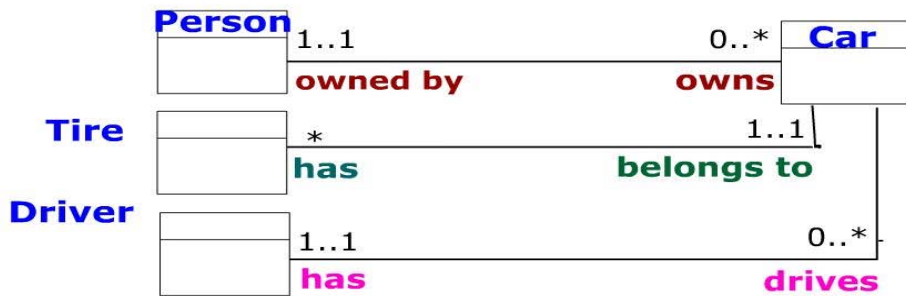
#### C.1.1 The sub-menu of the *Show Exercise (Flash)* menu option

Online Database Course (ODC)
<b>Flash developed databse-exercises</b>
<ul style="list-style-type: none"><li>• UML Quiz (.swf)</li><li>• NIAM Quiz (.swf)</li><li>• Alphabeth Quiz (.swf)</li></ul>
<b>XML integrated with Flash-MX</b>
<ul style="list-style-type: none"><li>• Question &amp; Answers (XML by Flash)</li></ul>
<b>A Flash animation &amp; a short Flash MX tutorial</b>
<ul style="list-style-type: none"><li>• Learning Flash by Flash (animation)</li></ul>

C.1.2 Menu option: UML Quiz (.swf)

