Experience with Teaching Electronic Commerce

Alfred C. Weaver
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
weaver@cs.virginia.edu

Abstract - CS453, Electronic Commerce Technologies, is an elective course taken primarily by computer science and computer engineering students in their senior year. The course was originally offered using a traditional lecture plus homework approach, but with support from NSF DUE CCLI we added a hands-on programming laboratory that met three hours per week and provided mentored learning for HTML, CSS, JavaScript, Perl, CGI, SQL, and PHP. Students liked the material but objected to the inflexible weekly format. With additional university support we transitioned to virtual labs that better aligned the students' experience with the 24/7 nature of the Internet. In fall 2005 we will compare one group using these virtual labs with another group using Tablet PCs, Conference XP, and OneNote.

Index Terms - electronic commerce, software laboratory, virtual labs.

BACKGROUND

In 1995 the University of Virginia's Department of Computer Science offered a seminar course in what was then a new subject—electronic commerce. The topic proved durable and in 1999 the seminar transitioned into a permanent course, CS453, Electronic Commerce Technologies, which is now offered each fall semester and attracts 50-60 students; about 90% of those students are computer science or computer engineering majors in their third or fourth year of study. Today's typical student is an accomplished C++ and/or Java programmer who has little difficulty learning new languages and is undeterred by the prospect of using four new programming languages in one semester.

Each offering of the course was accompanied by mid-semester and end-of-course evaluations, which involved whole-class discussions as well as anonymous paper (and later electronic) surveys. It became increasingly apparent that the traditional approach of lectures augmented with bi-weekly homeworks was an inadequate method of preparing students to create e-commerce applications that could safely and securely conduct business and financial transactions over the Internet. Our conclusion was that too much lecture time was being devoted to introducing languages and tools (e.g., Perl pattern matching) and too little time introducing broader and more durable concepts (e.g., cryptography and how it is used in secure electronic transactions). Our solution was to shift much of the programming detail to a laboratory experience.

Thus we proposed and the NSF DUE CCLI program funded [1] a proof-of-concept experiment to create a physical programming laboratory that would meet three hours per week and would provide practical, hands-on programming experience with the languages and techniques most often used to develop e-commerce applications.

Over the spring and summer of 2002, the instructor and two assistants developed instructional material for HTML, CSS, JavaScript, Perl, CGI, SQL, and ASP. The instructor and four teaching assistants used those programming exercises during weekly labs in fall 2002 for a class of 53 students. Although 100% of the students rated the material as “very helpful,” 87% objected to the fixed time of the lab. As one student opined in his end-of-course evaluation, “My schedule is very hectic. The Wednesday night lab hours were inconvenient in view of assignments in other classes and extracurricular activities. I want to use the lab resources on my schedule, not yours.”

Upon reflection, we agreed with that assessment. We were also mindful that devoting a staff of five persons for a class of 53 (which was possible one time only using the CCLI funding) was an unsustainable instructional model, and so we proposed a new approach based upon virtual laboratories as opposed to physical labs. With support from the U.Va. Teaching + Technology Initiative [2], we converted our physical labs into virtual labs by adding the supporting material that would be needed for 24/7 operation when the course staff was not available. We envision a slowly increasing collection of teaching modules, and indeed in the past two years we have added new modules for PHP, XML, C#, and web services.

In the following sections we discuss the choice of lecture content and how it is supported by in-class activities and the virtual labs, what homework exercises are assigned to reinforce the topics presented, the virtual lab content and usage, and our plans for future development. In fall 2005 we will try yet another experiment, this time using Tablet PCs, Conference XP and OneNote.

LECTURE COVERAGE

Over the ten-year history of the course we have developed a variety of lecture topics, some of which have proved enduring (e.g., cryptography) and some fleeting (e.g., Flash). While about 75% if the course is devoted to technical topics, the remaining quarter covers material not often seen in computer science or engineering courses (e.g., case study of eBay’s
business practices, creating an e-business and raising venture
capital, protecting intellectual property), and perhaps for that
very reason the non-technical topics receive consistently high
evaluations.

