

## **INFORMATION TO USERS**

**This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.**

**The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.**

**In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.**

**Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.**

**Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.**

**Bell & Howell Information and Learning  
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA**

**UMI<sup>®</sup>**  
**800-521-0600**



# **Distance Education on the World-Wide Web**

*by*

**Simone Spiller**

**School of Computer Science**

**McGill University, Montreal**

**November 1997**

**A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH IN PARTIAL  
FULFILLMENT OF THE REQUIREMENTS OF THE DEGREE OF**

**MASTER OF SCIENCE**

**Copyright © Simone Spiller, 1997**



National Library  
of Canada

Acquisitions and  
Bibliographic Services

395 Wellington Street  
Ottawa ON K1A 0N4  
Canada

Bibliothèque nationale  
du Canada

Acquisitions et  
services bibliographiques

395, rue Wellington  
Ottawa ON K1A 0N4  
Canada

*Your file Votre référence*

*Our file Notre référence*

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-44287-X

**Canada**

# Abstract

The Internet's growing acceptance as a key medium to educate people is the main focus of this thesis. Although distance education has been a popular learning method since the end of the 19<sup>th</sup> Century, never before has technology made it so easy to disseminate knowledge by linking different media formats – e.g. text, sound, video. In fact, this huge library accessed by a universal interface, is one of the key contributions that the Internet and the World-Wide Web have brought to the learning process.

The main goal of this thesis is to encourage instructors to create and deliver courses using the Internet and, most of all, to show that the process can be simple and effective. In order to support this study, four major Course Management tools are presented and analyzed: Pathway by Solis-Macromedia, LearningSpace by Lotus, WebCT by The University of British Columbia, and Virtual-U by Simon Fraser University.

As a result of this thesis, a Grades Application was developed using the Internet protocol. This application is an uncomplicated, yet effective solution for using the Web to manage, calculate, and view students' marks. With the open architecture of the Web and standard programming languages such as JavaScript and Perl, the system will execute in most computers available in universities around the world.

# Résumé

L'utilisation de plus en plus fréquente de l'Internet comme outil d'apprentissage est le thème principal de cette thèse. Même si l'éducation à distance est un outil utilisé depuis la fin du 19<sup>e</sup> siècle, c'est grâce aux nouvelles technologies de l'information et des communications que la dissémination du savoir est devenue une réalité. En combinant images vidéo, sons et textes sous une seule interface-utilisateur, l'Internet et, plus particulièrement son volet multimédia – le Web, ont apporté une contribution significative à l'éducation.

Le but principal de ce document est d'encourager les enseignants à se servir de ces nouveaux outils pour la création et la livraison de cours, ainsi que de leur démontrer comment ce processus peut être, en même temps, simple et efficace. Pour illustrer cet étude, quatre outils de Gestion de cours sont présentés: Pathway de Solis-Macromedia, LearningSpace de Lotus, WebCT développé à l'Université de la Colombie Britannique et Virtual-U par l'Université Simon Fraser.

Comme produit de cette thèse, une application concrète pour la gestion de notes a été développée. Malgré son caractère peu compliqué, cette application offre une solution efficace pour la gestion, le calcul et l'affichage des notes. En utilisant l'architecture ouverte de l'Internet, ainsi que des langages de programmation standards tels que JavaScript et Perl, le système peut être exécuté sur la plupart des ordinateurs de la majorité des universités au monde.

*Para Silvino, minha cara metade e  
aos meus pais por tudo...*

# Acknowledgements

First, I would like to express my deep gratitude to my supervisor and guru, Professor Gerald Ratzer who supported and provided me with ideas, experience, encouragement, and inspiration to accomplish this thesis.

Second, I want to thank Silvino Mezzari Júnior, my partner in life, who read and re-read, discussed, listened, enlightened, and much more, the many long evenings and weekends I (and as a consequence, we) dedicated to this thesis. Third, my dear friend Lynne Chalmers who also read and re-read this work and who helped with suggestions, comments, coffee, and even goodies.

Un gros merci à Charles Boulos et Michèle Carrier pour les suggestions et la révision du résumé!

Um baita obrigada ao meu manão, Angelo Spiller e à mana Andréia Luz (soon to be Spiller), for the word "tesanda" and their support.

Indirectly, as part of my daily motivation, some professional women inspire my work and accomplishments: my grandmother, my mother, and three other women who are also a source of inspiration to my career: Maria Janilce Almeida, Angela Brodbeck, and France Nicole.



# Contents

<b>Abstract</b>	<b>i</b>
<b>Résumé</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>iv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation	1
1.2 Thesis Contributions	3
1.3 Thesis Layout	4
<b>2 Distance Education on the Web</b>	<b>6</b>
2.1 The Waves of Distance Education	6
2.2 Web-Based Training Universe	7
2.3 Potential Advantages of WBT	8
2.3.1 WBT is Easy to Build (and Maintain)	9
2.3.2 WBT is Dynamic	10
2.3.3 WBT is Interactive and Collaborative	10
2.3.4 WBT is Platform Independent	11
2.4 Possible Disadvantages of WBT	12
2.5 Examples	12
2.5.1 Computers in Engineering, McGill University	13
2.5.2 Online English 101, Spokane Community College	13
2.5.3 Environmental Law, Manhattan College	14
2.5.4 Internet Resources, McGill University	14
2.5.5 Other References	15
<b>3 Commercial Solutions for Web-Based Training</b>	<b>17</b>
3.1 Overview	17
3.2 Pathway by Solis and Macromedia	18
3.3 LearningSpace by Lotus	23

<b>4</b>	<b>Academic Solutions for Web-Based Training</b>	<b>26</b>
	4.1 Overview	26
	4.2 WebCT by The University of British Columbia	27
	4.3 Virtual-U by Simon Fraser University	31
<b>5</b>	<b>Comparative Study of Four WBT Tools</b>	<b>33</b>
<b>6</b>	<b>The Grades Application</b>	<b>37</b>
	6.1 Overview and Architecture	37
	6.2 Procedures	38
	6.2.1 Phase I – Course Registration and Student Registration	40
	6.2.2 Phase II – Marks Entry and Marks Report	42
	6.2.3 Phase III – Final Marks Calculation and Marks Report	43
	6.3 File Structure	47
	6.3.1 Initialization file: COURSE.INI	47
	6.3.2 Course List File : COURSE.LST	48
	6.3.3 Course Marks File: COURSE.MKS	49
<b>7</b>	<b>Conclusions and Future Work</b>	<b>52</b>
	7.1 Conclusions	52
	7.2 Future Work	53
	<b>Bibliography</b>	<b>55</b>
	<b>A The Grades Application – Forms and Programs</b>	<b>58</b>

# List of Figures

1.1 Iceberg	3
2.1 The Three Waves of Distance Education	3
2.2 WBT Universe	8
2.3 WBT Basic Steps	9
2.4 English 101 Course Home Page	14
2.5 Altavista Search Network, October 30, 1997	15
3.1 Pathway Architecture	19
3.2 Pathway (Pathmaker) Designer Module	21
3.3 Pathway Course Enrollment	22
3.4 LearningSpace Demo Schedule	24
4.1 WebCT and Virtual-U Architectures	27
4.2 WebCT Sample Home Page	29
4.3 WEBCT Sample Chat Line	30
4.4 Virtual-U Demonstration Campus	31
6.1 Grades Application Overview	38
6.2 Grades Application Client-Server Architecture	39
6.3 Course Registration Page	40
6.4 Student Registration Page	41
6.5 Marks Entry	42
6.6 Student Marks Report	43
6.7 Final Marks Calculation (Assignments)	44

<b>6.8 Final Marks Calculation (Mid-term Exams)</b>	<b>45</b>
<b>6.9 Final Marks Calculation (Display Marks)</b>	<b>46</b>
<b>6.10 Final Marks Report</b>	<b>47</b>
<b>6.11 Course Initialization File (COURSE.INI)</b>	<b>48</b>
<b>6.12 Course List File (COURSE.LST)</b>	<b>49</b>
<b>6.13 Course Marks File (COURSE.MKS)</b>	<b>50</b>
<b>6.14 Sample Final Mark Calculation</b>	<b>51</b>
<b>7.1 Instructional Methods</b>	<b>52</b>

# List of Tables

5.1 Comparison of WBT Technical Specifications	34
5.2 Comparison of WBT Tools (Part I)	35
5.2 Comparison of WBT Tools (Part II)	36
5.3 Comparison of WBT Prices	36

# Chapter 1

## Introduction

### 1.1 Motivation

Distance education or, distance learning, is widely defined as *any* approach to education delivery that replaces the same-time, same-place, face-to-face environment of a traditional classroom [1]. Surprisingly, this learning method has existed since the 19<sup>th</sup> Century. For over one hundred years many universities have offered classes by mail or, more recently, by televised lectures [2]. Self-paced learning and geographic independence are considered some of the most important advantages of distance learning. Indeed, it did not take long for instructors and companies to realize the impact that the Internet, particularly the World-Wide Web, would have on distance education.

New information and communications technologies, such as the Internet, may lead towards astonishing levels of downsizing in higher education. “In the next five years, virtually every major institution of higher education in the United States will offer at least a portion of their classes over the Internet.” [2] If Peter Drucker – a highly respectable management consultant – is right “universities won't survive thirty years from now” [2]. This alone could be the motivation for this and many other such works.

Teaching post-secondary courses certainly represents an excellent “lab” experience which

permits a deep analysis of the role of instructors and the ways technology can support their tasks. This has been the case with the "Computers in Engineering"\* course at McGill University where the Internet has been used as a support tool for the past few years.

Another factor that encouraged this work is my sound programming experience with the "CM and CI - Course Management and Course Information" applications developed on MUSIC/SP [3]. Due to the enthusiasm generated by the Internet, Course Management tools are continuously advancing towards more effective and efficient ways to support teaching. This study will present, analyze, and compare four such tools currently available on the market.

Do students learn more and better using the Internet? This is the hardest question to answer. Studies show that school children seem to learn just as well in less technologically sophisticated ways. Technology alone, no matter how futuristic or exciting, does not automatically improve the learning process.

What types of distance learning methods work best? We still have neither solid research nor results on the methods. However, one fact is for sure – the Internet and the Web are undoubtedly excellent vehicles to deliver and support courses. Moreover, the next-generation network (also known as Internet2) is under way with promises of speeds 100 to 1,000 times greater than the current Internet, reaching up to 622Mbps. This will permit students in different campuses to sit in on interactive two-way video lectures, and universities to share online digital libraries of video and audio content [4].

The next-generation network is the last, but not least, motivation for this work. Accomplishing a project on this thrilling topic (the Web) is by itself very exciting.

---

\* "Computers in Engineering" is a first-year three-credit course taken by an average of 170 students per term from different engineering and science disciplines. I am lucky enough to conclude this project after being a Faculty Lecturer at McGill University where I taught this course for three semesters.



Figure 1.1 – Iceberg

The image of an iceberg can be immediately associated to the subject of this thesis. So far we have only seen the tip of what the Internet will represent for educators and students. This work reflects my full belief that the Internet may dramatically change the way we teach and learn.

## 1.2 Thesis Contributions

This thesis has three important contributions: evaluation and comparison of four Course Management tools, guidelines on how to deliver successfully a course using the Internet, and finally, the use of JavaScript and the CGI programming environment to calculate grades for a course.

The major goal of this thesis is to encourage instructors to implement courses using the Internet and, most of all, to show that it can be simple and effective. Four major Course Management tools currently available that deliver material over the Internet were chosen for this work. A comparative study is presented in this document and it may be the basis to the development of a framework – set of principles and ideas – to implement a course using the Internet.

Based on the actual needs of the "Computers in Engineering" course, a series of JavaScript and Perl programs was developed to accomplish grade reporting. My goal was to develop a grade system that would be as much as possible platform independent, which is accomplished by using standard JavaScript on the client side and standard Perl on the Web server.



An instructor looking for guidelines and references on how to implement a course on the Internet can fully count on this project. The scope of this work is limited to the post-secondary level (college or university) education.

## **1.3 Thesis Layout**

Chapter 2 – Distance Education on the Web – contextualizes our study. In this chapter, two important concepts are introduced – the Distance Education Waves and the Web-Based Training Universe. Through examples and references we analyze the potential advantages and possible disadvantages of this innovative teaching medium.

In Chapters 3 and 4 we investigate four Course Management tools. Chapter 3 – Commercial Solutions for Web-Based Training – analyzes two commercial products developed by two solid companies: Pathway by Solis-Macromedia, and LearningSpace by Lotus. Both are called proprietary systems offering combined solutions with the open architecture of the Internet. A system is defined as proprietary when it generates data in its own unique format, which, as a consequence, makes portability and interoperability difficult.

Chapter 4 – Academic Solutions for Web-Based Training – concentrates on the other two applications, academic alternatives developed by two well-known and respected universities: WebCT by The University of British Columbia and Virtual-U by Simon Fraser University.

Chapter 5 – Comparative Study of the Four Tools – shows the similarities and dissimilarities among the tools described in the previous chapters. One may want to refer to this final analysis before reading about the full features of the tools described in the previous chapters.

Chapter 6 – The Grades Application – presents a straightforward approach which can be used to deliver a course using the Web. It also fully describes the Grades Application system (developed using Java Script and Perl) to calculate final grades.

Chapter 7 – Conclusions and Future Work – finalizes the thesis and indicates future work which can be accomplished.

This document is entirely available on the Web where it uses hyperlinks between the chapters and direct links to the bibliography. The *URL* for this document is:

**<http://www.cs.mcgill.ca/~spiller/thesis/index.htm>**

The userid to access the document is **vip** and the password is **super** (both in lower case).

This document was entirely developed using Microsoft FrontPage 97 and Paint Shop Pro 4.1.

## **Chapter 2**

# **Distance Education on the Web**

### **2.1 The Waves of Distance Education**

One could define education on the Web as the third wave of distance learning. The first wave can be represented by those courses delivered by mail and television, where technology did not play a significant role. This is the most passive learning method where, as mail recipients or TV spectators, students do not participate at all in the lectures. The single most popular form of distribution technology for distance education is TV broadcast. In the United States PBS has 350,000 subscribers to its open university with 75 accredited business schools that offer degree granting programs primarily through the use of broadcast [2]. Given all the fuss about the Internet it may seem the first wave is obsolete. However, this is not the case. One of the main reasons why the first-wave is not dead is how widespread and firmly incorporated into people's lives are both mail and TV networks in most countries.

The incorporation of technology represents the separation between the first and second waves, where the advent of personal computers played a major role, thus Computer-Based Training (CBT) was born. CBT provides self-paced instruction, with anytime and anyplace access. It may be interactive, incorporate examples, demonstrations and simulations, and

provide self-evaluation quizzes. What it does not provide to students, though, is the possibility of interacting with instructors and the collaborative learning with colleagues. CBT is limited in that the student only interacts with the technology [1] and this can be a significant drawback in learning effectiveness.

