Design and Evolution of An Undergraduate Course on Web Application Development

Kwok-Bun Yue and Wei Ding
Computer Science and Computer Information Systems Programs
School of Science and Computer Engineering
University of Houston – Clear Lake
Houston, TX 77058 USA
{ Yue | Ding }@cl.uh.edu

ABSTRACT

Web technologies have become essential in the computing curricula. However, teaching a Web development course to computing students is challenging because of large bodies of knowledge, rapidly changing technologies, demanding support infrastructures and diverse background of audiences. This paper presents the evolution and the experiences we have gained in teaching a Web development course for the past seven years. We incorporate selected leading edge Web technologies as soon as they become mature and stable. The course covers a broad spectrum of Internet technologies to provide a solid conceptual framework. It also includes an in-depth study of a selected technology to provide the necessary depth and knowledge to build realistic Web applications. This paper describes the course design, our choice of topics, programming assignments, course delivery and our experience in coping with the rapidly changing Web technologies.

Categories and Subject Descriptors: K3.2 **[Computers and Education]:** Computer and Information Science Education-*Computer Science Education;* D.2.11 [Software Engineering]: Software Architectures-client/server.

General Terms: Design, Experimentation, Languages.

Keywords: Web Programming, Internet Technology, Web Architecture, Pedagogy, Computer Science Education, JSP, Java Servlets, ASP, Cold Fusion, Perl, HTTP, CGI, HTML, JavaScript, CSS, XML, XSL, XHTML, .NET, C#, J2EE.

1. INTRODUCTION

Web technologies have become essential in the computing curricula. However, teaching a Web development course to computing students is challenging because of large bodies of knowledge, rapidly changing technologies, demanding support infrastructures and diverse background of audiences.

ITICSE'04, June 28–30, 2004, Leeds, United Kingdom.

Different approaches of delivering courses on Internet technologies have been reported [5, 10, 11, 14, 15]. Nearly all of them include some coverage of both server-side and client-side Web development. Although there are many Web server-side technologies, most attempts focus on only one of them, such as CGI-Perl [5, 10, 15], or Java/JSP/Servlet [11]. With the exception of [11], all of them do not cover Web development deeply. Instead, the courses devoted much time on other Web-related topics, such as compression, Web security, Web server administration, copyright and ethics. However, as Web applications become more complex, an advanced Web development course is desirable.

In the University of Houston-Clear Lake (UHCL), a senior level course on Web application development has successfully been taught for seven years. The course was first introduced in fall 1996 as "Software Tools" and has then been renamed to "Internet Application Development" (IAD).

IAD may serve as an example for other educators who would like to design a similar course. The course covers a broad spectrum of Internet technologies to provide a solid conceptual framework. It also includes an in-depth study of a selected technology to provide the necessary depth and knowledge to build realistic Web applications.

This paper describes the evolution and the experience we have gained in teaching IAD. Section 2 presents the design goals of IAD. The criteria for selecting choices of topics and how they affect the course evolution are presented in Section 3. Section 4 describes our programming assignments. Course materials and delivery are discussed in Sections 5 and 6 respectively. Our experience on coping with rapidly changing Web technologies is described in Section 7 and we draw our conclusions in Section 8.

2. COURSE DESIGN

The course began in the fall semester of 1996 with the title "Software Tools" to cover emerging and leading practical tools and techniques for computing students. As the Web was becoming very popular, more than 60% of the topics of the first offering were already devoted to Web techniques, including HTML, CGI-Perl, JavaScript and Java applets.

The course has then evolved quickly to a serious Web programming course with the title changed to "Internet Application Development" in 2000. This emphasis in programming is similar to that of [11] but in contrast with other courses that include significant Web architecture topics such as compression, Web security, Web administrations and intellectual properties [5,10,15].

Besides covering popular topics such as Web standards and client-side technologies, IAD has many distinctive goals throughout its evolution:

1) It covers a broad spectrum of Web technologies. For example, at one point, four Web server-side technologies were covered: CGI-Perl, ASP, Cold Fusion and Java Servlets/JSP. This broad coverage not only provides a survey of the landscape so that students can gain insight by comparing and contrasting each technology, it also assists them to gain a head start in their jobs. Many have reported that specific projects done in selected technologies were the major reasons companies hired them. Programming and project assignments will be discussed in Section 4.