The lecture material used in fall 2004 is shown in figure 1. No set of lecture topics is “right”; these are the topics of interest in 2004, and ten years of experience has shown that the lecture topics exhibit about a 20% turn-over from year to year in an attempt to keep the class modern and relevant. Other instructors will unquestionably prefer a different mix of topics, and the modular nature of the lecture and virtual labs enables an instructor to do exactly that.

Some topics worked better than others. For example, an established topic such as symmetric key cryptography is now supported by well-rehearsed slides and a Java-based DES simulator that allows the student to see each and every transformation required to convert plaintext into ciphertext (see http://intercom.virginia.edu/crypto/crypto.html). In contrast, our introduction of C# and web services proved exceptionally challenging. Because neither the university nor the CS department provided adequate infrastructure support for .NET, this burden fell on the students (and hence on the instructor and the teaching assistant) and consumed an inappropriate amount of time for all concerned.

### Homework

Lectures and labs were augmented with six homework assignments. Each homework was of two weeks duration, except web services for which we budgeted three weeks.

#### Lectures Lecture Topic In-class support Virtual lab exercises

<table>
<thead>
<tr>
<th>Lectures</th>
<th>Lecture Topic</th>
<th>In-class support</th>
<th>Virtual lab exercises</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>History of Internet, web, and e-commerce</td>
<td>PPT slides</td>
<td>HTML, CSS</td>
</tr>
<tr>
<td>1</td>
<td>HTML and CSS</td>
<td>live programming</td>
<td>archived examples</td>
</tr>
<tr>
<td>1</td>
<td>Web innovation; what’s new; how do they do that?</td>
<td>live exploration</td>
<td>eBay.com</td>
</tr>
<tr>
<td>1</td>
<td>Case study of eBay</td>
<td>live programming</td>
<td>JavaScript</td>
</tr>
<tr>
<td>3</td>
<td>Cryptography: history, symmetric key, DES, 3DES, public key, RSA, IDEA, AES</td>
<td>PPT slides</td>
<td>Java simulators for DES and RSA</td>
</tr>
<tr>
<td>1</td>
<td>SET, SSL, and PGP</td>
<td>reference manuals</td>
<td>Perl, CGI</td>
</tr>
<tr>
<td>2</td>
<td>Perl and CGI</td>
<td>e-auction example</td>
<td>SQL, PHP</td>
</tr>
<tr>
<td>1</td>
<td>Forming an e-business; venture capital</td>
<td>slides, business plan</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Copyrights, trademarks, patents</td>
<td>PPT slides</td>
<td><a href="http://www.uspto.gov">www.uspto.gov</a></td>
</tr>
<tr>
<td>1</td>
<td>Patents and patent infringement</td>
<td>famous cases</td>
<td>patent files</td>
</tr>
<tr>
<td>1</td>
<td>XML and C#</td>
<td>live demonstrations</td>
<td>XML, C#</td>
</tr>
<tr>
<td>1</td>
<td>Web services</td>
<td>live demonstrations</td>
<td>web services</td>
</tr>
<tr>
<td>1</td>
<td>Electronic payment systems</td>
<td>secure transactions</td>
<td><a href="http://www.paypal.com">www.paypal.com</a></td>
</tr>
<tr>
<td>1</td>
<td>Biometric identification and authentication</td>
<td>fingerprint scanner</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Internet access technologies, wired and wireless</td>
<td>802.11 demo</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Website security and reliability; viruses and worms</td>
<td>examples</td>
<td>code review</td>
</tr>
</tbody>
</table>

### HTML and CSS: Implement a feature-rich webpage (including graphic images, checkboxes, radio buttons, selection boxes and forms) by hand (i.e., without using FrontPage). In an age of WYSIWYG editors, one might question the relevance of hand-coding HTML and CSS (the argument is reminiscent of whether to teach square roots by hand when the world is full of calculators or whether to teach assembly language in an age of high-level languages). Our opinion is that someone has to know HTML/CSS at a fundamental level, and e-commerce students who will ultimately be responsible for privacy, security, and reliability are those people.

