Web-Based Courses: The Assiniboine Model

Stephen Downes, Information Architect, University of Alberta; former distance education design specialist, Assiniboine Community College.

Note: This article is based on a paper presented to NAWEB September, 1997. The article, in its original format, may be found at Downes' website: http://www.atl.ualberta.ca/downes/naweb/am.htm

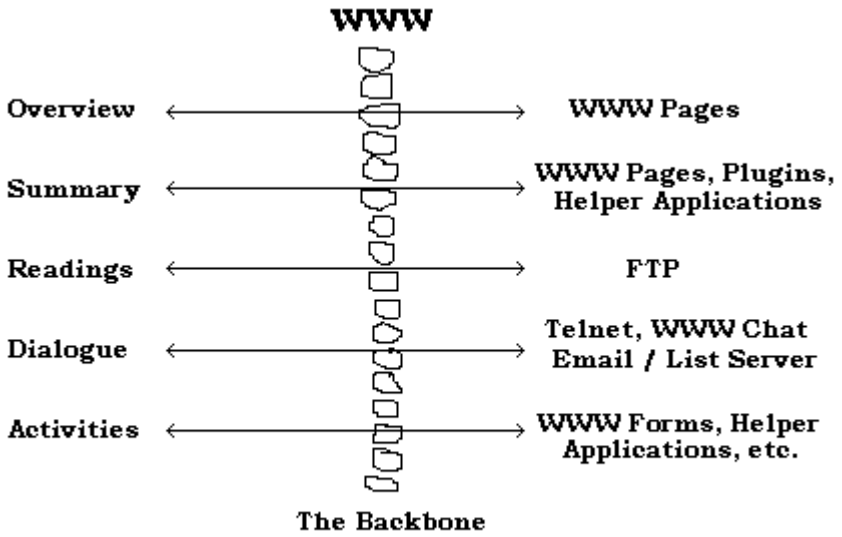
Overview

This paper describes the model used at Assiniboine Community College for the creation of web-based courses. This model was developed based on our experience developing on-line courses, on information gathered from other on-line courses, and on discussions with other people engaged in on-line course development.

To provide a contrast with existing models, this paper follows the structure of the *East-West Project Course Developer's Standards Guide* (http://teleeducation.nb.ca/eastwest/standards). This guide describes a format with which we have had experience (see, for example, http://www.assiniboinec.mb.ca/cae/banner.htm and http://www.assiniboinec.mb.ca/ic15) and from which we have chosen to depart for what we feel are some well-founded reasons.