2) It covers a selected Web technology in more depth than many other comparable courses. For example, in the current offering, there are 12 hours of lectures and a term project in ASP.NET with C#. This allows the students to understand the intricacies of building sophisticated real-world Web applications. More importantly, Web programming and design theory can be better understood, appreciated and applied with the emphasis on in-depth details.

3) Leading edge Web technologies are selected to keep abreast of the practices of the industry, providing students an understanding of the current and future trends and practices of problem solving techniques. A newer technology usually has the advantage of providing a better and more elegant model of solving problems encountered in the older technologies. Thus, using leading edge techniques is not just about keeping up with the latest and greatest, but also for introducing new models to enrich student's repertoire of problem solving techniques.

   Many educators have reported difficulties in keeping up with the huge collection of quickly changing Web technologies and have proposed different coping approaches such as focusing on concepts [10] or using a seminar format [14]. Although the decision to keep abreast of industrial practice is very beneficial, we have same difficulties as other educators. We will discuss our experience in Section 7.

4) Projects and homework assignments are designed to be actually useable in real-world applications with no or minor modifications. Section 4 discusses our related experience.

The first three goals are related to the choices of topics to be included, which are discussed below.

3. CHOICES OF TOPICS

Selecting from the very large collection of Web technologies to include into a course is a significant challenge. Like many other educators, we use two sets of criteria to guide the evolution of our course: a set of technology criteria that may have more universality and a set of resource and curricular criteria that is more specific to UHCL.

A Web technology is measured by three main criteria for inclusion:

1) Popularity: A model of Web application development embraced by many developers is more likely to be robust and significant. Thus, the relative popularity of a Web technology affects its inclusion and coverage. For example, at the early offerings of the course, Perl and CGI-Perl programming represented more than one third of the course contents. They are now covered in two weeks.

2) Maturity and stability: Many educators have reported problems in insufficient, outdated and incorrect documentation and difficulties in maintaining stable supporting platforms [11, 15]. We decided to alleviate these problems by covering technologies only when they become relatively mature and stable. For examples, although we have lightly covered ASP.NET for more than two years, it has only become the main thrust in spring 2003.

3) Features and innovativeness: A Web technology is also judged on how well it highlights various Web techniques and approaches. For example, despite its popularity, PHP has not been selected since many of its features are similar to ASP and CGI-Perl. Instead, PHP is included in a separate graduate course.

The initial inclusion of Cold Fusion in 1998 and its exclusion in 2000 may serve as an example of the interaction of these criteria. When Cold Fusion was first selected, it was relatively popular. More importantly, with the Cold Fusion Markup Language (CFML), it was the leading Web server-side technique using embedded declarative tags. It was decided that the embedded tag is an innovative and important approach to complement the scripting approach used by CGI-Perl, PHP and ASP. After its inclusion into the course, Cold Fusion had become less popular. Its embedded tag approach has also been embraced in different formats by other Web technologies, such as Taglib in JSP and server-side controls in ASP.NET. As a result, it was replaced by Java Servlets/JSP, which provides a model of using a strong object-oriented architecture, through J2EE, to build complex Web applications.

The set of local criteria for course coverage can be classified into two categories: curricula and resources. Important curricular criteria include other available computer science courses, core requirements and prerequisite structures at UHCL. Important resource criteria include existing computing platforms and laboratories, and background and expertise of the instructors. As an example, we are currently de-emphasizing Java Servlets and JSP since the department is now offering two undergraduate courses in software development in Java. That makes room for the increased coverage of the .NET framework. Since different institutions have different sets of local criteria, we will not further elaborate them.

Using these three criteria, IAD has steadily evolved in the past seven years. Table 1 lists the approximate time when a Web

technology became relatively stable and available, and when it was incorporated into IAD.