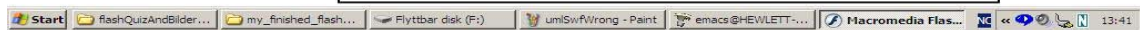


**Exercise 1: Complete the following UML diagram:**  
 (Place each element in the right place)

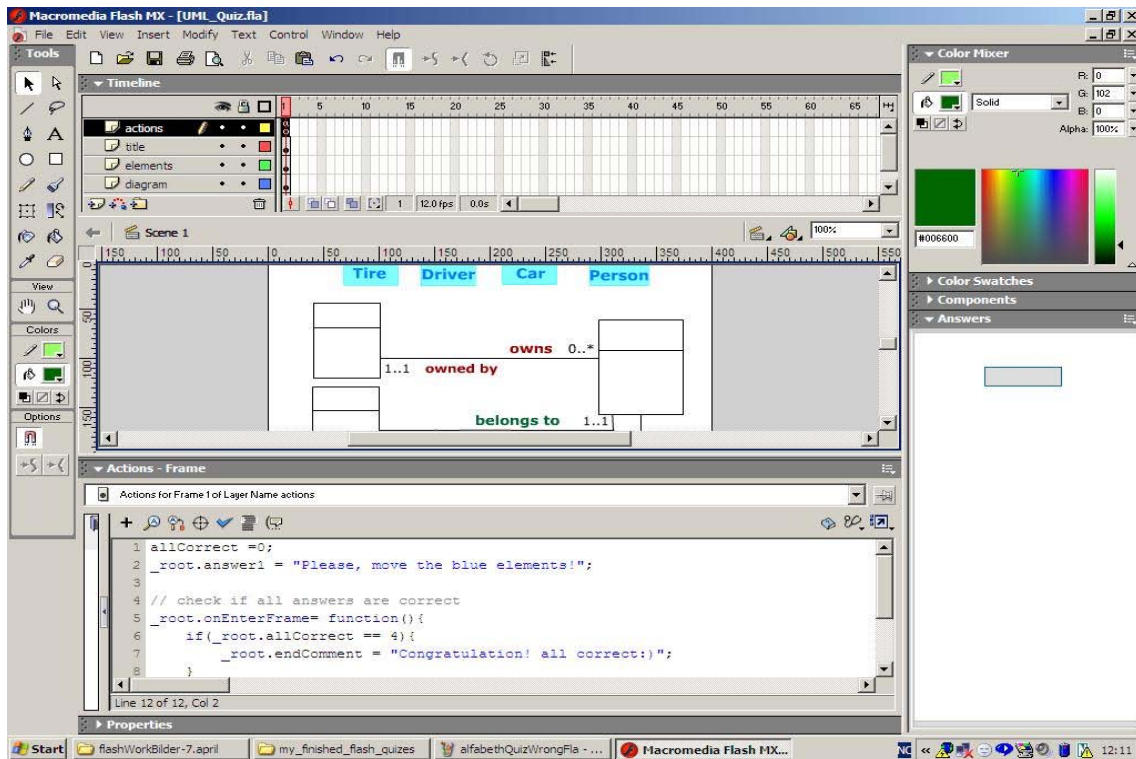
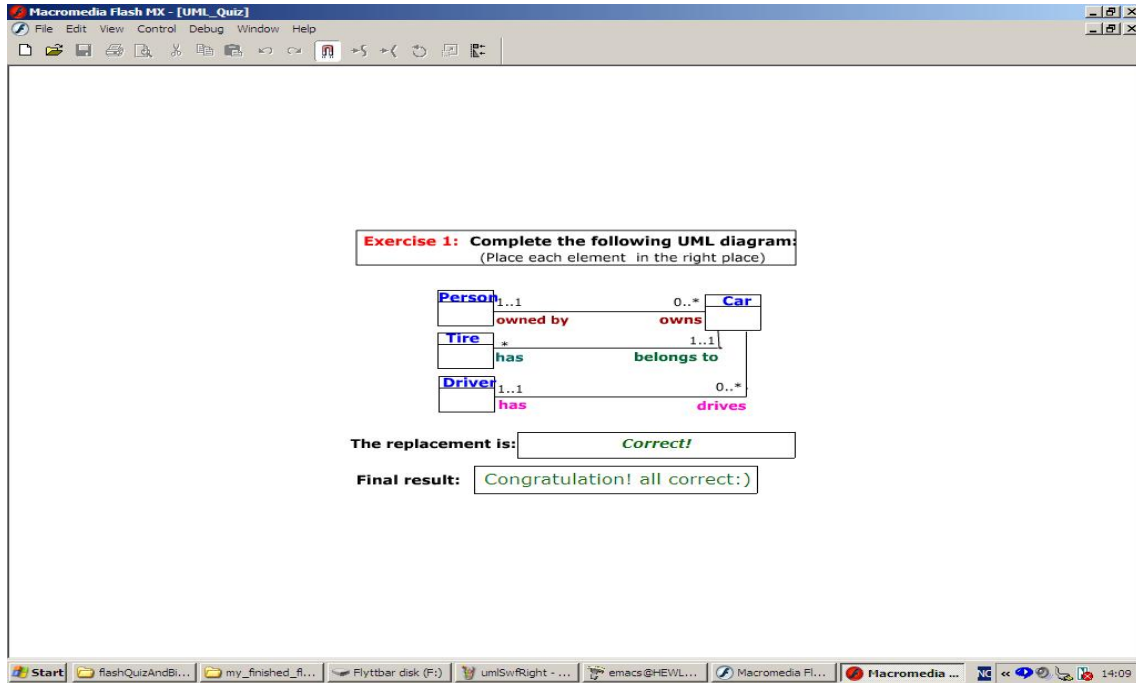


The replacement is: **Correct!**

Final result: **Moving process!**



C.1.3 Menu option: UML Quiz (.swf)

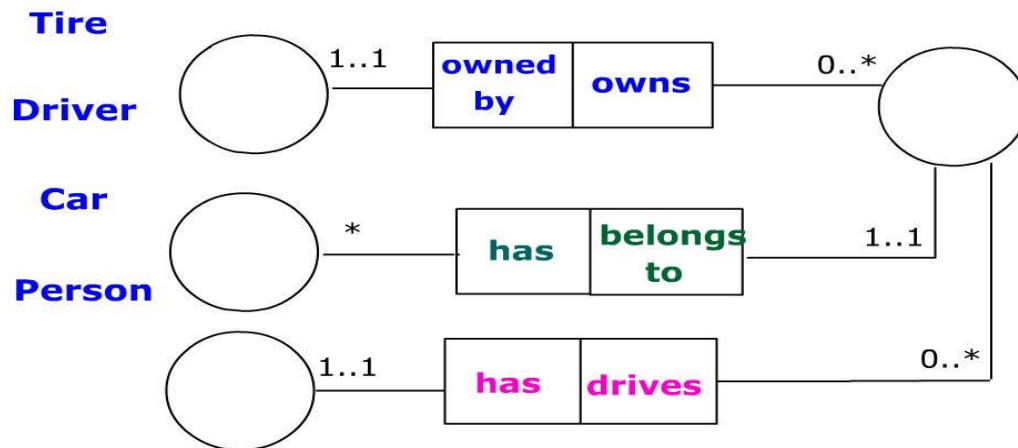


C.1.4 Menu option: NIAM Quiz (.swf)