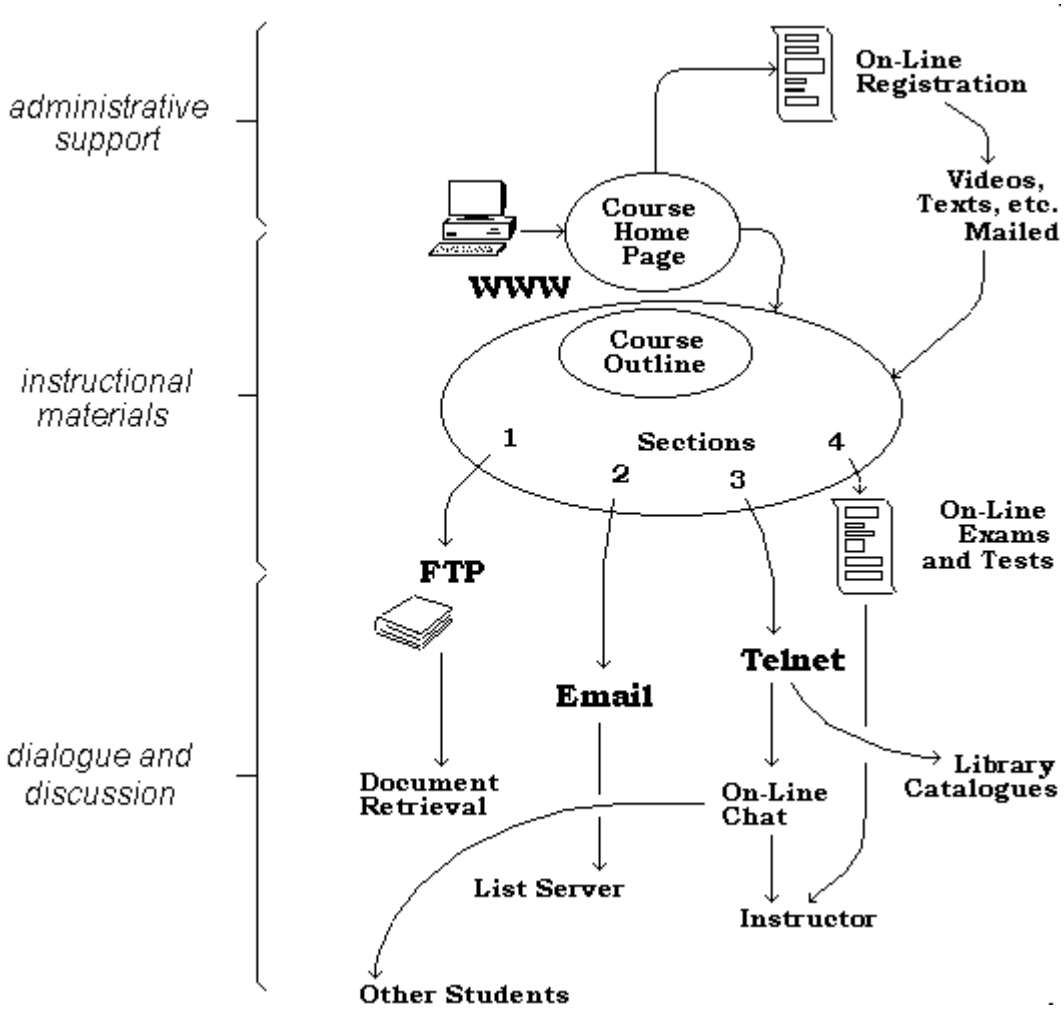
Purpose of the Web Pages

The East-West Guide describes the web page as the link for all elements in the course, guiding students through the course and directing them to course-related media. We concur. We view the series of web pages as the 'backbone' from which other materials may be accessed.



HTML allows users to access almost any other resource from a web page. The web is very easy for people to use. Moreover, web may be designed in an easy to read and visually appealing format.

The diagram below depicts this concept from the point of view of function:



From the central location of a series of web based pages, students can access any of the support functions required for on-line education, specifically, as follows:

FTP: **F**ile **T**ransfer **P**rotocol is used for bulk file transfer. Use an FTP site to distribute printed material such as course manuals, texts, large diagrams, work samples or templates, and any other material that may be printed. FTP servers may also be used as a means to allow students to submit text or other files. FTP is supported in HTML with the ftp tag.

Email: **E**lectronic **Mail** will serve as the primary communications link between instructor and students, and students with other students. Email is supported on the web with either the mailto tag or with a web-based form which either sends email directly or sends it through a CGI script. Another important use of email is the list server, which allows a user to send messages to many other people at once. This is useful for class announcements or general discussion.

Telnet: telnet programs, such as Ewan, allow users to interact directly with a remote program. Chat programs, such as MUDs, MOOs, or IRC are accessed using telnet. A user can access a telnet site directly from a web page using a telnet link. Telnet sites can be used to support on-line chat, simulations, bulletin boards, electronic library catalogues, and any other user-program interaction.

Course Structure

The East-West Guide recommends the following structure template:

- **Course** Introduction (Introduction, Overview, and Objectives)
- **Module #** Title (Overview and Objectives)
- **Section #** Title (Overview and Objectives)
- **Lesson #** (Objective)

This structure employs the **course** as the basic unit of instruction. We vary from this format. In the Assiniboine Model, the basic unit of instruction is the **module**. We are using this term in what is probably a non-standard manner. We define it as follows:

> A **module** is a unit of instruction equivalent to approximately two or three hours of classroom instruction. It is a self-contained learning unit, not referring in any way to structure or content of the course of which it is a part.

We employ this unit for a variety of reasons. First of all, it is intuitive for course developers. It allows them to see very clearly that the structure of a module is analogous to the structure of a face-to-face class. Second, it is a manageable unit of instruction for students. By breaking the course into modules, we are adding an element of pacing to the on-line course, even for students who are working in a non-paced environment.