Table 1. Technology's Incorporation into IAD

Technology	Available	Incorporated to IAD
Perl	1995	1996
CGI-Perl	1995	1996
JavaScript	1995	1996
Java	1995	1996
CSS	1996	1998
ASP	1997	1997
Cold Fusion	1997	1998
Java Servlet/JSP	1999	2000
XML	1999	2001
XSL	2001	2001
ASP.NET	2001	2003

The current syllabus of IAD includes:

- Introduction to Web application development: HTML, XHTML, CSS and XML: 3 weeks.
- Client-side programming with JavaScript and DOM: 3 weeks.
- Server-side programming with ASP.NET and C#: 4 weeks.
- Server-side programming with Java Servlets and JSP: 2 weeks.
- Server-side programming with CGI-Perl: 2 weeks.

Several additional points are worthy to be further discussed on this syllabus. Firstly, although not explicitly mentioned, important concepts and standards in Web development are scattered throughout this technology-oriented syllabus, including HTTP, cookies, sessions, database connectivity, regular expressions and string manipulation, concurrent programming, etc. Some of these concepts are covered in more than one technology for comparison and contrast. Each concept is covered in depth in at least one selected technology. For example, database connectivity is currently discussed using ADO.NET/C#, Perl/DBI and JDBC, with an in-depth treatment using ADO.NET. This allows the students to better understand the common features of the various approaches, which represent the underlying concepts. It also helps the students to gain insight of the various design options under different requirement constraints.

Secondly, it can be seen that IAD focuses more on the server-side rather than the client-side. This is in synchronization with industrial trends, especially since the client-side has increasingly suffered from compatibility and security issues. Even using standard JavaScript only may not be sufficient to reach all target users as some may turn off JavaScript entirely in their browsers.

Finally, the current selection of server-side technologies represents the three major platforms of Web development:

- Microsoft's ASP.NET under the .NET Framework
- Sun's J2EE/JSP/Servlet
- The open source platform LAMP: Linux, Apache, MySql, and Perl, PHP or Python (we selected Perl).

4. PROGRAMMING ASSIGNMENTS

We use the following goals to design programming assignments for IAD:

- Realistic: the assignments should be similar to useful real-world projects.
- Complete and ready: the products of the assignments should be Web applications that can be deployed with no or little modification.
- Technically important: the assignments should use important concepts and technologies.
- Illustrative and interesting: the assignments should be intellectually appealing and interesting.

To satisfy these goals, we have designed assignments in typical mainstream Web development, such as customized version of shopping carts and online auctions. We have also designed assignments that are more opportunistic to capture what was hot and interesting at the moment. For example, in 2001, Newsweek reported that the site *AmIHotOrNot* has come out from nowhere to quickly achieve more than 1 million hits per day. The site allows users to post their photos and others to rate them. We immediately designed a copycat programming assignment, "am my code hot or not," to allow users to post code for others to rate. The students rated the project as one of the best.

We have also experimented with different arrangements of programming assignments within the semester. They can be roughly classified into three categories:

1) A collection of unrelated assignments,
2) A tightly coupled term project composing of a sequence of related assignments, with one building on top of previous assignments, and
3) A loosely coupled project of a sequence of related yet independent assignments

We found that each has its relative merits. A collection of unrelated assignments provides the most flexibility to target more technologies and concepts. A tightly coupled project provides the experience of building a more sophisticated and complete Web application.. A loosely coupled project is somewhere in between.

Since one main goal of IAD is to provide a broad coverage of leading edge technologies, a tightly coupled project means that students will need to integrate separate technologies. This provides special challenges that are even greater than typical real world Web development projects, which tend to be more homogeneous.

For example, a tightly coupled project in Fall 2002 required students to design and implement an online registration system. HTML, CSS, JavaScript, regular expressions and DOM were used in the client-side to handle various forms. In the server-side, various components were implemented using CGI/Perl, ASP and Java Servlets/JSP respectively. Session tracking was used to integrate these components. However, although session tracking in one technology is easy, cross technology session tracking was much more complicated. For example, it is difficult to pass an ASP.Net session object to a CGI-Perl program. Eventually, we had to relax some of the session tracking requirements.

In our experience, no single arrangement is universally better than the other and we will continue to experiment. In Fall 2003, we will attempt a mixed approach where a term project and a sequence of smaller independent assignments will be required.

