An experiment with WWW interactive learning in university education

David R. McIntyre*, Francis G. Wolff

Department of Computer and Information Science, Cleveland State University, Cleveland OH 44115, USA

Received 20 August 1997; accepted 16 January 1998

Abstract

The World Wide Web coupled with user friendly Web browsers now provide access to multimedia Web pages in universally accepted formats that can be accessed world wide easily via inexpensive desktop computers. Everyone appears to agree that this technology will revolutionize how students, faculty, researchers, and the public access and use information. Consequently university educators are now enjoying, for the first time in history, a new way to customize and share their unique approaches to teaching and information resources in the form of text, graphics, and sound—to students both on and off campus and, with concern for the future, across time. In this paper we discuss our exploration with the use of interactive learning on the Web in an Introduction to C Programming Course taught in the Department of Computer and Information Science at Cleveland State University, and compare results with the same course taught a previous semester using no interactive WWW learning. © 1998 Elsevier Science Ltd. All rights reserved.

Keywords: Interactive learning; Teaching assistant; Cyber classroom; Active learning; Internet; World Wide Web; Java Script; Common gateway interface

1. Introduction

The tremendous increase in the ability to access hypermedia information has resulted from the convergence in the past few years of several complementary factors: 1) very powerful computers have become cheap enough to sit on desktops with gigabytes of hard disk storage within their cases; 2) the Internet has quickly become a ubiquitous open World Wide Web (WWW) of interconnected, global computer networks; and 3) the development of user friendly Web browsers which provide access to multimedia Web pages in universally accepted formats.

* E-mail: mcintyre@cis.csuohio.edu
It is apparent that we as educators are now just beginning to understand the current and future potential of this new Web technology on university courses, particularly in the areas of distribution of course material, enhancement of course material via hyperlinks to related WEB sites, multimedia lecture enhancement, electronic notes and books, and interactive learning.

We take a closer look at the last topic: interactive learning in a WWW environment. Until very recently, the typical Web educational systems were largely non-interactive in nature offering static tutorial or educational material on a variety of topics (Bignall, Bond, Rice & Windley, 1994; Campbell, Hurley, Jones & Stephens, 1995), but using the Web merely as a transmission medium. Currently some Web systems, with emphasis on the use of interactive components as fundamental parts of the learning environment, are beginning to be experimented with. Also, until very recently with the development of Java and JavaScript, the only means of achieving interactivity with a Web page was through use of the Common Gateway Interface (CGI) standard. However, CGI often leads to unacceptably slow user response time in this real time environment since the interactive execution must take place on the remote server and not on the client machine. Additional negative factors for CGI include display problems on different client platforms due to differences in the various client windowing systems, and potential security risks in running user made applications at the server end. Java and JavaScript, as we shall see, solve all these concerns. Still other papers focus on the development of user friendly Web facilities to aid the instructor in developing interactive Web pages (Jones, Ruehr & Salter, 1996). In our view, these ad hoc, special purpose Web facilities, while interesting, are often inflexible and hence incapable of keeping pace with the new features and capabilities rapidly occurring in the Web world.

In this paper we present a prototype interactive system developed using JavaScript which executes on the client machine providing fast response (avoiding the slower CGI technology). The ideas developed in this paper are general in nature and can be readily adopted to other courses (Computer Science or otherwise) and so the potential application is wide. Furthermore, no special ad hoc user friendly Web facilities are needed to develop an interactive Web system like ours. An outline of the paper is as follows. In Section 2 we discuss the suitability of WWW as a learning environment. Section 3 presents our learning system prototype. In Section 4 we present the results of administering the same quiz on C pointers to two introductory C classes, one having access to WWW interactive learning of C pointers but with no in-class examples, and the other having extra class time devoted to C pointers and examples. Finally, Section 5 gives our conclusions.

2. WWW as a learning environment

Students do not learn by simply receiving information (Brown, Bransford, Ferrara & Campione, 1983), but rather constructively through a process of reflection on material and interaction with it thereby creating an understanding (Farnham-Diggory, 1990). Thus interactivity is an important ingredient in effective learning by allowing the student to become engaged and to reflect on interesting problem issues. The syntax of a computer programming language can be learned from a book and sample programs. However the design skills and conceptual knowledge are more difficult to acquire from written materials. It has been shown
that interactivity in the form of lectures and tutorials is more effective in teaching such concepts than a book and simple programs alone can provide (Aveling, Smith & Wilson, 1992). However, in the standard lecture-style class, students do not always ask the questions that may occur to them throughout the course of the class. This may be due to a number of factors: inability to clearly formulate or express the question, disinterest in the topic, shyness in large groups, unsure of the appropriateness of the question etc. In addition, the instructor may not have the luxury of encouraging many questions throughout the class due to time constraints in order to cover the prescribed amount of material.