A course, by contrast, is a collection of modules. A typical 40-hour course would be composed of 15 or 20 modules. Any other division of a course (for example, into three or four sections) is, from a design point of view, completely arbitrary.

This structure allows us to allocate resources more efficiently. Courses frequently overlap content. For example, an Accounting 2 course might review some elements of Accounting 1. Or for example, a Financial Mathematics course and an Accounting course might each include instruction on the calculation of compound interest. It makes no sense to develop this instruction twice. So the module is developed as a stand-alone unit and inserted into each course as needed.

It also allows us to customize or upgrade courses more efficiently. We can create a customized course by assembling a set of modules constructed previously. And upgrading may proceed on a module by module basis as required.

Thus, our approach is to develop the instructional material as a series of self-contained modules. Courses are assembled from these modules. The basic unit of instruction is the module.

Organizational Pages

The East-West Guide presents a template for organizational pages which is typical of many on-line courses:

- **Home**

- **Index**
- **Help**
- **Introduction**
- **Course Sections and Lessons**
- **Resources**
- **Information**
- **Glossary**

The guide additionally recommends that another page provide links to the following information:

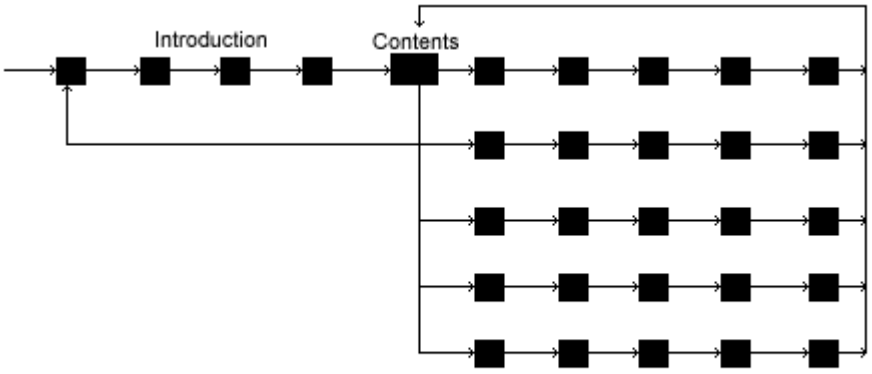
- **Module Outcomes**
- **Module Outline**
- **Assessment Information**
- **Instructor Information**
- **Instructor's Guide**
- **Student's Guide**
- **Hardware and Software Requirements**
- **Notes for Implementers**

While we believe that all of this information is necessary information, we do not believe that it is necessary for all people all the time. Additionally, a page with either set of links presents a complex challenge to the student. While HTML allows users to select between options, presenting too many options with no clear guide as to which option is useful in a given context leaves too many unknowns. Our experience is that students like to feel guided to some degree. A page of links does not offer this.

The Assiniboine approach is to present this material in its own module at the beginning of the course (called 'About this Course'). The content of this module is essentially a standard course outline. At Assiniboine, the following are standard elements of every course outline:

- **Course Name**
- **Course Number**
- **Program to which Course is taught**
- **Instructor**
- **Total Course Hours**
- **Total Credits**
- **Text (if any)**
- **Delivery Dates**
- **Costs / Fees**
- **Registration Information**
- **Goal**
- **Rationale**
- **Prerequisites**
- **Target Population**
- **General Objectives**
- **Specific Objectives**
- **Course Topics**
- **Delivery Method**
- **Method of Evaluation**
- **Details of Preparation**

This information should be presented to students prior to their taking the course. Additionally, it should be presented to students at the beginning of any course, to ensure that they are familiar with the ground-rules for course completion. Before viewing any course material, users are led through a series of instructional pages. The organization thus looks like this:



Some elements in the East-West Guide would not be placed on any specific page at all. A good example of this is the **glossary**. A glossary is a list of new words or technical jargon. The East-West Guide simply presents the glossary as a list.

Presented this way, it means that a student must access and scan the entire list each time a particular definition is required. This is not efficient. Glossaries in some subjects can be quite large and require significant download time, a disincentive. Additionally, when presented as a list, a glossary must be re-created each time a new course is developed. This adds extra work and increases the possibility of inconsistent definitions.

