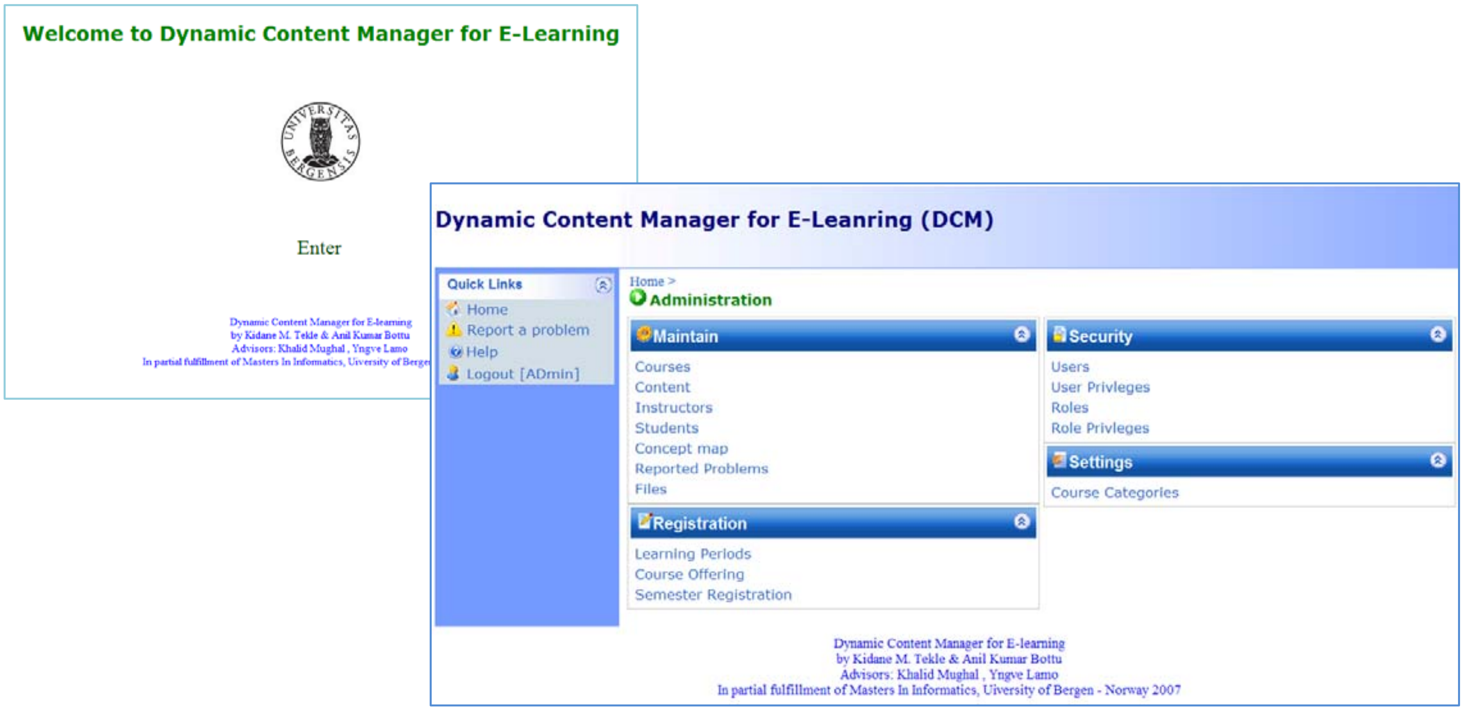
Dynamic Content Manager for E-Learning

By

Kidane M. Tekle

Supervisors: Yngve Lamo, Phd. & Khalid Azim Mughal, Phd.



Welcome to Dynamic Content Manager for E-Learning

Enter

Dynamic Content Manager for E-learning
by Kidane M. Tekle & Anil Kumar Bottu
Advisors: Khalid Mughal , Yngve Lamo
In partial fulfillment of Masters In Informatics, Uiversity of Berge

Dynamic Content Manager for E-Leanring (DCM)

Dynamic Content Manager for E-learning
by Kidane M. Tekle & Anil Kumar Bottu
Advisors: Khalid Mughal , Yngve Lamo
In partial fulfillment of Masters In Informatics, Uiversity of Bergen - Norway 2007

Department of Informatics

University of Bergen, November 2007

Preface

The DCM project is an e-learning system project done by Kidane M. Tekle(Me) and Anil Kumar Bottu in partial fulfillment for Masters in Informatics at the university of Bergen, Norway. The project was started in January 2006 and completed in November 2007.

This document gives a historical description of e-learning systems and explores some of the popular ones being used today. It states some shortcomings of most e-learning systems and describes in detail the pedagogical and technological points taken into consideration in the problem definition, analysis, design, development and pilot deployment of the DCM system. More emphasis is given to description of parts of the DCM project that I am responsible for while describing the overall solution.

I would like to extend my appreciation to my friend Anil Kumar Bottu for enduring with me and making the project so much fun. We are greatly indebted to our advisors Khalid Azim Mughal (Phd.), Yngve Lamo (Phd.) and Terje Kristensen (Phd.) for their guidance and support throughout the duration of the project. We are also equivalently grateful to Adrian Rutle (Phd. Candidate) for helping us put the DCM system to use. A special thanks goes to East Africa Systems (www.eastafricasystems.com) for allowing us use their database and user interface components. It wouldn't have been possible to make all the development of the DCM system in such a small duration of time if it was not based on their solid and proven framework of software development. My gratitude goes to all the members of St. Michael orthodox Tewahdo Church of Bergen for taking me in and giving me the love and support I needed and turned those two years into an enjoyment and a blessing.

I believe the DCM project has given me great understanding of the exciting field of E-Learning and provoked me to come up with ideas and solutions for the realization of the DCM system. I would like to consider this as a stepping stone of my future endeavors in the area of developing expert e-learning systems of the future.

I dedicate this Master’s thesis to my family who believed in me and …

Table of Contents

Table of Figures

Definitions and Acronyms

CBT – Computer Based Training

LMS – Learning Management System

DPG – Dynamic Page Generator

DCM – Dynamic Content Manager for E-Learning

UP – Unified process

XP – Extreme Programming

CLR – Common Language Runtime

IDE – Integrated Development Environment

CMS – Content Management System

DNN – Dot Net Nuke

1. Introduction

In the past few years, information technology has become very influential in our day to day activities. Many people cannot imagine living without the facilities of using computers and the internet. All major decisions of today are powered by statistical results and information that is the outcome of some computational processing.

The teaching-learning environment is one of the areas where information technology has showed its strong influence. Traditional lecture-based learning is not so attractive for students of today. This type of teaching may be more ineffective and usually creates more passive students in the learning process. Traditional classroom learning is mostly based on *behaviourism* learning theories where the learner is the object of assessment. The teacher initiates the learning process and the learner responds. Another learning approach, *constructivism*, focuses on the learner's abilities to develop her own mental models and learning concepts. By introducing web-based teaching systems one is able to create more constructive learning paradigms. The students will be more active in the learning process and more able to construct their own mental models of the learning objects, rather than doing only pure knowledge acquisition.

Electronic Learning (popularly called E-Learning) is a learning paradigm by which computer based learning methods and tools are put to use. It avoids time and space barriers in that content published by the educator can be available to the learner in practically no time. E-learning also gives new possibilities for making learning material available to the learner. It offers educators flexibility in their work and allows co-operation through content sharing and working together. Thus,an E-Learning system is a software system that is used by educators and learners as a means of courses administration, content delivery, progress follow-up, realization of educator learner interactions etc… Currently, there are many free and commercial e-learning systems in use. Each e-learning system has its own strengths and also its weaknesses based on its history, underlying philosophy, development methodology, usability and target audience.

1.1 Problem Description

E-learning systems have undergone tremendous evolution and have incorporated lots of functionalities. The two forces that dictate the changes in e-learning systems are research on the pedagogical understanding and modeling of e-learning and technological advances. These two factors do not always go side by side and it is not an easy task to find the correct mix on the two. Some e-learning systems are too much technology driven and incorporate functionalities that sometimes become responsible for their own downfall. Such systems lose the pedagogical grounds and may influence the teaching –learning process negatively. On the other hand, e-learning systems that focus too much on pedagogical structures are observed to be difficult to use since they lack the latest technological advances and ways of presentation that the user can find in most other places.

Most of today's e-learning systems are based on the constructionist pedagogical theory. As such, they focus on enhancing the interaction of the student and instructor via the availability of various resources and activities. The educator avails various resources to the learner during the running period of the courses she is giving. There is mostly little support for the educator while creating the learning materials and because most systems do not have an easy-to-use navigation mechanism of existing resources, much effort is duplicated. Another observed shortcoming is that learning content is closely tied to courses and that sharing and re-use of resources is restricted to file copying since there is no clearly defined model for it. The separation of learning content from specific courses and provision of visual navigation and arbitrary aggregation of the existing learning material helps the educator reuse existing resources and lets her focus on enhancing the quality of the learning material. Another shortcoming of most e-learning systems is that they are very much involved and do not provide flexibility enough for interfacing with other systems. This is an important feature where possibly more than one e-learning system is put to use and their parallel operation is desired.

1.2 **Justification and Motivations**

The Dynamic Content Manager (DCM) system is an e-learning system project that tries to define sound pedagogical modeling for removing the very tight coupling of learning material to specific courses. It defines a conceptual atomic unit of knowledge and builds up courses by organization of these knowledge units from the repository. This gives the ability to create knowledge elements at a finer granularity level which can be re-used across various courses. Resources like lecture notes, presentations, attachment files, questions etc… are attributed to the knowledge elements and hence can be imported while using existing knowledge elements.

Another motivation for the DCM project is the need to create an e-learning system that utilizes the technological advances made in the area of software development. Usage of proven design and development methodologies enables easy creation of a system that is highly customizable and expandable. Such a system could be very useful in addressing major interfacing issues that are displayed when more than one e-learning system is put to use in a learning environment. Creating such a highly customizable, extensible and light weight system can be a foundation on which more advanced and highly intelligent e-learning systems can be built.

1.3 Construction of Thesis

This document is organized into six chapters. Chapter 1 has given a general introduction, justification and modification for the DCM project. Chapters two begins by giving a historical background to E-Learning systems and describes some of the most widely used ones. The third chapter states observed shortcoming of existing systems and raises various points that have been given consideration in the development of the DCM system. Following, in chapter four, available technology, software development methodology and design issues used in the development of the system are described. Chapter five gives a detailed description of the DCM system solution to be followed by the last chapter that summarizes the project, describes the current status of the system and points out directions for further enhancement and extended development.

2. E-Learning Systems

2.1 Historical Background

The first modern learning technology emerged during World War II when the United States used film to train millions of service people around the world [1]. These military training films covered such topics as personal hygiene and weapons maintenance. The success of these films, and their later use through television, led the military to partner with universities to conduct research into modern learning techniques.

In the 1960s, the first types of teaching machines were developed, while instructional film became more creative and broadened its reach to children in schools.

Then came television as a new learning delivery method. But, because the expense was too great and the delivery of the information too dry, there were only a few successes. What evolved from this were videotapes, which were produced for use in corporate and school classrooms. Since then, e-learning systems have evolved dramatically to incorporate rich learning material delivery to learners with multimedia content and providing a virtualized class room environment.

Currently, there are many E-Learnings systems that have gained popularity. Some of these are : FirstClass, WebCT, Desire2learn, BlackBoard and Moodle. Out of these BlackBoard and Moodle are by far the most used systems. Blackboard is an LMS system that is available commercially and a yearly license fee is paid to use it. On the other hand, Moodle is an open source product distributed under the terms of the GNU General Public License.

Beginning from the early nineties, some research and development has been undertaken in the field of e-developing E-Learning systems in the Bergen region or Norway. About ten years ago web-based learning systems were constructed and used that had great impact on the development of e-learning systems of today. The “Gudmundstad” project [2,3] was started in 1994 and was a quite successful e-learning project for its time. It showed a learning system that started from fairly local initiatives and expanded to regional, national and even to an international project.

In 1997 the “Gudmundstad” project was followed up with another e-learning system, “Reidar”, developed on the Windows platform. The “Reidar“ system was very much used in both the curriculum and in distance education at Bergen University College. “It's learning” is another e-learning system that has been developed in Bergen, Norway [4]. The origin of this project system was a student project at the Bergen University College in 1999.

The Dynamic Presentation Generator (DPG) system project, was a project that was initiated in 2001, and aims at decoupling the storage and manipulation of learning material (content) from its presentation (learning structure)

Following, the Moodle learning management system, the “It’s Learning” system and the DPG system project are described.

2.2 **Moodle [5]**

The word ***Moodle*** was originally an acronym for ***M***odular ***O***bject-***O***riented ***D***ynamic ***L***earning ***E***nvironment. Moodle is by far the most widely used E-Learning system of today. Currently there are 22,387 registered Moodle sites from 172 countries of the world.

2.2.1 **Philosophy*[5]***

The design and development of Moodle is guided by a particular philosophy of learning, a way of thinking that is referred to as "*social constructionist pedagogy*". The four main concepts behind this philosophy are:

1. Constructivism - This point of view maintains that people actively **construct** new knowledge as they interact with their environment.
2. Constructionism - asserts that learning is particularly effective when constructing something for others to experience. This can be anything from a spoken sentence or an internet posting, to more complex artifacts like a painting, a house or a software package.
3. Social Constructivism - This extends the above ideas into a social group constructing things for one another, collaboratively creating a small culture of shared artifacts with shared meanings.
4. Connected and Separate - This idea looks deeper into the motivations of individuals within a discussion. Separate behavior is when someone tries to remain 'objective' and 'factual', and tends to defend their own ideas using logic to find holes in their opponent's ideas. Connected behavior is a more empathic approach that accepts subjectivity, trying to listen and ask questions in an effort to understand the other point of view. Constructed behavior is when a person is sensitive to both of these approaches

2.2.2 **Overall design**

The overall design of the Moodle is very simple, lightweight, efficient, compatible, low-tech browser interface. Its easiness makes the users to install on almost any platform that supports PHP. The database abstraction supports all major brands of database. One model site can support thousands of courses – courses can be categorized and searched and course listing shows description for every course on the server, including accessibility to guests. It has a very strong security throughout the system.

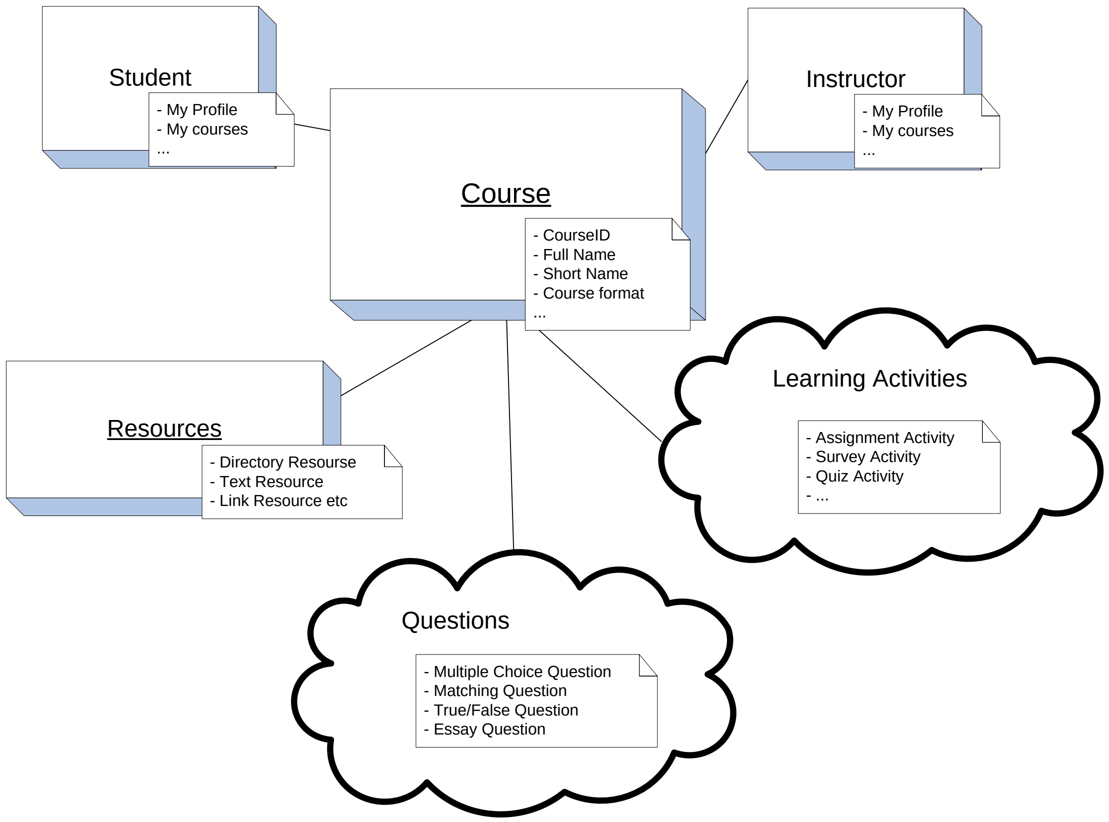


Figure 1 – a simplified model of the Moodle E-Learning system

2.2.3 **Site Management** *[5]*

A Moodle site is managed by an admin user, defined during setup. Plug-in "themes" allow the admin to customize the site colors, fonts, layout etc to suit local needs. Plug-in activity modules can be added to existing Moodle installations. Plug-in language packs allow full localization to any language. These can be edited using a built-in web-based editor.

2.2.4 **User Management** *[5]*

One of the design goals of Moodle is to reduce admin involvement to a minimum, while retaining high security. It supports a range of authentication mechanisms through plug-in authentication modules, allowing easy integration with existing systems.

An admin account controls the creation of courses and creates teachers by assigning users to courses. A course creator account is only allowed to create courses and teach in them. Teachers may have editing privileges removed so that they can't modify the course. Teachers can add an "enrolment key" to their courses to keep out non-students. They can give out this key face-to-face or via personal email etc and can enroll and un enroll students manually if desired.

2.2.5 **Course Management** *[5]*

Moodle incorporates easy to use facilities to manage the various courses to be offered. A course can be defined by selecting from the various available formats formats such as by week, by topic or a discussion-focused social format. Courses can be packaged as a single zip file using the Backup function. These can be restored on any Moodle server

2.2.6 **Learning Activities in Moodle** *[5]*

In Moodle learning activities are used to realize various interactions between students and instructors. Moodle has an extensible interface for adding more activity modules. Some of the commonly used activity modules of Moodle are:

- Assignment module – implements the interaction of giving assignments and giving feedback

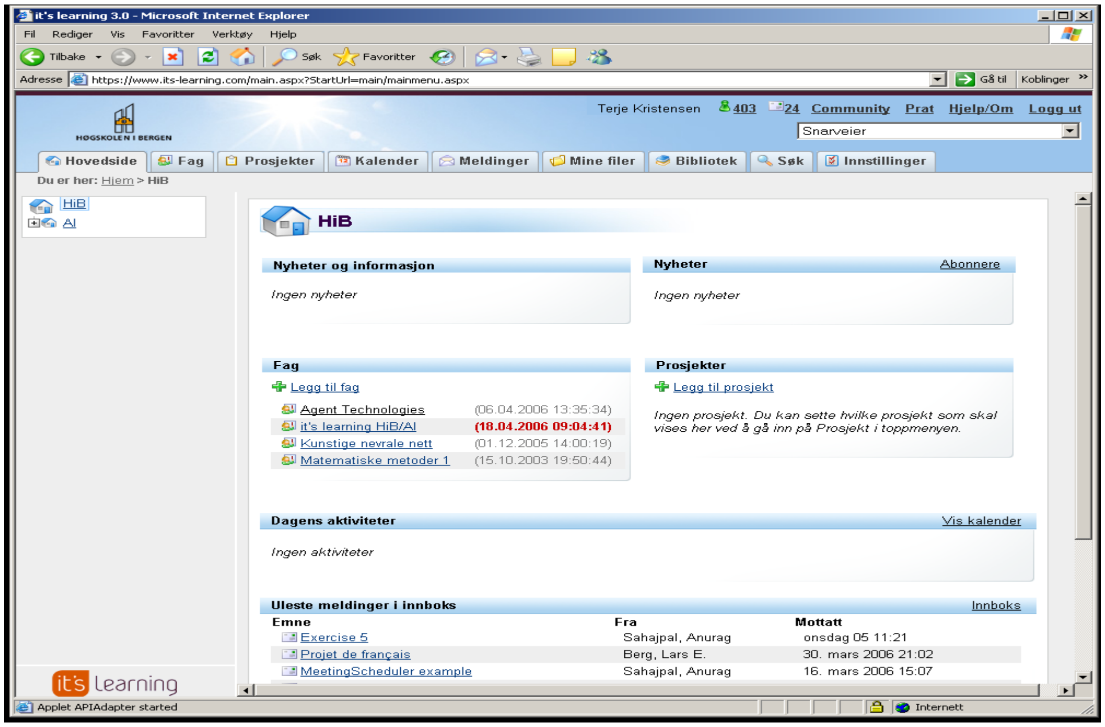
- Quiz module - Teachers can define a database of questions for re-use in different quizzes. Questions are stored in categories for easy access, and these categories can be "published" to make them accessible from any course on the site. The quizzes can be:

    - Multiple-choice questions supporting single or multiple answers Short
    - Answer questions (words or phrases)
    - True-False questions
    - Matching questions
    - Random questions
    - Numerical questions (with allowable ranges)
    - Embedded-answer questions (cloze style) with answers within passages of text
    - Embedded descriptive text and graphics

2.3 **It's Learning**

"It's learning" is an e-learning system that has been developed in Bergen, Norway [4]. It has had a great success in the Scandinavia market, with more than 450.000 users. The "It's learning" platform is designed for schools and universities. The origin of the "it's learning" system was a student project at the Bergen University College in 1999. "It's learning" is a tool for supporting and enhancing different learning activities, new teaching methods and also providing easy access to knowledge. The system uses a fixed learning platform.

"It's learning" has a variety of built-in tools for communication and cooperation such as internal messaging system, e-mail, chat, SMS notifications, discussion forums, etc.. This offers a lot of possibilities for the instructor of a course. However, on the other hand much of the tools are not necessary to use in a course and by ordinary users. The tools may appear as noise that distracts the user in a given learning situation. One problem is that the system gives the user too many possibilities. An ordinary user does not need all these options. One other problem is that the graphical layout and navigation are not consistent. This makes it difficult for the users to have a global overview and control of the learning objects.
"It's learning" is also a tool for course administrators and course leaders. The system provides a range of automatically generated reports that provide an overview of a group and individuals progress within the learning cycle. One problem is that the reports do not have a consistent design.



https://www.its-learning.com/main.aspx?StartUrl=main/mainmenu.aspx

HiB

Nyheter og informasjon

Ingen nyheter

Nyheter Abonnere

Ingen nyheter

Fag

Legg til fag

Agent Technologies	(06.04.2006 13:35:34)
it's learning HiB/AI	**(18.04.2006 09:04:41)**
Kunstige nevrale nett	(01.12.2005 14:00:19)
Matematiske metoder 1	(15.10.2003 19:50:44)

Prosjekter

Legg til prosjekt

Ingen prosjekt. Du kan sette hvilke prosjekt som skal vises her ved å gå inn på Prosjekt i toppmenyen.

Dagens aktiviteter Vis kalender

Ingen aktiviteter

Uleste meldinger i innboks Innboks

Emne	Fra	Mottatt
Exercise 5	Sahajpal, Anurag	onsdag 05 11:21
Projet de français	Berg, Lars E.	30. mars 2006 21:02
MeetingScheduler example	Sahajpal, Anurag	16. mars 2006 15:07

Figure 2 – It's learning system user interface

2.4 **Dynamic Presentation Generator (DPG)**

The Dynamic Content Manger (DPG) system is a project with an approach to generate online courses from presentation patterns. In order to simplify setting up new on-line courses one wants solutions that do not require particular programming skills. The main objective of the teacher is to develop and presents good learning material.

The structure of the learning content of the Dynamic Presentation Generator (DPG) system is specified in XML, and its visualization is dictated by a course pattern. The teacher needs only supply the contents of the learning material in order to create an online course. The system takes care of the rest; dynamically generating the web pages for the course and making them accessible to the users

2.4.1 **The work flow model**

The workflow model of the DPG system is shown in Figure 3. It comprises two phases:

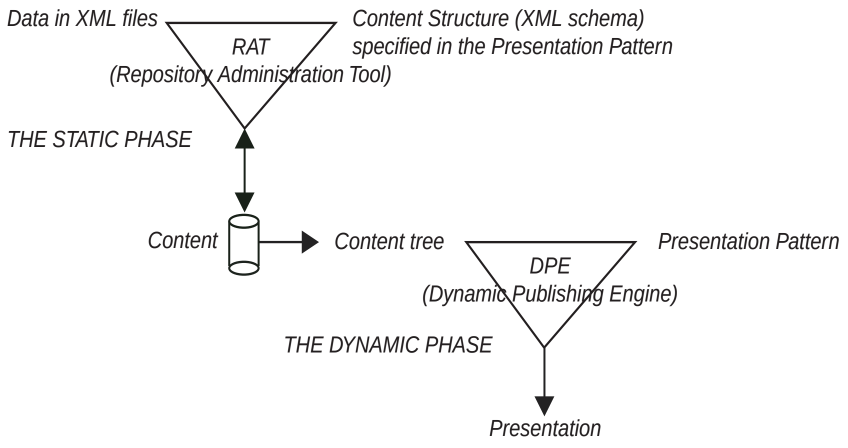

Data in XML files
Content Structure (XML schema) specified in the Presentation Pattern
RAT
(Repository Administration Tool)
THE STATIC PHASE
Content
Content tree
Presentation Pattern
DPE
(Dynamic Publishing Engine)
THE DYNAMIC PHASE
Presentation

Figure 3 – Workflow model of the DPG

• The static phase, and

• The dynamic phase

In the static phase, the Repository Administration Tool (RAT) validates the data in XML files against the content structure specified in the presentation pattern. Only validated data is stored in a XML database. RAT is also used to retrieve the data from the database for updating purposes.