#### JavaScript: Develop an airline webpage that dynamically calculates fares based upon the user’s selection of flights and routes. This illustrates client-side interactions that validates user input and provide information without burdening the server. Another good problem is syntactic validation of credit card numbers, email addresses, and expiration dates.

#### Cryptography: Complete a written homework covering substitution and transposition ciphers, one-time pads, and additional calculations requiring the use of our Java-based simulators for DES, 3DES, and RSA. To show the importance of using long keys, we asked students to assume some distributed brute-force key-cracking environment capable of trying one trillion (or any other number) keys per second. It is an eye-opening experience for them to calculate the time required to try all $2^{56}$ keys of DES or all $2^{256}$ keys of AES (3.6x10^57 years for AES-256).

#### Perl: Create “Five-Minute Auctions,” a website used for posting items for auction, making bids, tracking bid histories, and automatically awarding the item to the high bidder after five minutes duration.
**SQL and PHP:** Create a website for a small e-store that can display an e-catalog, accept and verify orders for items, and manage inventory.

**C# and web services:** Write a web service that can implement a two-factor authentication technique using a static password and a dynamic random number that changes every minute. To gain access to a protected resource (we used a grade book), the student wrote a GUI that collected a password and number from the user and verified whether the number provided was the correct one for the particular minute during which access was attempted (we provided the code that generated a new random number once per minute). This exercise mimicked the actions of a commercial product (RSA SecurID [3]) that we demonstrated to the class for purposes of motivation.

**Choose one:** participate on the plaintiff or defense team in a mock patent infringement trial; technical report on any two biometric techniques; technical report on the state-of-the-art of smartcard technology; write a credible e-business plan; or negotiate a customized assignment to accomplish a personal e-business goal.

**INTELLECTUAL PROPERTY**

The protection of intellectual property is a topic often deferred to commerce, business, or law students. Recognizing that many companies today receive substantial income from licensing their patent portfolios, we cover intellectual property at the level of two lectures and a mock patent infringement trial to illustrate how the legal process works. We used U.S. Patent 5,960,411 to Hartman et al. [4], assigned to Amazon.com, which is colloquially referred to as the “Amazon one-click patent” because its claims encompass the placing of an order over the Internet using any of several “one-action” techniques such as one mouse click, one keypress, one sound, etc. We identified a commercial product that might infringe the “one-click” patent—the eBay “Buy-it-now” feature. Well in advance of our mock trial, four students were chosen to represent the plaintiff and four to represent the defendant; they studied the patent, patent law, and the accused infringing mechanism with the instructor. During the mock trial the plaintiff presented its evidence of infringement, the defense presented its arguments of non-infringement and patent invalidity, and the plaintiff rebutted the invalidity contentions; the other members of the class served as the jury. Following the mock trial, we heard the true story of the Amazon v. Barnes and Noble patent infringement case from a Washington, DC, patent attorney (Bruce Barker) who actually participated in that case. In the three years that we’ve held the mock trial, students consistently rate it as their favorite activity.

**VIRTUAL LABORATORIES**

To date we have created 141 programming exercises in ten topical areas as shown in figure 2. Each topical area is divided into three levels of difficulty: beginner, intermediate, and advanced. Students were advised to peruse the appropriate virtual lab materials as each topic was introduced in lecture, read each programming exercise, skip those that they believed they could implement without assistance, and concentrate on those that introduced an unfamiliar or poorly understood concept. Figure 3 shows a programming exercise that illustrates the use of cookies within PHP.