Creating and maintaining computing servers and laboratories to support assignments of a wide spectrum of technologies is difficult. Continuous experimentation is usually necessary to provide the best support. For example, to support Java Servlets and JSP, we have used different configurations of Sun's Java Web server, Weblogic, JRrun and Tomcat, each with relative successes. Furthermore, additional measures are required for the necessary stability. For example, to increase load balancing and availability, we current use two platforms to support JSP assignments: Microsoft's Windows/IIS/JRun/Access and Linux/Apache/Tomcat/MySQL.

5. COURSE MATERIALS

Because of the large range of topics and rapidly changing technologies, it is a well-known problem to find a good textbook to meet the needs of a Web programming course. Most Web-related books are either professional technical reference books for a specific topic or do not provide sufficient support required by a textbook. Other universities faced similar problems: Treu [14] used Web resources as the primary source; Lee, Walker and Browne [11, 15] used textbook with tutorials and references on the Internet. The problem is even more challenging for IAD due to its broader coverage. Three different resources are used to assist students from different perspectives:

- Lecture notes and laboratory examples written by the instructors,
- A textbook as the major reference book, and
- Freely available tutorials on the Web carefully chosen by the instructors.

The best teaching materials are usually written directly by the instructors to specifically meet the course objectives. However, this is also a time-consuming task, as the instructors often need to use many reference books and Web resources to develop their courseware. Section 7 further elaborates our experience on coping with the rapidly changing technologies. Our current favorite reference books are [1, 3, 6, 7, 8, 9, 13].

A main textbook is the secondary resource used to complement instructor's notes. Three textbooks have been used at different time in the past: *Webmaster in a Nutshell* [12], *Internet & World Wide Web: How to Program* [4], and *The Web Warrior Guide to Web Programming* [2]. Textbooks have been changed mainly because the new ones had better coverage on the main technologies the course focused on.

Although there are many online tutorials on Web technologies, it is often difficult for students with little Web development experience to evaluate their values in terms of timeliness, correctness, and relevance. IAD encourages students to recommend online tutorials or articles. They are then filtered by the instructors for the class to use.

6. COURSE DELIVERY

IAD is a Web-assisted course based on traditional lectures. Students have widely diverse background, from beginners with only a few computing courses to highly experienced professional with solid academic credential. Lectures were thus designed to include little prerequisite knowledge to cater to the widest audiences initially. However, they usually also include deep treatment in various selected issues to keep the advanced students interested. Our experience also indicates that using a lot of examples with varying degrees of difficulty works well with a diverse audience.

IAD is nearly paperless: a course website [16] is used to host all related materials for easy and timely access. These materials include lecture notes, resources, examinations and programming assignments. Furthermore, to enhance learning by comparison, either instructor's solutions or selected student solutions were posted for the assignments. Students also have accesses to materials of the previous semesters. The use of an online forum has proved to be very popular among students, allowing students and instructors to participate in discussion anytime.

In addition, several seminars have been offered by the teaching assistants to assist students who need face-to-face laboratory instructions on selected topics, such as installation of Web server and usage of selected IDE tools.

7. COPING WITH RAPIDLY CHANGING WEB TECHNOLOGIES

Most other educators have reported difficulties in coping with the large collection of rapidly changing technologies. This is an even larger issue for us since IAD has a goal of incorporating a broad coverage of leading edge technologies. However, our experience shows that it is possible to achieve the goal, even for smaller universities where resources may be strained.

Our experience indicates that faculty members dedicated to learning the fast evolving technologies are crucial in developing such courses. Fortunately, with dedicated faculty members in an interesting subject with high demand, these courses usually result in very good student evaluations and very high student enrollment, which will benefit the career advancement of the faculty members. Furthermore, in many cases, the popularity of the courses ensure large number of sessions, increasing the reuse of the course materials and decreasing the number of separate preparations of the faculty members. For example, at UHCL, three sections of IAD are offered every regular semester and two sections are usually offered during the summer semester. In fact, IAD was the only course that one of the authors taught in a span of 1.5 years.