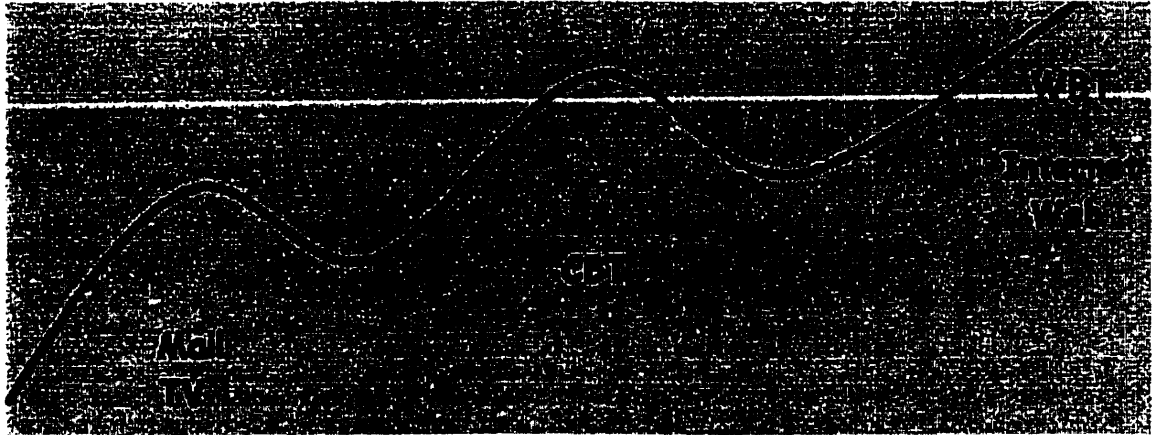


Figure 2.1 – The Three Waves of Distance Education

The third wave started with the Internet and, in particular, with the advent of the Web. The potential of this new medium is enormous, as one can verify by the impressive number of courses exploring it since 1994 – the booming year of the Web. Several acronyms have been established to define learning on the Internet. They include Internet-Based Instruction (IBI), Web-Based Instruction (WBI) and Web-Based Training (WBT). Basically, the three refer to the same framework and technologies. Web-Based Training (WBT) will be the acronym used throughout this document.

## 2.2 Web-Based Training Universe

The Web provides educators with an outstanding opportunity to deliver courses [5]. Figure 2.2 illustrates a comprehensive learning environment containing different information in

distinct formats or media – e.g. text, sound, and video. On the Web, all the course content is accessed through applications running on the TCP/IP protocol which is the transmission protocol of the Internet. It is not within the scope of this work to describe the protocol or its services.

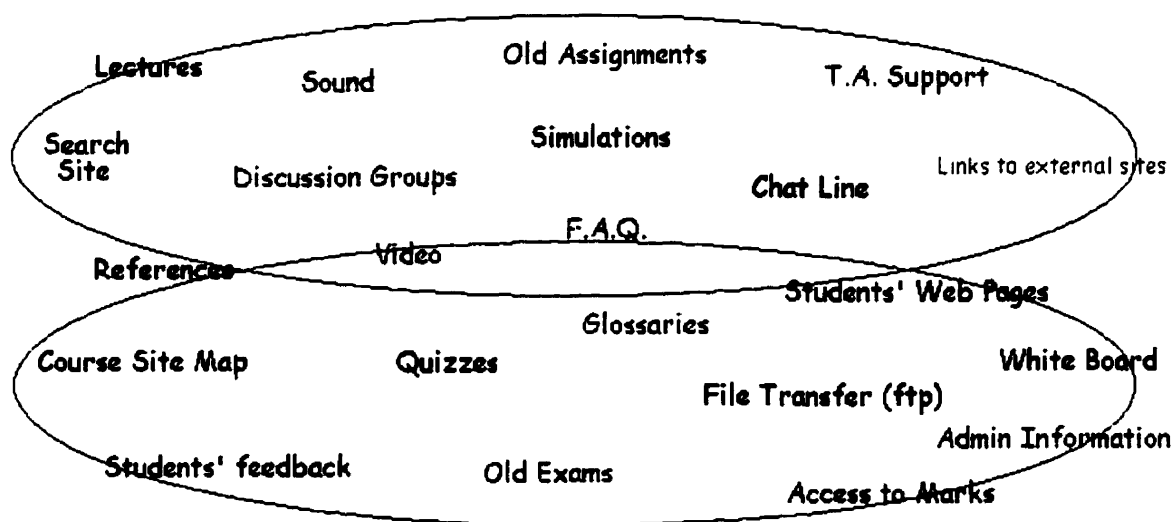


Figure 2.2 – WBT Universe

Figure 2.2 exemplifies the contents a typical post-secondary course may contain. Obviously, some examples may not apply to a specific course. One interesting point to observe is that, since the Web is a hypermedia environment, all the course content can be linked together. For instance, a student can access the Frequently Asked Questions (FAQ) about the course and then perform a site search for Quizzes. Another interesting aspect about this *universe* is the knowledge base that can be built based on previous editions of courses. A student taking the course in one semester may easily access assignments and exams from past semesters.

## 2.3 Potential Advantages of WBT

Never before has technology made it so simple to set up a distance learning program as with the Web. Web-Based Training is easy to build, dynamic, interactive, collaborative, and, last but not least, platform independent.

### 2.3.1 WBT is Easy to Build (and Maintain)

Basically, the process of making course material available to students involves the following three steps:

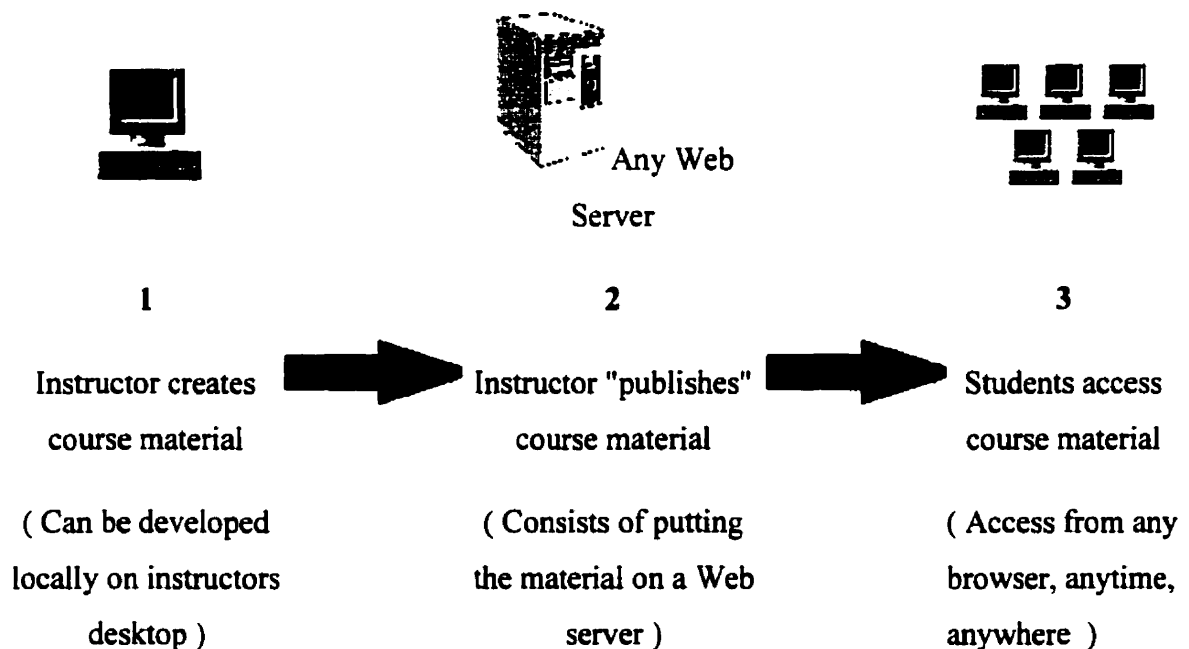


Figure 2.3 – WBT Basic Steps

Essentially, anyone can build a WBT course. The only technical ability required of instructors to create courses on the Web is the same basic skill required of students to use it. Knowing how to "point-and-click" suffices to build WBT; no programming is required which allows instructors to concentrate on course content. As a matter of fact, some of the most interesting WBT systems currently available were developed by instructors who are from schools other than Computer Science.

### **2.3.2 WBT is Dynamic**

Not only are Web-based courses easy to build, but they can also be dynamic. Instructors can add or change course material as needed, making the new information immediately available to students online, wherever they are.

WBT also brings great possibilities in terms of customization. For instance, the instructor can make available specific material only to a certain group of students who need extra information on a given topic. Moreover, with graphics, animation, colours, and sound, a course may have a more "student" look. In fact, some WBT offer the student the possibility of tailoring some of the course's Web pages, applying the style they prefer.

This hypermedia aspect of the Web which links different formats of media (e.g. sound, video, pictures) allows a smooth navigation not only through material located in the course's data repository, but also linking to related Web sites anywhere in the world.

### **2.3.3 WBT is Interactive and Collaborative**

Interactive exercises and simulations may form an integral part of the Web course, adding an experience that can hardly be matched in the classroom. In a typical class, only one or two such examples can be presented due to the time required to set up and evaluate them [7].

Contrary to Computer-Based Training, Web-Based Training facilitates group interaction: e-mail, newsgroups and chat, create new ways of communication and collaboration between professor and students and also among students. In addition, students can be guided through tutorials and quizzes that may provide them with better understanding of the topics as well as instant feedback.

As a way to promote collaboration, WBT encourages students to participate (most commonly by writing) in “class discussions” and share their opinions with their fellow colleagues. In many cases, students who are shy or physically challenged will find it easier to ask questions of their peers in this “virtual” condition which will eventually help them to better integrate within the group.

Moreover, the opportunity to reach a more geographically dispersed audience brings great benefits in terms of inter-cultural dialogue, not to mention the possibility of offering the latest information to students in remote locations.

Students’ comments, opinions, and questions can be registered in databases building a knowledge base for future students. Furthermore, by using access log files and *cookies* (files containing information about site visits), it is possible to monitor the progress of each student.

### **2.3.4 WBT is Platform Independent**

All distance education methods imply that students can learn at varying rates. CBT and WBT add an important advantage: students may explore the course material to whatever depth they desire.

However, what gives the biggest competitive advantage to WBT compared to second-wave CBT technologies is the multi-platform *universal browser* – it adds significant portability requiring no special software or tools installed on the client’s workstation other than the browser. Whether the student’s computer is a Mac, a PC, a Unix station, a Network



Computer or even a Web-TV, they all have standard viewers to browse the Web.

## **2.4 Possible Disadvantages of WBT**

When asked about what they liked least on a Web-based course, students at the University of British Columbia, most commonly answered: "the extra motivation and responsibility required" [7]. Would this be considered a disadvantage for WBT? One could argue that humans are motivated for a course by different reasons and that even self-organization could be another potential factor to a student's success or failure. Some disadvantages are difficult to qualify or quantify but based on previous experience [5] [7] [8], the following items were present:

- Teaching Assistants (TA) have to be trained to be able to support students;
- The restrictions of the technology (specially bandwidth) may limit the size and quality of some course material, e.g. video clips;
- Need for technical support staff to help students and instructors;
- Overload of e-mail messages or student demands;
- The difficulty to measure student failure or success.

In some cases, professors who are firmly established in their academic areas but are novices with technologies may not be able to motivate students when their course interest diminishes or to help them with technical problems.

## **2.5 Examples**

Examples of Web-based courses are bursting everywhere. Chapters 3, 4 and 5 will analyze important products currently available on the market that facilitate course management.

However, before introducing such complex systems one should inspect independent initiatives from different educational institutions.

### **2.5.1 Computers in Engineering, McGill University**

This course is one of the main motivations for this thesis: the Computers in Engineering course and Web site developed by Professor Gerald Ratzer (<http://www.cs.mcgill.ca/~ratzer/308208Welcome.html>) here at McGill University. The course material which is available to anyone who wants to consult it, includes all lectures material, old assignments, old mid-terms, and a complete set of quizzes with instant feedback. One of the most important features of this Web course is its *knowledge base*. For instance, students can refer to more than 70 previous assignments.

This Web course does not require powerful client machines or large bandwidth since one of Professor Ratzer's goals was to make the material easily accessible from Zimbabwe, Africa where he conducts some of his academic research and where computer resources and telecommunications infrastructure are limited.

### **2.5.2 Online English 101, Spokane Community College**

Another interesting and simple implementation was developed by Jan Strever from Spokane Community College, US. What captures one's attention in his Online English 101 (<http://www.ior.com/~jstrever/index.html>) course is the informal language he uses to introduce the course content. His site is also extremely dynamic and creative. For instance, from time to time he promotes, as the course's *home page*, a page entitled "Pause a moment for Poetry with weekly updated links to poetry, poetry ezines, and other resources for the poet". Figure 2.4, is an extract of the course's *site map*:

Daily and Weekly Assignments		
Click on the assignment below to read student produced essays and assignments. <a href="#">Compare/contrast essays</a>   <a href="#">Process essays</a>   <a href="#">Cause/effect essays</a>   <a href="#">Punctuation Handouts</a>		
<a href="#">Final Essay - Reflection</a> <b>NEW</b>		
<b>English 101 Exit Exam</b> —All English 101 students must pass the departmental exit exam. Please read the following information about the exam.		
<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>

Figure 2.4 – English 101 Course Home Page

### 2.5.3 Environmental Law, Manhattan College

The Manhattan College offers the Environmental Law - 417 course (<http://www.manhattan.edu/engineer/envir/matystik/envl417/envl417.htm>) which also implements a simple and efficient version of the WBT universe. Again, creativity plays an important role in the course. For instance, Mr. Walter F. Matystik includes a photo of the students in his course site. It may not help students in the learning process but it certainly gives a *personal* touch to the virtual course.

### 2.5.4 Internet Resources, McGill University

Another attractive example is the successful "Internet Resources" course from McGill University entirely taught using the Internet. This course was created and designed by Marcos Silva and Scott McKeown for the "Certificate in Educational Technology" program

in the Faculty of Education. The course material is limited to students enrolled in the course, one feature includes a Web page called the "Student Lounge - A place to sit back and relax. Get to know your classmates or just grab a coffee". The "Student Lounge" includes students' home pages, biographies and interests. This is a proven example that a virtual group may not meet *face-to-face* but can create the *ambience* of a course taught in a real classroom.