A better method is to hyperlink each word in what would be the glossary to a short page containing the relevant definition. When a student clicks on the link, a small box pops up over their web page with the word defined. The student clicks the 'close' button in the box to remove the glossary item. See the sample at http://www.assiniboinec.mb.ca/glossary/glos_test.htm .

This is a better method because it is a lot easier for students to access, and they do not need to leave their current page in order to get this information. It also allows for the construction of a *single* glossary used by all courses. This ensures consistency and removes duplication of effort. The pop-up boxes may be created using JavaScript, and therefore may be used by most current browsers.

In general, the approach taken by the East-West Guide is to assemble any information that might be needed during the course and to present this information at the beginning of the course. As pointed out above, this presents too many options to a new student. Instead, in the Assiniboine Model, we present the information *as it is needed*. Thus, we ensure that what students are viewing at any point in the course is relevant to that point of the course. Three examples illustrate this approach:

1. Chat and Discussion Forums: in most on-line courses, from the home page a link is provided to a chat or discussion forum. Additionally, in many on-line courses, links to the chat and discussion

forums are provided at the bottom of every page. This is overkill. A student does not need access to chat and discussion all the time. A student needs access only when chat or discussion is called for in the context of instruction. Thus, when a student is asked in the material to, say, add to a discussion, a link is provided *at that point*. Or better (and this is the format we've adopted) a form for input and a link to view previous comments are inserted on the instructional page itself. For example, see http://www.assiniboinec.mb.ca/user/downes/ua/intro3.htm .

2. Web-based resources: on-line courses may take advantage of the information available on the web by linking directly to an original source. We support this technique, however, it is important to do more than to merely list resources on a single page. Links are added throughout the course as determined by the content. Many of the links we add are predefined searches at, say, Alta Vista. Experience has proven that lists of links require a lot of time to maintain. For example, if the topic in question is "Bloom's Taxonomy" we may add the following link to that page: http://www.altavista.digital.com/cgi-bin/query?pg=q&what=web&fmt=.&q=%22bloom%27s+taxonomy%22

3. On-line help: students need instruction in the use of browsers, plug-ins, or other aspects of the course. The East-West model is to gather these into one location, offer a link from the index, and a link from wherever that help may be needed. However, our experience has shown that students get lost in the help pages. They jump, for example, from module 1, section 1, page 7 to Plug-In help amd then have difficulty finding their way back. Assiniboine's solution is to centralize all help files in one directory. A 'Help' button on each course page points to help topics appropriate for that page. When a student is in the Help directory, clicking on 'Exit Help' takes the student back to where they first entered Help. See the sample at http://www.assiniboinec.mb.ca/help/test.htm .

Module Structure

Module structure is not envisioned in the East-West guide because it is not a basic unit of instruction. The paradigm they appear to follow resembles that of a book or manual: there is an overall course, divided into sections. Each section is divided into lessons. Each lesson, in turn, is divided into a series of pages. This is the model that was used at Assiniboine in the past.

The new Assiniboine model breaks modules into four major components:

1. **Scaffolding**
2. **Static Pages**
3. **Dynamic Pages**
4. **Module Exam**

We borrowed the term 'scaffolding' from the Simon Fraser web-based course design project called Virtual U (http://kochab.cs.sfu.ca:8000), and we believe we use it in a similar manner. The scaffolding functions in the same manner a Course Guide would for traditional distance education students.

Essentially, the scaffolding leads the student through a series of *learning activities.* From the scaffolding the student is referred to other web pages, books, videos, or assigned tasks as appropriate.

To keep in mind the sort of information which should be contained in a scaffolding, we use what we call the **IDEAS** model (our apologies for the acronym, but Tranna got to talking, and we couldn't resist):

Information

Description

Exercises

Assignment

Summary

In the **Information** section we introduce the module. These pages contain a brief outline of module content, lists the module learning objectives, and states the estimated time of completion (again, usually 2 to 3 hours). If required, preparation, pre-requisites, or materials needs are also stated in this section of the module.

The **Description** component of the scaffolding leads students to learning materials. While it may contain brief descriptions or discussion of the topic to be learned, this section does not contain the learning material itself. Students are always referred to learning materials from description pages. For example, description pages may assign textual or other readings. Or they may assign a video, audio, or some other multi-media source. Additionally, they may refer a student to a set of static pages (discussed below).