Interactive learning on the Web may be a way to partially supplement the classroom learning experience by providing an interactive environment similar to the classroom but with more attention to individual student needs. One of the powers of interactivity in a Web environment is the capability to engage by providing rapid, compelling interaction and feedback to the student. Interactive multimedia technology can help motivate learners by providing information in a form that is concrete and perceptually easy to process. Engagement is also enhanced by problem-based presentation of educational material. An engaged student is a motivated student (Neorman & Spohrer, 1996). Also, unlike the classroom environment, each student can individually learn at the computer at his/her own time, location (lab, home, office), and pace and many more examples can be covered than might otherwise be possible in a classroom environment.

In a programming course, learning through code design, writing, stepping through code execution, and exploring with the results is a significant percentage of student out-of-classroom learning experience. This involves design, writing and running (debugging) program modules. Consequently the student must master several component pieces of software: an editor (to write code), a compiler (to compile the code), run/debug interface (to run code), which can often get in the way of learning for a novice student programmer.

In order to teach students the rather difficult task of program design, in a semester, typically a student is given a rather limited set (four or so) of programming assignments to design, write and debug. For each assignment, especially in an introductory programming language course, the student, whose time is somewhat limited (programming language courses are notorious for gobbling up “all” the student’s out-of-class time), often develops each program by writing and gradually incorporating several small (experimental) pieces of code in order to checkout his/her ideas on those aspects of the C language needed for that assignment. The limited time of the course means that a novice programmer in his/her effort to get the programming assignments completed on time, often has little or no time to explore by code writing other areas of the programming language not required in coding the programming assignments, but which might otherwise help him/her in better understanding the course material. Indeed, a large number of code experiments would have to be tediously developed (designed, written, debugged) by a novice programmer to fully explore their knowledge of the a programming language. The authors resolved to solve this problem by developing a WWW learning environment in which the student did not have to design, write or debug any code in order to explore the code and its effects. Instead the student would be presented with complete C programs along with various points in the code which he/she could select modifications. The student then could step-by-step execute the code by simply clicking on a button and observe the effects of the execution. This would create a pleasant, interactive environment in which the student is
unencumbered with the implementation details of the code and instead can focus upon the intended properties of the code.

It should be pointed out that often in top universities the Computer Science student is expected to pick up computer languages largely on his/her own. Such a Web learning tool should significantly help this group of students as well.

We chose the WWW over other available sophisticated multimedia authoring tools (Lynch, 1995) as the distribution mechanism for our interactive learning experiment even though Hypertext Markup Language (HTML) is not as polished. The primary reasons for not using these more sophisticated tools were expense, and the high learning curve for such highly specialized software. The factors that most influenced our choice of the WWW as the interactive out-of-classroom learning environment were:

1. accessibility: wide audience due to WWW browsers creating user friendly environment. Students can access Web pages in the laboratory, from their dorm rooms, from home through PPP or SLIP.
2. platform independence: runs equally well on DOS, Windows, UNIX, MacIntosh etc.
3. HTML: the standard at the heart of the World Wide Web. HTML is the prototype network application of SGML which takes great care to minimize the overhead in terms of storage space and more importantly in network bandwidth.
4. permanence of the Web in university education: educators will embrace the Web as a vehicle for interactive learning.

Since WWW is the chosen delivery system for our interactive learning tool, it follows that the interactive components of our system are constrained by capabilities of the HTTP protocol and WWW browsers. Until recently, the only way to provide interactive content on the Web was through the CGI standard in which a C program residing on the server was executed on the server machine. This necessarily resulted in slower response time at the user (client) end due to transmission delays. With the arrival of Java and JavaScript that has changed. When a Web page is downloaded from the server machine to the client machine, all attached Java and/or JavaScript code is also downloaded with that Web page to the client. Then when the embedded forms in the Web page are submitted, the associated Java/JavaScript code (now residing on the client) is executed on the client (without any interaction with the server). Fig. 1 illustrates the client/server relationship for Web pages containing JavaScript code.