The core of the dynamic phase is the Dynamic Publishing Engine (DPE). Given the content tree and the corresponding presentation pattern, the DPE renders the web pages that comprise the presentation. The DPE dynamically generates a web page in response to a browser request. The content tree is created from the content in the XML database at the start of the web application. Data for a browser request is retrieved from the content tree. Formatting of a web page is done according to the presentation pattern specification. The most obvious advantage of this workflow model is that different content and presentation patterns can be mixed and matched to create different presentations, as long as the content conforms to the presentation pattern.

2.4.2 The DPG system in use

The DPG system has been used since 2003 to create online Java programming courses at the Department of Informatics, University of Bergen and since 2004 at Bergen University College in a regular course in programming technology. The DPG system and the presentation specifications have now gone through a number of iterations and have provided the proof-of-concept for presentation patterns, as well as hands-on experience from running and maintaining the online courses.

The experience has shown that there are several advantages of using presentation patterns to create online courses. For instance, an initial investment in defining a suitable navigation structure and visually appealing layout can be capitalized on in later courses, as these aspects of a presentation are captured in the presentation pattern. From a course administration point of view, no programming experience is needed to prepare and update the content, and web-based tools are available for content generation and maintenance. In terms of cost and effort, the threshold to deploy this system is low compared to other such systems.

The current system is implemented by using Java and Tomcat and is available for installation on nearly all platforms. The RAT facility has a web-based GUI that allows uploading of initial content from XML files and its storage as a set of resources in the XML database. The tool also incorporates a general-purpose editor for content inspection and modification. For convenience, the database files and any associated resources (for example, images) are stored as part of the DPE web application.

If the administration tool modifies the content, the publishing engine automatically updates the presentation. One high-priority task is to create new presentation patterns. Typical examples of new patterns would be for slide shows, for interactive presentation of a lecture or for "webifying" articles and books. The main challenge will be achieving this goal through reuse of web-based presentation components.

3. Problem Analysis

3.1 Shortcomings of Existing Systems

Most of today's e-learning systems are based on the constructionist pedagogical theory. As such, they focus on enhancing the interaction of the student and instructor via the availability of various resources and activities. The educator avails various resources to the learner during the running period of the courses she is giving. There is mostly little support for the educator while creating the learning materials and because most systems do not have an easy-to-use navigation mechanism of existing resources, much effort is duplicated.

Most of the commercial e-learning systems of today have a lot of facilities, but the problem is that they are missing a solid underlying pedagogical structure. This means that the major challenge for e-learning systems is to develop a pedagogical structure of the system – not to develop further its technical functionality. Most existing E-Learning systems contain too much functionality, but not the possibilities to make presentations suited to specific end-users. All these functionalities may disturb the actual learning situation, both from a user and a course administration perspective. Systems like "It's learning", have evolved very much technologically, but the short comings are still there because the underlying pedagogical structure of these systems is not well structured. Another shortcoming that is common to most e-learning systems is that of non-uniformity in the quality and ease of use of the different modules.

Much of the effort duplication observed in today's systems could be avoided or kept to a minimum by defining of atomic small knowledge elements and building courses by assembling these elements in a structured manner. This results in a decoupling of content from course so that the same content material can be used by more than one course, in possibly more than one version. These knowledge elements could be organized according to the domain knowledge and desire of instructors to make various aggregations of inter-connected navigational entities giving a mesh that can be better described by use of a concept map. Furthermore, various course resources, questions and quizzes could be organized better by attributing them to the atomic knowledge elements.

Student modeling is a concept that is an active field of research. The level of modeling of the student in an E-Learning system shows its maturity and suitability for logical reasoning assertions. With a good conceptual student model, mathematical model can be defined for the learning and knowledge acquisition of the student in a manner than can be proven using theorem proving algorithms.

Following, a description of the DCM system and the requirements it tries to address will be described.

3.2 Decoupling of Content and Course

The major shortcoming of most existing E-Learning systems is that learning material is very tightly coupled to courses. This is present inherently since educators create courses that are availed to the students via the E-Learning system. Thus, the scope of the learning material developed will be limited to the defined course or to passing around of some shared files at the maximum.

The DCM defines an atomic knowledge element as a basic building block of all learning material. A course is defined as a hierarchical organization these knowledge elements. Such a model introduces an inherent decoupling between courses and learning content.

The instructor still defines learning material in more or less the same manner as most E-Learning systems but the DCM stores them as knowledge elements that can later be queried for re-use. The decoupling of course and content gives the possibility to develop a good domain knowledge with scoping of more than a single course.

3.3 Content reuse

Content reuse refers to any situation where a single piece of source content is written once, and then used in multiple locations or contexts [6]. There is not, however, a consistent understanding of what content reuse means in practice, and the term is used to mean many things, each of which may be met by different technology solutions

3.3.1 Types of content reuse

In theory, there are various scenarios where content reuse is applied. Some are:

Content is linked to from multiple locations – this is the simplest scenario where a single page is linked from more than one location

Content appears in multiple locations – in this scenario, a page appears in multiple locations within a site

Usage of standard elements – by this mechanism of content reuse, standard elements like headers, footers or disclaimers are shared among different content pages

Assembly from ‘components’ – this is the most complex scenario for content reuse. Content pages are assembled from a pool of content components from some repository. This model of content reuse is implemented in the case of the DCM with knowledge elements defining the atomic reusable content components

3.3.2 Benefits of content reuse

There are various benefits to content reuse. Some are:

Improved accuracy and consistency – content is written mostly once or a minimal number of times. This avoids the case of multiple copies being edited in a non consistent manner

Increased efficiency – using content material only once and reusing it in multiple locations reduces authoring effort required. With a good content reuse in place, lots of hours of authoring can be saved

Greater control – with the proper content reuse in place, the content material will be created, updated and generally managed in a predefined way such that there is more control over the content material.

3.3.3 Challenges of content reuse

In practice, content reuse is quite difficult to implement. Some of the challenges of content reuse are:

Increased complexity for authors – introduction of a content reuse management system introduces more constraints on authors/educators on their ways of creation of content. There is a learning curve before the system can be used with relative ease and in most cases, some people may even revert to other solutions unless they are able to understand and appreciate the advantages of content reuse.

Content versioning – this is also a challenge introduced in cases where content material is allowed to be edited by multiple authors to be used on different contexts. The content reuse management system should employ appropriate measures of content versioning and also authors need to be aware of the way the versioning scenario works

4. Available Technology and Methodologies

Often developing software is compared to building houses and bridges. Careful planning and understanding of what is actually needed is the starting point to the building of a house. The person needs to be clear on what she needs, like number of rooms, capacity, color and the like to some extent as a starting point. Then an architect or a designer is hired to put those requirements into a drawing that can be viewed and analyzed. After going through some enhancements and incorporating new ideas, the plan for the construction is laid out and then constructed. Development of software follows more or less the same steps as those needed in the construction of the house.

Any software development project is aimed at achieving some goal objective. The beginning point is always some domain problem the software is expected to model, assist or facilitate. The requirements of the domain should be analyzed at first and continuously revised during the development process for changes or enhancement. Different software development processes define different stages to be followed and address the requirements understanding, system modeling and system construction stages. Hence, the implementation of a software system is dictated by the development methodology and to some extent by the technology employed.

In this chapter, we will first consider popular software development practices will be stated with their relative advantages and shortcomings. Then, some of the popular development technologies will be described with justification as to the specific one that was used in the DCM project.

4.1 Object Oriented Software development

Object oriented software development is a software methodology that has gained great success and has been put to great use. [11], [12], [13], [14] This model follows the object oriented approach of modeling in that it focuses on identifying objects and their interaction to model the domain and the software system.

There are three stages to be followed when using the object oriented software development methodology. These are:

Requirements Elicitation

Requirements elicitation is mostly the first phase of the development process. The major task that is tried to be achieved is that of requirements understanding and modeling in an object oriented manner. During the requirements elicitation stage meetings are held, presentations conducted and various consultations followed so as to have a clear understanding of the requirements of the problem domain.

The requirement elicitation phase uses three types of objects to model the requirements of the system. The first type of objects is called “Entity Objects” which represent the data that is persistent. As an example, considering a requirement of registration of a student for a course, the details of students like name, age, and sex go under the category of entity objects. The second category of objects identified during the requirement elicitation stage is “Boundary Objects”. These objects represent the interface between the user and the software system developed. Considering our previous student registration example the web page that the student must open, the buttons that she must use to submit her desired operation requests fall under the boundary objects category. Clear definition and understanding of the boundary objects is crucial for the usability of the system and for achieving better user experience. The third type of objects that the requirement elicitation stage identifies is “Control Objects”. These objects define the constraints and business rules that have to be fulfilled by the system. Considering the student registration example, boundary objects should be defined to realize the constraints that the same student is not registered multiple times and other registration related requirements are met.

Requirement analysis (Elaboration)

After identifying the entity, boundary and control objects in the requirement elicitation stage, the requirements of the system should be further analyzed and elaborated. The requirements analysis phases focuses on further investigating the requirements of the system and developing detailed description and documentation. Standard documentation methodologies are employed in the form of activity diagrams and sequence diagrams. The outcome of this phase of the development process is clearer understanding and documentation of the requirements.

System design

In the Object oriented software development paradigm, the system design process stage defines object interactions, collaborations and prepares the grounds for the development. The system design describes the details of the software to be developed by use of use case diagrams, class diagrams, collaboration diagrams and deployment diagrams.

System development

The system development stage is the stage where the design is programmed to produce working software that meets its requirements. System development comprises of coding and testing at various levels of detail.

4.2 **Popular software development processes**

4.2.1 **Unified Process (UP)**

The Unified Process is an extensible framework which describes the various activities that must be followed in the development of software. It defines key points that are characteristic and that should be kept in mind throughout the software development process. The UP defines a concept called a "project lifecycle" that defines the stages that have to be followed. The UP is a highly generalized framework and hence, each software development institution or group should define its own adapted model as per the points of emphasis and domain model.

Project Life cycle

The UP defines the life cycle of a project (or iteration) as having four stages. These are the inception phase, elaboration phase, construction phase and transition phases. Following, a brief description of each stage and its output will be described.

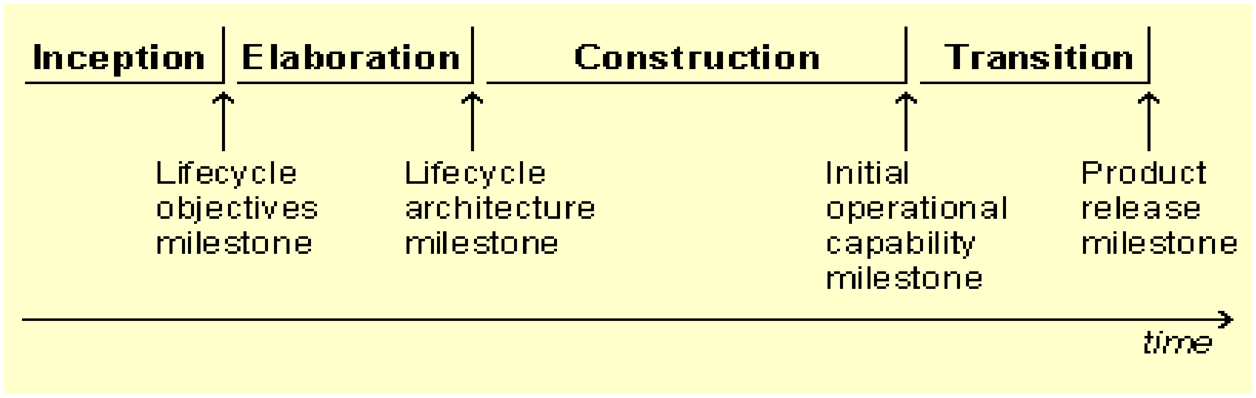

Inception
Elaboration
Construction
Transition
Lifecycle objectives milestone
Lifecycle architecture milestone
Initial operational capability milestone
Product release milestone
time

Figure 4 – software life cycle

Inception phase – this phase analyzes the risks and scope of the project. The necessary assumptions are made on the project (iteration). The inception phase is aimed at achieving concurrence among all stakeholders of the project on the lifecycle objectives. The output of this phase is a document called "Inception Report".

Elaboration phase – the elaboration phase starts by reviewing the output of the Inception phase and performs detailed analysis of requirements. Business requirements, conditions and constraints are investigated to depth and documented. The output of this phase is a document called "Requirement Analysis Document"

Construction phase – this phase is where the analyzed system is put into implementation. At the early stage of the construction phase, the requirement analysis document is reviewed and the overall design and architecture of the system is made. This result in a document called "Design Document". Then the software system is developed and subsidiary documents like system documentation and training manuals are produced during the construction phase.

Transition phase – the transition phase is responsible for porting of the developed software system from development platform to actual operational platform. Tests are done on the system by end users and the software is modified to reflect the outcome of the tests. During this phase, end users are given training and produced software is put to use

The Unified Process defines some qualitative characteristics that should be followed during all the phases of the development lifecycle. Some of the key characteristics of a UP process driven software development are:

Iterative and Incremental

This characteristics of UP defines that development should be performed in small volumes and that the various stages of the development cycle should be iterated. This is very helpful in that at each release, some working testing functional portion of the system is released it is as an increment to the previous developments.

Use case driven

A use case can be defined as a formal description and generalization of a specific requirement. Hence, each use case of a system captures some functional requirement of the system. By use case driven development the UP dictates that each iteration and each stage of the development process should be focused on identifying, elaborating, constructing and testing of use cases.

Architecture centric

The Unified Process insists that architecture sit at the heart of the project team's efforts to shape the system. Since no single model is sufficient to cover all aspects of a system, the Unified Process supports multiple architectural models and views. One of the most important deliverables of the process is the executable architecture baseline which is created during the Elaboration phase. This partial implementation of the system serves to validate the architecture and act as a foundation for remaining development.

The Unified process is a heavy weight process and as such it is most suited for big projects that involve many requirements and developers. Recently, other light weight software development processes have come into the picture and have gained quite an audience. Following, we consider a software development process called Xtreme Programming.

4.2.2 Agile software process (Xtreme Programming -XP)

In software development, the cost of change has an exponentially rising behavior. One reason that accounts for this behavior is the usage of heavy weight software development processes. The rigid and time consuming procedures of doing activities and the need to keep the body of related secondary documents accounts for most of the cost of making changes.

As a response to the increasing weight of processes, a group of software experts met in 2001 to discuss key principles of agile development. They came up with the following manifesto that says "*We are uncovering better ways of developing software by doing it and helping others do it*" [15].

Xtreme programming (XP) is one of the most popular Agile software development processes. It is a very light weight process that gives more emphasis to customer satisfaction than to producing documents and rigid methods. Some of the points that are given great values in the XP process are:

Communication – there should be very open and high level of communication between the project teams. This includes a representative of the customer that must actively participate in the overall process.

Simplicity – keep things simple. The main objective of software development is to produce working software and not documents. Also, the implementation should be kept to the bare minimum so that if some functionality is not needed at the moment, then it is not implemented. This helps keep the coding simple and avoids unnecessary complications.

Feedback – customer involvement is a key practice of the XP software development model. The customer's representative(s) are considered as part of the development team and should provide extensive and timely feedback.

Although extreme programming and agile methods have gained popularity over the past few years, they are not a replacement for the Unified Process. XP is better suited for projects of small size and where is not much difference in the knowledge and capability level of project teams. Although the Unified Process is a more rigid and some most of the times over bureaucratic, it gives a more deterministic process model and is better suited for software development projects of critical nature.

4.3 **Development Technologies**

The development technology chosen on a project greatly influences the flexibility, ease of development and in some cases ease of use by the end user as well. Thus, choice of development technology is one of the major factors that play an important role in the overall software development process. Some of the points to consider when choosing development technologies are:

Inbuilt functionality – most development frameworks and programming languages have inbuilt support for most of the routine operations in programming. But the ease of usage and extent of specialization of functionalities greatly varies from one framework to another. Primarily, the problem domain and need for supported functionality plays a big role in choosing a development environment.

Graphic support – ease of use in creating graphical user interfaces is one major quality to look for while choosing development technologies. The developers should spend most of their time on realizing the business logic of the system and making the system come to life. As such, a development environment that provides graphical support for creating user interfaces greatly reduces development effort and is preferred over other ones.

Support for component orientation – component oriented software development is a means of developing software as composition of different parts which could be developed by the same team or another. Effortless integration of third party of project developed components greatly reduces development time and also allows multiple users to work on parts of the project with seamless integration.

License issues – development and deployment software license is also one major factor to be considered when comparing choice of development technology. Decision on whether to use open source license products or proprietary technology has implications both on price and usability of the software system to be developed.

Development support tools – software development environments that give good support on runtime compilation, debugging and tracing greatly increase the productivity of the development team. Hence, the supporting tool an environment provides plays an important factor during selection of development technology.

Development team experience – another major factor to be taken into consideration when choosing a development framework is the level of experience and expertise of the development team. Most companies have a predefined development environment and work towards making their developers achieve higher efficiency by way of attending conferences and trainings.

Taking the above mentioned points into consideration, the following development technology was chosen for the implementation of the DCM project.

Development environment – Microsoft Visual Studio 2005

Programming language - asp.net for user front end and C# as core component development language and code behind logic for asp.net pages.

Database management system – Microsoft SQLExpress

Following, a description of some of the functionalities and tools of the visual studio 2005 development environment and the dot net framework will be described.

4.3.1 Dot Net framework and Visual Studio 2005

Dot Net Framework

The Dot Net framework is a platform for developing applications on the windows platform. It was first released in 2002 with the visual studio 2002. It provides a large body of pre-coded solutions to common program requirements, and manages the execution of programs written specifically for the framework. The dot net framework provides unified solution for developing windows applications, web applications and also XML web services. The Dot Net framework has two parts: the common langue runtime (CLR) and the dot net class library:

Common Language Runtime (CLR) – the CLR is part of the dot net framework that is functionally equivalent to the JVM (Java virtual machine) in a java environment. It performs the operations of: management of running code, verification of type safety, providing garbage collection and error handling, security and a unified type system. The CLR provides access to system resources through native API, COM interop and other facilities.

Dot net class library – this is a collection of unified classes that provide easy to use ways of access for the developer.

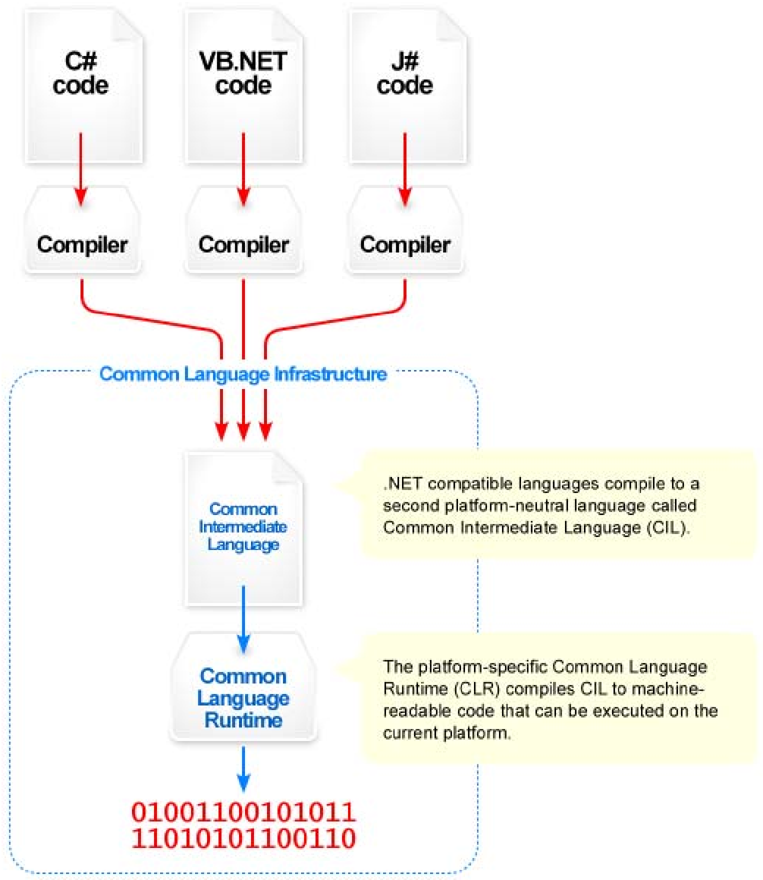

C# code
VB.NET code
J# code
Compiler
Compiler
Compiler
Common Language Infrastructure
Common Intermediate Language
.NET compatible languages compile to a second platform-neutral language called Common Intermediate Language (CIL).
Common Language Runtime
The platform-specific Common Language Runtime (CLR) compiles CIL to machine-readable code that can be executed on the current platform.
01001100101011
11010101100110

Figure 5 – The Dot Net framework

The Dot Net Programming Model

The dot net programming model is a multi language, same IDE (Integrated Development Environment) programming model with cross language inheritance and exception handling. It has tools that work in multiple programming languages and platforms. It introduces a zero impact installation model and also side by side execution of different versions of the same assembly. The dot net programming model provides seamless integration with the earlier technology of COM and gives easy access to system functions. It follows a unified type system where everything is an object and objects created in different languages can be cross inherited.