## 2.5.5 Other References

As of today's search on Altavista (one of the most popular Web search engines), the search for "Distance+Education" gives the result of "About 46073 documents match your query" :

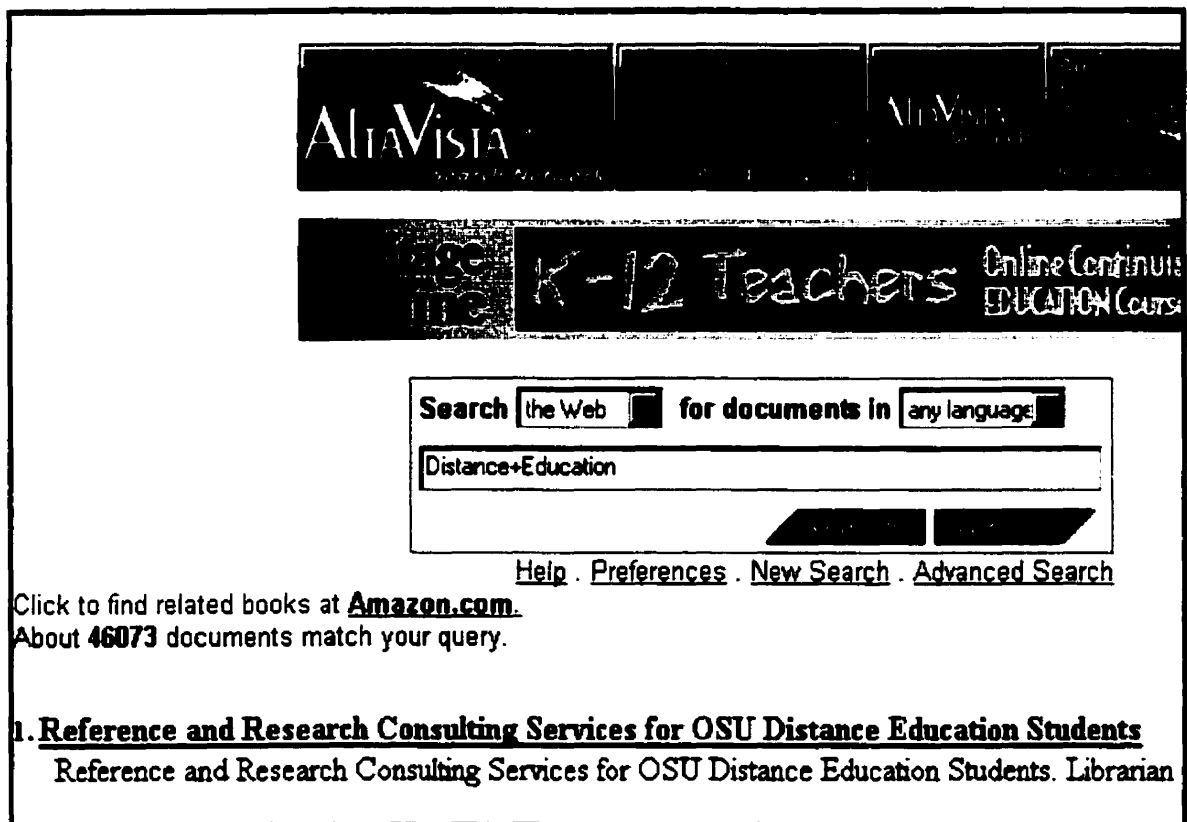


Figure 2.5 – AltaVista Search Network, October 30, 1997

Note that in Figure 2.5, one of the prime spaces reserved for advertising on the AltaVista page is about "Heritage Online" for K-12 Students, one example itself. Besides the *just in time* search option provided by search engines on the Web, there are several directories of Distance Education courses offered on the Web. The University of Texas, U.S., for instance, has created "The World Lecture Hall (WLH)" (<http://www.utexas.edu/world/lecture/index.html>) which contains links to pages created by faculty worldwide who are using the Web to deliver class materials.

## **Chapter 3**

# **Commercial Solutions for Web-Based Training**

### **3.1 Overview**

Different software companies are delivering products to manage course content, enrollment, access, and other administrative tasks related to accredited courses. This chapter presents the features of two such products which were primarily developed to answer the needs of corporate training: Pathway by Solis-Macromedia and LearningSpace by Lotus. These products offer complete solutions to implement a course to be delivered over Local Area Networks (LAN), intranets (networked systems inside companies' boundaries) or, over the Internet.

To stretch dollars, corporations are turning to technology for assistance in distributing flexible educational experiences. Technology can deliver learning experiences directly to employees' workstations, eliminating travel and other related expenses. For the business world, the impact of virtual learning is clear. By 1998, the Gartner Group predicts that 12 billion US\$ will be spent on online business training. Another well known research firm, Quality Dynamics, predicts that "half of all corporate training by the end of this century will be online" [2].

It did not take long for the developers of Course Managed Instruction (CMI) Pathway and LearningSpace to foresee the potential market in educational institutions. What the developers of these products must have in mind, however, is that educational institutions and business corporations have distinct missions. Their solutions must be tailored to match the particularities of the educational environment, where *nine-to-five* or *intensive-one-week* courses are not viable. The main goal of this chapter is to inform readers, in particular those in the academic community what is already available on the market especially since the software industry is willing to conquer and offer solutions to the educational *milieu*.

Although the majority of commercial products offer proprietary solutions, it is in their interest that standards be developed so that communication among different systems will be possible, improving as a consequence, their compatibility and interoperability. The Aviation Industry CBT Committee (AICC) oversees the only open standard in existence (API) for relaying courseware information between CBT and Course Management systems. The IEEE, one of the most respected standards bodies in the world, has recently begun the process of addressing standards in computer-based learning. The IEEE P1484 working and study groups are attempting to provide a top-down approach that covers every aspect of learning including, but not limited to, session management, authoring tools, learning objects, tool/agent communication, and the learner model [9].

## **3.2 Pathway by Solis and Macromedia**

Solis-Macromedia is the leading vendor of Computer Managed Instruction (CMI) software for enterprise-wide training and skills management [9], their product Pathway administers both CBT and WBT courses. Pathway has four general administrative functions: designing curricula; students' enrollment; managing and delivering courses; and finally, tracking and reporting student progress. Course content is encapsulated into the Pathway system where it may contain HTML files or any kind of material of the WBT Universe as described in Chapter 2.

However, note that the content management is done by the Pathway server instead of a standard Web server. The general product architecture is shown in Figure 3.1.

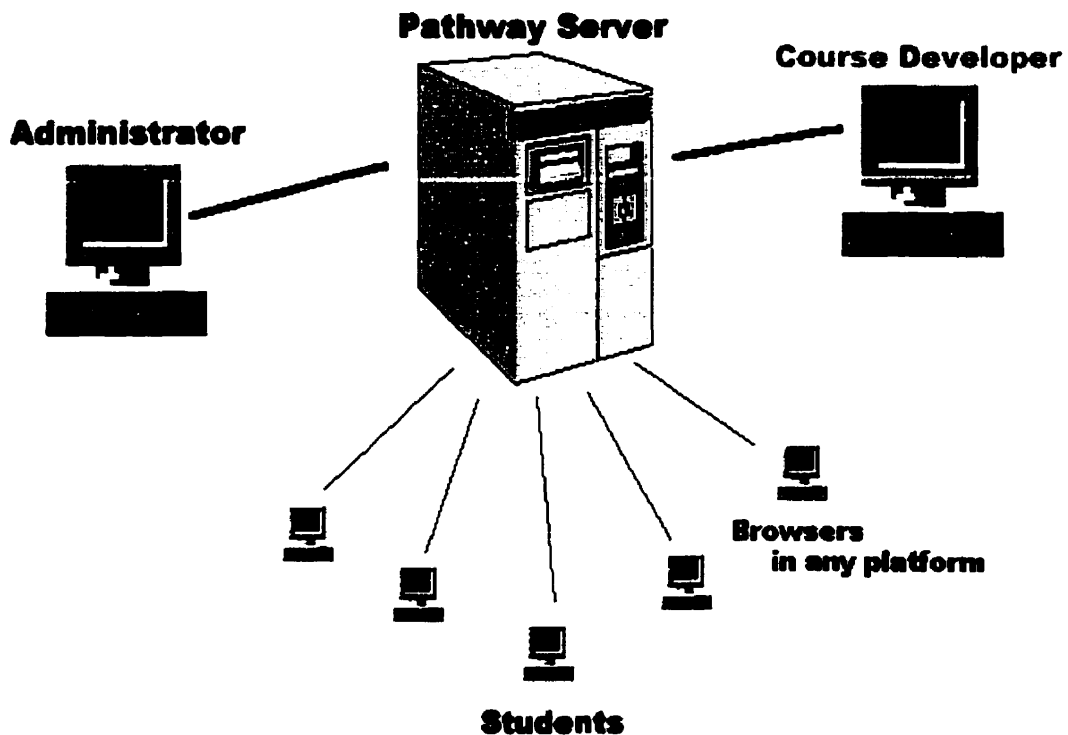


Figure 3.1 – Pathway Architecture

Since Pathway is a proprietary system, the data is kept in Pathway-specific format, requiring an administrator to know the product. To a certain extent, course developers are dependent on the administrator to manage the server. In its most recent version, Pathway incorporates Java Applets which permit courses to be accessed from the student's workstation without any Pathway client software – students can now use Netscape or Internet Explorer (Java enabled) to have access to the courses.

Some of the key features of Pathway are described below:

**Course Expiration** – Curriculum files can be assigned an expiration date that will notify students that course materials are out of date.



Relationships between objectives and activities - Learning objectives can be tied to learning activities and modules in any arbitrary fashion. Pathway's interface for this employs a grid to relate more than one objective to an activity or more than one activity to an objective.

**Required/Optional Assignments** – Learning activities, objectives, and modules can be either required or optional. The instructor can also require that the student complete a certain number of optional items in order to complete a course or module.

**Four Completion Options** – Each activity can be set as complete in a number of different ways. If the activity can communicate with Pathway, the activity will decide when it is complete. Another option is to have the student decide when it is complete. Other options assume completion the first time the student starts the activity. These options allow better integration of courseware or activities that do not communicate with Pathway.

**Provides time estimates** – The course designer can provide time estimates for all the course components. Pathmaker (a Pathway module) can automatically roll-up time estimates for activities into modules and courses. The time estimate allows the student to budget his or her time appropriately for each learning activity.

**Option to save interaction data** – Pathway can save detailed information about each interaction the student encounters in the instructional material. This information includes interaction identifier, objective relationship, student response, correct response, judgment, weight, and latency (how long it took the student to finish the interaction). The course designer decides whether or not to record this information.

**Enrollment "en masse" by course or by student** – Administrators can enroll all students in a course or one student in all courses with a single mouse-click by either clicking on the student or course name. This makes mass enrollment simple and fast. This function is specifically useful when several courses in one academic program are administered by Pathway.

**Four built-in reports** – Administrators have four reports available. These reports include a high level report which just shows completion status for any number of courses and

students, a summary report for a course and any number of students, a summary report for a student and number of courses, a detailed report for a student and a course, and an interaction data report for a student in an activity. All of these reports can be printed or exported to a text file for use in a spreadsheet, word processor, or a database.

A commercial package such as Pathway will normally include extra features that a specific course may not need. If those extra features do not add any complexity (or cost) to the course management, then the instructor should not worry should she decide not to use them.

The course design module uses an ordinary Windows multiple-document interface application where the course developer creates and links course content. Figure 3.2 shows an example of a possible screen where different course materials are being identified as part of the course program:



Figure 3.2 – Pathway (Pathmaker) Designer Module

As another example, the roster function gives administrators complete control over enrolling students. The roster screen shown in Figure 3.3 contains fields for all personal information about users:

Ann Admin  
James Baldwin  
Brad Bingly  
Terry Brentwood  
Ray Chuckles  
Mike Johnson  
Mary Mathews

Patty Ewing  
Patty Ewing  
112-22-2222  
Plastics  
Intern Tech Write  
All Classes  
All Seniorities  
All Skill Groups

Figure 3.3 – Pathway Course Enrollment

Figure 3.3 shows an example of a typical case where the enrollment fields will have to be adapted to the academic world (e.g. Employee ID to become Student ID).

The next section introduces another important commercial solution, LearningSpace by Lotus.

### 3.3 LearningSpace by Lotus

LearningSpace was developed by the Lotus Institute's Research & Development [1], the product uses the Lotus-Domino software as the server to accommodate the course content and the course management tools. The functional architecture of LearningSpace is very similar to Pathway's as shown in Figure 3.1. Instead of a Pathway Server, we have the Lotus-Domino server. As with Pathway, an administrator is necessary, since the system stores data in its own format which requires regular administrative tasks to be executed .

LearningSpace was also designed as a tool for business corporations, though it already has a significant presence in the educational market: University of Wisconsin, University of Maryland, Washington University, to name a few, are using LearningSpace to deliver courses. In Canada, there are more than twenty-five colleges and universities using LearningSpace [11].

LearningSpace has five modules that provide the tools and the framework needed for developing a course:

**Schedule** – From the instructor's point of view, "Schedule" contains information about each class. In this module, the course syllabus, including course description, course objectives, requirements and due dates are defined. Students use this module to access course information.

**Assessment Manager** – This module is only used by instructors to create and store tests, assessments, grades and class surveys, and to provide private evaluations and feedback to students or teams. Students will use "Schedule" to access this material.

**MediaCenter** – Contains material which may be rich in multiple forms, including text, audio, video, computer-based training, graphics, or whatever instructors have selected to make available. From the "MediaCenter" links to the Web can be made.

**CourseRoom** – Space reserved for the course interaction: it has discussion groups and other tools that facilitate collaborative learning.

**Profiles database** – This module may contain personal home pages covering students' background and interests.

With the latest version of the server, students do not need to install any piece of client software in their workstations in order to access the course material. They may use any Web browser – e.g. Netscape or Internet Explorer. In previous versions, "Notes Client" software was required. The main advantage to "Notes Client" is that students can pull the online information down to their desktops and then work entirely disconnected from the network.

Figure 3.4 shows a possible screen for a course. As in Pathway, the instructor uses a Windows multiple-document interface application to set-up and manage the course.

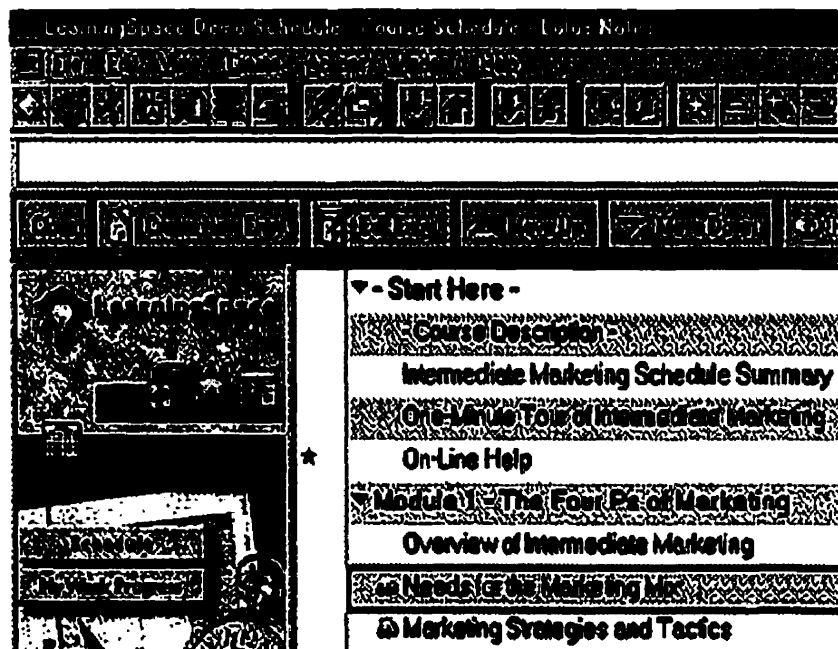


Figure 3.4 – LearningSpace Demo Schedule

The LearningSpace product Web site allows any Internet user to create a student account in a LearningSpace demonstration course. More information can found in: <http://www.lotus.com/education/learningspace.nsf>. In addition, an interesting aspect about the support an instructor may have from the LearningSpace team is the Lotus Institute, which provides training on how to create online classes and how to explore the instructional strategies for online course delivery.

Hopefully, this chapter provided the reader with a concrete idea of what such commercial packages can offer to instructors. In Chapter 5, we will compare both commercial products with applications developed by Universities.