The Web interactive features that met our implementation goals of fast user response, window compatibility, and security were all met by both Java and JavaScript. We chose JavaScript to implement our prototype since it was easier to quickly program than Java. Java, although unnecessary for our prototype, is more powerful than JavaScript and runs slightly faster (since it is compiled) and therefore might be preferable in building future systems. We want to emphasize that our purpose here is to explore the use of current and potential long term features of the Web in enhancing interactive learning in University education and not any particular implementation.
3. A prototype system for interactive learning

In this preliminary study we focused on learning of pointers in C for our test of the effectiveness of WWW interactive learning. This is considered by most students to be one of the hardest areas of the C language to understand and master. Several interactive exploratory practice sessions were placed on the WEB for the students to experiment with. Each allowed the student to experiment with C pointers in complete C programs. For example, in one session, the student is presented with a complete C program including the main function and two functions: trouble and double_trouble, in one frame on the top left quadrant of the computer screen. The top right quadrant contains variables in each module along with the contents and the address where the variable is located using the notation $A \rightarrow x = V$ where $A$ is the actual memory address where variable $x$ is located, and $V$ is the value of the variable at a particular instant in time. The bottom left quadrant shows any output printed so far by the program, and the bottom right quadrant contains the control buttons: STEP (to execute the highlighted instruction), RUN (to execute the entire program to completion), RESET (to reset the state of the simulation so that a new experiment can be tried), and QUIT (to end the session). Fig. 2 illustrates the initial screen.

In this learning session, the student is able to select one of many calls to the function double_trouble within the function trouble by clicking on the form box (which causes the form to be submitted and a JavaScript code to be called which will take the appropriate action). Then the student may click on the STEP button (the STEP form is submitted, a JavaScript code is called and appropriate action is taken) causing execution of the first instruction in the program. Initially, in the program on the screen that the student sees, all the variables are undefined. The instruction to be executed next is highlighted in red in the left-top quadrant, and the right-top quadrant shows the current addresses and variable contents. Then as the student continues to click on the STEP button the variables in the top right quadrant begin to assume addresses and values as the highlighted instruction in the top left quadrant is executed. Thus without designing, writing, compiling, or submitting a program, the student is able to easily modify an existing complete C program interactively and watch as the effects of the step-
by-step execution of the entire C program are displayed on the computer screen. Fig. 3 shows the screen after the instruction \( *p = 2*x - y \); in function \texttt{double\_trouble} is executed.

The authors feel that by restricting the student’s options to pull down menus containing carefully selected C source code fragments and then allowing the student, simply by clicking, to experiment with the results of step by step execution, focuses the student on the important issues embodied in the directed learning experiment.

4. Overview of implementation details

JavaScript is part of HTML (traditionally used for text formatting) which allows the ability to program executable functions on the local client computer. A web page contains text formatting commands and possibly JavaScript commands, both implemented via HTML tags. JavaScript provides the ability to dynamically (possibly through user interaction) create a web
page at the local client computer. Initially, when a hyperlink is clicked the browser downloads a new web page (called the parent page) from the remote server to the client computer. The browser then processes on the local client computer each HTML tag to display text or process JavaScript commands creating new web pages at the client computer.

In our case, the parent web page contains all global JavaScript variables and functions (needed to simulate the execution of the C source program) along with four HTML tags to load four more web pages in the top left and right, and bottom left and right quadrants (called code, variable, output and control web pages respectively). As each HTML tag is processed, the corresponding web page is loaded, and then the process repeats as each HTML tag within the loaded web page is processed. When the “code” web page is loaded, the HTML tags cause the C programming language source code to be displayed. The “variable” web page, when loaded, displays JavaScript variables which represent the C source code variables. The “output” web page, when loaded displays output produced by the execution of the C source program. Finally, the “control” web page displays the four hyperlinks (step, run, reset, quit) to

```c
#include <stdio.h>
void trouble(int *x, int *y, int z);
void double_trouble(int *p, int y);
int main(void) {
    int x = 10, y = 15;
    trouble(&x, &y, 128);
    printf("x = %d, y = %d ", x, y);
    return (0);
}
void trouble(int *x, int *y, int z) {
    what should be passed?
}
void double_trouble(int *p, int y) {
    int x;
    x = 14;
    *p = 2 * x - y;
}
```

![Fig. 3. Complete C program after *p = 2*x – y; executed.](image-url)
JavaScript functions which when clicked cause the corresponding JavaScript function located in the parent frame to execute. Once the parent and four web pages have been loaded the system waits for human interaction. The pull down menus within the C source code can be used to select from a variety of valid C code choices. The users can then click on the four hyperlinks to cause the associated Javascript code to be executed to make appropriate changes in the four web pages.