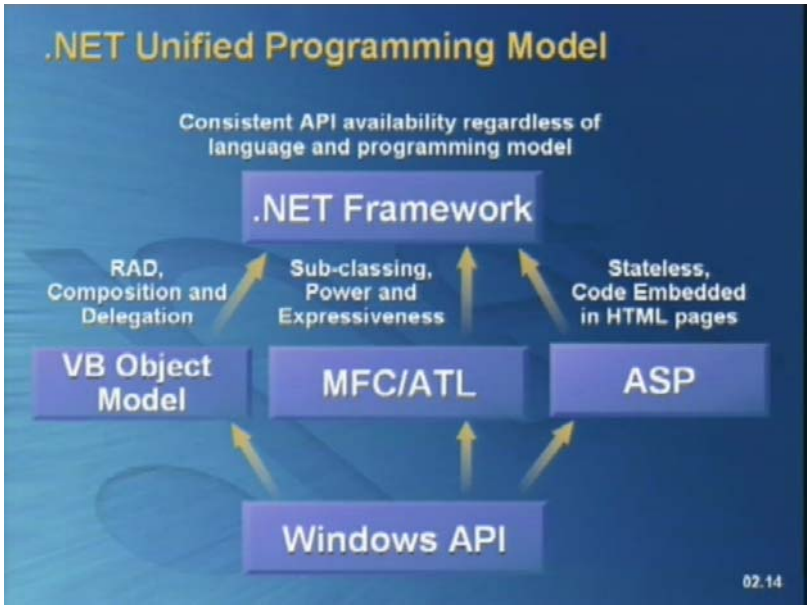

.NET Unified Programming Model
Consistent API availability regardless of language and programming model
.NET Framework
RAD, Composition and Delegation
Sub-classing, Power and Expressiveness
Stateless, Code Embedded in HTML pages
VB Object Model
MFC/ATL
ASP
Windows API
02.14

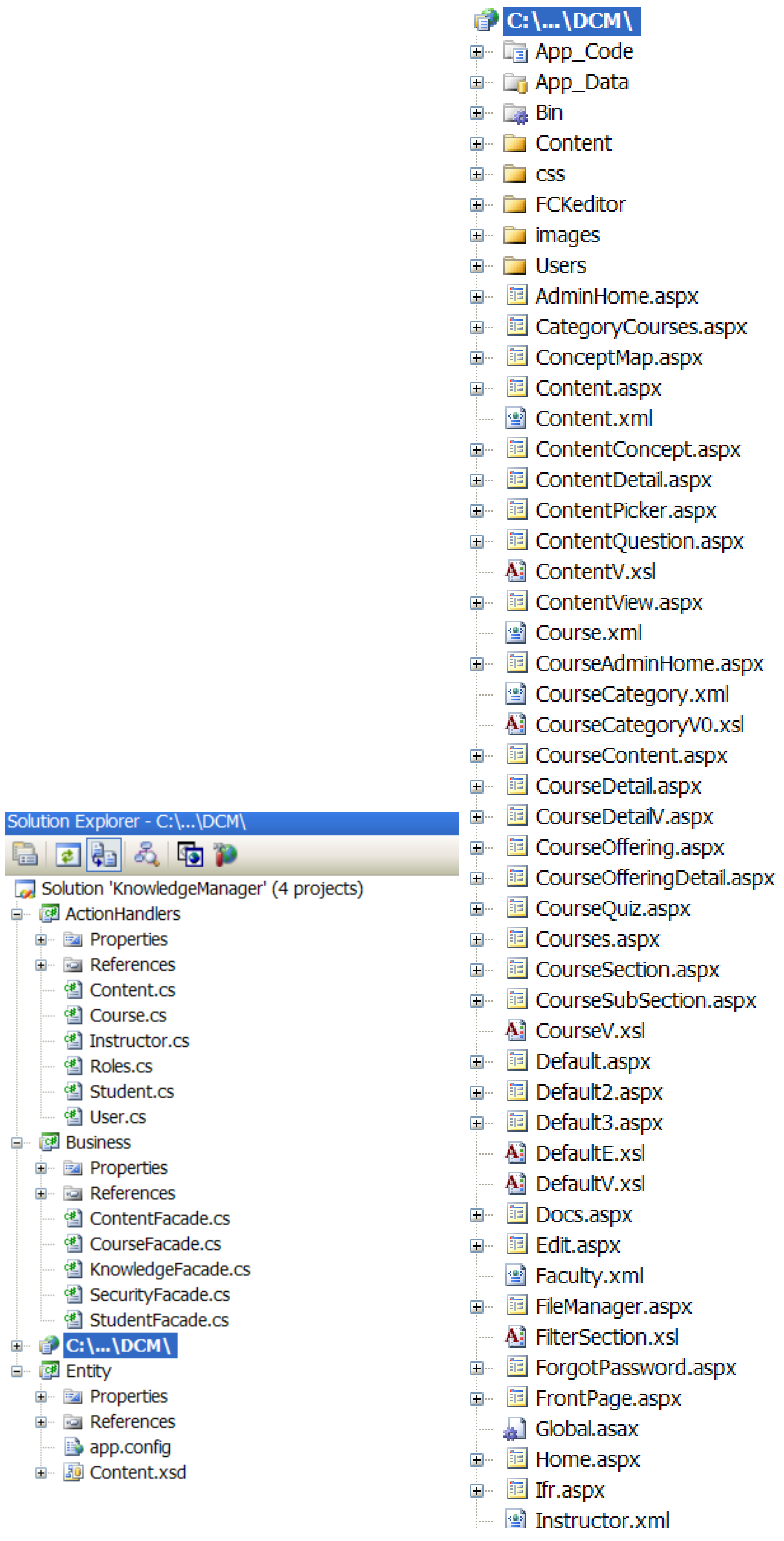
Figure 6 – dot net unified programming model

Visual studio 2005

Visual Studio 2005 was released online in October 2005 and. It is the current latest release of development environment in the .net line which runs the dot net framework 2.0. Visual studio 2002 was the first one in this product line containing version 1.0 of the dot net framework. Visual studio 2003, containing the dot net framework 1.1 introduced many changes as the framework matured.

The highest level of organization in visual studio 2005 is given the name "solution". A single solution can contain one or more projects of similar or different types. A project is a set of classes and files organized and outputs assemblies of different types depending on the project type. Some of the project types supported by visual studio 2005 are: class libraries, windows console applications, windows forms application, web applications, windows service application, deployment projects etc…

The component orientation inbuilt in the dot net framework provides the facility to distribute the system functionalities into different projects and also to use external assemblies. Following, a screen shot of the DCM solution is shown with its different projects:



```
Solution Explorer - C:\...\DCM\
Solution 'KnowledgeManager' (4 projects)
  ActionHandlers
    Properties
    References
    Content.cs
    Course.cs
    Instructor.cs
    Roles.cs
    Student.cs
    User.cs
  Business
    Properties
    References
    ContentFacade.cs
    CourseFacade.cs
    KnowledgeFacade.cs
    SecurityFacade.cs
    StudentFacade.cs
  C:\...\DCM\
  Entity
    Properties
    References
    app.config
    Content.xsd
```

```
C:\...\DCM\
  App_Code
  App_Data
  Bin
  Content
  css
  FCKeditor
  images
  Users
  AdminHome.aspx
  CategoryCourses.aspx
  ConceptMap.aspx
  Content.aspx
  Content.xml
  ContentConcept.aspx
  ContentDetail.aspx
  ContentPicker.aspx
  ContentQuestion.aspx
  ContentV.xsl
  ContentView.aspx
  Course.xml
  CourseAdminHome.aspx
  CourseCategory.xml
  CourseCategoryV0.xsl
  CourseContent.aspx
  CourseDetail.aspx
  CourseDetailV.aspx
  CourseOffering.aspx
  CourseOfferingDetail.aspx
  CourseQuiz.aspx
  Courses.aspx
  CourseSection.aspx
  CourseSubSection.aspx
  CourseV.xsl
  Default.aspx
  Default2.aspx
  Default3.aspx
  DefaultE.xsl
  DefaultV.xsl
  Docs.aspx
  Edit.aspx
  Faculty.xml
  FileManager.aspx
  FilterSection.xsl
  ForgotPassword.aspx
  FrontPage.aspx
  Global.asax
  Home.aspx
  Ifr.aspx
  Instructor.xml
```

Figure 7 – the DCM project in Visual studio 2005

5. DCM solution

5.1 Overview

The DCM (Dynamic Content Manager for E-Learning) is an e-learning solution that tries to remove the very tight coupling of learning material to specific courses. It defines a conceptual atomic unit of knowledge (given the name *Content*) and builds up courses by organization of these knowledge units from the repository. This gives the ability to create knowledge elements at a finer granularity level which can be re-used across various courses. Resources like lecture notes, presentations, attachment files, questions etc… are attributed to the knowledge elements and hence can be imported while using existing knowledge elements.

The DCM is designed with the functionality of knowledge elements, courses and resources versioning and history tracking so that changes made to the underlying knowledge elements is carefully tracked. This ensures that a specific course or some aggregation of the knowledge elements the educator has created, can appears unchanged as far as she is concerned while giving her a chance to follow the various revisions made on it.

The concept of dynamic presentation patterns tested in the DPG system is put to use in the DCM project. This gives the system the feature of being highly customizable in that the presentation of courses, resources and navigation can all be enhanced according to the configuration set by the site administrators. The instructor could, in future versions, be given the freedom to define the appearance of the courses she is giving and can use already used presentation patterns as well as define her own.

At the beginning of the DCM project, survey of various E-Learning systems was done. The survey was crucial in capturing the general operational functionality of E-Learning systems and in defining the direction of pursuit for the DCM project. After defining a domain model, an overall design of the system, subsystem decomposition and a decision on the development technology to be used were made.

Considering the complexity of the requirements of the DCM system and the short duration of the project development period, it was decided that the project be developed using a development environment giving good support on user interface and on project organization and componentization. The DCM project is implemented using the Dot Net Framework 2.0, Visual Studio 2005 using Asp.Net technologies and the C# programming language. It is optimized to work with Microsoft SQL Server 2000/2005 but its database abstraction layer also enables it to run on most of the major popular database providers. Extensibility is given great emphasis in the design of the system and hence provisions are made for the seamless addition of more external functionalities by possibly different developers

Following, a description of the system design, subsystem decomposition, and database abstraction is given. Afterwards, the dynamic processing and presentation schema designed to minimize coding efforts is described followed by description of the functionalities of the DCM system.

5.2 **Concept map**

When dealing with a huge collection of items stored in a repository, a storage and retrieval mechanism is desired that is based on sound technical as well as pedagogical grounds. The degree of usability of the information stored depends highly depends on the naturalness of storage, navigation and presentation of the content.

There are many researches being conducted worldwide on how to best model an arbitrary aggregation of small knowledge elements. The DCM system uses the ideas of Concept maps [7] to model the arbitrary relationship that can be created by the educators by their selection of the knowledge elements from the repository. Concept map is the concept of representing knowledge in graphs where nodes represent concepts and the links represent the relation between concepts. Fig 8 shows a sample concept map

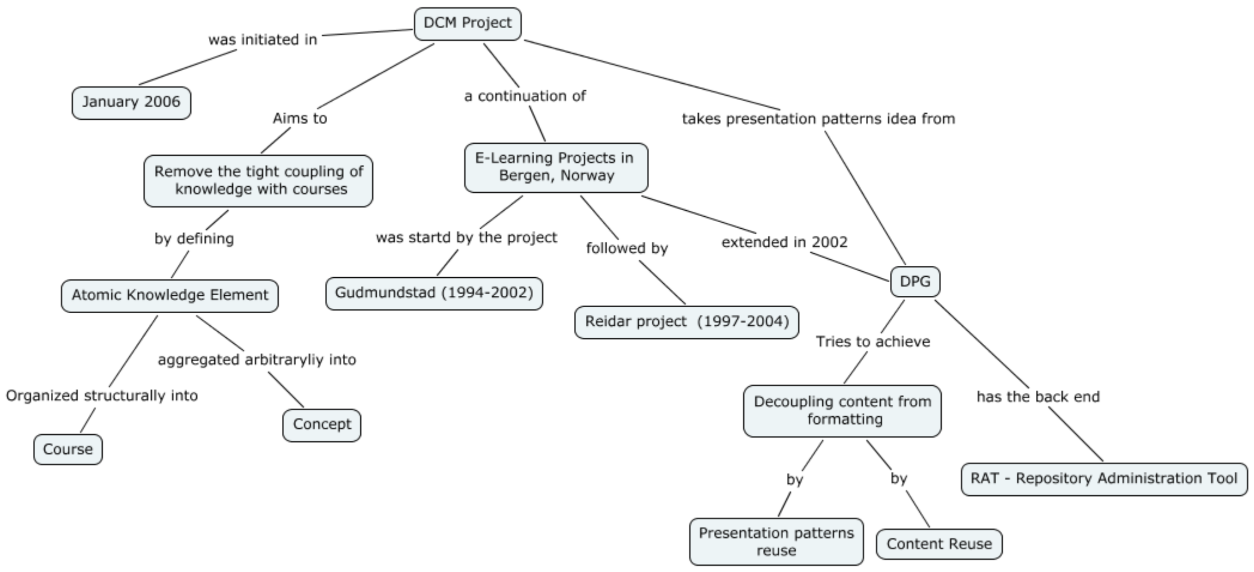

DCM Project
was initiated in
January 2006
Aims to
a continuation of
takes presentation patterns idea from
Remove the tight coupling of knowledge with courses
E-Learning Projects in Bergen, Norway
by defining
was startd by the project
followed by
extended in 2002
DPG
Atomic Knowledge Element
Gudmundstad (1994-2002)
Reidar project (1997-2004)
Tries to achieve
aggregated arbitraryliy into
Organized structurally into
Concept
Course
Decoupling content from formatting
has the back end
by
by
RAT - Repository Administration Tool
Presentation patterns reuse
Content Reuse

Figure 8 - A sample concept map created using IHMC CmapTools [10]

The DCM provides the educator with a graphical navigation of the knowledge repository in the manner depicted in fig 8 (although the display is not yet perfected). Such a visual presentation gives the educator a feel of the resources of the repository and enables her to create her own universe of knowledge elements collection. The educator can also specify different dependencies in the learning content that can influence the students' assimilation of the learning material. The interconnection between the various knowledge elements defined by many educators can be used for data mining purposes and to create a better structuring of the knowledge repository as a whole.

5.3 Student modeling and progress follow-up

5.3.1 Student modeling

Simply put, student modeling is the construction of a qualitative representation, called a student model, that, broadly speaking, accounts for student behavior in terms of a system's background knowledge [8]

Student modeling, as the model of a learner, represents the computer system's belief about the learner's knowledge [9]. Building a student model involves defining; the "who", or the degree of specialization in defining who is modeled, and what the learner history is; the "what", or the goals, plans, attitudes, capabilities, knowledge, and beliefs of the learner; the "how" the model is to be acquired and maintained; and the "why" , including to whether to elicit information from the learner, to give assistance to the learner, to provide feedback to the learner, or to interpret learner behavior.

In maintaining the student model, the factors that need to be considered include the fact that students do not perform consistently, forget information randomly, and then display large leaps in understanding. For an intelligent instructional system to be tailored to the student, the student model is the essential component in individualized learning. It is the student model that builds and maintains the system's understanding of the student.

The knowledge base of the student model takes both domain and pedagogical knowledge into account. The method to elicit a student model is closely tied to the approach. In the overlay approach, first the expert domain is modeled as a set of correct production rules. The learner is modeled as a subset of these correct rules, plus a set of incorrect production rules. Each new learner requires an individualized student model. In developing the student model, the type of knowledge (i.e., declarative, procedural) to be defined must be determined. It has to be decided whether or not to include student goals, and how to include these. The methods used include the users outlining their own goals, providing self-documentation, and submitting answers to a pre-test

5.3.2 Student modeling in the DCM

The DCM system defines a very simple model of the learner. The knowledge level of the learner is modeled on each content category. The system puts a learner initially into a default category with respect to the specific content she is studying. The updating category of the learner is provided by the DCM and based on assessments of the exercises delivered by the students.

The DCM supports questions and associates them to the knowledge element. These questions are the building blocks for the practice module where the learner goes through questions that have been associated to the knowledge elements of the course she is taking. The learner's category is taken into consideration and as a default optional filtering mechanism while providing the learner with the practice questions

5.4 **Overall system design**

Designing in software development is process of getting from a description of the properties a system should have (a specification) to a description of a system that has these properties (a design). Software Design focuses both on the significant structural elements of the system, such as subsystems; Use Cases, classes, components, Packages and nodes, as well as the collaborations that occur among these elements via interfaces.

The overall system modeling of the DCM system is shown in Fig 9. The system maintains the repository of knowledge elements which are organized structurally into courses and arbitrarily into concepts. Resources and questions are attributed to the knowledge elements.

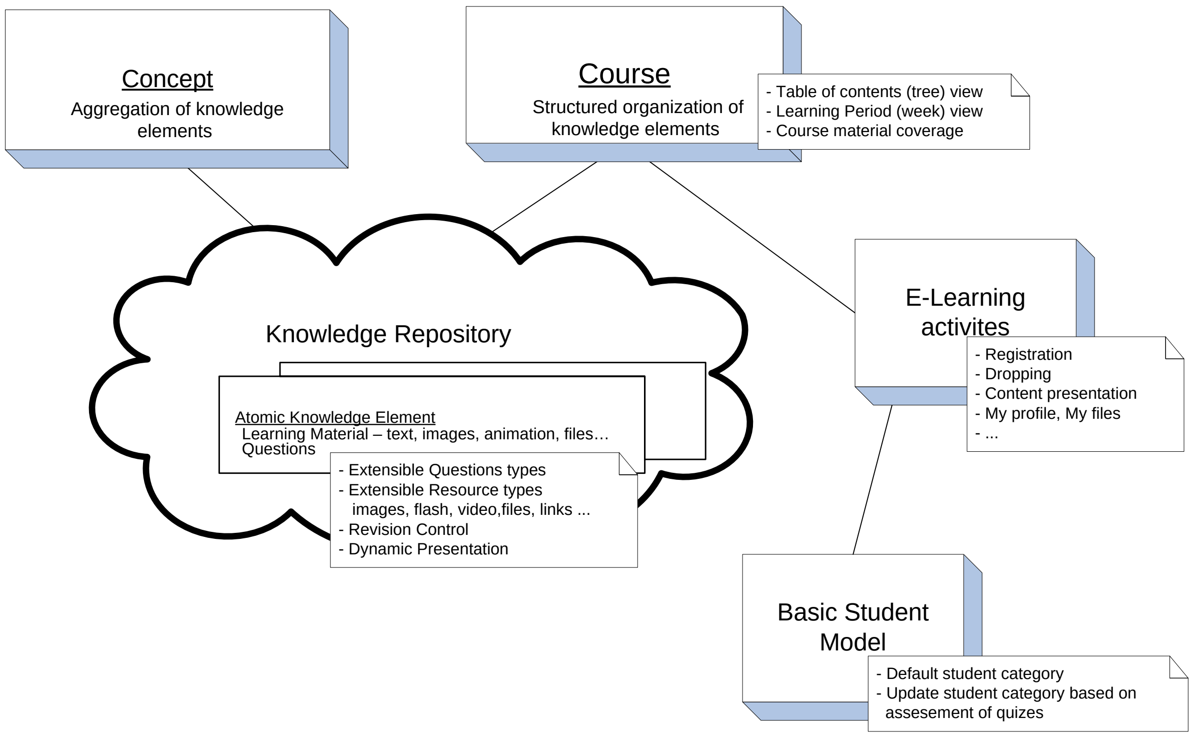

Concept
Aggregation of knowledge elements
Course
Structured organization of knowledge elements
- Table of contents (tree) view
- Learning Period (week) view
- Course material coverage
Knowledge Repository
Atomic Knowledge Element
Learning Material – text, images, animation, files…
Questions
- Extensible Questions types
- Extensible Resource types
images, flash, video,files, links ...
- Revision Control
- Dynamic Presentation
E-Learning activites
- Registration
- Dropping
- Content presentation
- My profile, My files
- ...
Basic Student Model
- Default student category
- Update student category based on assesement of quizes

Figure 9 – overall system design

5.5 **Subsystem decomposition**

In the development of systems that address many requirements, it is a software engineering principle that the system should be decomposed into smaller, more manageable and more closed subsystems. Subsystem decomposition helps ease the difficulty of addressing lots of requirements since each subsystem is created according to the natural, logical organization of the requirements. Interactions and dependencies between the subsystems are captured using interfaces of each subsystem with the others. By using subsystem decomposition, most of the operations and requirements become localized into one subsystem so that its detail implementation does not affect the other subsystems. This allows independent development of system components by possibly independent teams.

The DCM system was divided into five major subsystems: the *security subsystem*, *instructor subsystem*, *student subsystem*, *course, content and concept map subsystem* and the *question and quiz subsystem*. Following, the diagrammatical display of these subsystems with their interdependence is shows followed by description of each subsystem.

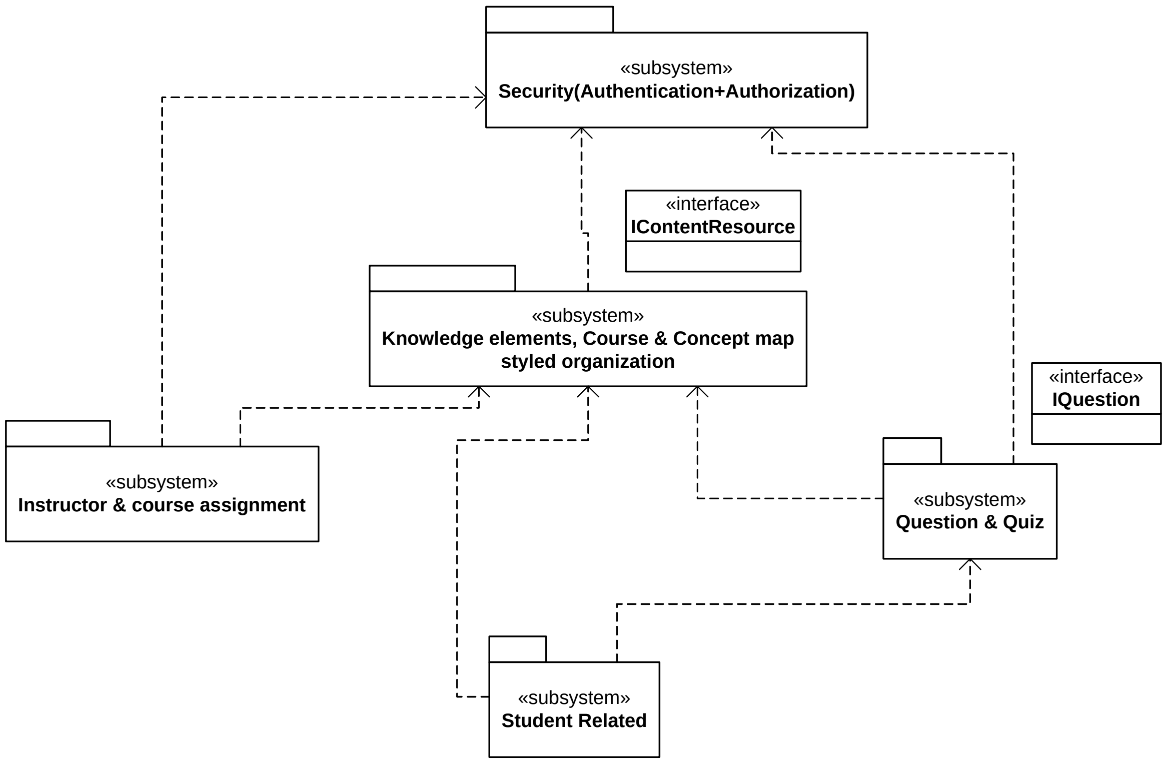


Figure 10 – subsystem decomposition of the DCM system

Security sub system

In a multi user environment, security is a very crucial requirement. Especially systems used in learning environment need to be extra secure since they are faced with a very dynamic, enthusiastically ambitions and creative student. The security subsystem of the DCM project handles the authentication and authorization of users, roles membership and operations assignment. User passwords are stored using a strong, irreversible one way encryption.

Role based security is a security schema where roles are defined into the system and users are given membership into the roles. Users inherit privilege levels from their roles and are subject to the rules defined on their roles. Role based security gives great ease in administration of systems so that the privileges of only a few well defined roles is required instead of the great number of users that are present.

The DCM system employs an additional security schema that can be called operation based security. This schema defines the functionality of the system as a list of operations and gives finer grained security on the operations by assigning/revoking different access levels to roles or users. To consider an example of usage of the operation based security, consider a situation where different instructors are given access update content of specific courses based on their offering assignment. As being a member of the Instructor role, they get privileges to the general *maintain course* operation whereas which specific course they can update is defined by the detail of the privilege which is set while instructors are assigned to specific courses during their course offering assignment.

Instructor sub system

The instructor subsystem of the DCM handles requirements relevant to instructors. Instructor registration is one major part of the system since instructors have to be registered into the system before they can be assigned courses to teach and can perform the creation and modification of content. The instructor subsystem also covers the process of assigning courses to instructors. Courses are assigned to instructors in a specific learning period. One or more instructors can be assigned to a single course.

The creation, usage and updating privileges are also addressed in the instructor subsystem. Instructors can update contents of only those courses that they have been assigned to in their course offering.

Student sub system

The student is the by far the most important individual of any e-learning system. Content is developed by instructors so that it can be availed to students. The student sub system addresses the basic student related activities. First and foremost, students have to be enrolled into the system and registered on each semester that they attend their education. Student registration can be carried out by administrator or by students themselves. When students register themselves, they should provide a registration key that will be given out in a classroom session by the instructor or via private mail. Their status will be pending and will require an activation approval by the system administrator before they can use the DCM system.

Content, Course and Concept map sub system

In the DCM system, the atomic knowledge element is given the name Content. Hence, a course is a structured organization of contents and a concept map is an arbitrary aggregation of content in a structure defined by the instructor.

This subsystem is the heard of the DCM system model. It addresses the requirements of defining, maintaining and teaching of courses. Also, creation, navigation and manipulation of concepts is addressed by this subsystem.

Question and Quiz sub system

One of the requirements of the DCM is that instructors should create and maintain questions and quizzes. The question and quiz subsystem focuses on the requirement of creating different types of questions and on the definition and manipulation of quizzes.

5.6 Dynamic Processing and Presentation

When implementing systems with different functionality as can be found in E-Learning systems, the implementation becomes very tedious because the system consists of too many pages and functionalities.