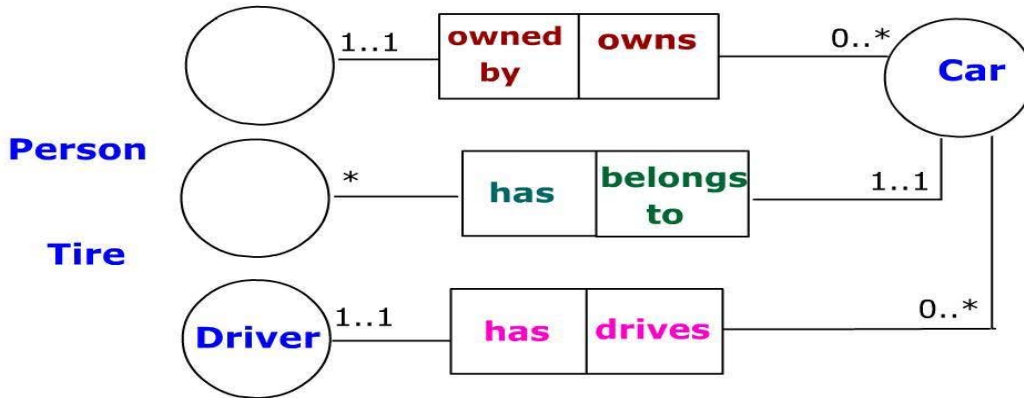
**Exercise 1: Creating NIAM diagram**

Drag and put the name of each element in the right place.



**Exercise 1: Creating NIAM diagram**

Drag and put the blue element names in the right place.

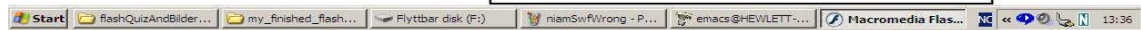


The replacement is:

*Correct!*

Final result:

**Please continue!**





C.1.5 Menu option: NNIAM Quiz (.swf)

**Macromedia Flash Player 6**  
File View Control Help

**Exercise 1: Creating NIAM diagram**  
Drag and put the blue element names in the right place.

The replacement is: **wrong! Try again.**

Final result: **Please continue!**

Start Fariba my\_finished\_flash... Flyttbar disk (F:) Uten navn - Paint emacs@HEWLETT... Macromedia Flas... 13:33

**Macromedia Flash Player 6**  
File View Control Help

**Exercise 1: Creating NIAM diagram**  
Drag and put the blue element names in the right place.

The replacement is: **Correct!**

Final result: **Congratulation!**

Start flashQuizAndBilder... my\_finished\_flash... Flyttbar disk (F:) niamSwfWrong - P... emacs@HEWLETT... Macromedia Flas... 13:38

**Exercise 1: Creating NIAM diagram**  
 Drag and put the blue element names in the right place.

Tire (1..1) owned by Car (0..\*)  
 Car (\*) belongs to Driver (1..1)  
 Driver (1..1) drives Person (0..\*)

The replacement is:

Color Mixer: #990000

Color Swatches: Answers

Color Mixer: #990000

Color Swatches: Answers

```

1 on (press) {
2   startDrag(this);
3   _root.answer1="The movement is";
4 }
5
6 on (release) {
7   stopDrag();
8   if (this._droptarget == "/car") {
9     _root.answer1="Correct!";
10    _root.allCorrect +=1;
11   }
12   else {
13     _root.answer1 = "wrong! Try again.";
  
```



C.1.6 Menu option: Alphabet Quiz (.swf)



**Exercise:** Put the following letters in the right order.

**E D C B A**

--	--	--	--	--

**Answer:**

**Final result:** **Start to solve the problem!**

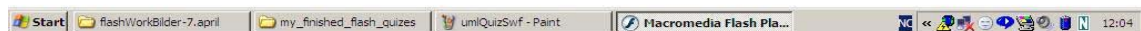


**Exercise:** Put the following letters in the right order.

<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>
----------	----------	----------	----------	----------