## Chapter 4

# Academic Solutions for Web-Based Training

### 4.1 Overview

In this chapter, two other Course Management alternatives developed by well-known and respected universities are presented: WebCT by The University of British Columbia and Virtual-U by Simon Fraser University. These authoring and management tools are intended to be used either as a complement to an existing lecture-based course, or as a self-contained distance alternative to a lecture-based offering. Due to their local success, both solutions are currently being packaged for commercialization as products. Since these tools were developed "in-house", *by instructors and for instructors*, based on post-secondary course needs, one could argue that these solutions could be easily implemented in other educational institutions.

One of the most important aspects of the architecture of both academic solutions is that they use a standard Unix Web server to create, store, and manipulate the course data. Unix machines are heavily used in universities, and, in most cases, no hardware upgrade is necessary to accommodate the courses.

Another key aspect in WebCT and Virtual-U is that the instructor uses a standard browser as the interface to manage the course. The commercial products presented in Chapter 3

use a Windows application in the course design modules. This requires the instructors to install new software in their desktops.

In addition, note that both solutions require minimal technical expertise therefore the role of a course administrator role is not necessary. If the server that hosts the course tools and content has standard backup procedures, no extra maintenance activity is required.

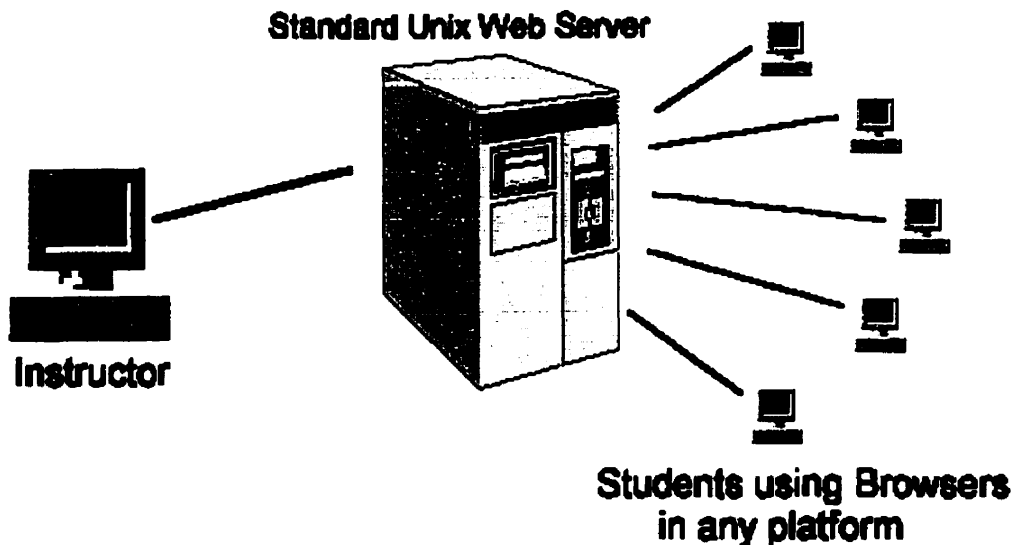


Figure 4.1 – WebCT and Virtual-U Architectures

## 4.2 WebCT by The University of British Columbia

WebCT stands for World-Wide Web Course Tools. The program was initially developed in 1996 as an educational resource focused on a third year Computer Science course taught by Murray W. Goldberg. Since then, Mr. Goldberg has published many articles and participated in several conferences about WBT. He has conducted various surveys and collected enormous feedback from students with positive results showing a high degree of student satisfaction and improved academic performance [7].



To date, approximately 500 institutions around the world have installed WebCT and are either testing or using it. The WebCT at The University of British Columbia server houses approximately 140 courses [5]. Minimal technical expertise and effort on the part of the course designer were among the main objectives of WebCT.

Some of the main features of WebCT are listed below:

**Student Home Page Creation Tool** – Students can use WebCT to build personal home pages containing images, text and links to other pages. The interface is similar to that presented to designers for home page construction. No knowledge of HTML is required. For security reasons, the instructor is also able to edit students' home pages.

**Timed Quizzes** – Quizzes can be written by the instructor and delivered on a predetermined day. While answering the quiz, the student is presented with the questions and a count-down clock displaying the remaining time. Once completed and marked or auto-marked, the assigned grade, with or without comments, is made available online to the student. Another module for self-evaluation using multiple-choice questions could also be made available to students where their performance is not recorded in any way.

**Student Tracking** – Besides information stored about timed quizzes, several types of information are provided by WebCT that allow the instructor to monitor student participation and progress. Student tracking is important since one disappointment in delivering Web-based courses is the instructor's inability to follow the progress of his or her students [5]. The evidence which is available in face-to-face instruction such as attendance or apparent interest is not available in a Web-based learning environment. Several types of information are provided by WebCT that allow the instructor to monitor student participation and progress, including a student-tracking summary page which consists of a table with one row presented for each student. For each student an indication of the date of first access to the course, most recent access to the course, and the total number of accesses to the course is presented. Clicking on any of the students links, the instructor is provided more detailed information for that student including a histogram of accesses made to various course components, and other detailed information such as the extent to which the

student has progressed through the course content.

The WebCT server to which the users connect can be installed by the course developer or by the network administrator at the developer's institution. Alternatively, some Internet Service Providers (ISP) with WebCT on their servers can provide instructors with the facility of storing courses.

An illustrative course home page built with WebCT is available in the WebCT site (<http://homebrew1.cs.ubc.ca/webct/webct.html>). Figure 4.2 shows that sample page. Note the main options displayed in the student's browser:

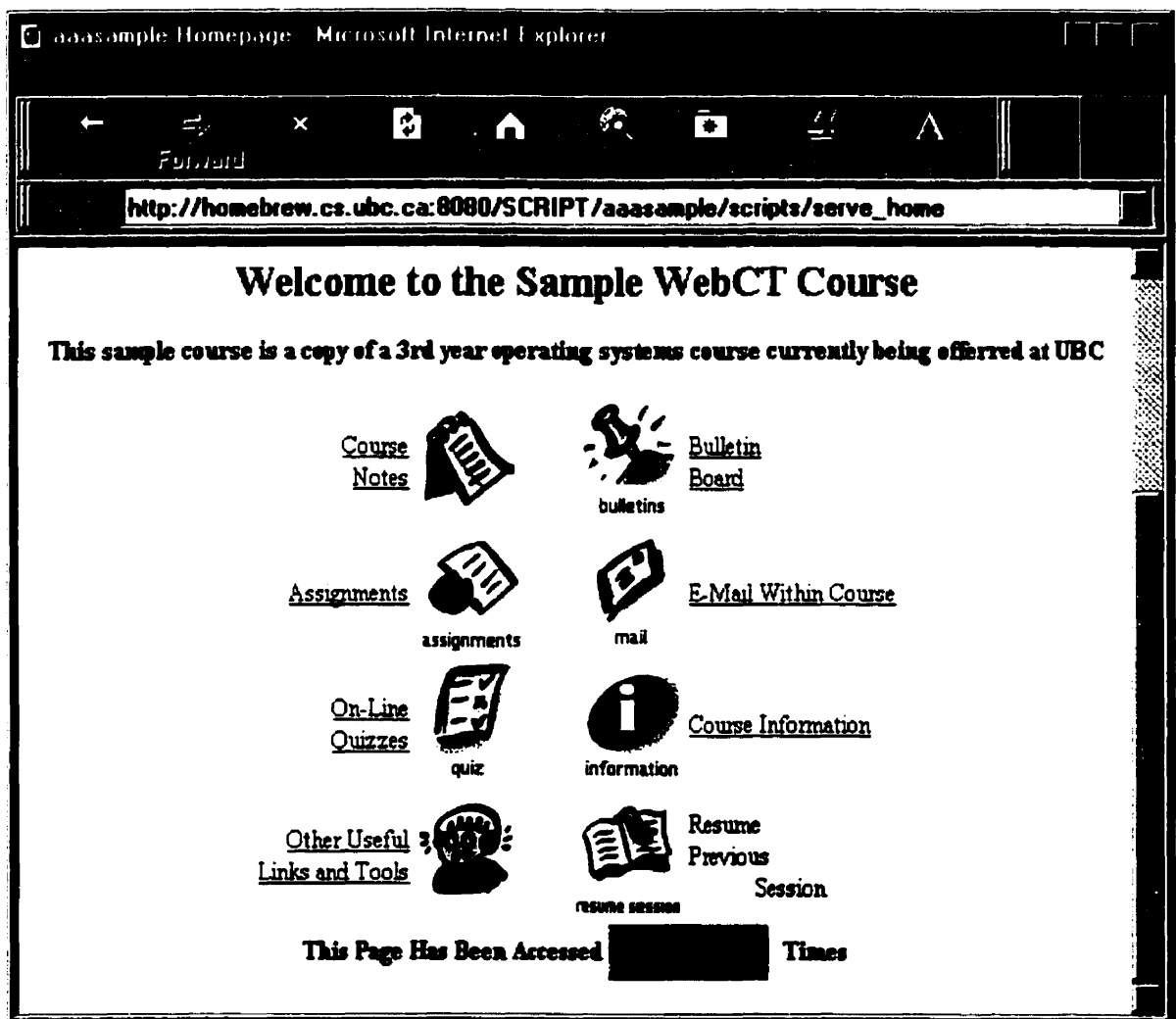


Figure 4.2 – WebCT Sample Home Page

As in both commercial tools (Pathway and LearningSpace), WebCT and Virtual-U allow asynchronous and synchronous interaction and collaboration among all the participants in the course (including the possible interaction with teaching assistants and instructor). Asynchronous communication consists of interactions, exchange of data and files where the correspondents are not online at the same time. For instance, discussions groups are an example of asynchronous collaborative communication. On the other hand, synchronous interactions require real-time communication among participants. Synchronous communication allows both casual conversations among students, as well as real-time tutorial sessions held by teaching assistants or the instructor. In WebCT, a Chat tool is available that allows this kind of synchronous collaborative communication:

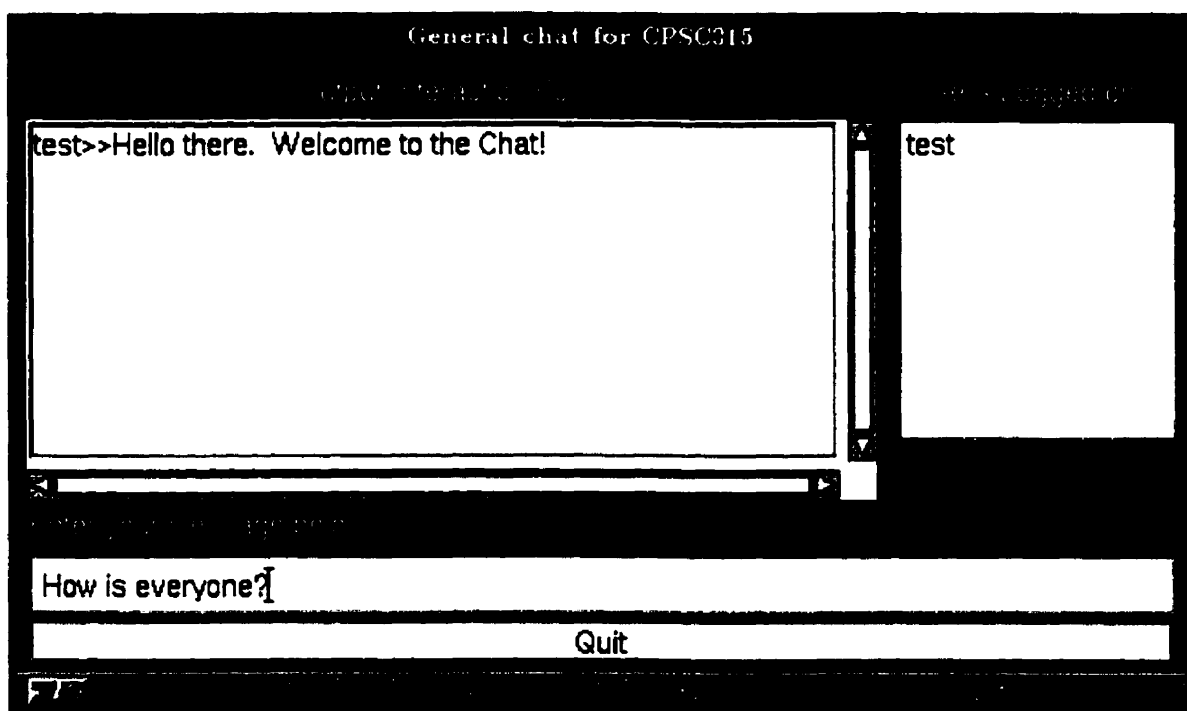


Figure 4.3 – WebCT Sample Chat Line

This section introduced a very successful Course Management tool. To complete the analysis of academic applications developed for Web-Based Training, the next section presents Virtual-U developed at Simon Fraser University.

## 4.3 Virtual-U by Simon Fraser University

As in WebCT, Virtual-U became a Course Management tool that can be purchased by other institutions. In fact, Virtual-U is now a product of the Virtual Learning Environments Incorporation (VLEI). The Virtual-U Research Project is part of the TeleLearning Network of Centres of Excellence and is funded in part by CANARIE. In an ongoing effort to understand how to achieve effective online learning, the research project brings together specialists from Computer Science, Education, Psychology and Engineering to merge approaches and knowledge in their search for answers. The fruit of this research is Virtual-U. One of the main contributors to this project is Dr. Linda Harasim, a recognized pioneer in online course design and delivery [12]. Using the geographical metaphor of a campus, the student can *surf* through different *buildings* and enter classrooms which transcend space and time of ordinary classrooms:

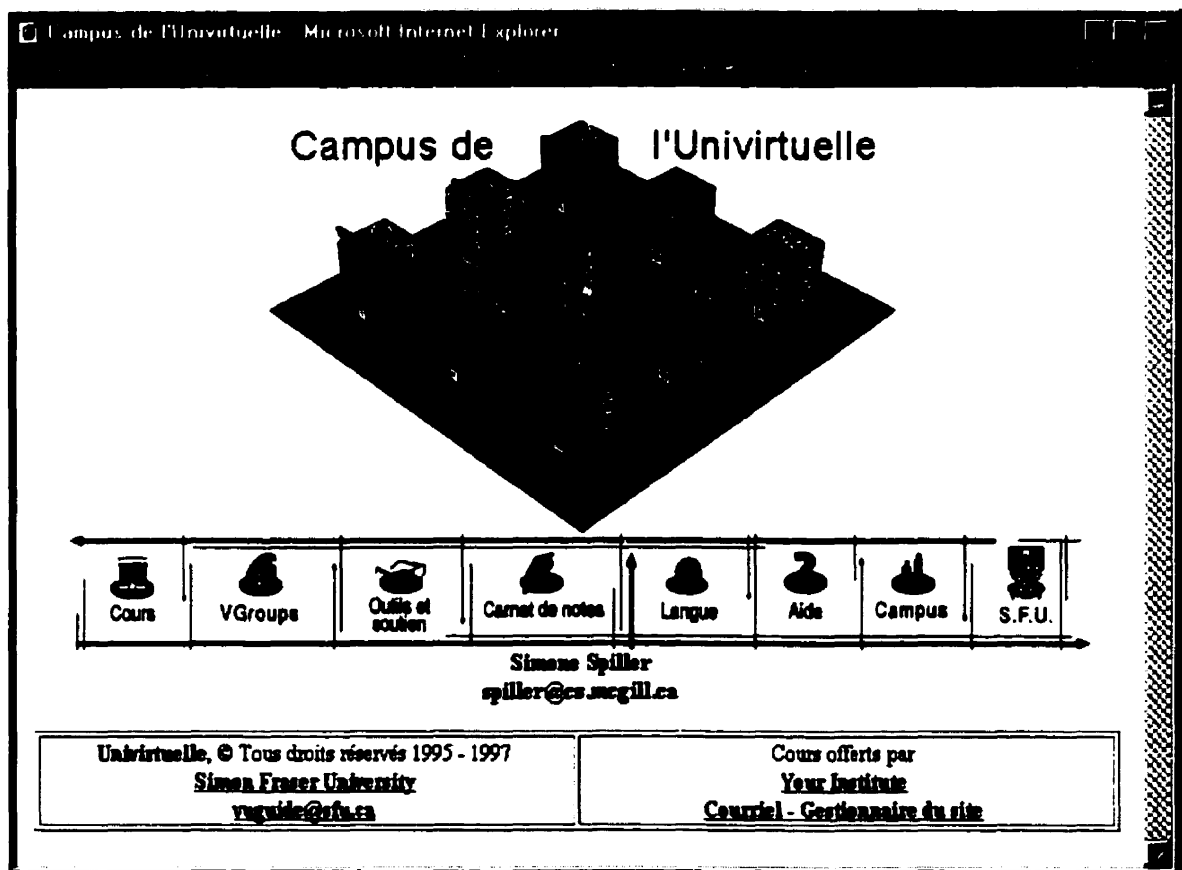


Figure 4.4 – Virtual-U Demonstration Campus

The Virtual-U modules are available in English or French, this customization feature may be a competitive advantage against other products (e.g. WebCT) which are available only in English.