To avoid this, the DCM defines a declarative way of defining pages and rendering using a dynamic engine. The DCM dynamic processing model classifies all pages of the system into four major categories. These are:

Process pages - pages that are used to implement realize some business logic

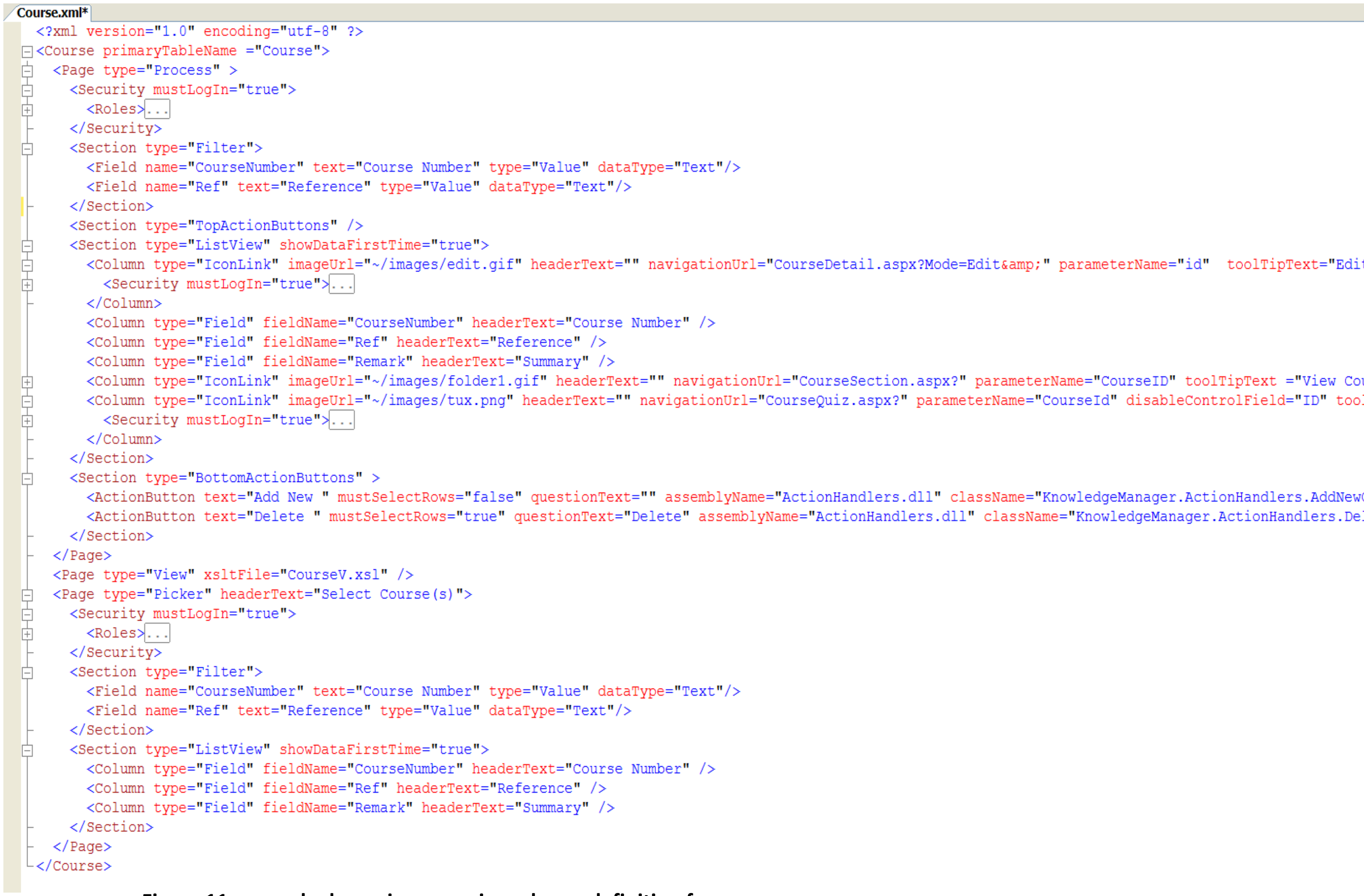
Editor pages – pages that are used for editing details of records. These pages can be a direct map to a single database entity or to multiple ones

View pages – pages that fall under this category are those that are used for displaying of existing information in some way. Mostly, view pages display results of queries that may be simple or complicated

Picker pages – when working with a relational database model for persistence of data of a system, there are always relations among the various database entities. Picker pages are basically defined so as to realize an inter dependence in data defined by the business logic. The basic function of a picker page is to provide a reasonably easy way of selecting some item to be used by way of reference in some other processing.

The DCM provides default implementations of the above 4 page types so that if the requirements of the system are not very specialized, they can be handled using these. Hence, for processing, editing, viewing and or picking, the DCM uses the default implementations and avoids much work.

The dynamic processing and presentation of entities is processed by the dynamic engine by taking a declarative XML schema (see fig 11). The schema optionally contains four high level Page elements of the types *Process*, *Edit*, View and *Picker* that represent the four types of dynamic pages generation. Following the XML schema of fig 11, description of each section and its relevance in the DCM system is described.


Course.xml*

```xml
<?xml version="1.0" encoding="utf-8" ?>
<Course primaryTableName ="Course">
  <Page type="Process" >
    <Security mustLogIn="true">
      <Roles>...
    </Security>
    <Section type="Filter">
      <Field name="CourseNumber" text="Course Number" type="Value" dataType="Text"/>
      <Field name="Ref" text="Reference" type="Value" dataType="Text"/>
    </Section>
    <Section type="TopActionButtons" />
    <Section type="ListView" showDataFirstTime="true">
      <Column type="IconLink" imageUrl="~/images/edit.gif" headerText="" navigationUrl="CourseDetail.aspx?Mode=Edit&amp;" parameterName="id"  toolTipText="Edi
        <Security mustLogIn="true">...
      </Column>
      <Column type="Field" fieldName="CourseNumber" headerText="Course Number" />
      <Column type="Field" fieldName="Ref" headerText="Reference" />
      <Column type="Field" fieldName="Remark" headerText="Summary" />
      <Column type="IconLink" imageUrl="~/images/folder1.gif" headerText="" navigationUrl="CourseSection.aspx?" parameterName="CourseID" toolTipText ="View Cou
      <Column type="IconLink" imageUrl="~/images/tux.png" headerText="" navigationUrl="CourseQuiz.aspx?" parameterName="CourseId" disableControlField="ID" tool
        <Security mustLogIn="true">...
      </Column>
    </Section>
    <Section type="BottomActionButtons" >
      <ActionButton text="Add New " mustSelectRows="false" questionText="" assemblyName="ActionHandlers.dll" className="KnowledgeManager.ActionHandlers.AddNew
      <ActionButton text="Delete " mustSelectRows="true" questionText="Delete" assemblyName="ActionHandlers.dll" className="KnowledgeManager.ActionHandlers.De
    </Section>
  </Page>
  <Page type="View" xsltFile="CourseV.xsl" />
  <Page type="Picker" headerText="Select Course(s)">
    <Security mustLogIn="true">
      <Roles>...
    </Security>
    <Section type="Filter">
      <Field name="CourseNumber" text="Course Number" type="Value" dataType="Text"/>
      <Field name="Ref" text="Reference" type="Value" dataType="Text"/>
    </Section>
    <Section type="ListView" showDataFirstTime="true">
      <Column type="Field" fieldName="CourseNumber" headerText="Course Number" />
      <Column type="Field" fieldName="Ref" headerText="Reference" />
      <Column type="Field" fieldName="Remark" headerText="Summary" />
    </Section>
  </Page>
</Course>
```

Figure 11 – sample dynamic processing schema definition for a course

5.6.1 Process pages

Process pages are pages that are used to implement realize some business logic. Basically, any page that performs operation on entities that is persistent falls under this category.

The DCM defines a declarative default implementation of process pages. This implementation of a process page consists of an optional filter section, a list section that displays rows of records and optional top and bottom action buttons that can be used for business processing. The flow chart of activities the dynamic engine follows while producing a dynamic process page is given in fig 12.

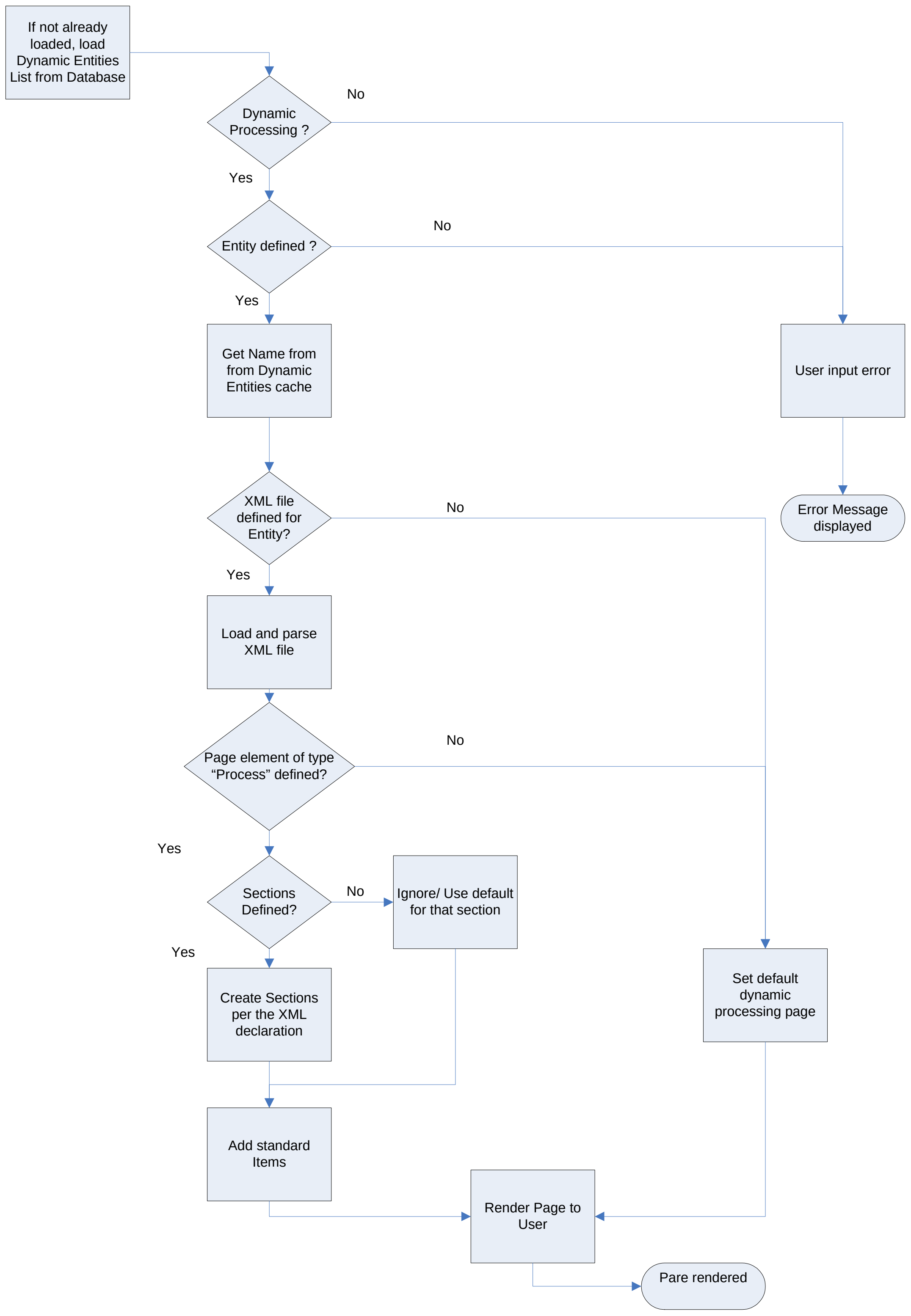

If not already loaded, load Dynamic Entities List from Database
Dynamic Processing ?
No
Yes
Entity defined ?
No
Yes
Get Name from from Dynamic Entities cache
User input error
Error Message displayed
XML file defined for Entity?
No
Yes
Load and parse XML file
Page element of type "Process" defined?
No
Yes
Sections Defined?
No
Ignore/ Use default for that section
Yes
Create Sections per the XML declaration
Set default dynamic processing page
Add standard Items
Render Page to User
Pare rendered

Figure 12 – flow chart of producing a dynamic process page

5.6.1.1 Filter section

Filtering is a means by which a selection is made using the desired criteria. Filtering criteria will be dependent on the type and complexity of the entity being manipulated.

In the XML declarative schema manipulation of the DCM, the filter section is an optional section of the process and picker pages category. When defined, this section dictates the type of filtering to be applied to the entity being dynamically processed. Fig 13 shows a sample filter section defined for the dynamic course processing.

```
<Section type="Filter">
  <Field name="CourseNumber" text="Course Number" type="Value" dataType="Text"/>
  <Field name="Ref" text="Reference" type="Value" dataType="Text"/>
  <Field name="IDs" type="Range" text="Registration Date">
    <Field text="From" dataType="Text"/>
    <Field text="To" dataType="Text"/>
  </Field>
</Section>
```

Figure 13 - filter section for course process page

When the XML schema of the defined filter section is encountered by the dynamic engine of the DCM, the appropriate user interface is generated for it so that the user can apply the desired filtering on the entity to be displayed. Fig 14 shows the user interface that is generated using the declaration of fig 13.

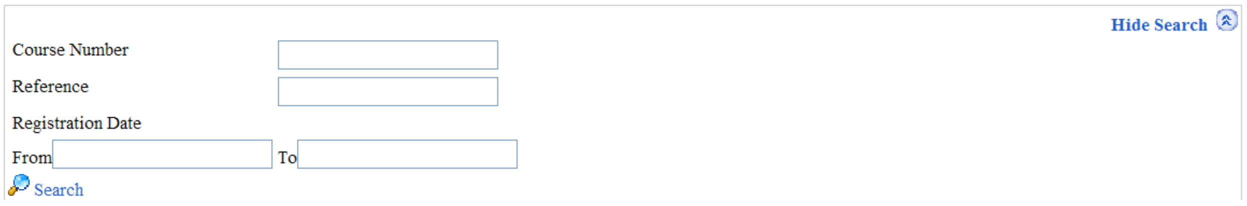


Figure 14 - generated filter user interface governed by filter XML schema of fig 13

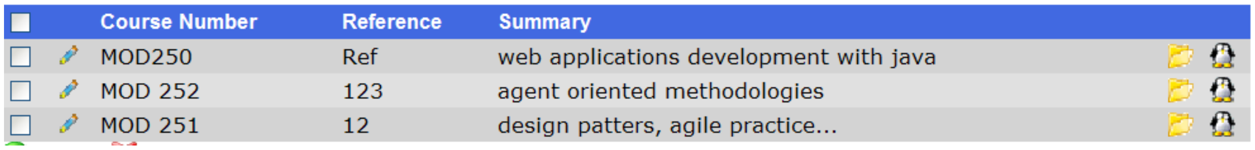
Currently, the DCM dynamic filter section support only direct value type and range types as shown in the figure above. In the future, this can be extended to have dynamically populated dropdown filters and much more for advanced declarative filtering. Also, enhancement can be done on the data types of the filter fields so that user entered values can also be validated dynamically.

5.6.1.2 List section

This section is a declarative description for the dynamic engine with regards to displaying a list of records. Fig 15 shows a list section of the course dynamic processing page. The resulting generated dynamic display of courses is shown in fig 16.

```xml
<Section type="ListView" showDataFirstTime="true">
  <Column type="IconLink" imageUrl="~/images/edit.gif" headerText="" navigationUrl="CourseDetail.aspx?Mode=Edit&amp;" parameterName="id"
    <Security mustLogIn="true">
      <Roles>
        <Role name="Administrators"/>
      </Roles>
    </Security>
  </Column>
  <Column type="Field" fieldName="CourseNumber" headerText="Course Number" />
  <Column type="Field" fieldName="Ref" headerText="Reference" />
  <Column type="Field" fieldName="Remark" headerText="Summary" />
  <Column type="IconLink" imageUrl="~/images/folder1.gif" headerText="" navigationUrl="CourseSection.aspx?" parameterName="CourseID" tool
  <Column type="IconLink" imageUrl="~/images/tux.png" headerText="" navigationUrl="CourseQuiz.aspx?" parameterName="CourseId" disableCont
    <Security mustLogIn="true">
      <Roles>
        <Role name="Administrators"/>
        <Role name="Instructors"/>
      </Roles>
      <Operations>
        <Operation name="EditContent" requireDetail="False"/>
      </Operations>
    </Security>
  </Column>
</Section>
```

Figure 15 - list section of the course process page



☐		Course Number	Reference	Summary		
☐	✎	MOD250	Ref	web applications development with java	📁	🐧
☐	✎	MOD 252	123	agent oriented methodologies	📁	🐧
☐	✎	MOD 251	12	design patters, agile practice...	📁	🐧

Figure 16 – dynamic rendering of courses governed by the XML schema of figure 15

Currently, there are four types of fields supported by the list view section columns. These are:

Field - this is a display of a direct field of a row record. Formatting can be applied to the filed data using declarative entry as shown below

```xml
<Column type="Field" fieldName="ToDate" headerText="ToDate" DataFormatString="{0:dd MMMM,yyy
```

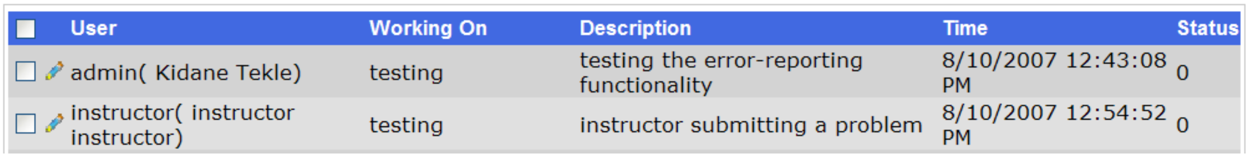
Figure 17 – declarative application of formatting on field

Icon link – an icon link is a column that appears as an icon in on the list view with the navigation URL being specified declaratively. It is very helpful in extending and redirecting to related pages/things. Looking at the declarative description of the editing column in fig 15 and its rendering in fig 16, the user will be redirected to the page defined in the ***navigationUrl*** parameter with the defined parameters.

Mapped field – a mapped field is a special field where the display of the column will be displayed from another data source defined declaratively. Taking an example,

```
<Column type="MappedField" fieldName="UserId" headerText="User" sourceTable="Users" expression="Use
```

Figure 18 – **mapped field declaration for the UserId column**



☐	User	Working On	Description	Time	Status
☐	admin(Kidane Tekle)	testing	testing the error-reporting functionality	8/10/2007 12:43:08 PM	0
☐	instructor(instructor instructor)	testing	instructor submitting a problem	8/10/2007 12:54:52 PM	0

Figure 19 – **dynamically rendered output governed by the XML schema of fig 18 with mapped field**

Security of columns

All the columns of the dynamic list view section can be guarded by a declarative security requirement. As the security model of the DCM is both Role based and Operation based, these can be defined as shown in the fig 20 below.

```
<Column type="IconLink" imageUrl="~/images/tux.png" headerText=""
  <Security mustLogIn="true">
    <Roles>
      <Role name="Administrators"/>
      <Role name="Instructors"/>
    </Roles>
    <Operations>
      <Operation name="EditContent" requireDetail="False"/>
    </Operations>
  </Security>
```

Figure 20 – **declarative role based and operation based security on list view column**

5.6.1.3 Top/Bottom action buttons section

Top/bottom action button sections are features provided by the DCM system as an extensibility interface for other developers and realization of additional functionality.

The DCM provides an interface that must be implemented by all button action handlers. Fig 21 shows the declaration of the interface.

```
public interface IActionButtonHandler
{
    string Image { get;}
    void HandleEvent(Page page, ArrayList args);
}
```

Figure 21 – the IActionButtonHandler interface

Interface IActionButtonHandler

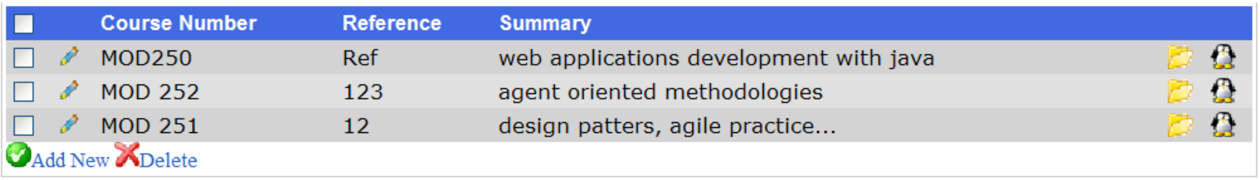
- Description: interface to be implemented by all action handlers to be declaratively bound as providing business logic operations
- Field – Image
    - A read only property that returns the path to the image to be used when rendering the action button
- Method – handle event
    - Takes the page and list of parameters to be used for processing. In most cases, the list of arguments will be id numbers of the rows selected from the list view display section of a process page

Fig 22 shows the XML schema used to declare an *Add New* and *Delete* action buttons on the bottom section of the courses dynamic processing page.

```
<Section type="BottomActionButtons" >
  <ActionButton text="Add New " mustSelectRows="false" questionText="" assemblyName="ActionHandlers.dll"
                className="KnowledgeManager.ActionHandlers.AddNewCourse"/>
  <ActionButton text="Delete " mustSelectRows="true" questionText="Delete" assemblyName="ActionHandlers.dll"
                className="KnowledgeManager.ActionHandlers.DeleteCourse"/>
</Section>
```

Figure 22 – XML Schema of dynamic bottom action buttons for course

When the dynamic engine reaches the declaration shown in fig 22, it loads the declared assembly, creates an instance of the defined class and makes a call to the Image property string. This string is used as the path for the image to be used in the action button and is rendered as an image and a link that the user can click and perform the desired action. Fig 23 shows a rendering of the AddNew and Delete action buttons defined in the course XML schema.

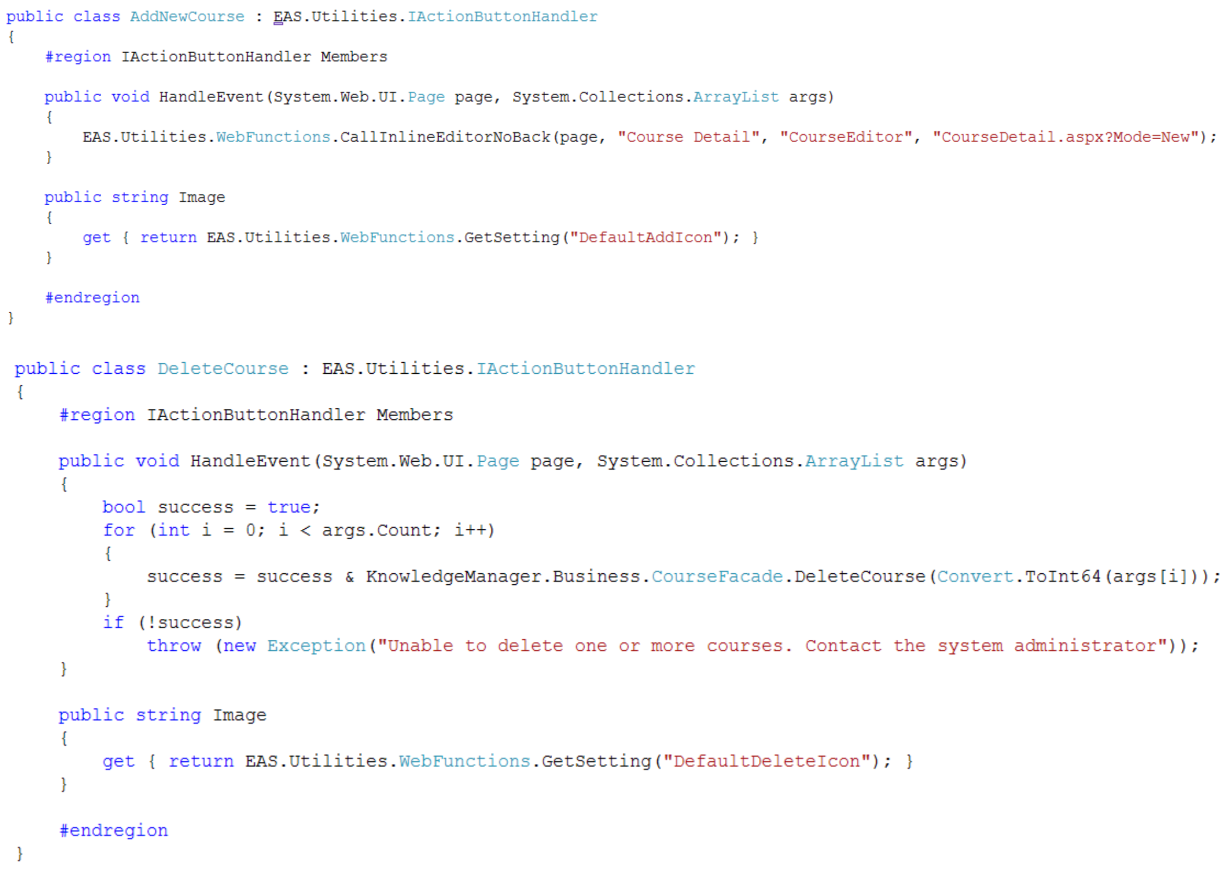


☐		Course Number	Reference	Summary		
☐	✎	MOD250	Ref	web applications development with java		
☐	✎	MOD 252	123	agent oriented methodologies		
☐	✎	MOD 251	12	design patters, agile practice...		