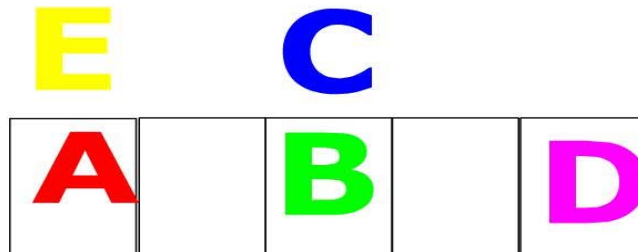
**Answer:**

**Final result:** **5 correct! Congratulation:-**



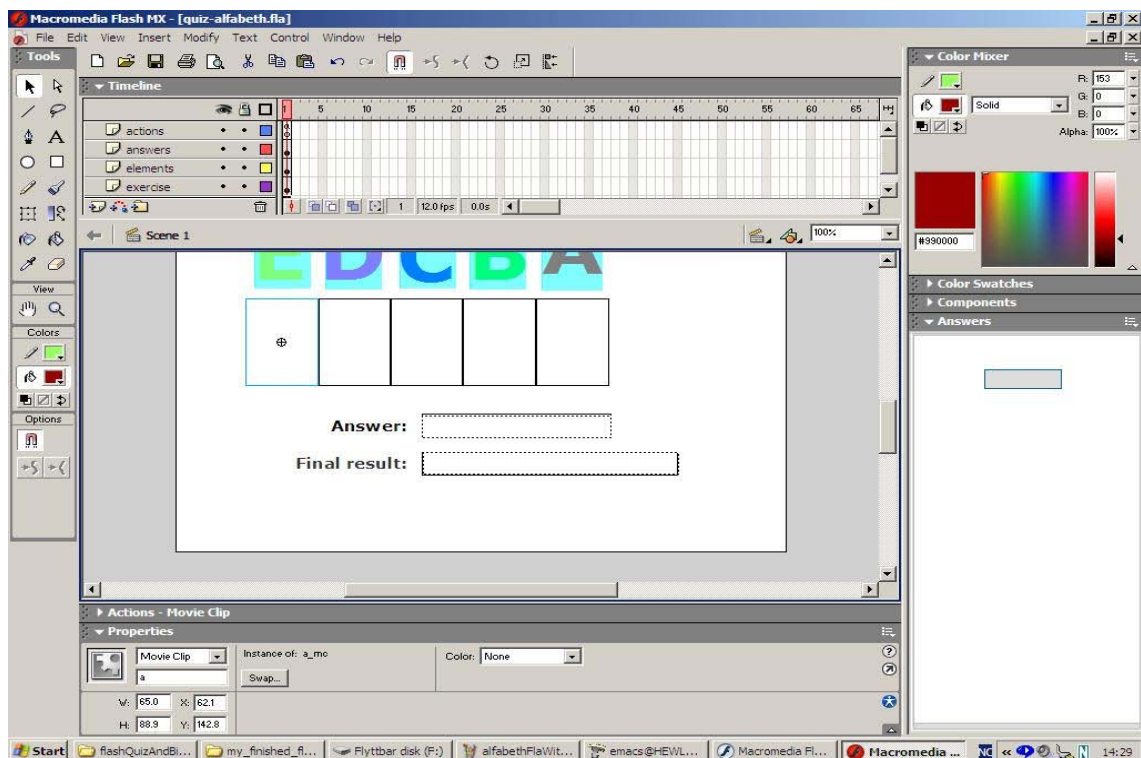
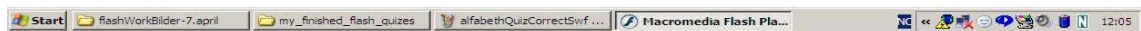