The philosophy of the virtual labs is the same as that of the lecture: there was no attempt to create a definitive set of topics or exercises that would be right for all instructors. Rather, we hope to build over time an extensive inventory of subject matter such that subsets of that material can be used as desired. All materials developed are freely available at http://iis.cs.virginia.edu/webweavers/ec and we welcome feedback, improvements, and/or contributions of any modules.

As each topical area was completed, we conducted a self-assessment of each student via our electronic survey tool called SurveySuite [5]. By the end of the semester, each of 53 students had answered 82 questions regarding the student’s view of how each topic was presented, whether the lecture had been effective, whether the virtual lab support for that topic truly resulted in new learning, whether the homeworks reinforced the concepts being taught, etc. Space does not permit a full discussion of the students’ views, but five questions answered by every student are worthy of some discussion.

**Should Virtual Labs be Graded?** The course staff took the view that assigning virtual lab materials as homeworks would require replacing the current, integrative homework exercises with some sequence of smaller, independent lab problems. The staff saw the integrative homework as being more valuable and thus virtual labs were presented as learning opportunities rather than required exercises. In our evaluations we asked the students outright for their opinion, and in all three years they agreed: the virtual lab exercises were valuable for teaching individual concepts (e.g., how to validate a credit card), but not for integrating disparate concepts into a single e-commerce site. The students believed that the homeworks did accomplish that goal and recommended that the virtual labs continue to be viewed as a resource rather than a required activity.

**Team Homeworks.** One year we allowed students to pursue homeworks independently or in teams of two, three, or four. Those students who chose to work independently proved to be among the most capable programmers in the class. Teams of two and three produced homeworks of similar quality. Without exception, teams of four generated some degree of internal friction, and it was a common complaint that one or more team members were not doing their fair share (thus teams of four were never used again). In this past year, all homeworks could be completed individually or in teams of two. In the fall class with 51 students and 6 homeworks, there was not a single complaint regarding team management, workload, or ineffective participation.
<table>
<thead>
<tr>
<th>LAB TOPIC</th>
<th>NBR OF EXERCISES</th>
<th>LAB EXERCISES</th>
</tr>
</thead>
<tbody>
<tr>
<td>HyperText Markup Language (HTML)</td>
<td>12</td>
<td><em>Beginner</em>: structural setup; page layout; text manipulation; special characters; images; links. <em>Intermediate</em>: image maps; tables; frames. <em>Advanced</em>: forms; meta tags; web forms.</td>
</tr>
<tr>
<td>Cascading Style Sheets (CSS)</td>
<td>12</td>
<td><em>Beginner</em>: embedding/linking; HTML element selectors; classes; ID selectors. <em>Intermediate</em>: text manipulation; background; borders and spacing; layout; context selectors and grouping. <em>Advanced</em>: pseudo-classes; pseudo-elements.</td>
</tr>
<tr>
<td>JavaScript</td>
<td>15</td>
<td><em>Beginner</em>: writing your first script; creating HTML tags; user input and output; loops and tables; payroll calculator. <em>Intermediate</em>: forms and text fields; validating an email address; radio buttons; check boxes; self-grading tests. <em>Advanced</em>: image rollovers; slide shows; real-time clock; controllable clock; working with cookies.</td>
</tr>
<tr>
<td>Perl/CGI</td>
<td>10</td>
<td><em>Beginner</em>: sample Perl operations; random numbers; lists; dealing four poker hands; time manipulation; subroutines. <em>Intermediate</em>: hash tables; files; string matching. <em>Advanced</em>: CGI; registration lists; surveys.</td>
</tr>
<tr>
<td>SQL and regular expressions</td>
<td>24</td>
<td><em>Regular expressions</em>: basics; repeating; positioning. <em>Beginner</em>: select; where; order by; insert; update; delete. <em>Intermediate</em>: like; between; in; distinct; group by; aliases; aggregate functions; create table; alter table; drop table. <em>Advanced</em>: nested selects; SoundEx; join; deterministic functions; non-deterministic functions.</td>
</tr>
<tr>
<td>ASP</td>
<td>25</td>
<td><em>Beginner</em>: structural setup; response. write; retrieving from forms; retrieving from querystring; variables; control constructs; subroutines and functions; session state; application variables; server variables; debugging. <em>Intermediate</em>: reading and writing cookies; server-side includes; response object methods; VBScript functions; error handling; debugging. <em>Advanced</em>: browser details; CDONTS; files; output from a recordset; global.asa; setup instructions for using IIS and ASP.</td>
</tr>
<tr>
<td>Flash</td>
<td>3</td>
<td>Create Flash movies of moving and interactive objects.</td>
</tr>
<tr>
<td>C#</td>
<td>9</td>
<td><em>Beginner</em>: Visual Studio .NET; GUI elements and form designer; common syntax. <em>Intermediate</em>: ASP .NET web forms and SQL; MySQL databases using ADO .NET; integrating C# with Excel. <em>Advanced</em>: creating an XML web service using C#; language interoperability using C# and web services; dynamic control creation in web applications using C#.</td>
</tr>
<tr>
<td>XML</td>
<td>16</td>
<td><em>Beginner</em>: creating an XML document; XML parsers; advanced components; attributes vs. elements; displaying XML. <em>Intermediate</em>: legal building blocks; advanced DTD; linking DTD and XML; W3 XML schemas; advanced schemas. <em>Advanced</em>: extensible stylesheet language (XSLT); simple API for XML parsing; W3C document object model; simple object access protocol (SOAP); introduction to XML web services.</td>
</tr>
</tbody>
</table>