Exercise pages engage the student in learning activities. An exercise page may contain a self-test quiz (easily constructed with Martin Holmes's multiple-choice web quiz making program; see http://www.net-shopper.co.uk/creative/education/languages/martin/jquiz.htm). We like to use web-based forms, which a student completes and submits to the instructor. In courses where discussion and group-work are required, the exercise page may link the student to a discussion board or chat conferencing area. Or the student may be sent from the computer altogether to interview local personalities, perform an experiment, or any other suitable learning activity.

The **Assignment** page contains information about work to be graded. Any of the activities contained in the exercise pages may equally well be assigned for grading. Additionally, this page may be used to prepare students for the module test, if one is required. Since modules are stand-alone units, it is anticipated that each module will have a self-contained testing component. Courses that involve longer projects, such as term papers, should assign a module specifically for that term project.

The **Summary** page offers a chance to relax and recap. A description of module highlights should be included. At Assiniboine, we also like to include a celebration page in recognition of the student's efforts. View one of our silly celebration pages at http://www.assiniboinec.mb.ca/ic15/s3/m4/end.htm . In other of our courses, we advise students to take a coffee break. We don't want them working for eight hours without a break. Including a coffee break tells them explicitly to slow down and let the information sink in.

Please see http://www.assiniboinec.mb.ca/module/linear_equations/intro.htm for a sample scaffolding using this concept, keeping in mind, of course, that this module is under construction. Hey - it's a new model.

While the **IDEAS** acronym is used to identify the parts of a scaffolding, there is no requirement in the model that the pages spell the word "ideas". True, it is generally a good idea to place the introduction at the beginning and the summary at the end. However, any combination of description, exercise and assignment pages will do.

What's key is that each page in a scaffolding:

- Be brief - little or no scrolling should be needed
- Address only one topic - each reading assignment should be given its own page
- Be clear - stating step-by-step exactly what is expected of the reader

The scaffolding is a series of *directions* - nothing more, nothing less.

Static Pages

Static pages constitute the informational core of on-line learning materials. Static pages are, as the name implies, pages that are for the most part unchanging. They include such timeless information as *"Paris is the capital of France"* and *"Triangles have three sides"*. In other words, they contain the content which is likely to remain the same no matter who is teaching the course and no matter when the course was taught.

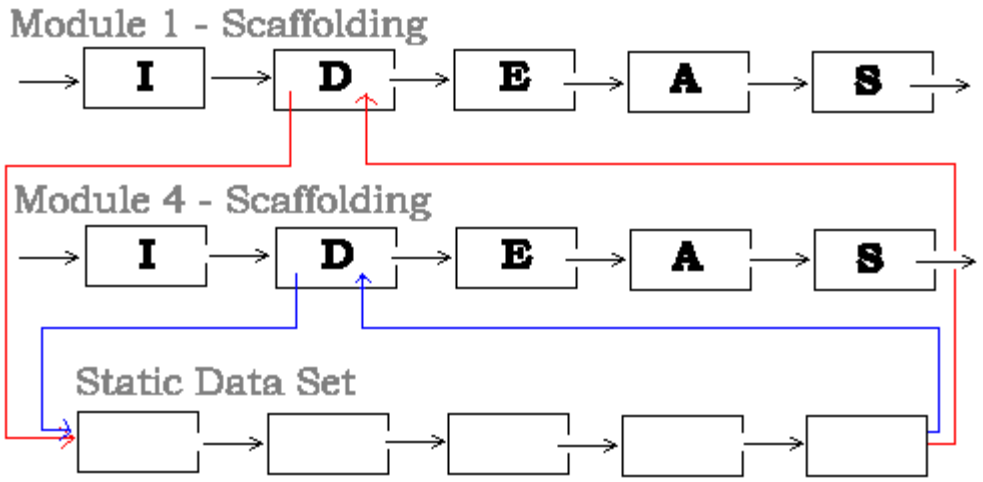
Static pages are not assigned to any particular module. Instead, static pages are arranged by *topic*. This is important, because it means that if two different modules, possibly from two different courses, address the same topic, then the scaffolding for each module links directly into the *same* set of static pages.