The modules of Virtual-U are the following:

**The Conferencing System** – This module gives instructors the ability to set up collaborative groups and define structures, tasks and objectives. Any user can learn to moderate conferences and to create subconferences.

**Course Structuring Tool** – Enables instructors to create courses online without programming knowledge. Templates prompt the instructor for relevant information such as weekly overviews, required readings, grading standards and group conferencing assignments.

**Gradebook** – Manages the database of students' grades for each course delivered with Virtual-U. The grade book displays text and graphical representations of student performance including personal grades and distribution of marks for evaluated activities.

**System Administration Tools** – Assists the system administrators in installing and maintaining Virtual-U. These include functions such as creating and maintaining accounts, defining access privileges and establishing courses on the system.

Both Virtual-U and WebCT administration modules require a standard Unix Web server such as Sun OS 4.1.x or Solaris 2.5 or higher.

This chapter completes the study of four Course Management tools that may well answer the needs of a post-secondary course delivered on the Web. Chapter 5 – Comparative Study of the Four Tools – presents a final analysis highlighting the similarities and dissimilarities among the tools described in this and the previous chapters.

## Chapter 5

# Comparative Study of Four WBT Tools

This chapter presents a comparative study of the four tools presented in Chapters 3 and 4: Pathway by Solis-Macromedia, LearningSpace by Lotus, WebCT by The University of British Columbia, and Virtual-U by Simon Fraser University. In the format of comparison tables, the tools are analyzed according to several essential criteria which are extremely important when choosing a product to answer the needs of a Web-based course.

The following tables are built based on the analysis made in this thesis in addition to three other studies published by The University of Manitoba [8], the PcWeek Magazine [13] and the SCOET/CCTT/OLT (Standing Committee on Educational Technology, the Centre for Curriculum, Transfer and Technology and the Office of Learning Technologies) [14].

The order of the columns follows the order in which the tools were presented in Chapters 3 and 4. The features are grouped by the following two categories:

**A- Technical Specifications:** does the solution use a proprietary architecture? Is it necessary to install software on instructor's desktop? These and other technical aspects to be considered when choosing the right WBT tool are summarized in this category. Although it is important to check the disk space and RAM requirements when installing the tools, note that the necessary disk space to accommodate the course material is directly

dependent on the course size and content (number of lectures, quizzes, etc.).

**B- Tools:** in this category, the term "tools" is used in its broader sense: Does the system provide the instructor with the tools to follow students' progress? Can a student follow his or her own progress? Are there tools to build quizzes? In other words, how much does the system help the instructor and the student, respectively, to teach and learn?

Proprietary system - Data stored in system's specific format	Yes but open to any ODBC compliant platform	Yes	No	No
Requires software installation on server	Yes	Yes	Yes	Yes
Required disk space (on the server)	200-300Mb	300Mb	10Mb	500Mb
Required RAM (memory required to run the system on the server)	64Mb	64Mb	32Mb	24Mb
Requires software installation on instructor's desktop	Yes (Pathway Client)	Yes (Notes Client)	No	No
Requires software installation on student's desktop other than a browser	Optional (Pathway Client)	Optional (Notes Client)	No	No
Types of operating systems	Pathway within Unix or NT	Notes/Domin o within Unix or NT	Most common Unix servers	Most common Unix servers
Allows access authentication (security)	Yes	Yes	Yes	Yes

Table 5.1 – Comparison of WBT Technical Specifications

Knowledge of HTML is required	No	No	No	No
When developing quizzes, knowledge of HTML is an asset	No	No	Yes	Yes
Allows instructor to follow student progress tracking	Yes	Yes	Yes	Yes
Allows student to follow his or her progress	No	No	Yes	Yes
Allows self-evaluation quizzes and tutorials	Yes	Yes	Yes	Yes
Allows timed quizzes	No	No	Yes	No
Allows student to personalize course view	No	No	Yes	No
Allows student to make private annotation about linked to course material	No	No	Yes	No
Instructors can assign specific material on specific time and to different groups	Yes	No	Yes	No
Allows instructor to mark assignments or quizzes online	No	No	Yes	No
Allows asynchronous communication (e.g. email, discussion groups)	Yes	Yes	Yes	Yes
Allows synchronous communication (e.g. chat line, whiteboard)	Yes	Yes	Yes	Yes

Table 5.2 (Part I) – Comparison of WBT Tools



Context search and automatic index building	No	No	Yes	No
Incorporates help pages and documentation	Yes	Yes	Yes	Yes
Allows changes in the help pages (customization)	No	No	Yes	Yes

**Table 5.2 (Part II) – Comparison of WBT Tools**

The price of the products may also be a key factor in the decision-making process. The following prices were extracted from the SCOET/CCTT/OLT [14] study:

1. The first group of respondents (Group 1) consisted of 100 individuals who were randomly selected from a list of all individuals who had been employed by the company in the past 12 months. The second group (Group 2) consisted of 100 individuals who were randomly selected from a list of all individuals who had been employed by the company in the past 12 months. The third group (Group 3) consisted of 100 individuals who were randomly selected from a list of all individuals who had been employed by the company in the past 12 months.

Table 5.3 – Comparison of WBT Prices

The information provided in this chapter may simplify the research process (and reduce research time) for instructors and administrators who are investigating Web-Based Training tools.

## **Chapter 6**

# **The Grades Application**

### **6.1 Overview and Architecture**

Most post-secondary courses, delivered on the Web or not, have a fundamental element – grades management. Web-based courses may incorporate in its universe a grades management system, such as the products presented in Chapters 3 and 4. Another example was developed for this thesis – The Grades Application.

Grades Application is an uncomplicated, yet effective solution for using the Web to manage, calculate and consult student's marks. This system is based on the actual needs of the undergraduate course "Computers in Engineering", offered by McGill University and previously mentioned in Chapters 1 and 2.

By using the open architecture of the Web and standard programming and mark-up languages such as JavaScript, Perl and HTML, the system will run in most computers available in universities around the world. Furthermore, since a JavaScript-compliant browser is the only piece of software required for instructors and students to use the programs, there is minimal, if any, installation, training, or money required to start using the system.

As illustrated in Figure 6.1, the system has three distinct phases during the semester: an initial setup, the marking procedure and the final marks calculation.

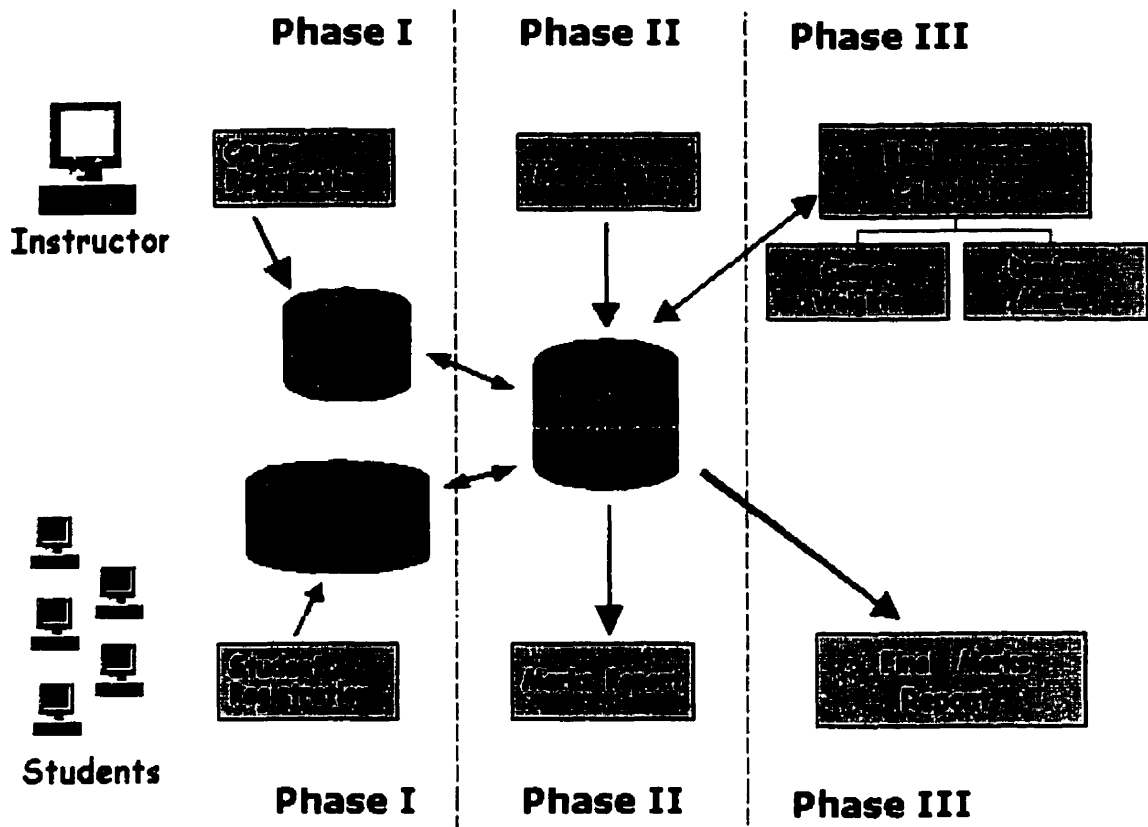


Figure 6.1 – Grades Application Overview

From the instructor's point of view, a Web page linking all modules of the Grades Application can be created. Using hyperlinks on the course home page, instructors give students access to specific modules of the Grades Application, such as Student Registration and Marks Report. The following sections describe in detail the procedures and files that form the Grades Application's architecture.

## 6.2 Procedures

As shown in Figure 6.2, the Grades Application uses the client-server approach with some of the procedures, particularly forms validation, being executed on the client side using JavaScript. This approach optimises the use of the network, by doing most of the

calculations and validation locally on the client without server intervention.

Once the data is considered correct, it is sent to the server through standard Common Gateway Interface (CGI). On the server, CGI scripts written in standard Perl language update the system files. CGI scripts are programs that receive their starting commands and parameters from a Web page that uses an HTML form. The CGI program takes the request from the user, performs server-side operations such as files updates and sends back responses in HTML format to the Web client.

Figure 6.2 illustrates the JavaScript and CGI programming environment used by the Grades Application.

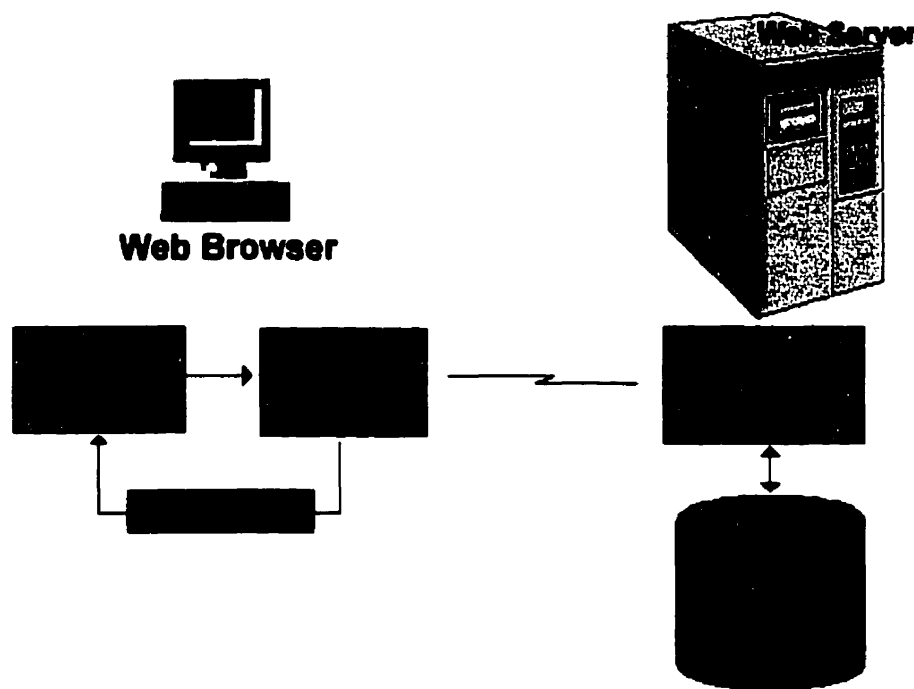


Figure 6.2 – Grades Application Client-Server Architecture

From instructor and student perspective the sole program they use is a Web browser, from where they fill in HTML forms and point and click throughout the system's pages.

## 6.2.1 Phase I - Course Registration and Student Registration

At the beginning of the term the instructor must set up the course by defining the following parameters:

- course name
- number of assignments
- number of mid-terms
- if there is a final exam
- number and description of other evaluation criteria, e.g. participation in class

Figure 6.3 illustrates an example of the Course Registration Web page:

Course Registration Microsoft Internet Explorer

### Course Registration

Use the following form to setup a course. Press the **Register** button to send the registration.

Course Name:	<input type="text"/>
Assignments:	<input type="text" value="0"/>
Mid-terms:	<input type="radio"/> Zero <input checked="" type="radio"/> One <input type="radio"/> Two <input type="radio"/> Three
Other Criteria:	<input type="text" value="0"/> Names: <input type="text"/> <input type="text"/>
Final Exam:	<input checked="" type="radio"/> Yes <input type="radio"/> No
	<input type="text"/> <input type="text"/> <input type="text"/>

Copyright Simone Spiller  
Last revised: November 14, 1997

Figure 6.3 – Course Registration Page

Before submitting the parameters to the server, JavaScript will validate the data. On the server side, a Perl script receives the validated data, creates the course initialization file and

stores the data provided by the instructor. In addition, the other two files required by the system are created: the course list and the marks file.