On the other hand, even with reduced remaining teaching preparations, we still used a cautious incremental approach to incorporate new technologies into the course. For example, ASP.NET was first surveyed in a one-hour lecture in fall 2001. Since it has become mature in 2003, the coverage expanded to three hours in spring and six hours in summer. In fall 2003, ASP.NET was selected as the in-depth technologies with a 12-hour treatment with a term project assignment.

This incremental approach provides an opportunity for the instructor to become more proficient in the technologies to know what and what not to incorporate into the courses. It also provides the instructors with sufficient time to accumulate resources, examples and assignments.

However, since another goal is to incorporate important leading edge technologies only after they become mature as soon as possible, an incremental approach requires spotting the promising technologies early, a difficult task requiring the instructors to pay constant attention. Fortunately, as Web techniques become more mature and consolidated, this has become a little easier.

Even though there is usually an abundance of course materials in the Web to create our courseware, we have encountered two major problems. The first is the quality of these materials, many of which may be obsolete, incorrect and simply of low quality. The second is their copyright, which may prohibit usage or may be difficult to figure out. The instructors have many times drastically modified good examples found in the Web to ensure good quality with no copyright violation. Thus, a repository of high quality course materials with an open source license should be very beneficial to this and other courses. Both authors of this paper are now involved in developing such a collaborative open courseware repository.

8. CONCLUSIONS

Despite of the various difficulties we have encountered in IAD, we believe that the design goals have successfully been met. Our experiences indicate that a Web programming course that has both broad and in-depth coverage of important lead edge technologies are both feasible and desirable.

About 1,300 students have taken the course since its inception. Student evaluations and enrollment have consistently been excellent. The course was nearly always full and was the most popular senior level course. A very high percentage of students have reported that they have found their first jobs specifically because of the course. More importantly, students found the course interesting and exciting. The course has prepared them well to take other related Web courses, such as Web database development and XML application development. With the addition of a faculty member in this area, we plan to introduce a continuation E-Commerce development course in fall 2004.

9. REFERENCES

[1] Allamaraju S., et al. Professional Java Server Programming J2EE 1.3 Edition. Wrox, 2001.

[2] Bai X., et al. The Web Warrior Guide to Web Programming. Thomason Learning Course Technology, 2003

[3] Buser D., et al. Beginning Active Server Pages 3.0. Wrox, 1999.

[4] Deitel H., Deitel P., and Nieto T. Internet & World Wide Web: How To Program. Prentice Hall, 2001.

[5] Finkel D. and Cruz I. Webware: A Course about the Web. ACM SIGCSE Bulletin, Proceedings of the 4th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education, Volume 31 Issue 3, June 1999.

[6] Flanagan D. JavaScript The Definitive Guide. O'Reilly, 2002.

[7] Goode C., et al. Beginning ASP.NET 1.0 with C#. Wrox, 2003.

[8] Hall, M. Core Servlets and JavaServer Pages. Prentice Hall, 2000.

[9] Kauffman J. Beginning ASP Databases. Wrox, 1999.

[10] Klassner F. Can Web development courses avoid obsolescence? ACM SIGCSE Bulletin, Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education, Volume 32 Issue 3, July 2000.

[11] Lee A. A Manageable Web Software Architecture: Searching for Simplicity. Proceedings of the 34th SIGCSE technical symposium on Computer science education, February 2003

[12] Sainhour S. and Eckstein R. Webmaster in a nutshell. O'Reilly, 1999.

[13] Siever E., Spainhour S., and Patwardhan N. Perl in a nutshell. O'Reilly, 1999.

[14] Treu, K. To teach the unteachable class: an experimental course in web-based application design. ACM SIGCSE Bulletin, Proceedings of the 33rd SIGCSE technical symposium on Computer science education, Volume 34 Issue 1, February 2002.

[15] Walker, E. and Browne, L. Teaching Web development with limited resources. ACM SIGCSE Bulletin, The proceedings of the thirtieth SIGCSE technical symposium on Computer science education, Volume 31 Issue 1, March 1999.

[16] Yue, K. and Ding, W. UHCL CSCI 4230 Course Website. http://dcm.cl.uh.edu/yue/courses/csci4230/Fall2001/index.asp. Fall 2001. http://sce.cl.uh.edu/ding/classes/4230. Fall 2003.