FIGURE 2
TOPICS AND PROGRAMMING EXERCISES FOR THE VIRTUAL LABORATORY
EXERCISE I4 – PHP SESSION VARIABLES

Session Variables allow you to use variables across multiple scripts, but unlike cookies, these variables are stored in the server's memory. This is useful for adhering to user-specified preferences, or for remembering the user's name. To create a session, run session_start() at the beginning of every PHP script that will use session variables. When a session is started, variables can be created with the function session_register('variable name'). When you are finished using the variables, they can be deleted by calling session_unregister('variable name'), or the entire session, along with all associated data, can be destroyed by calling session_destroy().

Sample Code

```php
<?php
    //Filename: I4_sample.php
    session_start();
?>

<html>
<head>
    <title>Sample Code I4</title>
</head>
<body>
    <p><font color="<?php echo($color);?>">Select a font color:<br></p>
    <form method="get" action="I4_sample2.php">
    <select size="1" name="mycolor">
        <option selected>Black</option>
        <option>Blue</option>
        <option>Orange</option>
        <option>Red</option>
    </select>
    <input type="submit" value="submit" name="btnSubmit">
    <input type="reset" value="reset" name="btnReset">
    </form>
    <p>
</body>
</html>

<?php
    //Filename: I4_sample2.php
    session_start();
    $color = $mycolor;
    session_register('color');
?>

<html>
<head>
    <title>Sample Code I4</title>
</head>
<body>
    <font color="<?php echo($color);?>">
        Using a session variable, you have just set your font color to be <?php echo $color; ?>.
        Click <a href="I4_sample.php">here</a> to change it.
    </font>
</body>
</html>

Cautions

1. session_register() will not work inside a function definition unless the variables are defined as global inside the function, e.g. global $session_variable
2. Session variables rely on cookies, so session variables will not work with browsers that do not support cookies or do not have cookies enabled.
3. session_start() and session_register() must be called before anything is sent to the browser, even a newline.
4. Once a variable is registered, it will not be available for the remainder of the script. It will be available for subsequent scripts in the same session.
Session T2H

**Self-Assessment of Learning.** For each topic, each student was asked to make a self-assessment of how much the various course components (lectures, virtual labs, homeworks) had contributed to that student’s knowledge of each subject. The scale used was: 1 – none; 2 – a little; 3 – a moderate amount; 4 – a substantial amount; 5 – an enormous amount. The student responses across the ten topics for each of the three course components ranged from 3.37 to 4.10, suggesting that the students credited the course with a moderate to substantial increase in their knowledge of these subjects.

**Improvements to Existing Modules.** As part of the evaluation of each lab topic, students were asked to identify ambiguities and errors in all the assignments and answers provided. The responses pointed to specific exercises that seemed unclear or incomplete (e.g., "The lab exercise on Perl hash tables did not show how to extract an entire table row as a list"), as well as to some inadequacy in the setup instructions (e.g., "I could make ASP connect to an Excel spreadsheet, but then had difficulty changing that code to connect to the SQL database"). Although several minor errors and ambiguities were uncovered in this first offering, none were severe enough to disrupt the lab in any major way.

**Virtual Lab Support.** The challenge of a virtual lab is not so much in the lab content as it is in the requisite ancillary resources. A student working a lab at 3 a.m. is not going to have reliable access to the course staff. At least five alternatives come to mind:
- traditional, physical, fixed-time office hours for the course staff