Why would we do it this way? There are several compelling reasons:

- It ensures consistency in course materials. Similar triangles are described the same way in every mathematics course covering them.
- It allows efficiency of structure. The section on similar triangles is constructed only once. And once constructed for one course, it does not need to be moved, altered, or changed in any way to be used by another course.
- It allows designers to assign review reading without the risk of a student getting lost. Suppose similar triangles are introduced in Module 1. Suppose then that in Module 4 a new concept is introduced and the instructor would like the student to review similar triangles. Sending a student back to module 1 *via* a link introduces navigational problems - how does the student return to module 4?
- It enables easy upgrading. If somehow the concept of similar triangles should happen to change, then changing the static data updates all courses linking to that static data.

The key to making static data work is the employment of the following design concept: users entering a static data set are always returned to the point from which they entered. So, for example, if a student enters the *Similar Triangles* discussion from module 1, they are returned to module 1. If the student enters from module 4, they are returned to module 4. This allows designers to link readers to any static data from anywhere with no fear of their becoming lost.

Graphically, the structure just described looks like this:



A user links to the static data set from Module 1 (red line) and when completed, returns to module 1. Later, a user links to the same static data set from Module 4 (blue line) and when completed, returns to Module 4.

Static data sets need to be relatively short. This is because they must fit as a component of a module that takes only 2 to 3 hours to complete. In general, static data sets should take about an hour to complete.

The static data model also anticipates future needs. As the demand for more dynamic content increases, developers will feel the urge to include audio tracks, video clips, slide shows, and other multimedia elements into their courses. However, these may take a long time to download. One answer is to place the course on a CD-ROM. While this makes the course load a lot faster, it is impossible to change.

When Assiniboine starts placing course materials on CD-ROM, we will place only *static data* on the disc. The scaffolding will remain on-line. Hence, courses may be revised even for people who have already received their CD-ROM. This allows greater flexibility while allowing for high-bandwidth content.

A Note About Writing Static Data Sets

The temptation is great to treat static data sets as data dumps. Don't. Large amounts of textual are perhaps best handled as textbooks or downloadable files that may be printed by the student. The purpose of a static data set is to lead a student, step-by-step, through a learning activity in which a concept is revealed.

For example, suppose the topic is *Writing HTML Pages*. Many tutorials simply introduce and describe HTML tags. A better approach is to have the student construct a web page as the tags are being introduced. So, for example, when list tags are being introduced, have the student add a list to their web page. See an example of this concept in action at http://www.assiniboinec.mb.ca/cae/tech/html.htm (it takes students about 2 hours to complete this 21 page set).

Dynamic Pages

From the point of view of the reader, dynamic pages look and feel exactly like static pages. They may be entered and exited from any module at any time, and students who are working in different modules may enter the same set of pages.

The difference is that, while static pages may reflect timeless truths, like those involving similar triangles, dynamic pages may reflect ongoing and changing circumstances, like Princess Diana's death. The content of static pages is expected to remain the same, while the content of dynamic pages may change on a daily, weekly, or monthly basis.

We developed the concept of dynamic pages in response to instructor requests. They pointed out, reasonably, that teaching is a fluid process. In many classes, such as English or philosophy, discussion often centres around current events. Instruction also reflects the instructor's growing awareness of the subject matter or understanding of the student's needs and backgrounds.

Some systems of internet-based instructional delivery provide very strong support for dynamic content. We have experience with First Class (http://www.softarc.com).

In this system, course material is centred around a set of discussion forums. Instructors used this forum to set daily assignments, record class discussions, and even to muse on the events of the day.

In a web based environment, such fluidity is more difficult, since it is more difficult to post a web page than it is to enter a comment into a First Class system. And First Class - at least when we used it; there's a new version - was not web compliant. It required extra software and had a customized interface, adding to the technological complexity of the course offering. About two weeks had to be spent introducing students to the system.

Using HTML and CGI, other conferencing solutions are possible. We used CGI scripts in PERL obtained from Matt's Script Archive (http://worldwidemart.com/scripts) and set up chat and discussion boards. This didn't suit our needs either. A separate instance of each script had to be installed for each dynamic page. We did what many people have done: we established a main discussion board for the course and linked to it from the course home or index page. But this meant that dynamic discussion was not inserted into the course materials. Students focused on the static content and ignored the discussion board. Sending the student to the discussion board from the course material meant a lengthy excursion back to the original entrance point.

