Teaching PASCAL with WWW based MCQ exercises

Henry M. K. Warkentyne, Eddy Forte, I. Smith

TEACHING PASCAL WITH WWW BASED MCQ EXERCISES

Henry M. K. Warkentyne, Eddy Forte

LEAO, Department of Informatics, EPFL

I. Smith

LIA, Department of Informatics, EPFL

Abstract : *This paper presents an overview of the design, first implementation and evaluation of hypertext based Multiple Choice Question (MCQ) Pascal courseware. The courseware is implemented using the WebDidact system, which is based on the World Wide Web. The MCQs are written in a simple language that allows the author to use HTML markup tags within the question structures. Using WWW places some constraints on the hypertext system and gives rise to some navigational difficulties. From the experience developing and using the courseware, we conclude that it would be better to use a new SGML DTD to define the MCQ authoring language. The navigational problems may be partially overcome by exploiting mechanisms provided by the latest technological developments in the WWW, notably Java and Javascript.*

INTRODUCTION

This paper presents an overview of the design, first implementation and evaluation of World Wide Web (WWW) hypertext based courseware to teach introductory Pascal programming to first year Civil Engineering students at the École Polytechnique Fédérale de Lausanne (EPFL). At EPFL, courses in computing are the responsibility of the Department of Informatics (DI). In recent years, the demand for such courses has risen considerably, placing a large burden on the department, and it is hoped that the use of courseware will alleviate this to some extent.

The courseware consists of a set of Multiple Choice Questions (MCQs). Learners navigate from question to question in a strictly hierarchical fashion. However, an individual question may be decomposed into several units such as the question itself, explanations related to concepts in the question, the response, and further explanations of aspects of the response. Thus it is interesting to model MCQs as hypertext.

OVERVIEW OF THE WEBDIDACT SYSTEM

WebDidact, developed at EPFL's Laboratoire d'Enseignement Assisté par Ordinateur (LEAO), is a system that provides facilities to deliver courseware over WWW. According to the schema presented in figure 1, when a learner selects an exercise to perform, the WebDidact server reads in the exercise file and generates a page in Hypertext Markup Language (HTML) (Graham, 1995), which is then returned via the WWW server to the learner's browser. The learner then selects a response, which is delivered to the WebDidact server for processing. After the processing, the WebDidact server may send another exercise to the learner, and so on. An earlier version of WebDidact was used to create a Unix tutorial for Rural Engineering students at EPFL (Warkentyne, Delafontaine, 1995).

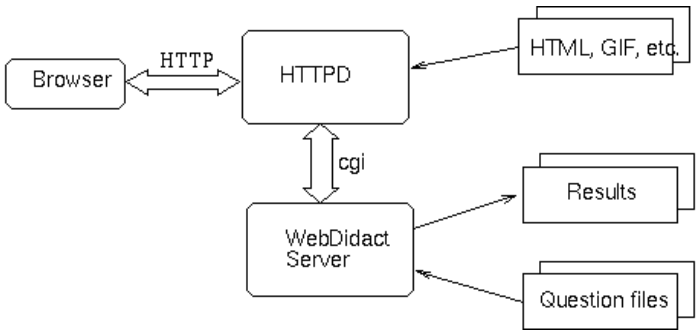

Figure 1: WebDidact System

THE WEBDIDACT PASCAL COURSEWARE

The Pascal courseware questions are written in a simple language created specially for WebDidact. This language allows authors to design multiple choice questions and use HTML to markup the text. For example, the following code fragment is from a question used the Pascal course.

```
( MSELECT,
"<H3>Fonction ou procédure?</H3>
Sélectionnez les définitions qui devraient être réalisées
   comme des fonctions.",
   6,
   {
     ("Le dessin d'un rectangle",
     F, "(<a href=fonctions/q1.r1.html>procédure</a>)"),
     ("Le calcul de la flèche d'une poutre cantilever",
     T, "(<a href=fonctions/q1.r2.html>fonction</a>)"),
     ("La vérification qu'une poutre est capable de résister à
     une charge appliquée",
     T, "(<a href=fonctions/q1.r3.html>fonction</a>)"),
...
   }
);
```

In the above code, the keyword `MSELECT`[1] is followed by the text of the question, the number of choices, and a list of triples where each triple contains the text of the response, its truth value, and the feedback text. On the browser, the question is displayed as shown in figure 2.