✓Add New ✗Delete

Figure 23 – rendering of dynamic bottom action buttons for course

When a user clicks on any of the action buttons, the appropriate action button handler is loaded and executed by the dynamic processing engine. As a follow up on our example, the following figure shows implementations of the AddNewCourse and DeleteCourse button action handler classes.



```
public class AddNewCourse : EAS.Utilities.IActionButtonHandler
{
    #region IActionButtonHandler Members

    public void HandleEvent(System.Web.UI.Page page, System.Collections.ArrayList args)
    {
        EAS.Utilities.WebFunctions.CallInlineEditorNoBack(page, "Course Detail", "CourseEditor", "CourseDetail.aspx?Mode=New");
    }

    public string Image
    {
        get { return EAS.Utilities.WebFunctions.GetSetting("DefaultAddIcon"); }
    }

    #endregion
}

public class DeleteCourse : EAS.Utilities.IActionButtonHandler
{
    #region IActionButtonHandler Members

    public void HandleEvent(System.Web.UI.Page page, System.Collections.ArrayList args)
    {
        bool success = true;
        for (int i = 0; i < args.Count; i++)
        {
            success = success & KnowledgeManager.Business.CourseFacade.DeleteCourse(Convert.ToInt64(args[i]));
        }
        if (!success)
            throw (new Exception("Unable to delete one or more courses. Contact the system administrator"));
    }

    public string Image
    {
        get { return EAS.Utilities.WebFunctions.GetSetting("DefaultDeleteIcon"); }
    }

    #endregion
}
```

Figure 24 - course action handler class implementations

5.6.2 Edit pages

In the DCM solution, edit pages are pages that are used for editing details of records. These pages can be a direct map to a single database entity or to multiple ones. Basically, there are two types of edit pages

5.6.2.1 Simple edit pages

Simple edit pages are mapping of a single database entity onto a page for editing. Hence, such pages edit a single record in a database entity. DCM provides a dynamic editing scenario for such tables. Edit pages are defined by the Page type ***Edit*** in the dynamic XML schema. The following figure shows a dynamic edit schema for the course entity.

```
<Page type="Edit" xsltFile="CourseE.xsl" />
```

The work flow defined for dynamic editing is as follows:

- Retrieve the desired record
- Check if an XSLT translation is defined in the Edit page definition schema
    - If defined – apply the XSLT transformation on the record row and render the resulting output
    - If not defined – apply the default XSLT transformation for rendering

Fig 25 shows the default XSLT transformation for edit pages and figure 26 shows a resulting rendered output for the dynamic processing of the entity ***Learning Period***

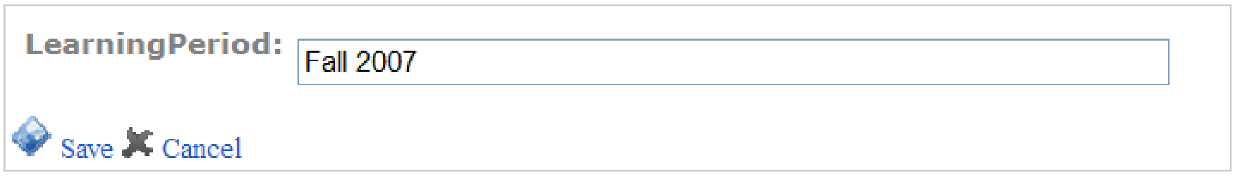
```
<?xml version="1.0" encoding="utf-8"?>

<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:asp="remove">
  <xsl:template match="/">
    <html>
      <body>
          <xsl:for-each select="//Table1">
            <table width="100%" align="left" >
              <xsl:for-each select="field">
                <xsl:if test="@name!='ID'"> <!--don't allow the id column to be edited-->
                  <tr>
                    <td align="left">
                      <h3>
                        <xsl:value-of select="@name" />:
                      </h3>
                    </td>
                    <td >
                      <asp:TextBox id="{@name}" runat="server" value="{@value}" Width="400px"/>
                          <asp:RequiredFieldValidator  runat="server" ControlToValidate="{@name}"
                            ErrorMessage="*"></asp:RequiredFieldValidator>
                    </td>
                  </tr>
                </xsl:if>
              </xsl:for-each>
            </table>
          </xsl:for-each>
      </body>
    </html>
  </xsl:template>

</xsl:stylesheet>
```

Figure 25 – the default editing XSLT transformation

The default XSLT schema is applied to the data of any database entity for which a specialized transformation is not defined. Fig x shows the resulting dynamic editing form for the database entity of *Learning Period*.



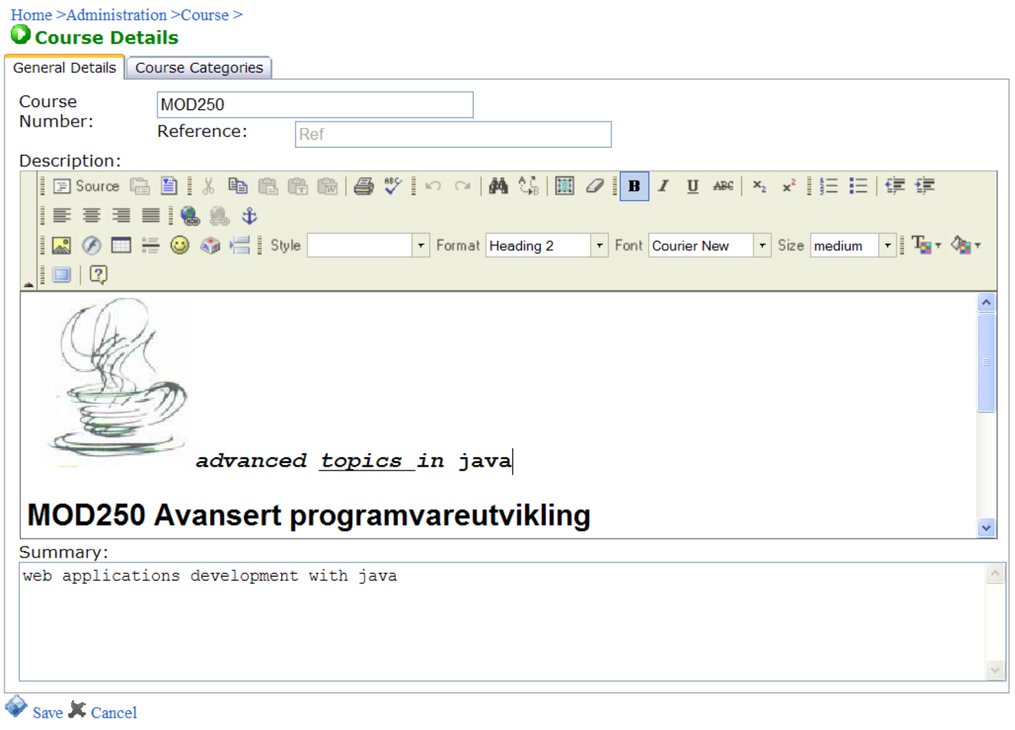
LearningPeriod:	Fall 2007

Figure 26 – default editing of Learning Period

By defining XSLT transformations in the manner the user desires, entities can be edited in an interface that is defined declaratively. This feature can further be extended in the future by allowing the educator to define her own look and feel (using some easy to use graphical tool) for editing of the courses she teaches.

5.6.2.2 Complex edit pages

Not all the functionality of pages could be generalized and defined declaratively. For such cases, the DCM consists of edit pages that are complex in nature. These pages manipulate data of more than one entity of the database in a manner that is easy for the user. Fig 27 shown the course edit page as an example of complex editing pages.



Home >Administration >Course >

Course Details

General Details | Course Categories

Course Number: MOD250

Reference: Ref

Description:

Style: Format: Heading 2 | Font: Courier New | Size: medium

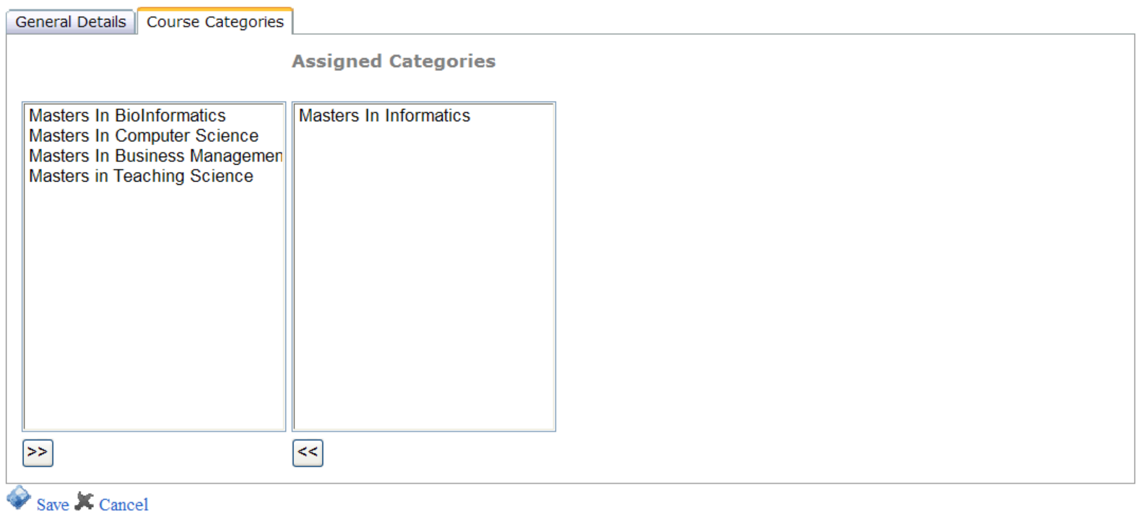
advanced topics in java

MOD250 Avansert programvareutvikling

Summary:

```
web applications development with java
```

Save Cancel



General Details | Course Categories

Assigned Categories

Masters In BioInformatics	Masters In Informatics
Masters In Computer Science	
Masters In Business Managemen	
Masters in Teaching Science	

>> <<

Save Cancel

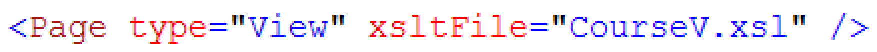
Figure 27 – course details editing page as an example of complex editing page

5.6.3 **View pages**

In the DCM solution, view pages are pages that are used for viewing details of records. These pages can be a direct map to a single database entity or to multiple ones. Basically, there are two types of view pages

5.6.3.1 Simple view pages

Simple view pages are mapping of a single database entity onto a page for viewing. Hence, such pages show a single record in a database entity. DCM provides a dynamic viewing scenario for such tables. View pages are defined by the Page type ***View*** in the dynamic XML schema. The following figure shows a dynamic view schema for the course entity.

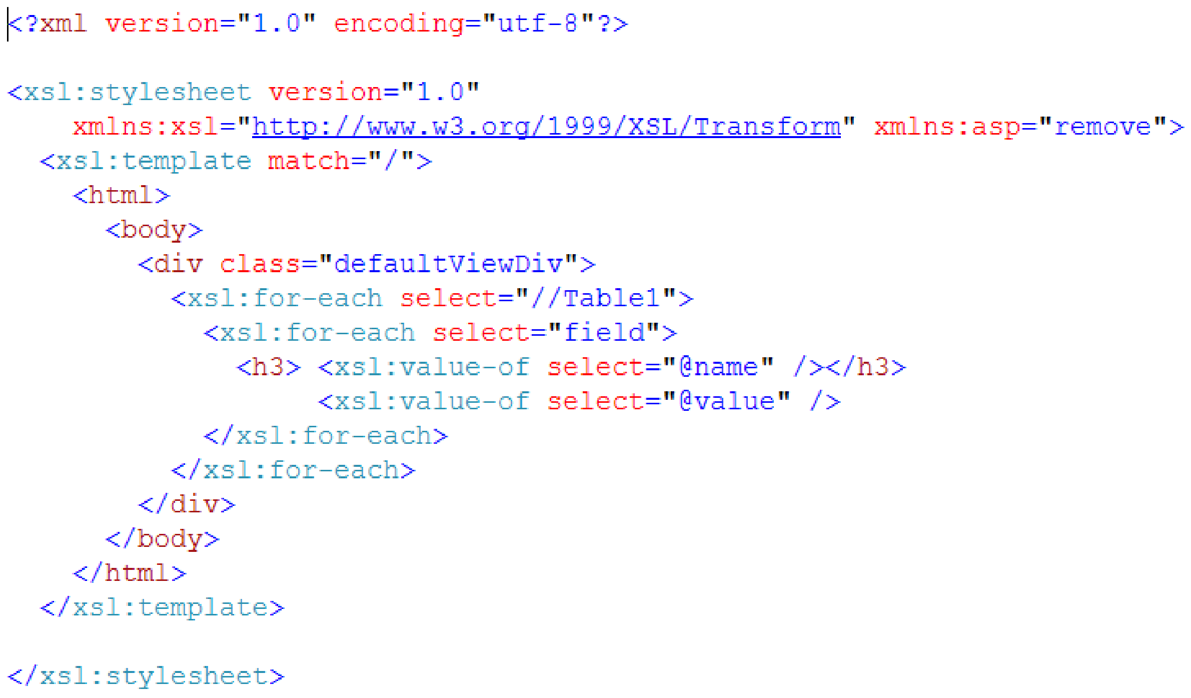


```
<Page type="View" xsltFile="CourseV.xsl" />
```

The work flow defined for dynamic viewing is as follows:

- Retrieve the desired record
- Check if an XSLT translation is defined in the View page definition schema
    - If defined – apply the XSLT transformation on the record row and render the resulting output
    - If not defined – apply the default XSLT transformation for rendering

The following figure represents the default view XSLT transformation schema of the DCM system.



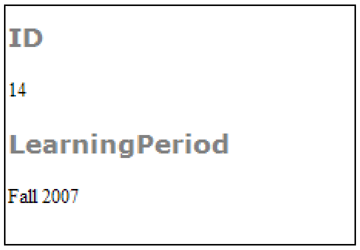
```
<?xml version="1.0" encoding="utf-8"?>

<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:asp="remove">
  <xsl:template match="/">
    <html>
      <body>
        <div class="defaultViewDiv">
          <xsl:for-each select="//Table1">
            <xsl:for-each select="field">
              <h3> <xsl:value-of select="@name" /></h3>
                   <xsl:value-of select="@value" />
            </xsl:for-each>
          </xsl:for-each>
        </div>
      </body>
    </html>
  </xsl:template>

</xsl:stylesheet>
```

Figure 28 - the default viewing XSLT transformation

Considering the entity **Learning Period** as an example, with no special viewing XSLT transformation file defined, the default one will be applied and would result in a display similar to the figure shown below.



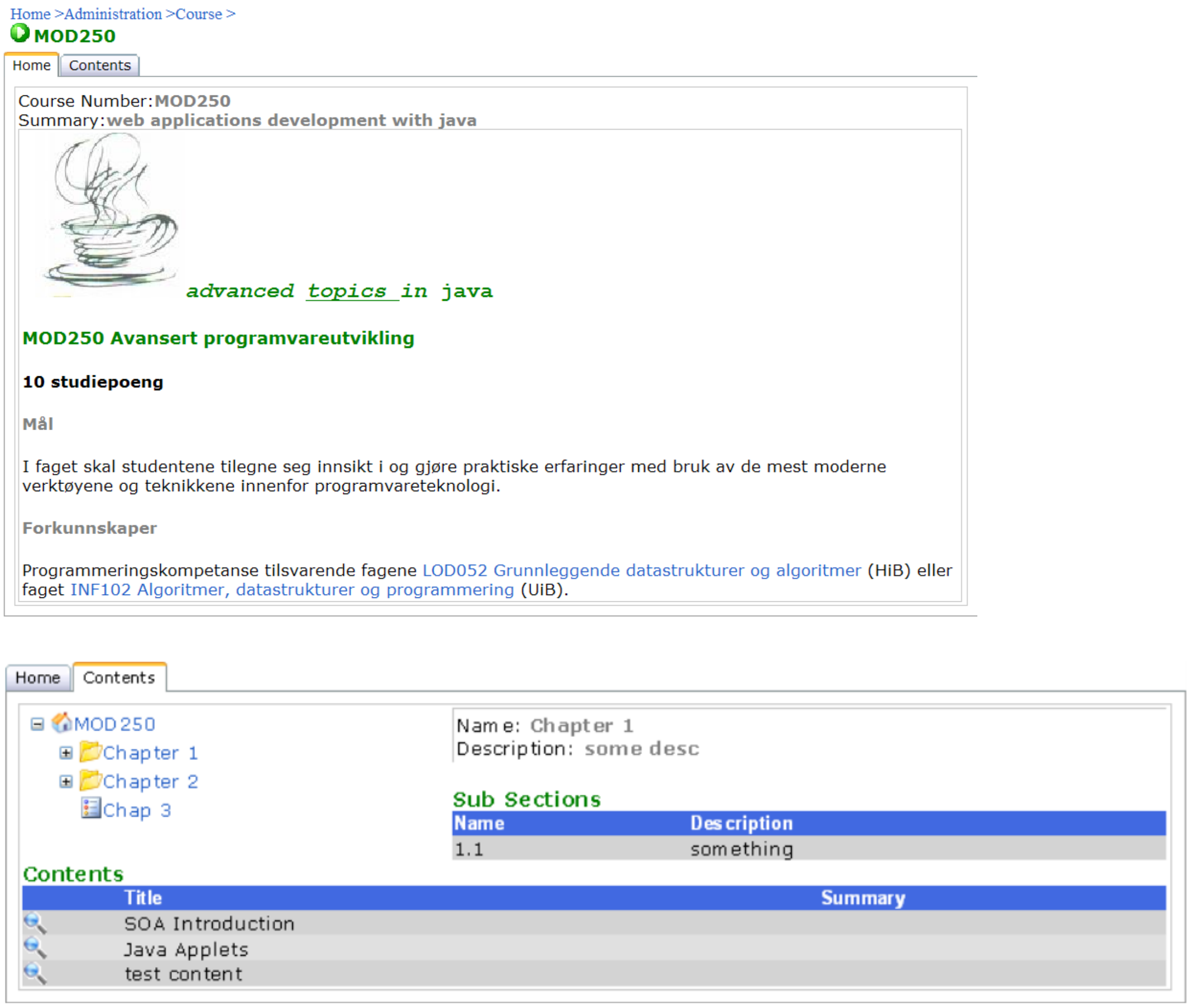
ID
14
LearningPeriod
Fall 2007

Figure 29 – default viewing applied to Learning period

By defining XSLT transformations in the manner the user desires, entities can be displayed in an interface that is defined declaratively. Any valid XSLT transformation file can be applied so that the resulting output can be made as good as desired. This feature can further be extended in the future by allowing the educator to define her own look and feel (using some easy to use graphical tool) for the presentation of courses she teaches

5.6.3.2 Complex view pages

Complex view pages are those that display data collected from more than one database entity and that have highly customized presentation. In the DCM, such pages are defined on their own with the formatting and display of the appropriate data being dedicated to that page. Fig 30 shows a specialized complex display page of course.



Course Number:MOD250
Summary:web applications development with java

advanced topics in java

MOD250 Avansert programvareutvikling

10 studiepoeng

Mål

I faget skal studentene tilegne seg innsikt i og gjøre praktiske erfaringer med bruk av de mest moderne verktøyene og teknikkene innenfor programvareteknologi.

Forkunnskaper

Programmeringskompetanse tilsvarende fagene LOD052 Grunnleggende datastrukturer og algoritmer (HiB) eller faget INF102 Algoritmer, datastrukturer og programmering (UiB).

Name: Chapter 1
Description: some desc

Sub Sections

Name	Description
1.1	something

Contents

Title	Summary
SOA Introduction	
Java Applets	
test content	

Figure 30 – course display page as an example of complex view page

5.6.4 Picker pages

Due to the normalized design of the DCM database structure, information is found distributed in different tables. Hence, in order to perform certain activities, one needs to select items from another entity for defining relations.

The DCM defines a concept called picker pages provide a reasonably easy way of selecting some item to be used by way of reference in some other processing. Based on their complexity, the picker pages can be classified as simple and complex. According to the number of records they pick, the picker pages of the DCM can also be either single selection only or allowing multiple selection. In effect, the DCM defines a picker interface which can be implemented by any page so that it can be used as a picker of some basic entity.

5.6.4.1 Simple picker pages

A dynamic picker page is structured in a similar manner to the dynamic processing pages defined in section 5.4.1. The flow chart is basically the same with processing pages with the following limitations on sections

Filter section – same as dynamic processing page

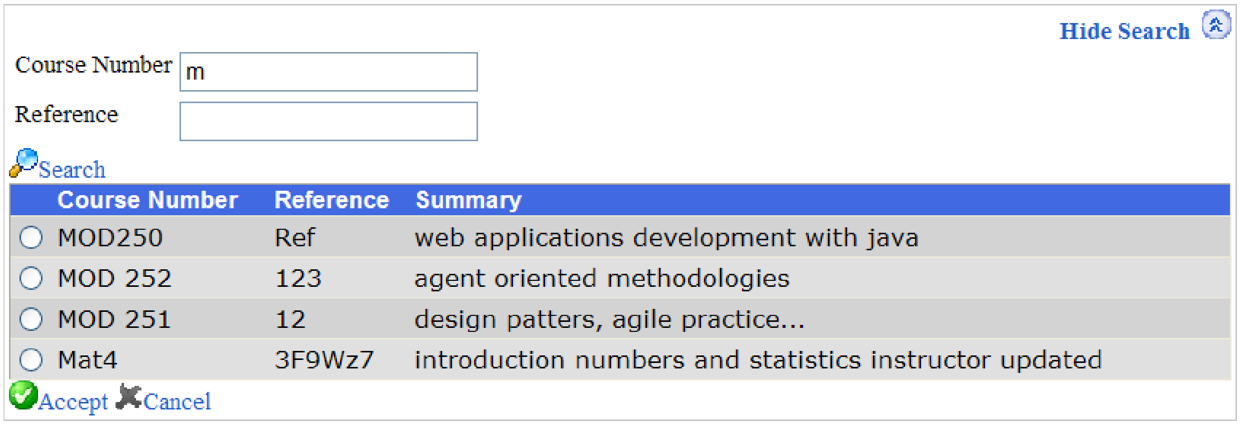
List view section – limitation that Icon columns are not allowed

Bottom/top action buttons section – removed since the only action required on pickers is that of accepting the selection.

```
<Page type="Picker" headerText="Select Course(s)">
  <Security mustLogIn="true">
    <Roles>
      <Role name="Administrators"/>
      <Role name="Instructor"/>
      <Role name="Student"/>
    </Roles>
  </Security>
  <Section type="Filter">
    <Field name="CourseNumber" text="Course Number" type="Value" dataType="Text"/>
    <Field name="Ref" text="Reference" type="Value" dataType="Text"/>
  </Section>
  <Section type="ListView" showDataFirstTime="true">
    <Column type="Field" fieldName="CourseNumber" headerText="Course Number" />
    <Column type="Field" fieldName="Ref" headerText="Reference" />
    <Column type="Field" fieldName="Remark" headerText="Summary" />
  </Section>
</Page>
```

Figure 31 – XML schema description for dynamic course picker

When the dynamic engine reaches the above declared picker schema for a course, it generates the appropriate user interface creating the filtering criteria. It enables the user to make a selection (single or multiple based on the business in question). Fig 32 shows a user interface rendered to the user from the processing of the XML schema of fig 31



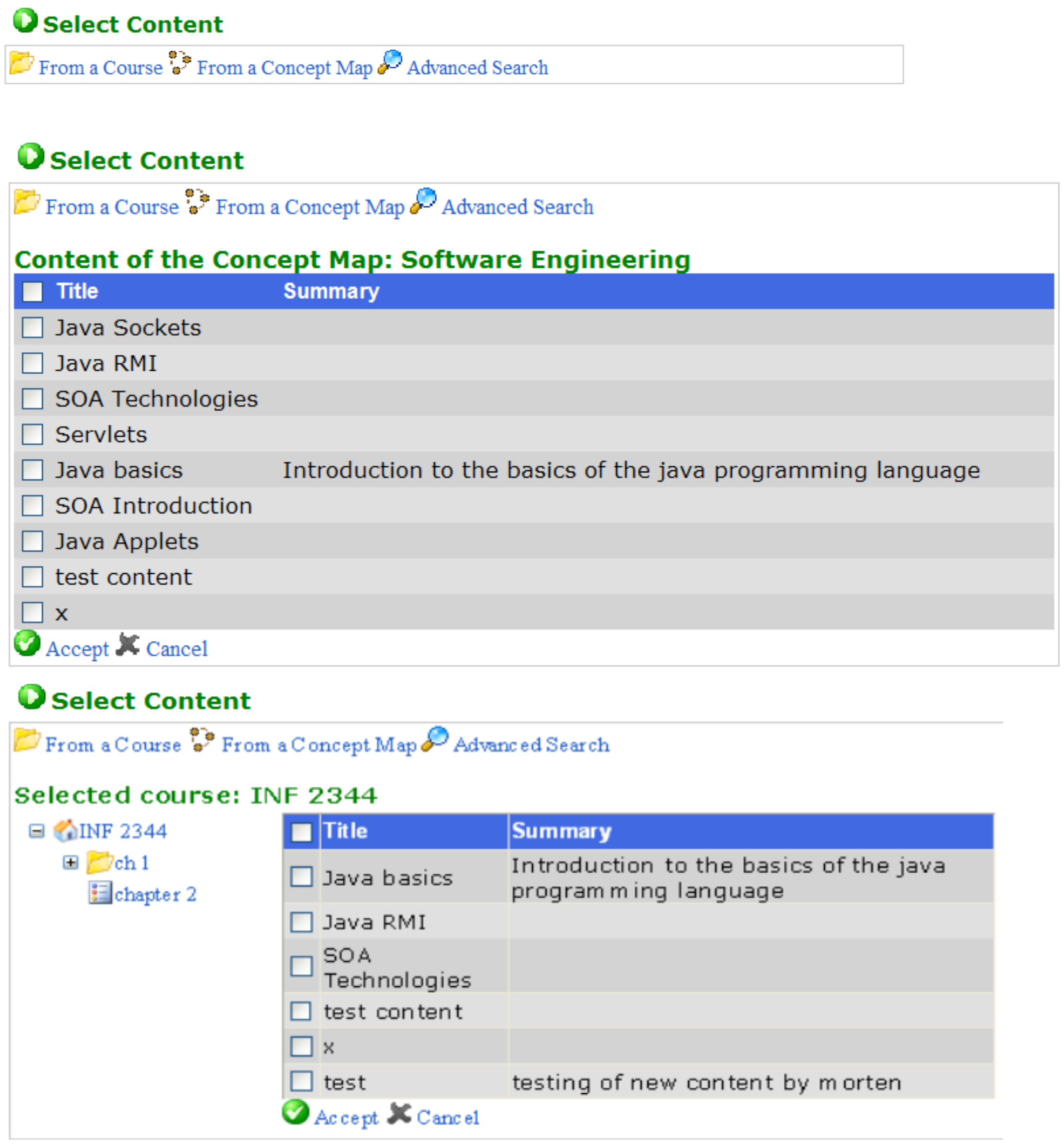
Course Number	m
Reference	

	Course Number	Reference	Summary
○	MOD250	Ref	web applications development with java
○	MOD 252	123	agent oriented methodologies
○	MOD 251	12	design patters, agile practice...
○	Mat4	3F9Wz7	introduction numbers and statistics instructor updated