What we wanted were fixed points within the course where topical and dynamic discussion occurred. For example, in a module dealing with interest rate calculations we wanted to place a discussion of current interest rates and their implications for consumers, businesses and the economy.

Our solution was to write module scaffoldings in such a way as to link to dynamic pages at certain fixed points. Thus, for each module, there is a relatively fixed set of dynamic pages. On a regular basis, instructors are expected to update this content. So when students progress through a module, they are exposed to current discussion as well as timeless truths.

Dynamic pages are created by uploading current content inside a standard template. The template is the same as that surrounding the content of static pages. The manner of uploading may vary. Instructors may be provided with the template and instructed where to place the content. Or instructors may be provided with an interface screen, where a CGI script takes their input and

places it into a template shell.

We recommend using dynamic pages as a primer for course discussion. This is a technique used effectively by Hotwired (http://www.hotwired.com). Readers at the Hotwired site enter through a main page, read through one of the daily articles provided, and are then sent to a discussion forum (called *Threads*) and invited to comment. Despite a relatively small user base (Hotwired contains about 5500 user pages) the site's forums are contain thousands of messages on hundreds of topics.

But we are just beginning to explore the use of dynamic pages. Anything which in a classroom environment changes on a regular basis may be used as grist for the dynamic mill. As more instructors employ this model at Assiniboine, we expect that we will be surprised by the uses to which dynamic pages are put.

Module Exam

As mentioned above, we view modules as stand-alone entities. This is to allow a single module to be used in more than one course. Since modules are stand-alone, they should be self-sufficient so far as evaluation is concerned. Hence, at the end of each module we propose a module exam.

The purpose of a module exam is very narrow. It is intended only to establish whether a student has achieved the skills or learning set out in the module objectives. We recognize that there may be wider ranging evaluative components in a course. These should be treated as separate modules. A term paper module, for example, may be defined early in the course, with the expectation that the student will complete the assignment near the end of the course. No rule states that modules must be studied in order, or one at a time.

We decided to employ web-based exams. This allows us to insert module exams into the course materials. Students thus the same interface to write module exams as they do to read content and submit exercises. At Assiniboine, we offer students a link to the exam both at the beginning and the end of a module (the link at the beginning may be disabled in cases where participation is a course requirement). This allows students to skip content with which they are already familiar. Since students sometimes misjudge, they are allowed more than one attempt at an exam.

Placing exams on the web poses a set of problems. JavaScript cannot be used, since a student may easily view the source and obtain the answers. And it is important that the exams be different for each student. Otherwise course exams will no doubt be distributed among students.

Assiniboine generates its exams dynamically. All possible questions are placed into a single directory. From this set of questions, a subset is assigned to an individual module. From this subset, a CGI script creates an exam by selecting random questions according to a set of rules defined by the instructor. Password entry is required to write the exam, and each attempt to reload the exam page, since it reactivates the PERL script, counts an attempt (it is important to warn students of this *before* they try it).