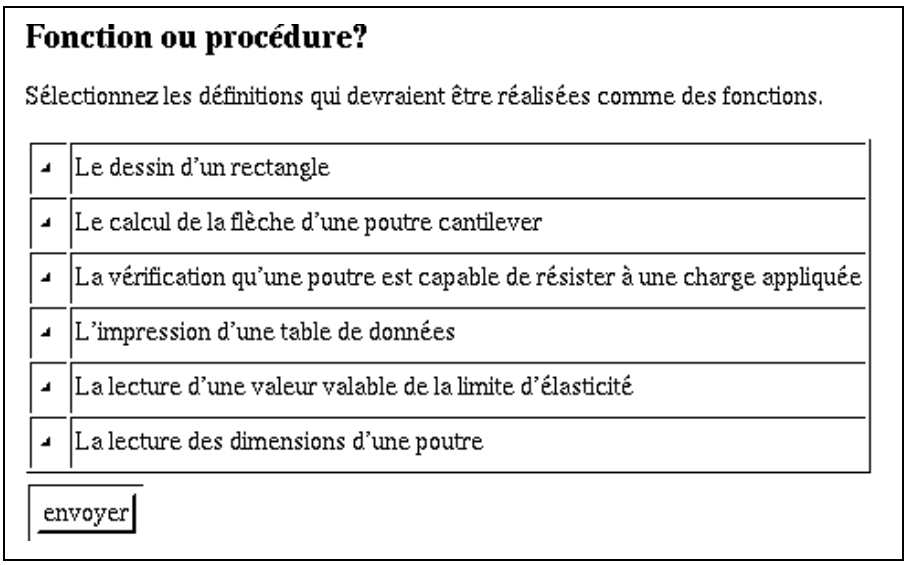

Figure 2: Sample Question

After selecting some of the answers and pressing the "envoyer" (send) button, the question server generates a feedback page indicating the correct and incorrect choices (see figure 3). In this page, there are no explanations of the errors. On the other hand, the student can follow hypertext links (anchored in the words "procédure" and "fonction") to pages that provide detailed explanations of the corresponding error. These explanations are also used as comments or expansions to correct answers.

This form of *progressive disclosure* is a significant benefit of basing our courseware on a hypertext system. Instead of being put off by the appearance of a dauntingly large page of explanations, a student can allow his natural curiousity to lead him to discover the extra information.

1 This keyword specifies that the structure is a MCQ with the particularity that the learner may select zero or more choices. We will not discuss the other types of structure in this paper.

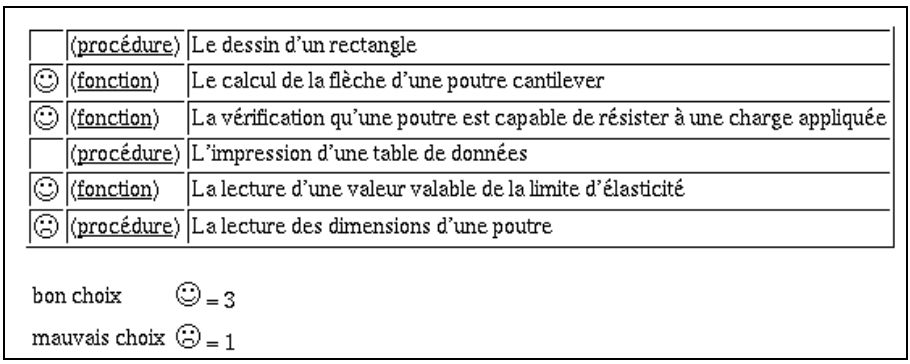

Figure 3: Feedback Page for Sample Question

There are approximately 30 questions divided into five sections :

1. Structure of a Pascal program
2. Conditions
3. Loops
4. Procedures
5. Functions

A main menu provides links to the submenus for the sections.

HYPERTEXT NAVIGATION ISSUES

There are two quite different problems with hypertext navigation as implemented on WWW. The first arises from a mismatch in the services offered by WWW and the special needs of the MCQ courseware, which require dynamic updating of the node content. The second concerns the different roles of hypertext links and whether it is necessary to explicitly distinguish them.