Figure 32 – a dynamic course picker page rendered by processing XML schema of fig 31

5.6.4.2 Complex picker pages

Depending on the complexity of the required item to be selected, it might be necessary to define its own specialized page. The DCM defines some pages specialized for advanced picking of entities. Considering an example, when user wants to pick content the following complex picker page is put to use



Select Content

From a Course From a Concept Map Advanced Search

Select Content

From a Course From a Concept Map Advanced Search

Content of the Concept Map: Software Engineering

☐ Title	Summary
☐ Java Sockets	
☐ Java RMI	
☐ SOA Technologies	
☐ Servlets	
☐ Java basics	Introduction to the basics of the java programming language
☐ SOA Introduction	
☐ Java Applets	
☐ test content	
☐ x	

Accept Cancel

Select Content

From a Course From a Concept Map Advanced Search

Selected course: INF 2344

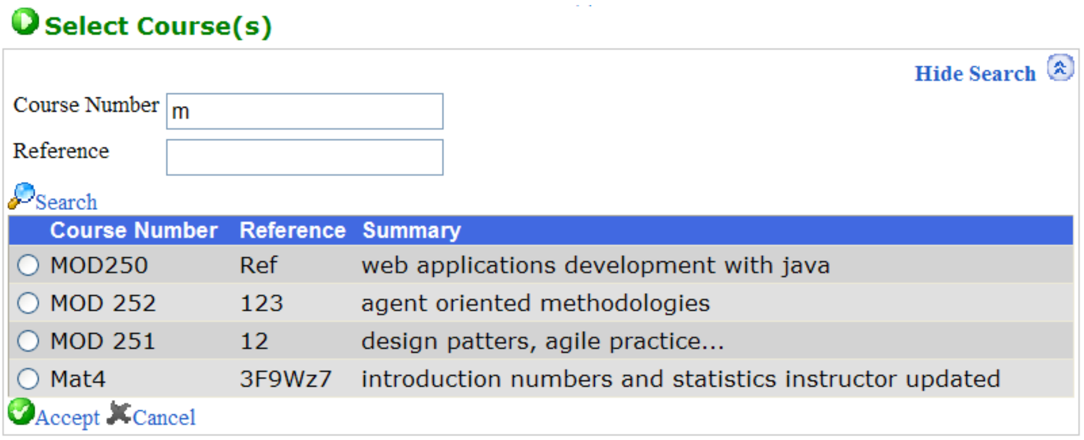
☐ Title	Summary
☐ Java basics	Introduction to the basics of the java programming language
☐ Java RMI	
☐ SOA Technologies	
☐ test content	
☐ x	
☐ test	testing of new content by morten

Accept Cancel

Figure 33 – content picker page on as an example of a complex picker page showing different picker modes

5.6.4.3 Single selection and Multi -selection picker pages

Depending on the desire of the business rule, the DCM provides two ways of picking items when working with dynamic picking.



Select Course(s)

Hide Search

Course Number: m

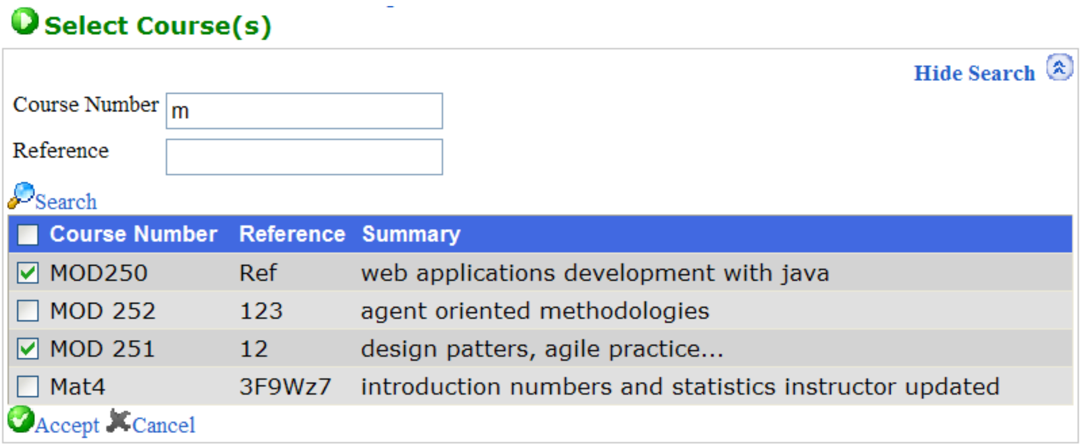
Reference:

Search

	Course Number	Reference	Summary
○	MOD250	Ref	web applications development with java
○	MOD 252	123	agent oriented methodologies
○	MOD 251	12	design patters, agile practice...
○	Mat4	3F9Wz7	introduction numbers and statistics instructor updated

Accept Cancel

Figure 34 – course picker on single selection mode



Select Course(s)

Hide Search

Course Number: m

Reference:

Search

☐	Course Number	Reference	Summary
☑	MOD250	Ref	web applications development with java
☐	MOD 252	123	agent oriented methodologies
☑	MOD 251	12	design patters, agile practice...
☐	Mat4	3F9Wz7	introduction numbers and statistics instructor updated

Accept Cancel

Figure 35 – course picker on multi-selection mode

5.7 Navigation

The word navigation implies the path followed by users of a website/web application in order to accomplish their desired task. The most important parts the user needs to be aware of while working on a site is:

- Where am I and where have I been? - The user in most occasions will lose where she is. Also, users will need to back track their paths in order to cancel a started operation or to go to a logically higher level within the site navigation. Mostly on database driven use of back operations of the browser and of no use or can give an undesired result

- Minimal choice - it is always advised to keep choices to a functionally minimal set of options

- Logical structure – the contents and items of a site/web application should be structured logically so that the user does not need any or minimal training to make best use of the system

The DCM implements its own navigation structure so that the user always knows the active page and can backtrack to any of her previous visited pages. This is done by following an object oriented approach to the pages and keeping a stack of visited pages by a navigation manager. The work flow involved is:

- Each page decides if it is to be put on the navigation stack. It makes a call to the navigation manager with its current path and a display name that appears when used. This means the navigation becomes more logical and can take data driven names

- The navigation manager searches if the page requesting registration exists on the navigation stack
    - If it exists – navigation manager removes all pages that were visited on top of it since the user has navigated back to a previously visited page
    - If it does not exist – the navigation manager registers the page on top of the stack as the last visited one

Figures 36 and 37 show sample navigation path displays

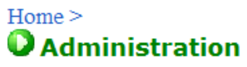


Figure 36 – administrator on home page

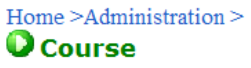
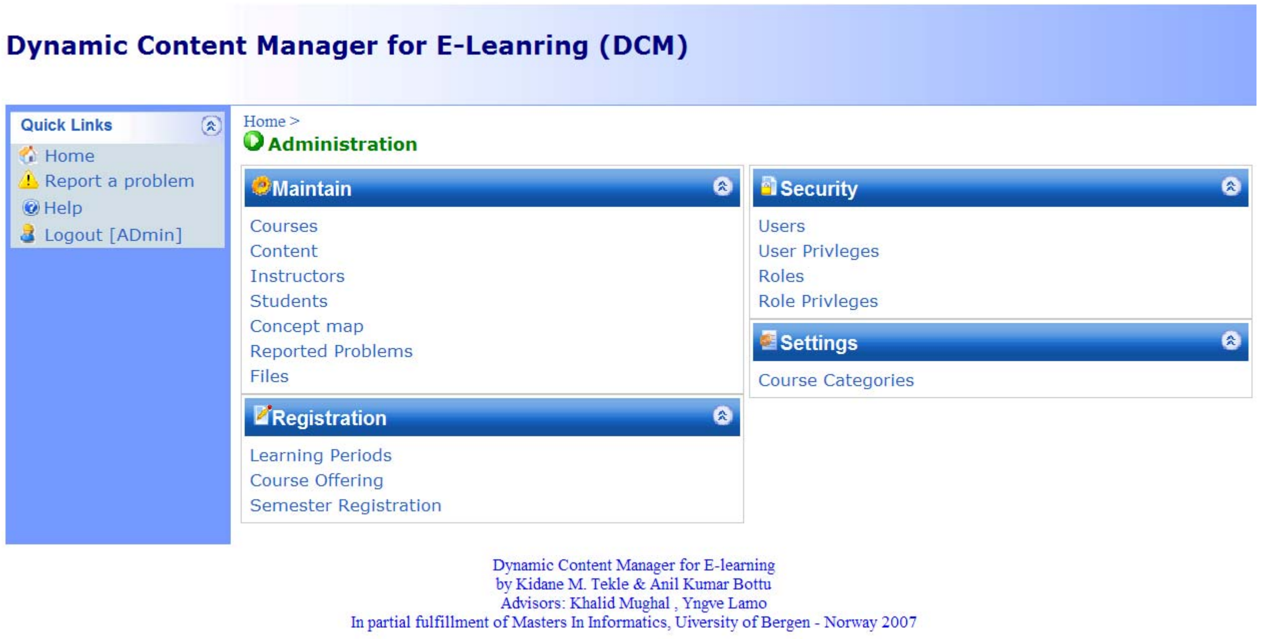


Figure 37 – administrator selected courses administration page

5.7.1. Home page of Administrator

The home page of the administrator is the stat up page for all administration operations. Administrator has full control in the system. The operations are organized so that it is easy to navigate and operate. Fig 38 shows the homepage of the administrator



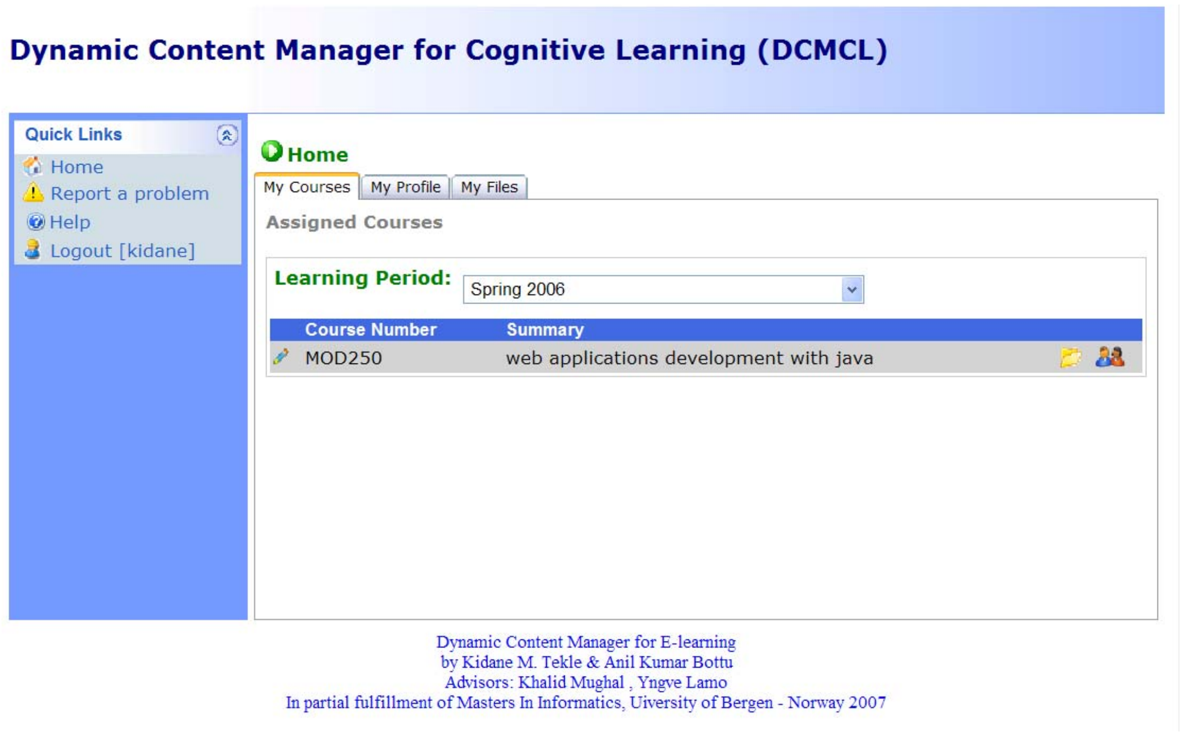
Dynamic Content Manager for E-learning
by Kidane M. Tekle & Anil Kumar Bottu
Advisors: Khalid Mughal , Yngve Lamo
In partial fulfillment of Masters In Informatics, Uiversity of Bergen - Norway 2007

Figure 38 – home page of the administrator

The operations are organized into four major groups so that it is easier for the administrator to find the desired operation easily.

5.7.2. Home page of Instructor

The home page of the instructor has very easy to use navigation capability. Fig 39 shows the home page of an instructor


Dynamic Content Manager for Cognitive Learning (DCMCL)

Quick Links
Home
Report a problem
Help
Logout [kidane]

Home

My Courses | My Profile | My Files

Assigned Courses

Learning Period: Spring 2006

Course Number	Summary
MOD250	web applications development with java

Dynamic Content Manager for E-learning
by Kidane M. Tekle & Anil Kumar Bottu
Advisors: Khalid Mughal , Yngve Lamo
In partial fulfillment of Masters In Informatics, Uiversity of Bergen - Norway 2007

Figure 39 – the home page of the instructor

The home page of the instructor has three tabs these are:

My courses – this tab shows list of courses the instructor is assigned to. The instructor selects the learning period that she wants to view courses of. By default, the current learning period is selected. On any of the courses she has been assigned to, she can do the following operations

- Edit description of course – following the icon
- Manage content of course - following the icon
- Manage students registered for the course – following the icon

My profile – this gives the instructor access to her login details and her personal details as shown in fig – 40 below

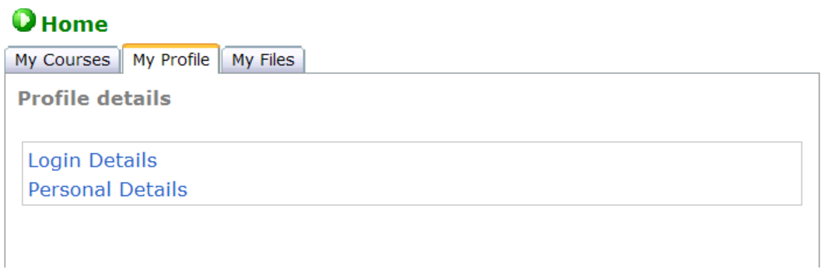
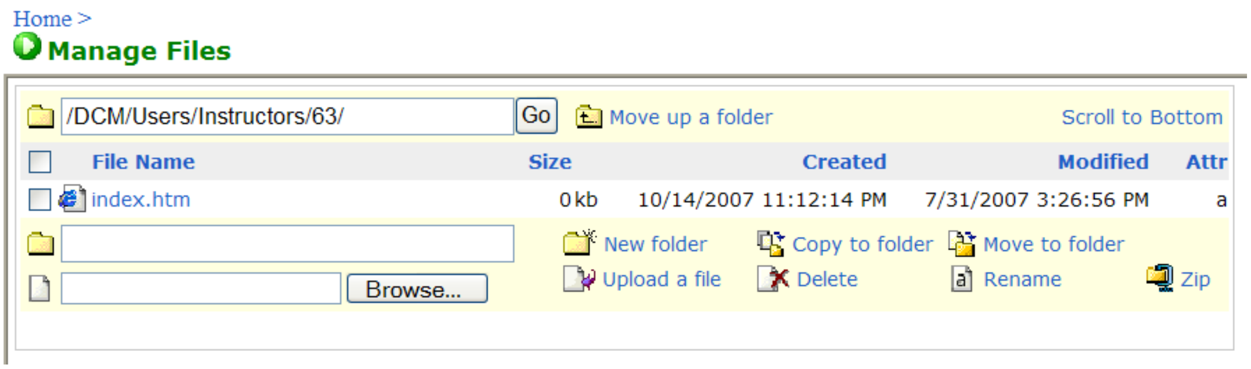


Figure 40 – my profile tab of instructor home page

My files – this tab represents the files of the instructor uploaded to the server. The system administrator of the DCM defines the maximum disk quota allowed for an instructor. Fig 41 shows the interface used to manage files by instructor

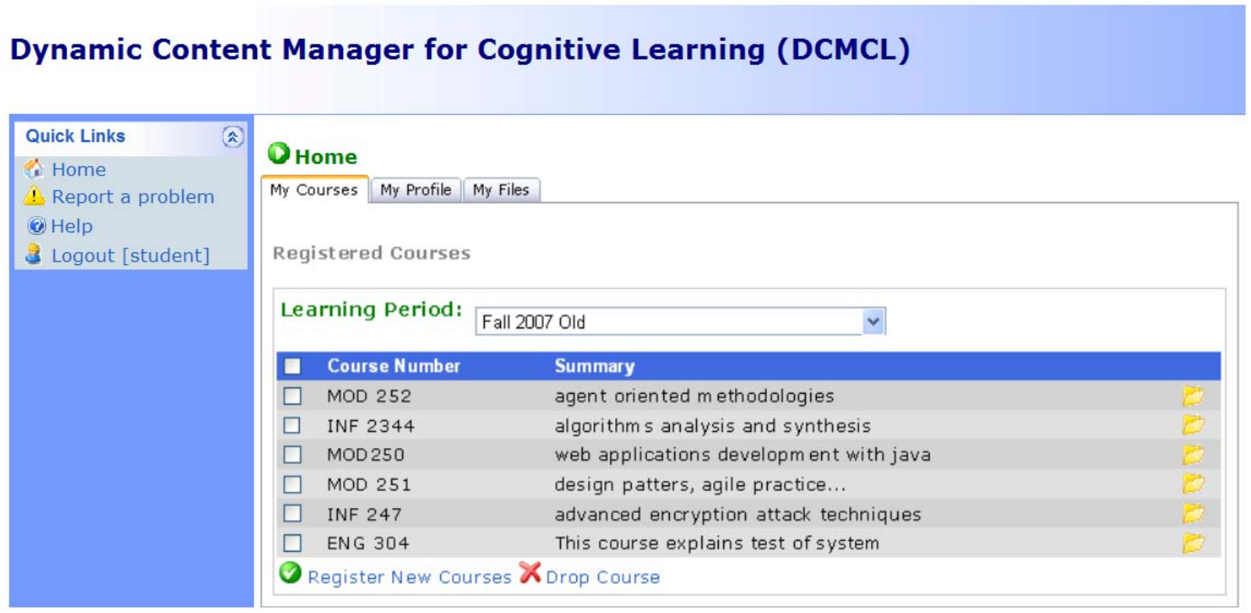


File Name	Size	Created	Modified	Attr
index.htm	0 kb	10/14/2007 11:12:14 PM	7/31/2007 3:26:56 PM	a

Figure 41- files management for the instructor

5.7.3. Home page of Student

The homepage of the student is designed so that the student does not need training to use the DCM system. Fig 42 shows the home page of the student


Dynamic Content Manager for Cognitive Learning (DCMCL)

Learning Period: Fall 2007 Old

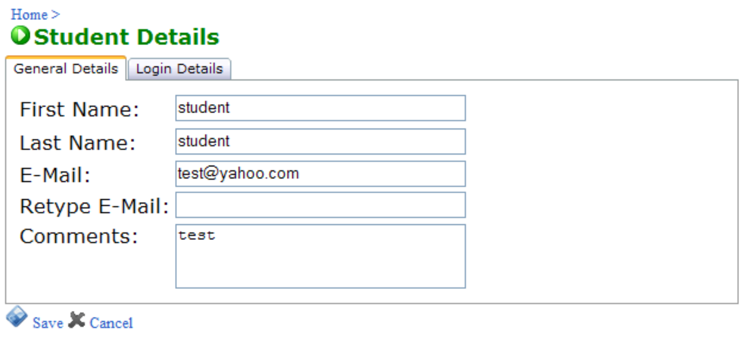
	Course Number	Summary
☐	MOD 252	agent oriented m ethodologies
☐	INF 2344	algorithm s analysis and synthesis
☐	MOD250	web applications developm ent with java
☐	MOD 251	design patters, agile practice...
☐	INF 247	advanced encryption attack techniques
☐	ENG 304	This course explains test of system

Figure 42 - **student home page**

The home page of the student has three tabs:

My courses – this is the first and most important tab of the student. The student selects the learning period that she wants to view courses of. By default, the current learning period is selected on this tab, the student can Register for new courses (only on current learning period) by following the Register New Courses menu found under the list of courses. The student can also drop from registered courses by selecting the course(s) she wants to drop out from and following the Drop Course menu found under the list of registered courses. By clicking on the icon next to a desired course, a student goes in to see the content material of the course.

My profile – this gives the student access to her login details and her personal details as shown below.


Home >

Student Details

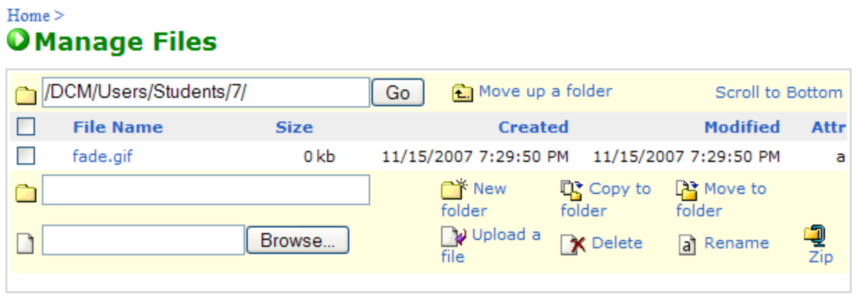
General Details | Login Details

First Name:	student
Last Name:	student
E-Mail:	test@yahoo.com
Retype E-Mail:	
Comments:	test

Save Cancel

Figure 43 – detail of student

My files – this tab represents the files of the student can save securely on her allocated space on the server. The system administrator of the DCM defines the maximum disk quota allowed for a student. Fig 44 shows the interface used to manage files by student


Home >

Manage Files

/DCM/Users/Students/7/ Go | Move up a folder | Scroll to Bottom

	File Name	Size	Created	Modified	Attr
☐	fade.gif	0 kb	11/15/2007 7:29:50 PM	11/15/2007 7:29:50 PM	a

New folder | Copy to folder | Move to folder

Browse... | Upload a file | Delete | Rename | Zip

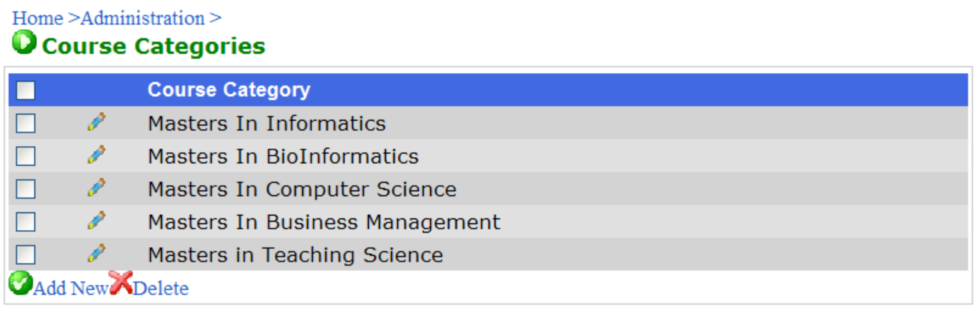
Figure 44 - files management for the student

5.8 Course category, Learning periods and Course Offering

In the DCM system, students register for various courses and go through the content material defined within. Instructors work on creating course material of the courses they are assigned to teach.

5.8.1. Course category

In the day to day learning operations, courses are categorized into different fields of study. To represent this situation, the DCM defines an entity called course category. A course category is defined by a name and courses are assigned as belonging to the category of choice.