- electronic, asynchronous contact with the course staff via email
- audio/video consultation with the course staff via Microsoft NetMeeting or similar
- continuously monitored and rapidly updated (by the staff) FAQs for each lab topic
- student-operated free-form electronic interchanges (bulletin boards and email forums)

The last option shifts the burden of course support from the staff to the students, as they discover questions and problems and exchange techniques and solutions. An implementation, comparison, and evaluation of these (and other) methods of student support would be instructive.

**New Topics.** While the students made no objection to the initial set of topics covered, there was much enthusiasm for additional coverage (even though implementation of this idea would require a follow-on course to achieve). Suggested additional topics included:
- implementing cryptographic algorithms
- creating and verifying digital signatures
- experimenting with Secure Electronic Transactions
- utilizing Secure Sockets Layer (SSL) code
- processing email protected using Pretty Good Privacy
- biometric identification devices (e.g., fingerprints, iris scans, handwriting)
- ethical hacking
- handling multimedia: audio, pictures, bitmaps, streaming video
- experimenting with speech recognition and speech synthesis
- Virtual Reality Modeling Language (VRML)
- mobile e-commerce applications (e.g., PDAs and cell phones)
- e-commerce applications to medicine (e.g., secure electronic prescriptions)
- e-commerce accessibility issues for the disabled

**PLANS FOR THE FUTURE**

Our CS faculty continually question whether new or different technology in the classroom will lead to an improvement in learning outcomes (no doubt many other faculties have this same discussion). With the support of Microsoft Research [6], we are planning an experiment in fall 2005 to determine whether using Tablet PC technology (including Conference XP and OneNote) will alter learning outcomes. We will run one session of CS453 with the students taught in the conventional way as described above, and another section of approximately the same size in which all work will be done in teams using one Tablet PC per team. We do not presuppose that the use of the Tablets will be a positive influence; instead, our goal is to determine how that technology affects learning (as measured by pre- and post-tests, homeworks, exams, self-assessment, and instructor opinions). One issue to be studied is whether the powerful multimedia archiving capabilities of Conference XP encourages unintended side-effects such as reduced lecture attendance or reduced student interaction and teaming.

**ACKNOWLEDGMENT**

We gratefully acknowledge the support of the National Science Foundation’s DUE CCLI program, the University of Virginia’s Teaching + Technology Initiative, Microsoft Research, and the programmatic contributions of many students, including Andy Snyder, Tim Mulholland, Logan McKinley, Chris Adams, Sean O’Connor, Rahul Gupta, Kevin Thomas, Andrew Marshall, and Paul Bui. Special thanks are always due to Kim Gregg, our group’s fabulous administrative assistant.

**REFERENCES**