Dynamic Hypertext Nodes

In a normal WWW hypertext document, the information is organized in static nodes. If the reader returns to a previously visited node, he will see exactly the same page displayed. In a set of questions, on the other hand, it is necessary to modify the contents of the nodes and their interlinking to avoid confusion over which questions have been answered and which have not. Although WebDidact is capable of modifying the node content or access, it is a standard WWW browser feature to memorize pages locally and provide local navigation functions to go backwards and forwards between them. This means that, through inadverdant use of the "back" button, a student can return to a previously answered question with no indication on the page that this is the case. A partial solution is to provide a guide to the use of the courseware that explains this situation and how to avoid it.

Link Types

In the Pascal courseware, we use links for three different purposes :

- to link keywords in feedback pages to detailed explanations,

- to link concepts in the text of a question to extra explanations and
- to link menu topics to submenus (as per the sections listed above) or MCQs.

Although the visual context is usually distinct for each type of link, there is no distinction at the WWW level (i.e. in the anchor markup tag). But such a distinction could be useful at the browser level where, for example, fetching the pages whose links designate them as “extra explanations” could be displayed in a popup window instead of replacing the contents of the main browser page.

EVALUATION

Here we provide an brief evaluation of the courseware as a hypertext from the author's and learner's points of view.

Courseware Authoring

While developing the Pascal MCQs, the authors encountered two difficulties. The first was the visually confusing mix of the question structure syntax with the HTML tags. The second was the management of two separate sets of files, question files (served by the WebDidact server) and HTML files containing supplemental explanations (served by the WWW HTTP server). It was felt that it would be better to have a unified syntax wherein MCQs and the explanations are defined in a single mark up language, which in turn has motivated new work in defining a more coherent MCQ authoring language based on SGML (see below).

Courseware Utilisation

We used two sources of information to evaluate the courseware from the users perspective, logfile information generated by the WebDidact server and a short questionnaire distributed to the students (to be filled in anonymously).

The logfile generated by WebDidact records information about each student's session including the connection time, question responses, and disconnection time. From this information, we were able to observe that quite a few students repeated several questions while doing a particular set. We speculated that this was due to the navigational difficulties cited above, and this was confirmed verbally by the students. The questionnaire focused on the didactical quality of the courseware rather than the browser interface. On a scale from zero to two, the mean “satisfaction” level was 1.4. On the other hand, several individuals had strong negative feelings about the courseware, using terms such as “waste of time” and “useless game”.

FUTURE DEVELOPMENTS

There are clearly two aspects of the WebDidact system that require further improvement: the MCQ authoring language and the navigation system.

An SGML DTD for MCQs

As mentioned above, we believe that the current MCQ authoring language suffers from the blend of SGML markup tags and a C-like question definition framework. This problem can be eliminated by using a single markup language to write questions. Indeed, if we view MCQs as special types of documents, it makes a great deal of sense to create a dedicated MCQ DTD. The Finish University EVITech has already began work in this area and will be collaborating with the LEAO in the European Telematics Application project ARIADNE (Alliance of Remote Instructional Authoring and Distribution Networks for Europe) (Forte et al., 1995).

Navigational Improvements

Some of the navigational problems cited above can probably be ameliorated by using the latest developments in WWW technology. In particular, the Java language will permit the creating of “dynamic” pages, thus preventing the browser from memorizing out-of-date questions and simplifying the problem of returning to the current active page. Finally, Javascript, an extension to HTML proposed by Netscape Corporation, may allow the document author to specify clearly the type of a given link with text that is displayed in a reserved area at the bottom of the browser's main window.

Bibliography

Graham Ian S. (1995). *The HTML Sourcebook*. John Wiley and Sons, New York.

Warkentyne Henry M. K. and Delafontaine G.E. (1995). “WWW Based Courseware for Teaching Unix.” in A.N. Johnson, editor, *Hypermedia in Sheffield '95*. SEFI.

Forte Eddy, Wentland Maia, and Duval E. (1995). “ARIADNE: Basic Concepts for a Computer Based and Telematics Supported Instructional Framework”, *European Journal of Engineering Education*.