Once the course is created by the instructor, students will be able to register in the course. Students are asked to type their name, student number and e-mail address. In order to be able to verify their marks through the system later on, they are also required to specify a password.

Figure 6.4 shows a sample registration form presented to students.

**Student Registration**

Enter your information below and click on **Register** to

<b>Last Name:</b>	Smith
<b>First Name and Middle Initials:</b>	John
<b>E-Mail Address:</b>	smith (e.g. john@lan.mcgill.ca)
<b>Student #:</b>	9898557 (e.g. 9612345)
<b>Password:</b>	smith007
<b>Confirm Password:</b>	smith007
	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

Figure 6.4 – Student Registration Page

As with all electronic forms within the system, a validation routine (JavaScript) analyses if all the fields were correctly completed and sends them to the server, where a CGI script will update the course list file (COURSE.LST).

## 6.2.2 Phase II - Marks Entry and Marks Report

Throughout the semester, as assignments and other items are evaluated, instructors (or markers) can enter the marks.

Figure 6.5 shows an example of the form used by instructors or markers to enter the marks into the Grades Application.

CS 208 - Computers in Engineering

Marks Entry for Smith, John (9898557)

Assignment 1 :	100
Assignment 2 :	80
Assignment 3 :	
Assignment 4 :	
Mid-term Exam :	70
Final Exam :	
Other (Quizzes) :	

Figure 6.5 – Marks Entry

As marks are entered on the system via the Marks Entry procedure, students can use the Marks Report module to obtain their respective marks. In order to ensure privacy, a student will only be able to check his or her own marks by providing a valid password which was defined at registration time.

As an example, Figure 6.6 shows a page with the results obtained by student 9898557 after two assignments and one mid-term have been marked.

Netscape [Marks Report]

## CS 208 - Computers in Engineering

---

Marks report for **Smith, John (9898557)**

Assignment 1 :	100
Assignment 2 :	80
Assignment 3 :	
Assignment 4 :	
Mid-term Exam :	70
Final Exam :	
Other (Quizzes) :	

Figure 6.6 – Student Marks Report

### 6.2.3 Phase III - Final Marks Calculation and Final Marks Report

Once all evaluation criteria are entered on the system, the instructor can proceed with the last phase of the Grades Application, the Final Marks Calculation.



Figure 6.7 shows the main form of the Final Marks Calculation procedure:

**Final Marks Calculation**

Assignments: 20 %

Mid-term Exams: 20 %

Other: 10 %

Final Exam: 50 %

**Total: 100 %**

Assign.01: 25 % Assign.07: %

Assign.02: 25 % Assign.08: %

Assign.03: 25 % Assign.09: %

Assign.04: 25 % Assign.10: %

Assign.05: % Assign.11: %

Assign.06: % Assign.12: %

**Total: 100 %**

Figure 6.7 – Final Marks Calculation (Assignments)

In order to calculate the final marks, the instructor must first specify the weights for each of the evaluation criterion. By default, the system will equally distribute the weight amongst all criteria. The instructor can, however, change them at anytime during the final marks calculation phase by using the marks Change Weight form as shown in the previous figure.

Note that in the example shown in Figure 6.7 the assignments are worth 20 per cent, the mid-term exam 20 per cent, the final exam 50 per cent and, finally, other criteria account for 10 per cent of the final mark. Also note on the right that each assignment has the same weight - 25 per cent. By clicking on the hyperlink Assignments, Mid-term, Other, or Final on the left side of the form, the weight distribution for the respective criteria will appear on

the right side of the form, allowing the instructor to change it.

Figure 6.8 shows an example of an hypothetical case where two mid-terms were assigned, both having the same weight:

**Final Marks Calculation**

Component	Weight (%)
Assignments	20
Mid-term Exams	20
Other	10
Final Exam	50
<b>Total</b>	<b>100</b>

Component	Weight (%)
First mid-term	50
Second mid-term	50
Third mid-term	
<b>Total</b>	<b>100</b>

Figure 6.8 – Final Marks Calculation (Mid-term Exams)

All forms on the Weight Marks procedure are first validated on the client side by JavaScript procedures and then sent to the server, as shown in Figure 6.2 previously in this chapter.

After defining how much each component of the final mark is worth, the instructor can ask the Grades Application to display the marks, including the final marks, calculated based on the weights distribution as previously defined.

Figure 6.9 shows the list of final marks and their composing marks.

□ Marks Calculation Microsoft Internet Explorer

### Final Marks Calculation

Student	A 1	A 2	A 3	A 4	Mid-term	Quizzes	Final Exam	Final Mark
9901010 - <u>Alves, Bert</u>	100	90	80	70	85	78	60	71.8
9998999 - <u>Baldwin, Charlie</u>	80	80	70	80	80	75	90	84
9809091 - <u>Brown, James</u>	30	75	90	90	90	80	80	80.25
9809987 - <u>Chuckles, Ray</u>	70	90	50	100	80	100	90	86.5
9800901 - <u>Johnson, Mike</u>	45	80	70	70	80	80	100	87.25
9914576 - <u>Mathews, Mary</u>	87	90	90	80	80	70	60	70.35
9909765 - <u>Richards, Ann</u>	90	100	70	60	90	90	70	78
9911111 - <u>Simpson, Homer</u>	70	45	70	50	60	40	20	37.75
9898557 - <u>Smith, John</u>	100	80	100	80	70	100	90	87

Figure 6.9 – Final Marks Calculation (Display Marks)

Since the marks are not yet uploaded to the server, the instructor can change the weights and ask the system to recalculate the final marks.

Once the final marks have been calculated and the instructor is satisfied with the results, she can upload the marks to the server, making marks automatically available to the students. Again, using a password-validated procedure, students will check their results.

Figure 6.10 shows an example of the Final Marks Report presented to students at the end of the semester.

Final Marks Report Microsoft Internet Explorer

## CS 208 - Computers in Engineering

---

Final Marks report for **Smith, John (9898557)**

<b>Assignment 1 :</b>	100
<b>Assignment 2 :</b>	80
<b>Assignment 3 :</b>	100
<b>Assignment 4 :</b>	80
<b>Mid-term Exam :</b>	70
<b>Final Exam :</b>	90
<b>Other (Quizzes) :</b>	100

Figure 6.10 – Final Marks Report

## 6.3 File Structure

Three files are created on the server side as part of the Course Registration procedure: the initialization file (COURSE.INI), the course list file (COURSE.LST) and the course marks file (COURSE.MKS).

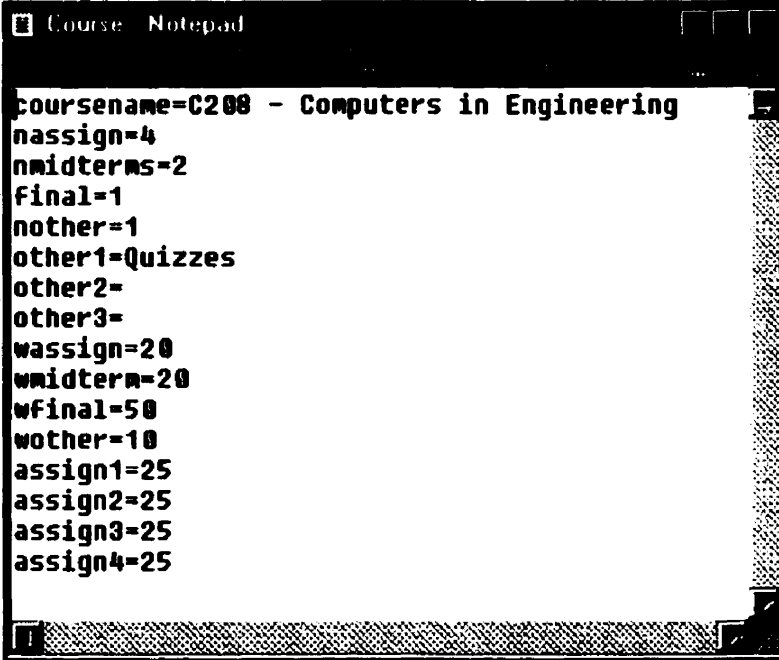
### 6.3.1 Initialization File: COURSE.INI

The course initialization file contains the parameters specified by the professor for that course. They are:

- course name
- number of assignments
- number of mid-terms

- if there is a final exam
- number and description of other evaluation criteria, e.g. participation in class

Figure 6.11 shows an example of an initialization file for a course named Computers in Engineering, where students will submit four assignments, make one mid-term and one final exam. They will also be evaluated by an extra criterion called Quizzes.



```
coursename=C200 - Computers in Engineering
nassign=4
nmidterms=2
final=1
nother=1
other1=Quizzes
other2=
other3=
wassign=20
wmidterm=20
wfinal=50
wother=10
assign1=25
assign2=25
assign3=25
assign4=25
```

Figure 6.11 – Course Initialization File (COURSE.INI)

The weights for each evaluation criteria in the example shown on Figure 6.12 have been defined by the instructor via the Course Registration procedure as described in the topic 6.2.1. Note, for example, that the final exam is worth 50 per cent of the final mark, while assignments are worth 20 per cent, with each assignment having the same weight.

### 6.3.2 Course List File: COURSE.LST

The course list contains information about students registered in the course. The class list

contains:

- Student's name
- Student's number
- E-mail address
- Password to view marks

Figure 6.12 shows an extract of a class list with the above fields colon delimited.

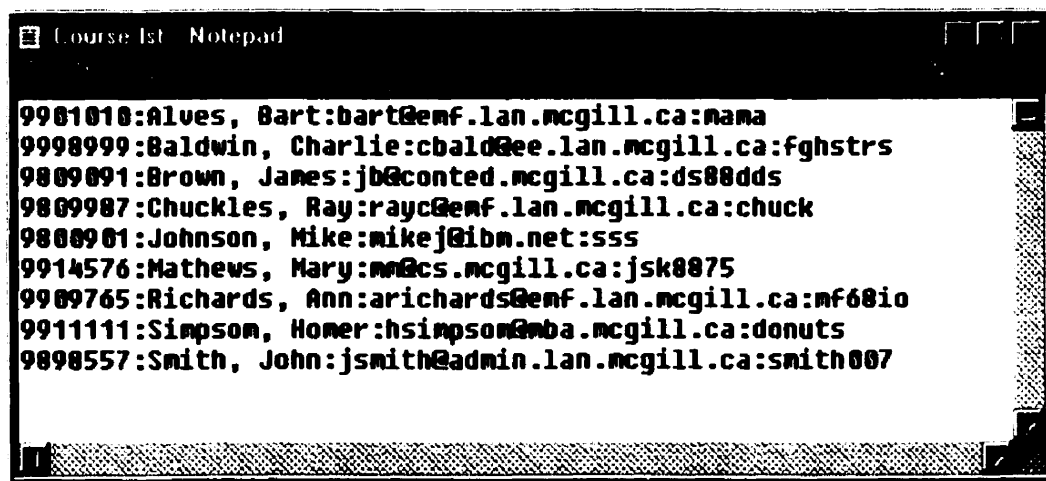


Figure 6.12 – Course List File (COURSE.LST)

### 6.3.3 Course Marks File: COURSE.MKS

The layout of this file is determined by the number of evaluation criteria defined by the instructor at the Course Registration procedure.

Anytime an instructor or a marker marks an assignment or any item, the marks file is updated to reflect the new mark. Also, at the end of the semester, this file will store the final mark calculated through the Marks Calculation procedure described in topic 6.2.3.

Figure 6.13 shows an example of the marks file for the Computers in Engineering course with marks delimited by colons. Please note that the number of evaluation items conforms with the parameters of the initialization file (COURSE.INI). Also note that the final mark – the last one for each line – is the result of the weights (also specified at COURSE.INI) applied to each of the assignments, exams and quizzes.

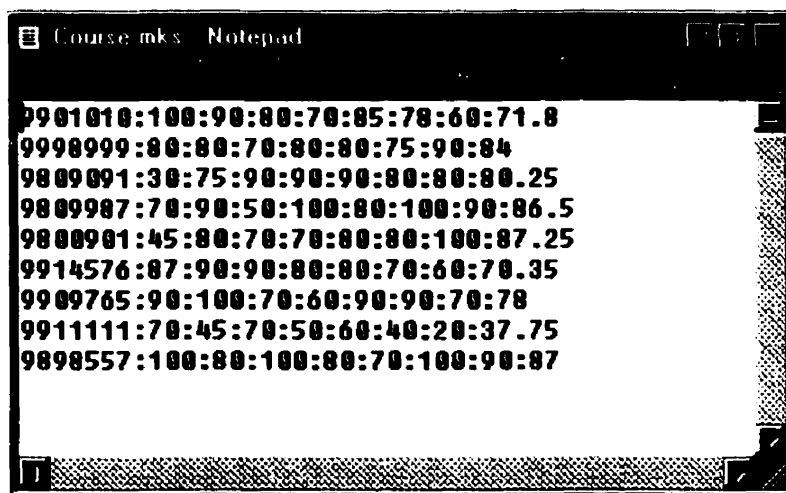


Figure 6.13 – Course Marks File (COURSE.MKS)

Let us look at the results for student number 9898557. An analysis of his situation according to the marks file shown in the previous figure indicates that he obtained the following marks for each respective assignment: 100, 80, 100 and 80, which gives a 90% overall result. Applying the 20% weight to the assignments total, 20% to the mid-term exam, 10% for the quizzes and 50% for the final exam, the student obtained a final mark of 87 per cent.

Figure 6.14 details this calculation.

Assignment 1:	100	x	25%	=	25
Assignment 2:	80	x	25%	=	20
Assignment 3:	100	x	25%	=	25
Assignment 4:	80	x	25%	=	20
					----
					90
Assignments Total :	90	x	20%	=	18
Mid-term exam :	70	x	20%	=	14
Other (Quizzes) :	100	x	10%	=	10
Final-exam :	90	x	50%	=	45
					----
					Final Mark : 87

Figure 6.14 – Sample Final Mark Calculation

This chapter presented the interface, programs and files that make up the Grades Application. Some of the forms and the programs written in JavaScript and Perl are listed in the Appendix. All modules of the Grades Application are available online for FTP at the following address: <http://www.cs.mcgill.ca/~spiller/thesis/GradesApplication.htm>.



## Chapter 7

# Conclusions and Future Work

### 7.1 Conclusions

By highlighting the interactive and collaborative characteristics of the Web throughout all chapters, this thesis encouraged readers to use the World-Wide Web as a medium to create and deliver courses. Positively, the Web allows for the accomplishment of learning objectives combined with the best instructional model as analyzed in [1], from where the illustration shown in Figure 7.1 was taken.