5. Experiment and results

The experiment involved learning C pointers in two introductory C programming classes, one using the WWW interactive learning tool and the other not. Both classes were given the same quiz on C pointers during classroom time following the learning process. The basic underlying definitions and concepts of C pointers were discussed in both classes. However, in the class not using the WWW interactive learning tool, the instructor (as in the past) presented and discussed with the students a few (limited due to classroom time constraints) variants of the future quiz (the students of course were not aware of this). The instructor in the other class spent no class time at all on the variant examples, but rather told the students to explore the WWW interactive learning tool and practice with the several interactive examples (again variants of the quiz). In both classes the students were advised to try and resolve any problems or difficulties on the C pointer material (without instructor help) by rereading the textbook and other literature, and through discussions with friends. In the case of the students with the learning tool, they were encouraged to intermix interactive WWW learning sessions, with textbook reading, and discussions with friends (possibly sitting around the screen) over a period of several days. The interactive sessions identified material not learned in class more thoroughly than is possible in class time and without requiring the student to do any programming. Thus the learning process is an interactive process on the WWW to identify issues not well understood, combined with textbook reading, and discussions with friends to clarify problem areas.

The results of the same in-classroom quiz on C pointers given to the two classes are given in Fig. 4.

While the experiment involved only two classes, it is clear from the results that the ability to leisurely interact with several examples during out of classroom time appeared to significantly raise grades on the identical quiz.

<table>
<thead>
<tr>
<th>Grade distribution on Quiz</th>
</tr>
</thead>
<tbody>
<tr>
<td>≤ 50's</td>
</tr>
<tr>
<td>Class on WWW</td>
</tr>
<tr>
<td>Class not on WWW</td>
</tr>
</tbody>
</table>

Fig. 4. Grade distribution on pointer quiz in two classes.
The students in the class using WWW interactive learning were surveyed after taking the in-class quiz on C pointers. The survey was unstructured with no leading questions and simply asked the student to assess the benefits/disadvantages of WWW interactive learning on C pointers in their learning experience. The students unanimously felt that introducing WWW interactive learning as a supplement to in-classroom learning was a significant benefit to their individual learning process. In particular, the following opinions were frequently expressed:

- could leisurely review the step-by-step tracing of code related to pointers
- 75% of the respondents indicated that WWW interactive learning should not replace in-classroom learning, the remaining 25% made no comment upon this issue
- easier, fun way of learning complex issues
- allows student to thoroughly explore a large number of troublesome issues with feedback. Such depth of coverage would not be possible in limited classroom time
- convenient to use Internet from home/office to access WWW interactive learning environment
- clarified difficult issues in the literature
- often several fellow students would practice WWW interactive learning sessions together and discuss results.

The fact that WWW interactive learning lends itself to collaborative sessions fits nicely with the long established belief of the educational community in the effectiveness of cooperative learning (Johnson, Johnson & Smith, 1991).

6. Conclusions

We developed a prototype to explore from a functional viewpoint the efficacy of WWW interactive learning as an enhancement to classroom learning. Our goal was to restrict ourselves to use only those Web features supported by all current Web browsers. We were not concerned in this paper with implementation issues other than: 1) the long term availability of the Web features we used; and 2) the use of client-based software which yielded acceptable user response time, avoided potential security issues and display problems due to different client platform windowing systems. Our implementation choice of JavaScript nicely satisfied these restrictions.

As reported, the results were very encouraging and we believe that many of the ideas discussed in this paper can be readily adapted to other university courses. Nonetheless, again it is worth emphasizing that while the students unanimously found WWW interactive learning a significant enhancement to classroom learning, nonetheless 75% of the students specifically suggested in their open ended responses that it should not be considered as a total replacement. To paraphrase one student: “One of the most integral parts of learning for me is the in-classroom interaction of students and instructor. It is amazing what questions come up (in a classroom) that one would not have thought of but may have to be faced with later”.

References