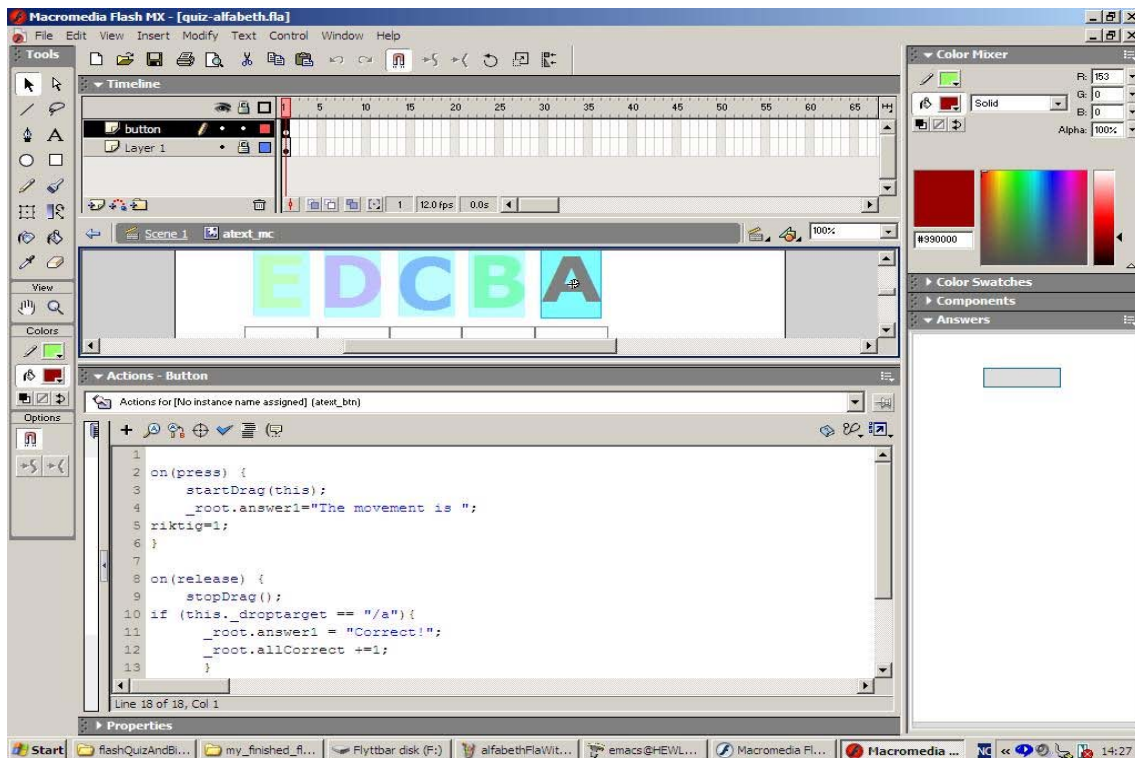
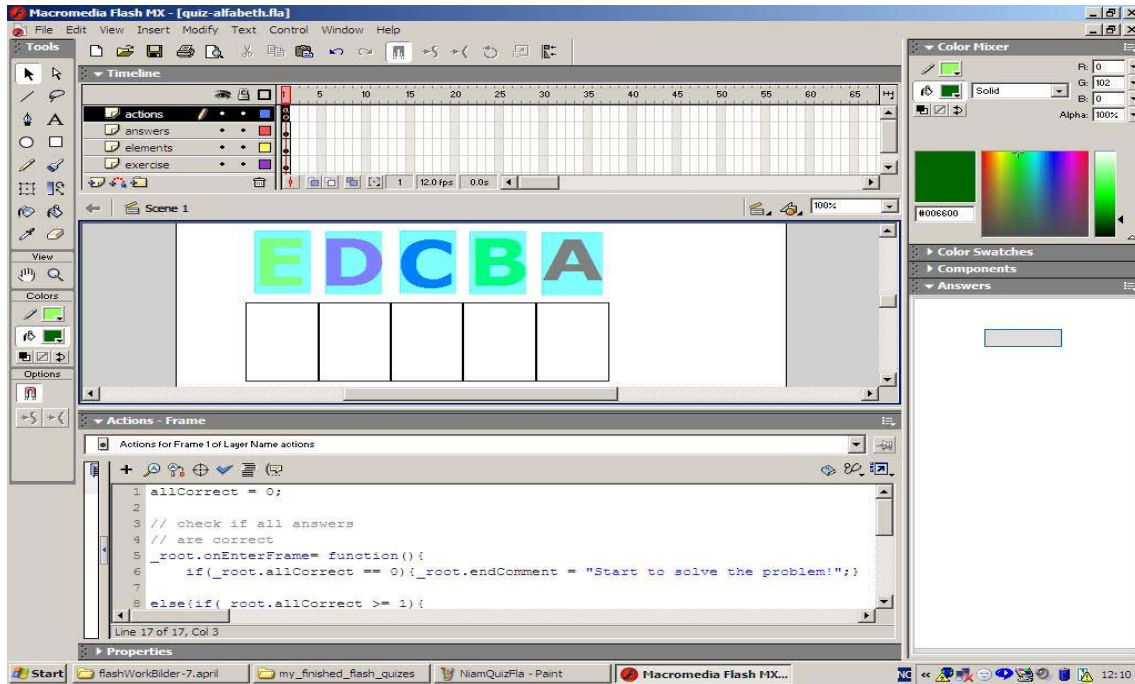
**Exercise:** Put the following letters in the right order.



**Answer:** Wrong! Try again.

**Final result:** Please continue!





**C.1.7 Menu option: Question & Answers (XML by Flash)****Online Database Course (ODC)****Question number 1:**

**Subject:** Semistructured DB **Content:** What does XML stand for?

**Comment:** XML needs XSL technology! **Answer:** Extensible Markup Language

**Question number 2:**

**Subject:** Database **Content:** What kind of database is Mysql?

**Comment:** The question was discussed in the class! **Answer:** Mysql is a relational database

**Question number 3:**

**Subject:** Extracting data **Content:** What does SQL stand for?