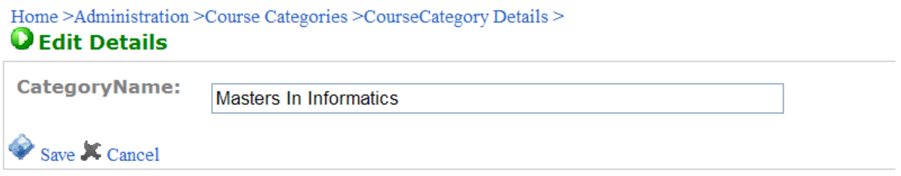
Home >Administration >

Course Categories

☐		Course Category
☐		Masters In Informatics
☐		Masters In BioInformatics
☐		Masters In Computer Science
☐		Masters In Business Management
☐		Masters in Teaching Science

Figure 45 – x list of category of courses

Being a simple entity of the system, “course category” does not have a specialized format of display and processing. Hence, the default processing, editing and viewing options described in section 5.4 are used. The following picture shows the editing of the course category.



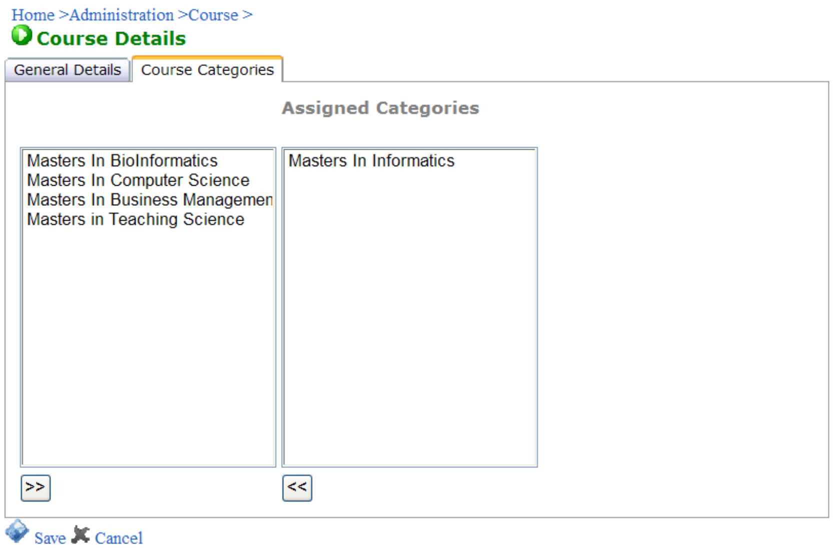
Home >Administration >Course Categories >CourseCategory Details >

Edit Details

CategoryName: Masters In Informatics

Figure 46 – editing a course category

In the DCM system, one course can be part of more than one category and most of the time; a course category contains more than one course. To be able to have this flexibility, the user interface for assignment of courses to categories is put as part of the details of a course. Fig 47 shows a sample display of the category assignment for a course.


Home >Administration >Course >

Course Details

General Details | Course Categories

Assigned Categories

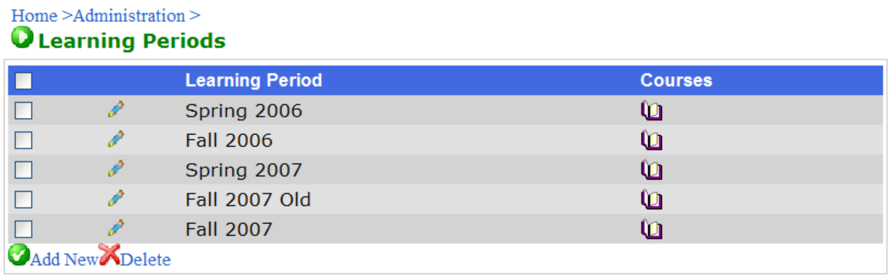
Masters In BioInformatics
Masters In Computer Science
Masters In Business Managemen
Masters in Teaching Science

Masters In Informatics

Figure 47 – assigning courses to categories

5.8.2. Learning Period

Learning periods define the different semesters that students attend courses in. In the DCM system, the following figures show the manipulation of learning periods

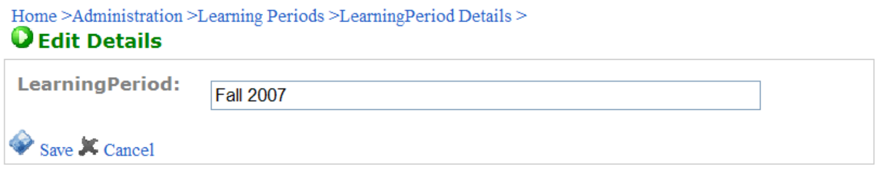

Home >Administration >

Learning Periods

		Learning Period	Courses
☐		Spring 2006	
☐		Fall 2006	
☐		Spring 2007	
☐		Fall 2007 Old	
☐		Fall 2007	

Figure 48 – learning periods

Since “Learning period” is a simple entity of the DCM system, it does not have a specialized format of display and processing. Hence, the default processing, editing and viewing options described in section 5.4 are used. The following picture shows the editing of the course category


Home >Administration >Learning Periods >LearningPeriod Details >

Edit Details

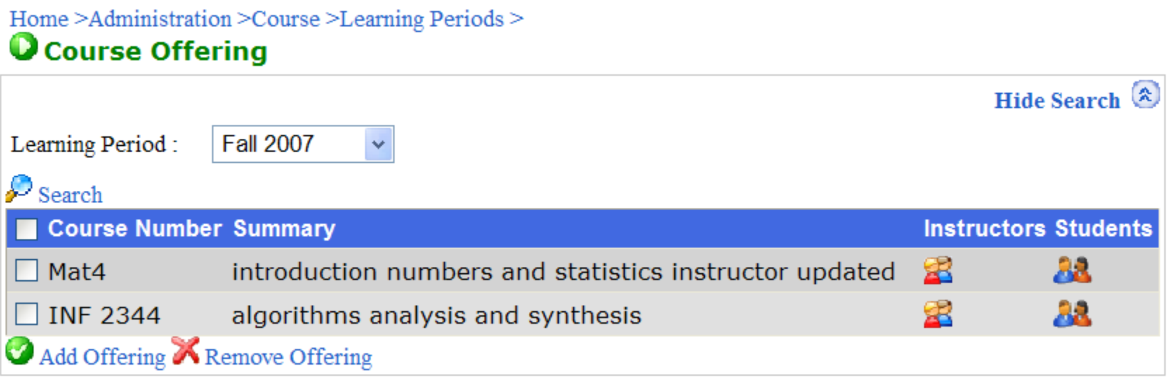
LearningPeriod: Fall 2007

Figure 49 – editing learning period details

5.8.3. Course Offering

In a learning environment, a single course can be given every semester or every other semester. It is also possible that a course is not given any more due to removal from the curriculum of the university or due to being replaced by other course(s).

The name course offering is given for the courses being taught in specific semesters. When a course is offered, it means that the course is to be given in the assigned semester and that students can register for it. The DCM administrator assigns instructors that teach the course. Fig 50 below shows the course offering startup page.



Home >Administration >Course >Learning Periods >

Course Offering

Hide Search

Learning Period : Fall 2007

Search

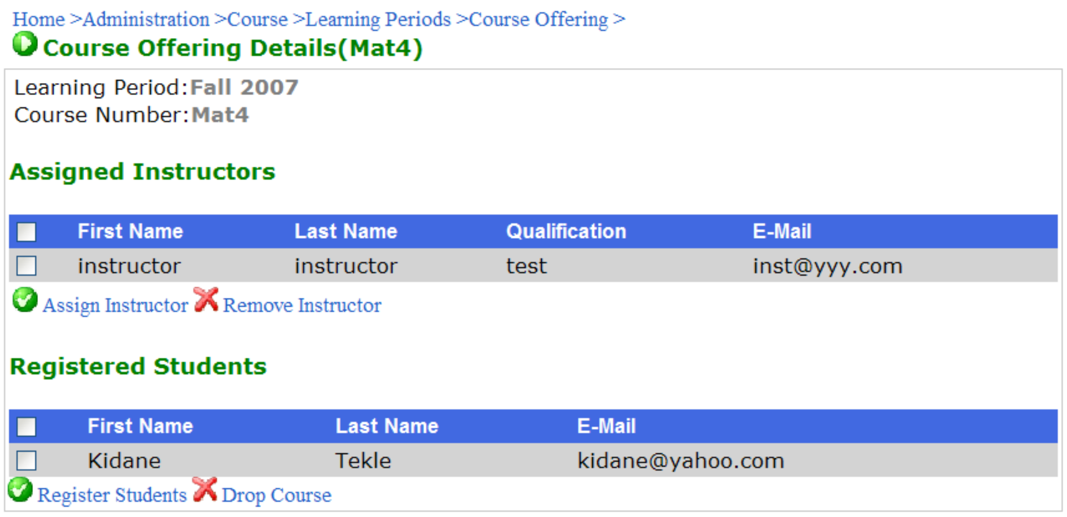
	Course Number	Summary	Instructors	Students
☐	Mat4	introduction numbers and statistics instructor updated		
☐	INF 2344	algorithms analysis and synthesis		

Add Offering Remove Offering

Figure 50 – course offering startup page

The DCM admin can add a course to be offered in the selected semester by following the Add Offering action button defined below the list of offered courses. Offered courses can be removed from offering provided that no student has been registered to that offering and that no instructor has been assigned. This is done by following the Remove Offering action button found below the list.

Assignment of instructors to courses is done by following the icon found on the row of the desired course offering. The instructors' assignment page and also an alternative student registration page for the system administrator are shown below:



Home >Administration >Course >Learning Periods >Course Offering >

Course Offering Details(Mat4)

Learning Period:Fall 2007
Course Number:Mat4

Assigned Instructors

☐	First Name	Last Name	Qualification	E-Mail
☐	instructor	instructor	test	inst@yyy.com

Assign Instructor Remove Instructor

Registered Students

☐	First Name	Last Name	E-Mail
☐	Kidane	Tekle	kidane@yahoo.com

Register Students Drop Course

Figure 51 – instructors' assignment and student registration page for a course offering

From this instructors' assignment and students' registration page, the system administrator can perform the tasks of assigning new instructors to teach the offered course or remove instructors from their assignment.

Also, this page gives the DCM system administrator to be able to manage students registered in the specified semester. These operations are realized by following the ✓ Register Students link for registration and the ✗ Drop Course link for removing students from their registration if desired.

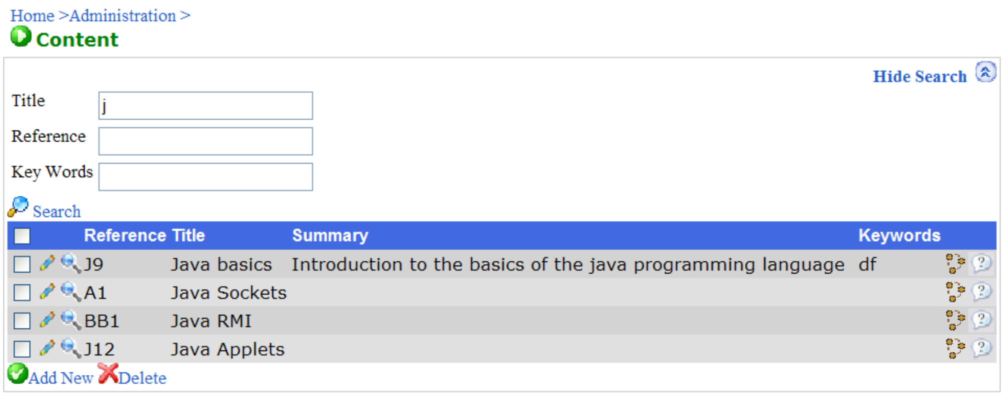
5.9 Content and Course Management

The DCM system focuses on defining atomic knowledge elements (given the name Content in the Project). These knowledge elements are to be reused and further enhanced as the material to be presented evolves through time.

The DCM supports advanced online html pages creation so that the educator can create learning material that contains text, links to external resources, images, flash, video and other content. Following, a description of the implementation for course and content (knowledge element) provided by the DCM system will be given.

5.9.1 Content Management

In the DCM system, content can be created through different paths. The first and straight forward way of creating and managing content is through the "Content Management" interface. This interface is a dynamic process page (see section 5.6.1 for detail) that manages content. Un-desired and un-used content can only be removed from the system through this interface. Fig 52 shows the content management interface.



Title	j
Reference	
Key Words	

Reference	Title	Summary	Keywords
J9	Java basics	Introduction to the basics of the java programming language	df
A1	Java Sockets		
BB1	Java RMI		
J12	Java Applets		

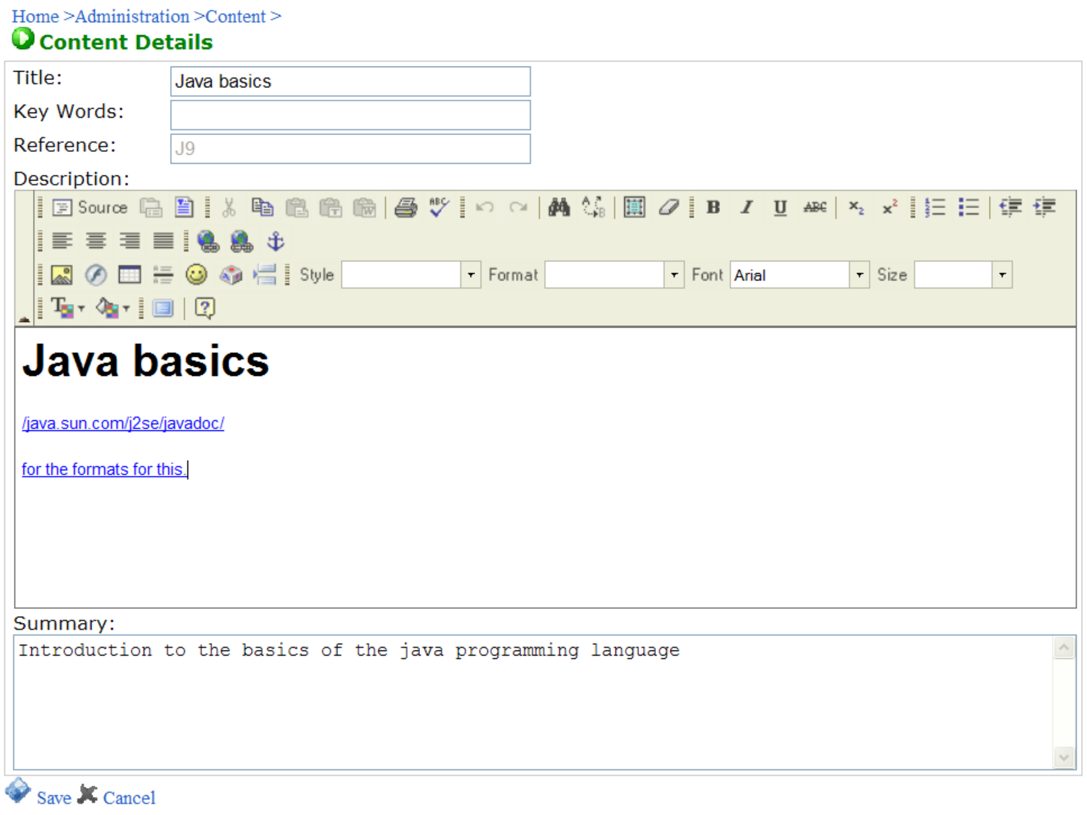
Figure 52 – content management interface

In fig 52, the icon takes the user to a page showing the concepts related to that content (see section 5.10) and the icon takes to questions management page for the desired content.

In addition to the content management interface, new content can be defined while defining learning material for a course (see section 5.9.2) and while managing content within a concept map (see section 5.10).

Editing content details

A content being the basic element of knowledge and crucial for reuse, is given a system generated reference number so it can be identified and searched easily. The content detail editing page of the DCM system is shown in fig 53


Home >Administration >Content >

Content Details

Title: Java basics

Key Words:

Reference: J9

Description:

Java basics

/java.sun.com/j2se/javadoc/

for the formats for this.

Summary:

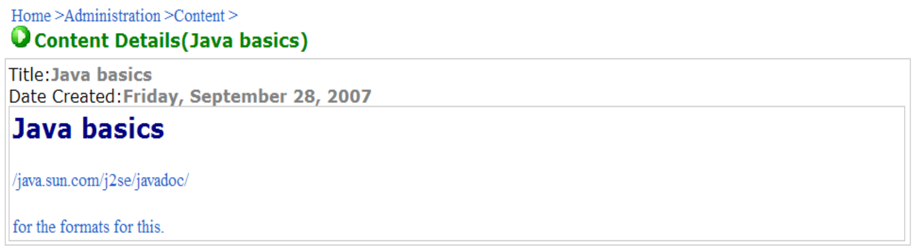
Introduction to the basics of the java programming language

Figure 53 – content detail editing page

The content creator can define key words that can be used in searches when trying to reuse existing content. The descriptions section is an advanced html editor where the user can create any rich html document containing the desired content and formatting. The DCM supports text, links, images, video, flash, documents, presentations and other file types during the development of content details.

Viewing content

Students are given only view privilege to the content created by the educator. For this reason, the DCM provides a content view page that shows the rich html content developed by the instructor. Fig 54 shows a sample view of one content


Home >Administration >Content >

Content Details(Java basics)

Title:Java basics

Date Created:Friday, September 28, 2007

Java basics

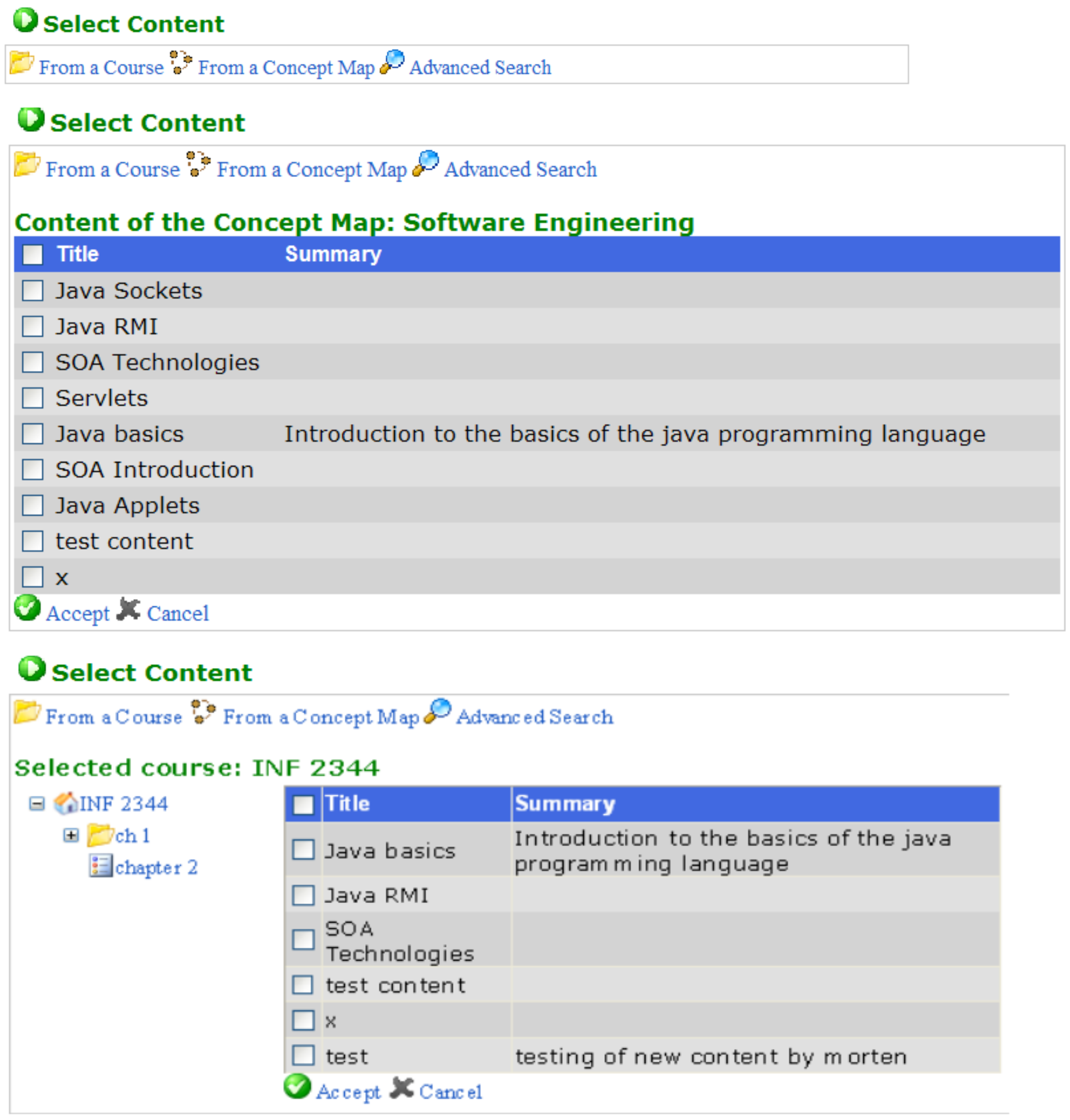
/java.sun.com/j2se/javadoc/

for the formats for this.

Figure 54 – sample view of an instructor created content

Selecting content for reuse

Content reuse is one of the key concepts that are at the heart of the DCM project. In the DCM system, courses are modeled as a structured organization of content and concepts are defined to be arbitrary aggregations of content elements. Hence, an easy to use content selection and reuse mechanism is desired. The DCM content selector has three modes of operation. The first mode is through an advanced search from the content elements. This is a generic dynamic picker implementation (see section 5.6.1 for details). The second and third modes of operation enable the user to select content from existing concept maps and from those that have already been assigned to existing courses.



Select Content

From a Course From a Concept Map Advanced Search

Select Content

From a Course From a Concept Map Advanced Search

Content of the Concept Map: Software Engineering

☐ Title	Summary
☐ Java Sockets	
☐ Java RMI	
☐ SOA Technologies	
☐ Servlets	
☐ Java basics	Introduction to the basics of the java programming language
☐ SOA Introduction	
☐ Java Applets	
☐ test content	
☐ x	

Accept Cancel

Select Content

From a Course From a Concept Map Advanced Search

Selected course: INF 2344

INF 2344
- ch 1
- chapter 2

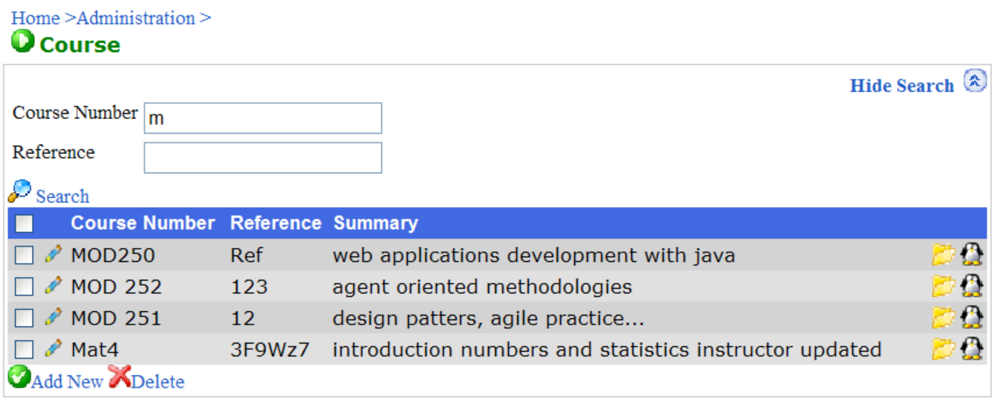
☐ Title	Summary
☐ Java basics	Introduction to the basics of the java programming language
☐ Java RMI	
☐ SOA Technologies	
☐ test content	
☐ x	
☐ test	testing of new content by morten

Accept Cancel

Figure 55 – content picker for reuse

5.9.2 Course Management

A course is created and managed through the "Course Management" interface. The course management page provided to the administrator is a dynamic processing page where she can create new and remove courses. Fig 56 gives a display of the course management page with courses filtered by a beginning letter of the course number.