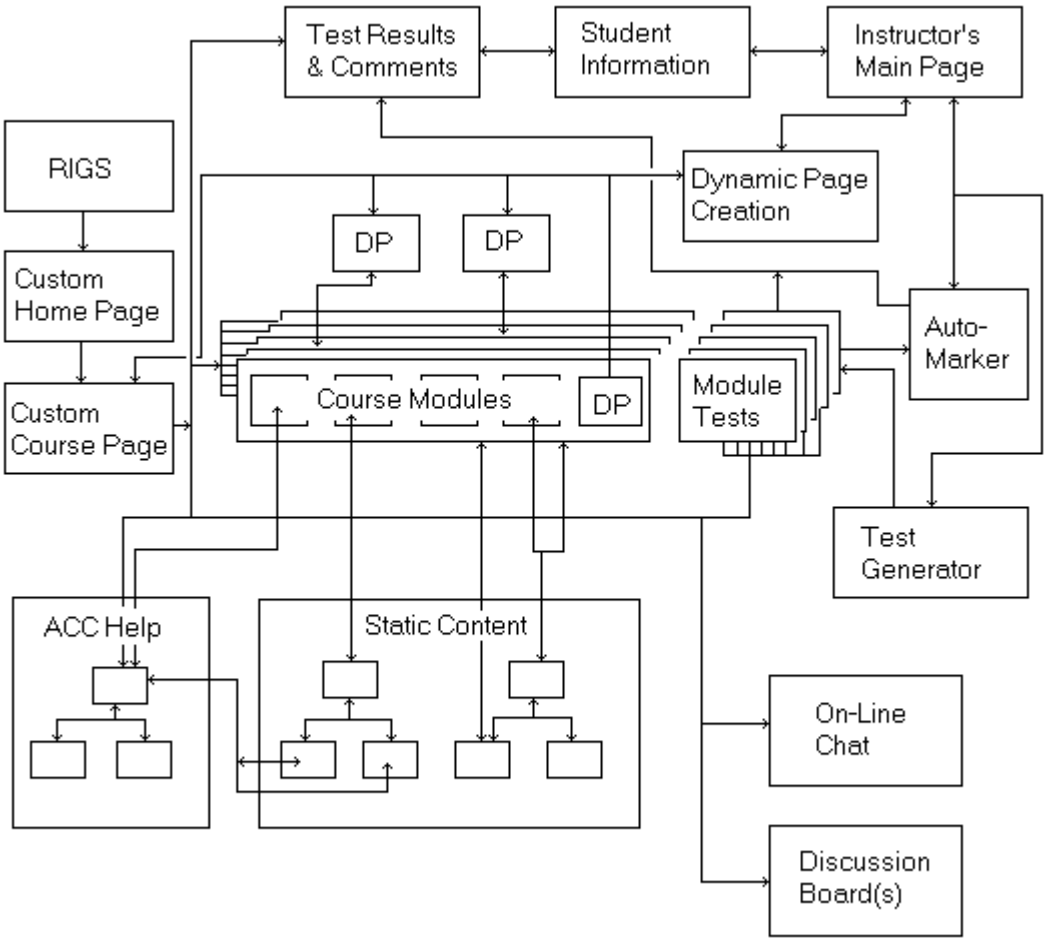
A copy of the completed exam is placed in a student's personal directory; the student may access the completed exam from a link in the course page. Another copy, with fields for marks and comments, is placed in the instructor's directory. When the instructor grades the exam, the graded version replaces the original in the student's directory. Thus, the student can access graded work from within the course itself.

Our module exam system is in early stages of testing. No doubt we may revise this model as

experience dictates. The question of invigilation, for example, remains an open one.

The Assiniboine Model: An Overview

The pieces described above result in a course structure which may be diagrammed as follows:



The diagram consists of the following components:

RIGS: The **R**egistration **I**nformation **G**athering **S**ystem is used to establish a student user id and password. Students enter data in a series of forms as they register for the course. This information is stored in a database, is available to instructors, and is used by a variety of subsequent course functions.

Custom Home Page: All website users, whether they are registered in a course or not, receive a user id and password. They may then log onto a custom home page. From this page they may order materials and register for courses.

Custom Course Page: a page listing each course module, and the student's status in each, is generated for each student. From this page they may access any module, and module exam, or important course information, such as the instructor's page, course outline, and the like.

Course Modules: as described in this paper, are composed centrally as scaffolding pages following the IDEAS model.

Static Content: students access static content from scaffolding pages. Note that the same static content set may be accessed from more than one module.

DP (Dynamic Content): dynamic pages are updated on a regular basis by the instructor. Note that dynamic pages may appear inside the scaffolding itself.

Chat and Discussion: these may be accessed directly from the custom course page, or alternatively, from within scaffolding, static pages or dynamic pages. We recommend priming students with dynamic content before sending them to a chat or discussion forum.

Module Tests: these are defined for each module. Tests may be marked by instructors or, in a multiple choice environment, automarked by the system. The results are stored as web pages, which may be accessed from the Custom Course Page.

Concluding Remarks

We have no doubt that many other models of on-line course design will propagate over the next few years. But while they may vary in shape and design, it is our contention that, at the very least, any online course will have to at the very least contain the *elements* described in this paper.