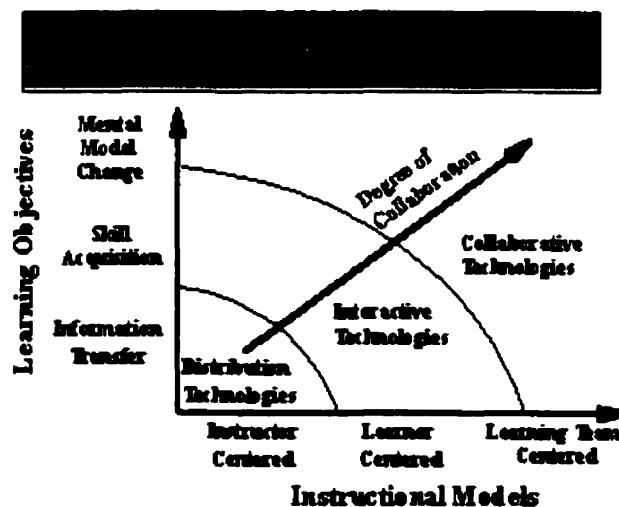


Figure 7.1 – Instructional Models

One may find that an important drawback in using the Web to disseminate knowledge is bandwidth. Indeed, given the multimedia character of the Web, an overload of texts, graphics, photographs, audio and video may endanger the effectiveness of the learning process. Fortunately, with promises of eliminating information highway traffic problems, the next-generation network – one of the main motivations for this work – is under development in North America.

CA\*net II, Canada's next generation Internet initiative, launched this year in Ottawa, represents the next crucial step in the development of the Internet in Canada. CA\*net II is geared to the needs of Canadian universities, research institutions, industry, and communications carriers. CA\*net II will provide these communities with dedicated high-speed access – separate from the public Internet, using a high performance IP/ATM network – to conduct research, develop applications and test new products and services that will add value for the public.

Clearly, even if it remains just an option, distance education will be an increasingly important learning method in the 21<sup>st</sup> Century. As retraining becomes more vital, and job skills change, distance education will also play a major role in continuing education for all professionals.

## **7.2 Future Work**

As presented in Chapter 6, a client-server Internet application to manage, calculate and view marks was developed. Taking advantage of the fact that the application uses the open architecture of the Web and standard programming languages, other modules can be easily incorporated to the system, as suggested below:

**Convert Perl programs into Java Applets** – This conversion, given its I/O capabilities, can eliminate server dependencies making the Grades Application completely platform-independent.

**Develop Graphical Reports, Analysis and Statistics** –Java or Perl programs could provide the instructor with graphical tools to better visualize the performance of students and to compare their results with those of students who have taken previous editions of the course. No changes are necessary to the format of the files currently in use by the system.

**Develop an export feature to EDI or to the McGill-EMS** – Develop an export feature to the standard EDI (Electronic Data Interchange) format commonly used in educational institutions, or to the McGill-EMS (Electronic Marks System) format. This would allow direct updates to the student record system.

**Evaluate and apply Electronic Performance Support Systems (EPSS)** – EPSS provide employees with the information, advice and learning experiences they need to upgrade their skills with minimal support from others [30]. EPSS provide the electronic infrastructure that captures, stores and distributes knowledge throughout an organization to enable its professionals to learn better and faster. As EPSS are becoming common in corporate environments, interesting research might be conducted analyzing the use EPSS in the educational environment.

# **Bibliography**

- [1] Lotus Learning White Paper - Distributed Learning: Approaches, Technologies and Solutions,  
<http://www2.lotus.com/education.nsf/641021c7cb140a3c852564060012ce06/e8f7e161c722898f8525643d00609cc1?OpenDocument>
- [2] Herther, Nancy K., Education over the Web - Distance Learning and the Information Professional, ONLINE Magazine p.63-72, September-October 1997.
- [3] MUSIC/SP Operating System, User's Reference Guide and World-Wide Web Support on MUSIC/SP, 1996.
- [4] Internet World Magazine, September 1997.
- [5] Murray W. Goldberg and Sasan Salari, "An Update on WebCT (World-Wide-Web Course Tools) - a Tool for the Creation of Sophisticated Web-Based Learning Environments", Proceedings of NAUWeb '97 - Current Practices in Web-Based Course Development, June 12 - 15, 1997, Flagstaff, Arizona.
- [6] Green, Kathleen. "Nontraditional Education: Alternative Ways to Earn Your Credentials", Occupational Outlook Quarterly 40, Summer 1996.
- [7] Murray W. Goldberg, "CALOS: First Results From an Experiment in Computer-Aided Learning". Accepted for publication in the Proceedings of the ACM's 28th SIGCSE Technical Symposium on Computer Science Education, 1997.

- [8] Simbandumwe, J.P., Tools for Developing Interactive Academic Web Courses, <http://www.umanitoba.ca/ip/tools/courseware/index.html>
- [9] Solis PathWay Home Page <http://www.solis.com/>
- [10] The Economist, Survey Universities, -  
<http://www.economist.com/editorial/freeforall/5-10-97/unil.html>
- [11] Lotus Notes, Domino and LearningSpace Slides - Presentation given by Lotus Sales Representatives at McGill University, May 1997.
- [12] Virtual-U Home Page <http://virtual-u.cs.sfu.ca/./vuweb/>
- [13] PC Week Labs evaluates Internet-based training systems August 18, 1997 <http://www8.zdnet.com/pcweek/reviews/ibt.html>
- [14] Home Page of The British Columbia Standing Committee on Educational Technology, The Centre for Curriculum, Transfer and Technology and The Office of Learning Technologies  
<http://www.ctt.bc.ca/landonline/>
- [15] Macromedia Home Page about Learning  
<http://www.macromedia.com/learning/>
- [16] Lotus Learning Home Page,  
<http://www.lotus.com/education/learningspace.nsf>
- [17] Book Chapter, "Using a Web-Based Course Authoring Tool to Develop Sophisticated Web-Based Courses", Editor: Badrul H Khan, Publisher: Educational Technologies Press. July, 1996.
- [18] Dertouzos, M., "What Will Be", HarperCollins, 1997.
- [19] PC Magazine Online  
<http://www.zdnet.com/pcmag/features/webedit/edchoice.htm>

- [20] Kilby, T., WBT Information Center - WBT Advantages and Disadvantages - [http://www.filename.com/wbt/\\_private/advdis.htm](http://www.filename.com/wbt/_private/advdis.htm)
- [21] Harvey, D., and Milheim, W. - Web-Based Instruction Resource Site, <http://www.personal.psu.edu/wdm2/main.htm>
- [22] Makulowich, J., Training the Internet Trainer - On The Internet, An International Publication of The Internet Society, July/August 1996, p.11.
- [23] Scientific American, The Computer in The 21<sup>st</sup> Century - Special Issue 1995 - Computers, Networks and Education, p.148.
- [24] Turcotte, S., Séminaire du CRIM. Aperçu des technologies d'aide à l'apprentissage, 1996.
- [25] van Hoff, A., Shaio, S., and Starbuck, O., Hooked on Java, Addison Wesley Developers Press, 1996.
- [26] Davis, Stephen R., Learn Java Now (Visual J++), Microsoft Press, 1997.
- [27] Wallace, Timothy's Home Page about JavaScript, <http://www.essex1.com/people/timothy/>
- [28] JavaScript Index, <http://www.sapphire.co.uk/javascript/docs.html>
- [29] Herrmann, E., Teach Yourself CGI Programing with Perl 5 in a Week, SamsNet Publishing, 1996.
- [30] The Electronic Performance Support System Web Site, <http://www.epss.com/>

# Appendix - The Grades Application - Forms and Programs

courseReg.htm

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">

<html>

<head>

<meta http-equiv="Content-Type"

content="text/html; charset=iso-8859-1">

<meta name="GENERATOR" content="Microsoft FrontPage 2.0">

<title>Course Registration</title>

<meta name="FORMATTER" content="Microsoft FrontPage 2.0">

</head>

<body bgcolor="#FFFFFF">

<script language="JavaScript">

<!--

//      *****

//      *****      The Grades Application      *****

//      *****      Copyright Simone Spiller, 1997      *****

//      *****      Last Update: 1997/Nov/14      *****

//      *****

/* The following JavaScript functions validate data related to

/* the course before submitting fields to the

/* CGI (http://www.cs.mcgill.ca/~spiller/cgi-bin/coursereg.cgi)

/****** Validate entries and submit form to CGI

function validate(){

    if ( isClassName() && isOther())
```

```

{
    msg = "Please, click on the button OK to submit the registration.\n"

    if (confirm(msg)){
        submitted = 1;

        document.form1.submit(); // passes control to the CGI program
    }
    else {
        alert("Registration cancelled. Please, re-submit.")
    }
}
}

//***** Validates the NAME field
function isClassName()
{
    var str = document.form1.coursename.value;
    // Return false if name field is blank.
    if (str == "")
    {
        alert("Please enter the Course Name.")
        document.form1.coursename.focus(); // focus = where cursor is placed
        return false;
    }
    return true;
}

//***** Validates the OTHER criteria
function isOther()
{
    var nother = document.form1.nother.selectedIndex;
    //If no other criteria then return true
    if (nother == 0) {
        return true; // no need to test below
    }
}

```



```

//***** Test first other criteria
if (document.form1.other1.value == ""){
    alert("Please enter the name of the first extra criterium.");
    document.form1.other1.focus();
    return false;
}

//***** Test second other criteria
if (nother > 1) {
    if (document.form1.other2.value == ""){
        alert("Please enter the name of the second extra criterium.");
        document.form1.other2.focus();
        return false;
    }
}

//***** Test third other criteria
if (nother == 3) {
    if (document.form1.other3.value == ""){
        alert("Please enter the name of the third extra criterium.");
        document.form1.other3.focus();
        return false;
    }
}

return true;
}

//***** Help - Under construction
function help(){
    alert("Help - Under Construction!")
}

// --></script>

<h1>Course Registration</h1>

<hr>

<p>Use the following form to setup a course. Press the <strong>Register</strong>
button to send the registration.</p>

```

```

<form
action="http://www.cs.mcgill.ca/~spiller/cgi-bin/coursereg.cgi"
method="POST" name="form1" onsubmit="return false">
  <div align="center"><center><table border="1" width="75%"
  bgcolor="#FFFFCC">
    <tr>
      <td align="right" width="30%"><strong>Course Name: </strong></td>
      <td width="70%"><input type="text" size="30"
      maxlength="50" name="coursename"></td>
    </tr>
    <tr>
      <td align="right" width="30%"><strong>Assignments: </strong></td>
      <td width="70%"><select name="nassign" size="1">
        <option selected value="0">0</option>
        <option value="1">1</option>
        <option value="2">2</option>
        <option value="3">3</option>
        <option value="4">4</option>
        <option value="5">5</option>
        <option value="6">6</option>
        <option value="7">7</option>
        <option value="8">8</option>
        <option value="9">9</option>
        <option value="10">10</option>
        <option value="11">11</option>
        <option value="12">12</option>
      </select></td>
    </tr>
    <tr>
      <td align="right" width="30%"><strong>Mid-terms: </strong></td>
      <td width="70%"><input type="radio" name="nmidterms"
      value="0">Zero <input type="radio" checked
      name="nmidterms" value="1">One <input type="radio"

```

```

        name="nmidterms" value="2">Two <input type="radio"
        name="nmidterms" value="3">Three </td>
</tr>
<tr>
    <td align="right"><strong>Other Criteria:</strong> </td>
    <td><select name="nother" size="1">
        <option selected value="0">0</option>
        <option value="1">1</option>
        <option value="2">2</option>
        <option value="3">3</option>
    </select> <strong>Names: </strong><input type="text"
    size="6" maxlength="30" name="other1"> <input
    type="text" size="6" maxlength="30" name="other2"> <input
    type="text" size="6" maxlength="30" name="other3"></td>
</tr>
<tr>
    <td align="right" width="30%"><strong>Final Exam: </strong></td>
    <td width="70%"><input type="radio" checked
    name="final" value="1">Yes <input type="radio"
    name="final" value="0">No</td>
</tr>
<tr>
    <td>&nbsp;</td>
    <td><br>
        <input type="button" value="Register"
        onclick="validate()"> <input type="reset"
        value="Reset"> <input type="button" value="Help"
        onclick="help()"></td>
</tr>
</table>
</center></div>
</form>
<hr>

```

```

<h5 align="center">Copyright Simone Spiller<br>
Last revised: <!--webbot bot="TimeStamp" s-type="EDITED"
s-format="%B %d, %Y" startspan -->November 14, 1997<!--webbot
bot="TimeStamp" endspan i-checksum="41285" --></h5>
</body>
</html>

```

studentReg.htm

```

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">

<html>

<head>

<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">

<meta name="GENERATOR" content="Microsoft FrontPage 2.0">

<title>Student Registration</title>

<meta name="FORMATTER" content="Microsoft FrontPage 2.0">

</head>

<body bgcolor="#FFFFFF">

<script language="JavaScript"><!--
<!--
// *****
// *****      The Grades Application      *****
// *****      Copyright Simone Spiller, 1997      *****
// *****      Last Update: 1997/Nov/12      *****
// *****
//
/* The following JavaScript functions validate data related to
/* the student before submitting fields to the
/* CGI (http://www.cs.mcgill.ca/~spiller/cgi-bin/studentreg.cgi)
/* Note that the colon character (:) is a special character for COURSE.LST so there
/* is validation in all procedures to avoid user to enter ":"

//***** Validate entries and submit form to CGI

function validate(){
    if ( (isLastName() ) && (isFirstName()) && (isEmail()) && (isStudenttn()) && (isPW()) )

```

```

    {
        msg = "Please, double-check ALL fields.\n\nClick on the button OK to submit your
registration.\n"
        if (confirm(msg)){
            submitted = 1;
            document.form1.submit();
        }
        else {
            alert("Registration cancelled. Please, re-submit your registration.")
        }
    }
}

//***** Checks the NAME field
function isLastName()
{
    var str = document.form1.lastname.value;
    // Return false if name field is blank.
    if (str == "")
    {
        alert("Please enter your Last Name.")
        document.form1.lastname.focus();
        return false;
    }
    if (document.form1.lastname.value.indexOf(':',0) != -1)
    {
        alert("The fields in this form cannot contain the colon (:) character.")
        document.form1.lastname.select();
        document.form1.lastname.focus();
        return false;
    }
    else
    {
        return true;
    }
}

```

```

    }
}

function isFirstName()
{
    var str = document.form1.firstname.value;
    // Return false if name field is blank.
    if (str == "")
    {
        alert("Please enter your First Name.")
        document.form1.firstname.focus();
        return false;
    }
    if (document.form1.firstname.value.indexOf(':',0) != -1)
    {
        alert("The fields in this form cannot contain the colon (:) character.")
        document.form1.firstname.select();
        document.form1.firstname.focus();
        return false;
    }
    else
    {
        return true;
    }
}

//***** Checks Email

function isEmail()
{
    // Return false if e-mail field is blank

    if (document.form1.email.value == "")
    {
        alert("Please enter your Email address.")
        document.form1.email.focus();
    }
}

```

```

        return false;
    }

    // Return false if e-mail field does not contain a '@' and '.'
    if (document.form1.email.value.indexOf('@',0) == -1 ||
        document.form1.email.value.indexOf('.',0) == -1)
    {
        alert("The E-MAIL field requires a \"@\" and a \".\" be used.\n\nPlease re-enter your
e-mail address.")

        document.form1.email.select();
        document.form1.email.focus();

        return false;
    }

    if (document.form1.email.value.indexOf(':',0) != -1)
    {
        alert("The fields in this form cannot contain the colon (:) character.")