Course Number	m
Reference	

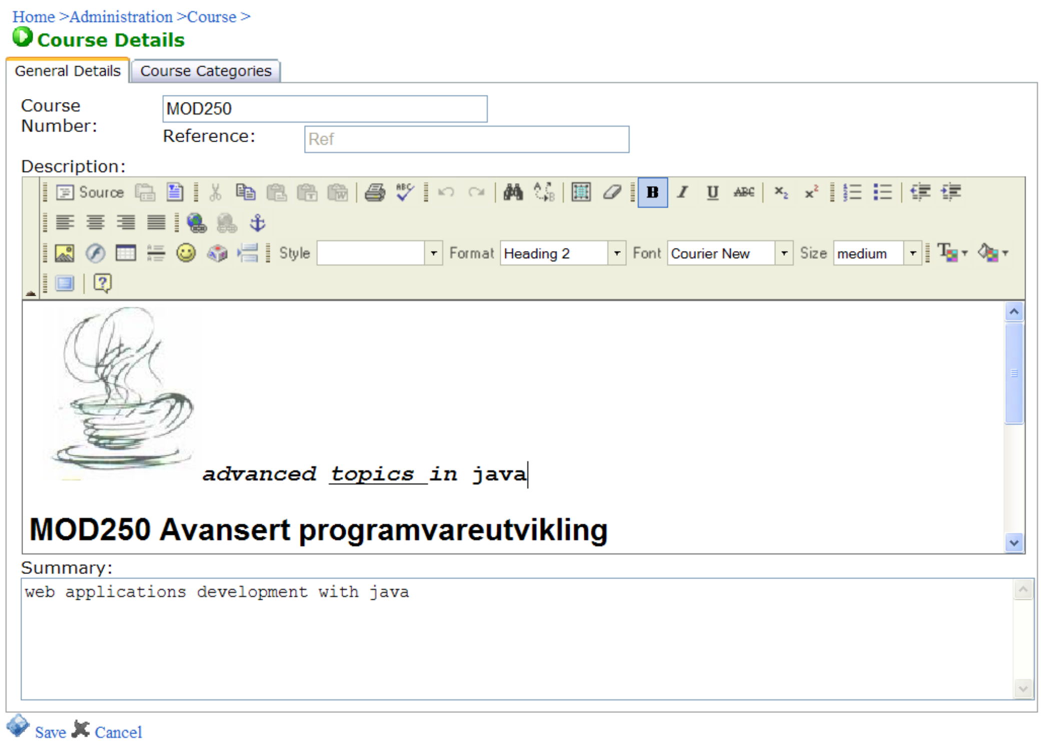
	Course Number	Reference	Summary
☐	MOD250	Ref	web applications development with java
☐	MOD 252	123	agent oriented methodologies
☐	MOD 251	12	design patters, agile practice...
☐	Mat4	3F9Wz7	introduction numbers and statistics instructor updated

Figure 56 – course management interface

The course management page also gives possibilities for the system administrator to manage contents of the course (by following the icon) and to manage quizzes related to the desired course (by following the icon)

Editing course details

A course is defined by a unique course number and course title that is being used in the university and a system generated reference number. It also contains sections for detail description and summary. In the DCM system, courses are created only by the system administrator. Educators are given the privilege of updating the description and summary of the course but not the course number and course title. Fig 57 shows the editing of a course


Home >Administration >Course >

Course Details

General Details | Course Categories

Course Number: MOD250

Reference: Ref

Description:

advanced topics in java

MOD250 Avansert programvareutvikling

Summary:

```
web applications development with java
```

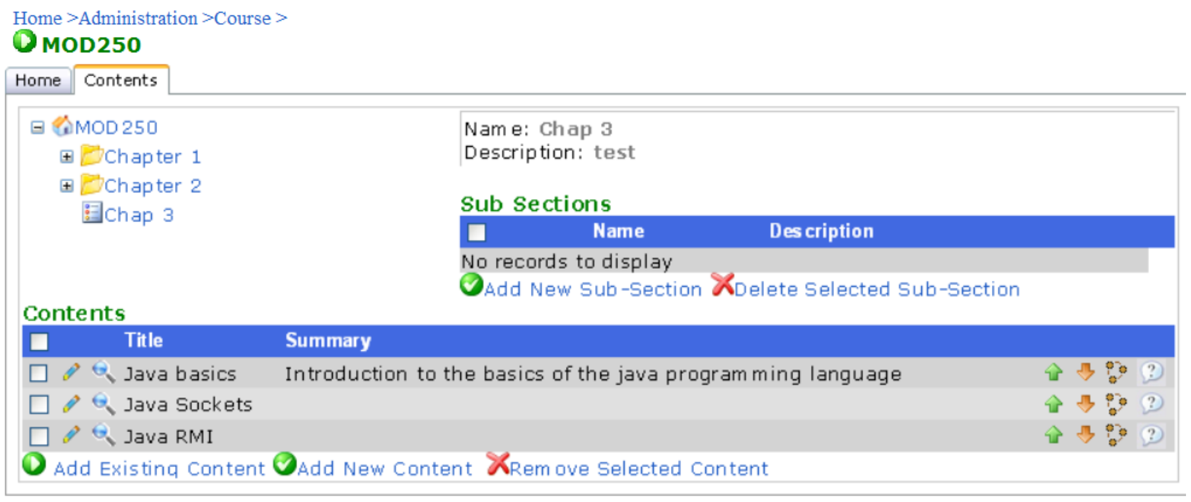
Figure 57 – editing of a course

The description section of the course is an advanced html editor that supports text, images, links, flash, video and other file types and gives the instructor full control of the look and feel of the description she develops.

Creating and managing course material

In the DCM system model, a course is a structured hierarchical aggregation of content materials. A course is composed of sub sections. And in turn these sub-sections can contain other sub-sections of their own. This gives a tree like view for the sections management. It can also be viewed as a "table of contents" view of the contents of the course.

A section contains one or more content elements depending on its complexity and content coverage. Fig 58 shows a sample course content tree display with the node chapter 3 selected. Note that chapter 3 has no sub-sections and contains three content elements to be displayed in the order shown.



Home >Administration >Course >

MOD250

Home | Contents

- MOD 250
  - Chapter 1
  - Chapter 2
  - Chap 3

Name: Chap 3
Description: test

Sub Sections

	Name	Description
No records to display		

Add New Sub-Section Delete Selected Sub-Section

Contents

	Title	Summary
☐	Java basics	Introduction to the basics of the java programming language
☐	Java Sockets	
☐	Java RMI	

Add Existing Content Add New Content Remove Selected Content

Figure 58 – course material creation and management window

Privilege to the course creation and management window is limited to the DCM system administrator or to instructors that are assigned to teach the course. From within this window, the educator (or administrator) can manage the sections tree by using the Add New Sub-Section and the Delete Selected Sub-Section menu items to add and remove sub-sections respectively. A sub-section is allowed to be removed only if it does not contain any content elements.

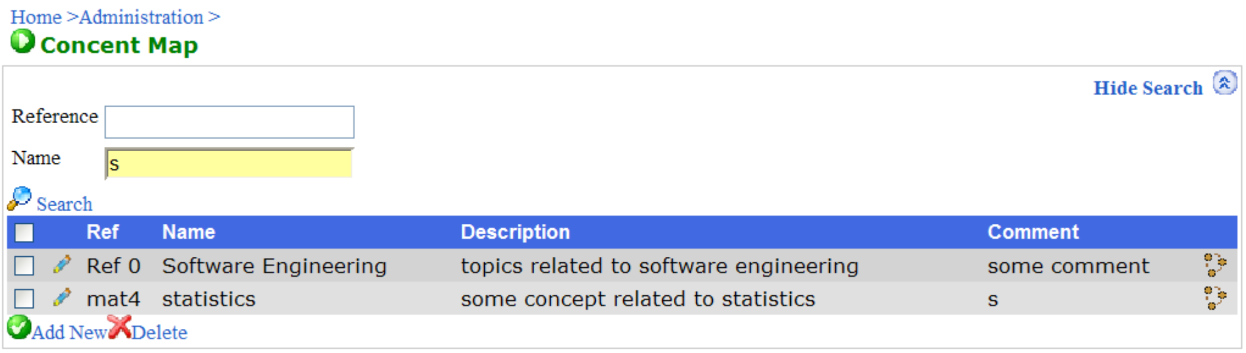
The content material of a sub-section is managed from the list shown at the bottom of the page of fig 58. here the educator can add new content or fetch existing content by use of the content picker by following the Add New Content and Add Existing Content links respectively. By using the up and down arrows, the educator can re-arrange the order by which the content elements are presented to the student.

The icon takes the user to a page showing the concepts related content defined by the row (see section 5.10) and the icon takes to its questions management page.

5.10 Concept map Implementation

In the DCM system, a concept map is an arbitrary aggregation of knowledge elements that can be defined by an educator to model a set of related contents. It is a means to give an easy navigation of the knowledge elements from the repository so as to achieve a much higher level of reuse.

The concept map management window gives the administrator with administration power over the different concepts. This page is a dynamic processing page (see section 5.6.1 for details) and provides an easy to use way of filtering, adding new and removing concepts from the system.


Home >Administration >

Concent Map

Hide Search

Reference

Name s

Search

	Ref	Name	Description	Comment
☐	Ref 0	Software Engineering	topics related to software engineering	some comment
☐	mat4	statistics	some concept related to statistics	s

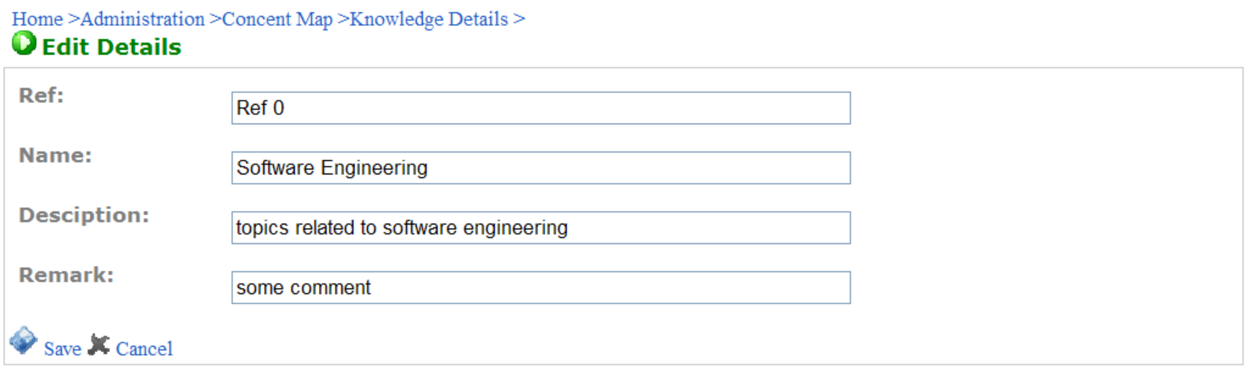
Add New Delete

Figure 59 – concept map management window

The concept map management window is the starting point for system administrators to define concepts by following the Add New menu element and removal of un-desired concepts by following the Delete menu. Concepts are removed only if there are no contents defined in it.

Detail of a concept

When a concept is created, the system generates a reference that is unique so that it can easily be located. The creator of the concept provides a name of her desire and a description as well as an optional remark. Fig 60 shows the editing page of a concept.



Home >Administration >Concent Map >Knowledge Details >

Edit Details

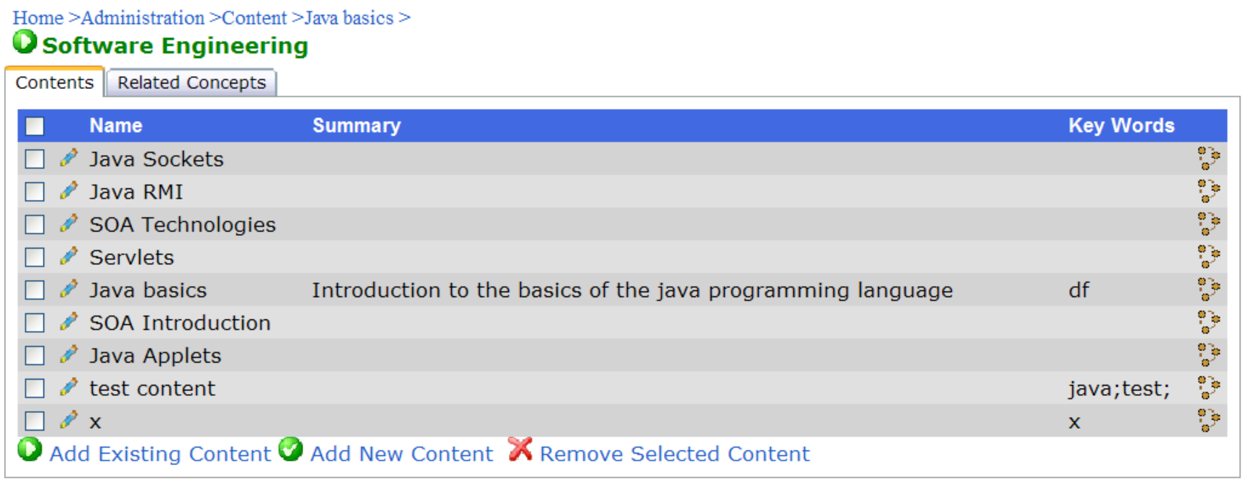
Ref:	Ref 0
Name:	Software Engineering
Description:	topics related to software engineering
Remark:	some comment

Save Cancel

Figure 60 – **editing a concept**

Concept contents

The concept as an arbitrary aggregation of content material is realized by clicking on the icon next to the desired concept from the concept management window shown in fig 59. Fig 61 below shows the content list manipulation window that results for the selected concept.



Home >Administration >Content >Java basics >

Software Engineering

Contents | Related Concepts

	Name	Summary	Key Words
☐	Java Sockets		
☐	Java RMI		
☐	SOA Technologies		
☐	Servlets		
☐	Java basics	Introduction to the basics of the java programming language	df
☐	SOA Introduction		
☐	Java Applets		
☐	test content		java;test;
☐	x		x

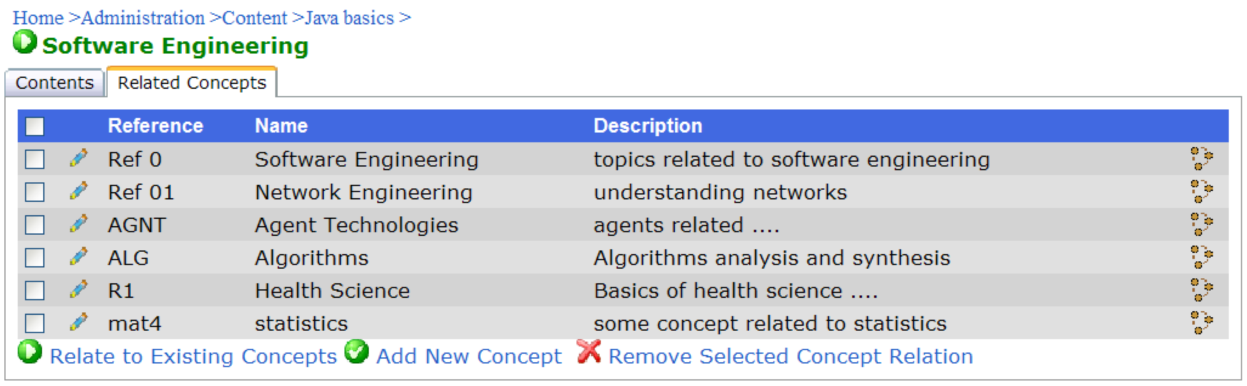
Add Existing Content Add New Content Remove Selected Content

Figure 61 – **content list of a concept**

From this list of content view of a concept, the educator or system administrator can add new content, add by selecting from existing content or remove a desired content from the listing by following the Add Existing Content , Add New Content and Remove Selected Content menu items consecutively. When removing contents from this listing, only their relation to the current concept is removed from the system so that the content still exists in the repository for later reuse and other usages of the content stay unaffected. When adding new content from this window, the new content is added by the system to the content repository and it is also added as having relation to the current concept.

Related concepts

In addition to consisting of content elements, a concept can have relation with other concepts which realizes the structure of the DCM solution model. To realize this, the educatory or system administrator has to use the “Related concepts” tab from the detail manipulation page of the concept. Fig 62 shows the implementation for this functionality


Home >Administration >Content >Java basics >

Software Engineering

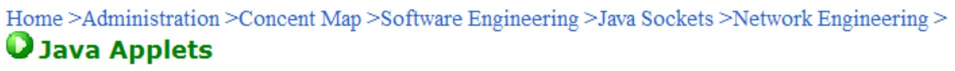
Contents | Related Concepts

	Reference	Name	Description
	Ref 0	Software Engineering	topics related to software engineering
	Ref 01	Network Engineering	understanding networks
	AGNT	Agent Technologies	agents related ....
	ALG	Algorithms	Algorithms analysis and synthesis
	R1	Health Science	Basics of health science ....
	mat4	statistics	some concept related to statistics

Relate to Existing Concepts Add New Concept Remove Selected Concept Relation

Figure 62 – related concepts manipulation of selected concept

From the related concepts manipulation page shown in fig 62, the educator or system administrator can relate the current concept with other concepts add new concept and related it to the current one or remove the current content’s relation from the list by following the Relate to Existing Concepts, Add New Concept and Remove Selected Concept Relation menu items respectively. Following the icon on any of the related concepts’ rows takes to the detail of the contents and related concepts for the selected concept. This gives an arbitrary navigation since the user can navigate from one concept to another and drill down to any of the content details she wants. The navigation link on the top of the page keeps track of the paths the user has navigated in case she wants to go back to any of her previously viewed in a manner shown in the figure below

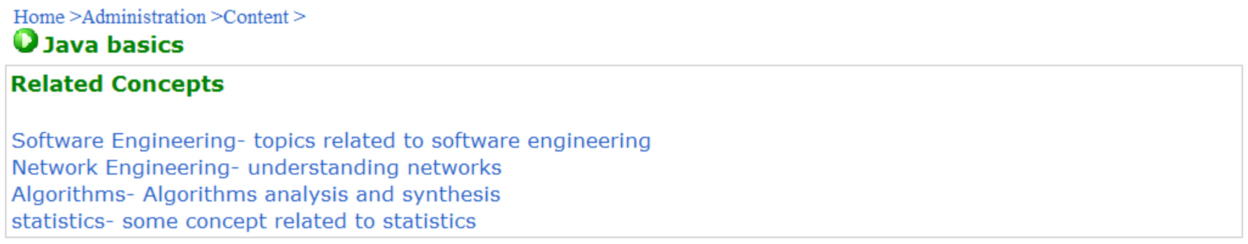

Home >Administration >Concent Map >Software Engineering >Java Sockets >Network Engineering >

Java Applets

Figure 63– sample navigation link display of some navigation of concepts and content elements

Concepts from a content element

The DCM also provides the navigational path to go up from within a specific content element to see concepts by which it is contained. Such a navigational facility greatly facilitates the navigational capability of the knowledge elements in the repository via a


Home >Administration >Content >
Java basics
Related Concepts

Software Engineering- topics related to software engineering
Network Engineering- understanding networks
Algorithms- Algorithms analysis and synthesis
statistics- some concept related to statistics

Figure 64 – view of concepts that contain the selected knowledge element

6. Summary and Conclusion

6.1 Status of the System

Development of the core functionalities of the DCM system was finalized and the system was deployed for testing in the fall semester of 2007. The course MOD 250 (modern software development techniques) was selected as a test course with a class of 25 students and two instructors. This sample usage of the DCM system served as a proof of concept for the logic behind the DCM solution and as a firsthand experience of usage in a real learning environment. It was observed that users did not need any training to start using the system.

6.2 Further Work

The DCM system will need to undergo some iteration through the incorporation of new requirements and the enhancement of implemented functionality. Currently, more emphasis is given to the realization of the core system functionalities and to achieving initial operational capability. There still remain lots of subsidiary functionality requirements that need to be implemented if the DCM system is to be used to its full potential. Following, some of the requirements that could be incorporated into the DCM system are described.

6.2.1 Student content coverage follow-up

Student content coverage follow up is very essential for the overall quality of the learning process. By incorporating this feature, the DCM system could give the educator more information on the content coverage of the student so that she can be able to prepare better suited learning material. It also enables the identification of learning trends and patters and could be a source for further analysis and research on student learning behavior.

6.2.2 University structure awareness

Currently, the DCM system is designed as a single installation, single deployment system. It uses a common students and instructors model without giving much emphasis to the structure of the university or the learning environment. The users modeling and security structure of the DCM system could be further enhanced so that it takes into account the different faculties and departments and the enrollment of students. Also, the DCM system could be enhanced so that it can be deployed in more than one institution with a regulated and standardized mode of content sharing.

6.2.3 More graphical presentation

The DCM system is designed in a way so that more enhancements can be done to the way the learning material is presented and updated. The inclusion of multiple views to a course is one graphical enhancement that could make the DCM more useful and user friendly to the learner. Weekly views, summary and categorized views of the learning material could be presented to the user in addition to the table of content view that is currently implemented in the system. A richer graphical presentation needs to be incorporated in the navigation and display of concept maps. Graphical representation of such a dynamic and arbitrary aggregation of knowledge elements is quite challenging and is a good area for further research and development.

6.2.4 Integration with other systems

The DCM system captures only the core operations and components of the content based, reusable E-Learning. There are lots more functionalities that need to be incorporated into the system if it is to be mature enough to be used in a large scale learning environment.

The DCM system is designed so that it can be integrated with other systems and use existing functionalities. One proposed integration mode is the implementation of security and utilities interface so that it can be integrated into existing popular content management systems (CMS). By integration with existing content management systems, the DCM benefits the usage of functionalities like: discussion boards, blogs, news, photo galleries, publications management and much more. Dotnetnuke (DNN) [19] is one of the most promising Content Management Systems into which the DCM can be integrated with relative ease.

6.2.5 Information exchange

There are many software systems that are put to use in the learning environment. As such, much information and resources are found distributed throughout these systems. Being able to extract information from existing systems and to make its resources and functionalities available to others is one measure of maturity and usability of a system. The DCM can be further enhanced so that it accepts students' information from dedicated student record management systems. Also, interfacing the DCM with systems currently being used for conduction of learning (MySpace in the case of University of Bergen) is one direction of enhancement to achieve more flexibility and usage.

6.2.6 Student modeling

The current DCM system uses very elementary modeling for students learning. It puts the student in a default category with regards to a certain course at first and the instructor updates the level of the student based on her assessment. We believe that the learning environment and the interactions within could be best modeled by employing the concepts of agent oriented software engineering. The learning environment could be modeled as an adaptive, collaborative multi-agent system [18]. Application of multi-agent system modeling and development in the area of e-learning is still at its infancy. Much work could be done on in this direction as a further enhancement of the DCM system.

6.3 Conclusion

Most of the commercial e-learning systems of today have a lot of facilities, but lack a solid underlying pedagogical structure. This means that the major challenge for e-learning systems is to develop a pedagogical structure of the system and develop a more structured and sound modeling – not to develop further its technical functionality.

Development of the DCM system has demonstrated its underlying foundation and has matured enough to be put to initial use in the short time period since its staring. It is developed using state of the art technology and developed following the most advised development techniques. Thus, it is very open for extension and further development to incorporate more functionalities and refinement. Also, provision for its integration with content management systems (CMS) and information exchange with other systems gives some directions by which the DCM can be made more useful.

I believe that open, flexible and interactive web-based learning in the future best can be achieved by using different types of intelligent agents supporting the users in the learning process [17, 18]. In such systems an intelligent agent, representing the learning profile of the users, could model each user. User profile agents collaborate with other agents to get the best-adapted learning materiel for each student. This means that the learning process can be realized as collaboration between different agents and users, constructed as a multi-agent system

References

[1] http://lyon.chin.gc.ca/tael-pte/module1/m01t03p01_e.asp - date accessed 02.11.2007

[2] Kristensen,T. The “Gudmunstad” School Project. Proceedings of The Fourteenth International Conference on Technology and Education, vol 2, Oslo 1997.

[3] Kristensen, T. The “Gudmundstad” School Project. The Learning Highway. Editors Owen,T., Owstone,R. Key Porter Books, Toronto, Ontario, 1998.

[4] http://www.itsolutions.no

[5] http://moodle.org/

[6] James Robertson , Content reuse in practice , Step Two Designs Pty Ltd - September 2004

[7] Joseph D. Novak , Alberto J. Cañas: The Theory Underlying Concept Maps and How to Construct Them

[8] Raymund Sison, Masamichi Shimura, Student Modeling and Machine Learning International Journal of Artificial Intelligence in Education (1998), 9, 128-158

[9] Karen Stauffer, Student Modeling and Web-based Learning Systems , Athabasca University http://ccism.pc.athabascau.ca/html/students/stupage/Project/initsm.htm

[10] Joseph D. Novak , Alberto J. Cañas: The Theory Underlying Concept Maps and How to Construct Them

[11] Ivar Jacobson, Magnus Christerson, Patrik Jonsson, Gunnar Overgaard Object-Oriented Software Engineering: A Use Case Driven Approach (ACM Press) Addison-Wesley, 1992, ISBN 0201544350

[12] I. Jacobson, M. Ericsson, A. Jacobson, The Object Advantage: Business Process Reengineering With Object Technology (ACM Press), 1994, Addison-Wesley, ISBN 0201422891

[13] Bernd Bruegge. Object-Oriented Software Engineering: Conquering Complex and Changing Systems, Prentice Hall, 1999. ISBN 0-13-489725-0

[14] Bernd Bruegge, Allen H. Dutoit. Object-Oriented Software Engineering: Using UML, Patterns and Java, Pearson Education, 2004. ISBN 0-13-191179-1

[15] Ian Sommerville, Software Engineering ,7th Edition

[16] http://www.agilealliance.org/articles

[17] Shoham, Y.. An Overview of Agent-Oriented Programming. In Jeffrey M. Bradshaw, editor, Software Agents. AAAI Press and The MIT Press, 1997.

[18] Michael Wooldridge ,An Introduction to Multiagent Systems. Published in February 2002 by John Wiley & Sons, ISBN 0 47149691X.

[19] http://www.dotnetnuke.com/