        document.form1.email.select();
        document.form1.email.focus();

        return false;
    }
else
    {
        return true;
    }
}

//***** Checks Student Number
function isStudentn()
{
    var str = document.form1.studentn.value;

    // Return false if number field is blank
    if (str == "")
    {
        alert("Please enter your Student Number.");

        document.form1.studentn.focus();
    }
}

```

```

        return false;
    }

    // Return false if characters are not '0-9' or '.'
    for (var i = 0; i < str.length; i++)
    {
        var ch = str.substring(i, i + 1);
        if ((ch < "0" || "9" < ch) && ch != '.')
        {
            alert("The Student number field accepts only numbers. Please re-enter your Student
Number.");
            document.form1.studentn.select();
            document.form1.studentn.focus();
            return false;
        }
    }

    // Return false if length <> 7
    str = document.form1.studentn.value ;
    if (str.length != 7)
    {
        alert("Please enter seven digits for the \"studentn\" field.");
        document.form1.studentn.select();
        document.form1.studentn.focus();
        return false;
    }

    return true;
}

//***** Checks the PW fields
function isPW()
{
    var str1 = document.form1.pw1.value;
    var str2 = document.form1.pw2.value;
    // Return false if PW field is blank.
    if (str1 == "")

```



```

    {
        alert("Please enter a Password")
        document.form1.pw1.focus();
        return false;
    }
// Return false if PW fields are different
if (str2 != str1)
    {
        alert("Both Passwords must match!\nPlease re-enter your password.")
        document.form1.pw2.select();
        document.form1.pw2.focus();
        return false;
    }
if (document.form1.pw1.value.indexOf(':',0) != -1 ||
    document.form1.pw2.value.indexOf(':',0) != -1)
    {
        alert("The fields in this form cannot contain the colon (:) character.")
        document.form1.pw1.select();
        document.form1.pw1.focus();
        return false;
    }
else
    {
        return true;
    }
}

//***** Help - Under construction
function help(){
    alert("Help - Under Construction!")
}

// --></script>
<h1>Student Registration</h1>
<hr>

```

```

<p>Enter your information below and click on <strong>Register </strong>to
submit registration. Please, fill in all fields.</p>

<p>&nbsp;</p>

<!--webbot bot="GeneratedScript" preview=" " startspan --><script
language="JavaScript"><!--
function FrontPage_Form1_Validator(theForm)
{
    if (theForm.studentn.value == "")
    {
        alert("Please enter a value for the \"studentn\" field.");
        theForm.studentn.focus();
        return (false);
    }
    if (theForm.studentn.value.length < 7)
    {
        alert("Please enter at least 7 characters in the \"studentn\" field.");
        theForm.studentn.focus();
        return (false);
    }
    if (theForm.studentn.value.length > 7)
    {
        alert("Please enter at most 7 characters in the \"studentn\" field.");
        theForm.studentn.focus();
        return (false);
    }
    return (true);
}
//--></script><!--webbot bot="GeneratedScript" endspan -->

<form
action="http://www.cs.mcgill.ca/~spiller/cgi-bin/studentreg.cgi"
method="POST" onsubmit="return FrontPage_Form1_Validator(this)"
name="FrontPage_Form1">

    <div align="center"><center><table border="1" width="90%"

```

```

bgcolor="#FFFFCC">

<tr>

  <td align="right" width="30%"><strong>Last Name: </strong></td>

  <td width="70%"><input type="text" size="30"

    maxlength="50" name="lastname"></td>

</tr>

<tr>

  <td align="right" width="30%"><strong>First Name and

    Middle Initials: </strong></td>

  <td width="70%"><input type="text" size="30"

    maxlength="50" name="firstname"></td>

</tr>

<tr>

  <td align="right" width="40%"><strong>E-Mail Address:

    </strong></td>

  <td width="70%"><input type="text" size="30"

    name="email"><strong> </strong><em>(e.g.

    john@lan.mcgill.ca)</em></td>

</tr>

<tr>

  <td align="right" width="30%"><strong>Student #: </strong></td>

  <td width="70%"><!--webbot bot="Validation"

    b-value-required="TRUE" i-minimum-length="7"

    i-maximum-length="7" --><input type="text" size="7"

    maxlength="7" name="studentn"> <em>(e.g. 9612345)</em></td>

</tr>

<tr>

  <td align="right" width="30%"><strong>Password: </strong></td>

  <td width="70%"><input type="text" size="15"

    maxlength="40" name="pw1"> <strong>Confirm Password :

    </strong><input type="text" size="15" maxlength="40"

    name="pw2"></td>

</tr>

```

```

        <tr>
            <td>&nbsp;</td>
            <td><br>
                <input type="button" value="Register"
                onclick="validate()"> <input type="reset"
                value="Reset"> <input type="button" value="Help"
                onclick="help()"></td>
        </tr>
    </table>
</center></div>
</form>
<hr>
<h5 align="center">Copyright Simone Spiller<br>
Last revised: <!--webbot bot="TimeStamp" s-type="EDITED"
s-format="%B %d, %Y" startspan -->November 14, 1997<!--webbot
bot="TimeStamp" endspan i-checksum="41285" --></h5>
</body>
</html>

```

---

weightmarks.htm

FR -

```

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
<meta name="FORMATTER" content="Microsoft FrontPage 2.0">
<meta name="GENERATOR" content="Microsoft FrontPage 2.0">
<title>Weights</title>
</head>
<body bgcolor="#FFFFCC" onLoad="getInitialValues()">
<script language="JavaScript">
<!--

```

```

// *****
// *****      The Grades Application      *****
// *****      Copyright Simone Spiller, 1997      *****
// *****      Last Update: 1997/Nov/12      *****
// *****

/* The following JavaScript functions validate data related to
the weights entry. The weights must be numbers and cannot sum more than 100%
The weights will be then saved in COURSE.INI
CGI (http://www.cs.mcgill.ca/~spiller/cgi-bin/courseregII.cgi)

***** Validate entries and submit form to CGI

function validate(){
    if ( isNumber(document.form1.assign.value) && isNumber(document.form1.midterm.value) &&
        isNumber(document.form1.other.value) && isNumber(document.form1.finalexam.value))
    {
        newtotal = 0

        var assign = 1 * document.form1.assign.value;
        var midterm = 1 * document.form1.midterm.value
        var other = 1 * document.form1.other.value;
        var finalexam = 1 * document.form1.finalexam.value;
        newtotal = assign + midterm + other + finalexam;

        alert(newtotal)

        if (newtotal == 100) {
            msg = "Please, click on the button OK to submit the changes.\n"
            if (confirm(msg)) {
                submitted = 1;
                document.form1.submit();
            }
        }
        else {
            alert("The sum must be 100.")
        }
    }
}

```

```

else {
    alert("All values must be numeric");
}
}

//***** Return false if characters are not '0-9'
function isNumber(str)
{
    if (str.length == 0){
        return false;
    }

    for (var i = 0; i < str.length; i++)
    {
        var ch = str.substring(i, i + 1);
        if ((ch < "0" || "9" < ch))
        {
            return false;
        }
    }

    return true;
}

//***** Get initial values
function getInitialValues()
{
    //    alert("To be implemented. Go get .ini on server");
    //    NEED:    initial values for the four fields
    //            nassign
    //            nmidterm
    //            nother
    //            final(YesorNo)

    // if mid-term does not exist, use hidden instead of text while creating field and
    // initialize it with 0 (this will prevent error in validate()) or use form1[n].elements[n]
    // up to n-elements in validate() (this is cleaner but more complicated to start...

```

```

//      var doc=parent.frames[0].document;
//      doc.open();
//      doc.write("<HTML><HEAD><TITLE>"+ "Default" + "</TITLE></HEAD>");
//      doc.write("<BODY BGCOLOR=FFFFFF LINK=FF0000 VLINK=FF0000 ALINK=FF0000>");
//      doc.write("<CENTER><H1><FONT COLOR=800000>"+ "Default Values from .INI Presented!"
+ "</FONT></H1>");
//      doc.write("</CENTER></BODY></HTML>");
//      doc.close();
    }
</script>

<form action="chweights.cgi" method="POST" name="form1" onsubmit="return false">
    <div align="left"><table border="0" width="100%">
        <tr>
            <td align="right" width="50%">
                <a href="FR-weight02.htm" target="criteria">Assignments</a> :
                <p><a href="FR-weight03.htm" target="criteria">Mid-term Exams</a> : </p>
                <p><a href="FR-weight04.htm" target="criteria">Other</a> : </p>
                <p>Final Exam : </p></td>
            <td width="50%">
                <input type="text" size="5" maxlength="5" name="assign">
                <strong>%</strong><p>
                <input type="text" size="5" maxlength="5" name="midterm">
                <strong>%</strong></p><p>
                <input type="text" size="5" maxlength="5" name="other">
                <strong>%</strong></p><p>
                <input type="text" size="5" maxlength="5" name="finalexam">
                <strong>%</strong></p>
            </td>
        </tr>
    </table>
    <p align="center"><strong> Total : &nbsp;&nbsp;100 %&nbsp;&nbsp;</strong></p>
</div>
<CENTER>

```

```

        <p align="center">
        <input type="button" value="Change" onclick="validate()">
        <input type="reset" value="Default"></p>
    </CENTER>
</form>
</body>
</html>

```

---

coursereg.cgi

```

#!/usr/local/bin/perl

#
# .....
# .....      The Grades Application      .....
# .....      Copyright Simone Spiller, 1997 .....
# .....      Last Update: 1997/Nov/14 .....
# .....
#
# COURSE.INI is generated using this CGI
# Variables are received from HTML Form courseReg.htm
#
# Indicates path and initializes variables

$inilocation = "http://www.cs.mcgill.ca/~spiller/course.ini";
$inipath = "../course.ini";
$cgilocation = "http://www.cs.mcgill.ca/~spiller/cgi-bin/coursereg.cgi";

# Tell browser

print ("Content-Type: text/html\n\n");

# Receive info from form (courseReg.htm)

read(STDIN, $buffer, $ENV{'CONTENT_LENGTH'});

```



# Process info from form - Splits buffer -

# Standard procedure described in

# Herrmann, E., Teach Yourself CGI Programing with Perl 5 in a Week, SamsNet Publishing, 1996

```
@pairs = split(/&/, $buffer);
```

```
foreach $pair (@pairs) {
```

```
    ($name, $value) = split(/=/, $pair);
```

```
    $value =~ tr/+// ;
```

```
    $value =~ s/%([a-fA-F0-9]{2})/pack("C", hex($1))/eg;
```

```
    $value =~ s/<!--(.|\n)*-->//g;
```

```
    $value =~ s/<([>]|\n)*>//g;
```

```
    $INPUT{$name} = $value;
```

```
}
```

# Write info to INI file

```
open (INI,">$inipath");
```

```
print INI ("coursename=$INPUT{'coursename'}\n");
```

```
print INI ("nassign=$INPUT{'nassign'}\n");
```

```
print INI ("nmidterms=$INPUT{'nmidterms'}\n");
```

```
print INI ("final=$INPUT{'final'}\n");
```

```
print INI ("nother=$INPUT{'nother'}\n");
```

```
print INI ("other1=$INPUT{'other1'}\n");
```

```
print INI ("other2=$INPUT{'other2'}\n");
```

```
print INI ("other3=$INPUT{'other3'}\n");
```

```
close (INI);
```

# Print Follow Up HTML

```
print ("<html><head><title>Course Registration</title></head>\n");
```

```
print ("<BODY>\n");
```

```
print ("<P align=center>Course $INPUT{'coursename'} has been created.</p>\n");
```

```

print("<P align=center>Students may now register.</p>\n");
print("</body></html>\n");

```

---

readlstcgi

```

#!/usr/local/bin/perl

#
# *****
# *****      The Grades Application      *****
# *****      Copyright Simone Spiller, 1997      *****
# *****      Last Update: 1997/Nov/14      *****
# *****
#
# COURSE.LST is read using this CGI
# Variables are read from COURSE.LST and an
# HTML file is then generated
# This is the basic program to the MARKS ENTRY module
# Indicates path and initializes variables
$lstpath = "../course.lst";
open($LSTFILE, $lstpath);
# Start HTML file
print("Content-Type: text/html\n\n");
print("<HTML>\n");
# Loop through .lst file
while (<$LSTFILE>) {
#     chop;
#     ($item, $content) = split(/=/, $_, 2) ;
#     $item_list{$item} = $content;
#     print("$item = $content<BR>\n");
#     print("$_ <BR>\n");
}

```

---

readini.cgi

```

#!/usr/local/bin/perl

#
# *****
#          The Grades Application          *****
#          Copyright Simone Spiller, 1997  *****
#          Last Update: 1997/Nov/14       *****
# *****

# COURSE.INI is read using this CGI

# Variables are read from COURSE.INI and a

# HTML file is then generated

# This is the basic program to the MARKS ENTRY and FINAL MARKS

# CALCULATION module

# Indicates path and initializes variables

$inipath = "../course.ini";
open($INIFILE, $inipath);

# Start HTML file

print ("Content-Type: text/html\n\n");
print ("<HTML>\n");

# Loop through .ini file
while (<$INIFILE>) {
    chop;
    ($item, $content) = split(/=/, $_, 2) ;
    $item_list{$item} = $content;
    print (" $item = $content<BR>\n");
##    print ("$_ <BR>\n");
}

_____ studentreg.cgi

#!/usr/local/bin/perl

#
# *****
#          The Grades Application          *****

```

```

# ***** Copyright Simone Spiller, 1997 *****
# ***** Last Update: 1997/Nov/14 *****
# *****

# COURSE.LST is generated using this CGI

# Variables are received from HTML Form studentReg.htm

# Indicates path and initializes variables

$lstlocation = "http://www.cs.mcgill.ca/~spiller/course.lst";
$lstpath = "../course.lst";
$cgilocation = "http://www.cs.mcgill.ca/~spiller/cgi-bin/studentreg.cgi";

# Receive info from form (studentReg.htm)
read(STDIN, $buffer, $ENV{'CONTENT_LENGTH'});

# Process info from form - Splits buffer -

# Standard procedure described in

# Herrmann, E., Teach Yourself CGI Programing with Perl 5 in a Week, SamsNet Publishing,
1996

@pairs = split(/&/, $buffer);
foreach $pair (@pairs) {
    ($name, $value) = split(/=/, $pair);
    $value =~ tr/+// ;
    $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
    $value =~ s/<!--(\\.\\n)*-->//g;
    $value =~ s/<([>]|\\n)*>//g;
    $INPUT{$name} = $value;
}

# Write info to Course List file

open (LST,">>$lstpath");

print          LST          ("{$INPUT{'studentn'}:$INPUT{'lastname'},
$INPUT{'firstname'}:$INPUT{'email'}:$INPUT{'pw1'}\\n");

close (LST);

# Print Follow Up HTML

print ("Content-Type: text/html\\n\\n");
print ("<html><head><title>Student Registration</title></head>\\n");

```

```
print("<BODY>\n");
```

```
print("<P ALIGN=CENTER>You have been added to the course.</P>\n");
```

```
print("<P ALIGN=CENTER>Please, remember your password ($INPUT{'pw1'})</P>\n");
```

```
print("</body></html>\n");
```

---