

A WEB-BASED QUIZ DELIVERY SYSTEM USING AJAX (ASYNCHRONOUS
JAVASCRIPT AND XML)

by

Rance Scott Harmon

A project submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

December 2009

©COPYRIGHT

by

Rance Scott Harmon

2009

All Rights Reserved

STATEMENT OF PERMISSION TO USE

In presenting this project in partial fulfillment of the requirements for a master's degree at Montana State University, I agree that the Computer Science Department shall make it available on the department's website.

If I have indicated my intention to copyright this project by including a copyright notice page, copying is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for permission for extended quotation from or reproduction of this paper in whole or in parts may be granted only by the copyright holder.

Rance Scott Harmon

December 2009

ACKNOWLEDGEMENTS

I would like to thank Dr. Rockford J. Ross for his support and patience throughout this project. In addition, I would like to thank Scott Dowdle, the Montana State University Computer Science Department's system manager, for his assistance in helping me set up the server resources for this project. In addition, I would like to thank my fellow students for their suggestions about the project, especially, Karen Schuller, Jon Hauer, and Swapam Dutta Roy. This work was supported, in part, by National Science Foundation Grant No. 0618744.

TABLE OF CONTENTS

INTRODUCTION	1
INITIAL DESIGN CONSIDERATIONS AND REQUIREMENTS	3
REVISED VERSION DESIGN CONSIDERATIONS AND REQUIREMENTS	5
USER GUIDE	8
FUTURE WORK	17
REFERENCES	18

LIST OF FIGURES

Figure	Page
1 Functional Requirements for Self-Check Quiz System Prototype.	4
2 Functional Requirements for Self-Check Quiz System Revised Version.	6
3 Appearance of the Biofilm Hypertextbook Quiz Delivery System With No Specific Quiz Loaded Initially.	9
4 Appearance of the Biofilm Hypertextbook Quiz Delivery System With a Specific Quiz Loaded Initially.	10
5 An Example of the File (Quizzes.xml) That Specifies the Available Quizzes.	11
6 A Sample Three-Question Quiz As It Appears in the Hypertextbook Quiz Application and the Corresponding XML File.	13
7 A Sample of the Feedback Generated by the Quiz Application for the First Question in Figure 6, in Which the User Selected Only the First Two Distractors.	16

ABSTRACT

This paper will discuss a web-based quiz delivery system for *Biofilms: The Hypertextbook*. Hypertextbooks differ from traditional textbooks by including interactive resources that can enhance student learning. While developed specifically for the biofilms hypertextbook, the quiz delivery system uses standard technologies and a modular design, so hypertextbooks containing different subject matter or other types of websites can easily add it to their offerings. The quiz delivery system uses Ajax (Asynchronous JavaScript and XML)—a combination of technologies that can make web applications more responsive to user requests. Online self-check quizzes can significantly enhance student learning and retention by mitigating the effects of student stress and enabling students to make adjustments before they take a graded assessment. The prototype quiz delivery system provided immediate feedback for each question. The feedback included both the correct answer and an explanation of why each distractor was correct or incorrect. A revised version, based on stakeholder input, includes the following improvements: allowing an arbitrary number of distractors for each question, enhancing the user interface, adding screen shots to the instruction page, permitting images as well as text to appear in questions, improving cross-browser functionality, and enabling entry to the quiz delivery system with a specific quiz loaded. We used the Google Web Toolkit (GWT) to develop the revised version. GWT enables programmers to write web applications using the Java programming language. The project includes a guide for website administrators, and the application contains an instruction page with screen shots for quiz takers. Future work could include adding a database component to store results and verify usernames and passwords, a widget that instructors or other authorized users could use to add quizzes, and a secure environment for administering for-credit quizzes.

INTRODUCTION

This paper discusses the development of a web-based quiz delivery system for *Biofilms: The Hypertextbook* [2]. Hypertextbooks differ from traditional textbook by, among other things, including interactive resources that can enhance student learning. The quiz delivery system is one of these resources. It uses Ajax (Aynchronous JavaScript and XML) to enhance user experiences. Ajax is a combination of technologies that can make web applications more responsive to user requests.

The Center for Biofilm Engineering (CBE) at Montana State University is targeting *Biofilms: the Hypertextbook* as a long-term education initiative [3]. A biofilm is “a complex aggregation of microorganisms marked by the excretion of a protective and adhesive matrix” [13]. While, I developed the quiz delivery system specifically for the biofilms hypertextbook, the application can also work with hypertextbooks containing different subject matter.

In a review of the literature, Rößling et. al. [9] found that the term “hypertextbook” has no standard definition. For this project, we define hypertextbook as a teaching and learning resource delivered via standard web browsers that extends traditional textual presentations by adding pictures and illustrations; video clips; audio; and interactive, active learning visualizations of key concepts [3], [10].

Active learning tools and models are key features of the biofilms hypertextbook. The tools and models extend the capacity for teaching and learning by requiring students to become actively involved in the learning process [3].

An additional opportunity for enhancing the active learning capacity of the biofilms hypertext is the inclusion of self-check quizzes linked to content sections. Online quizzes can significantly enhance student learning and retention by mitigating the effects of student stress, allowing students to “practice” before graded tests and quizzes, and motivating students to acquire the expected skills [12].

Self-check quizzes are a form a formative evaluation. They provide students with feedback on how well they understand the material before the end of the unit. Thus, self-check quizzes enable students to make adjustments before they take the assessment for a grade at the end of the unit, when it is often too late to improve learning. Self-assessment quizzes can strongly benefit student learning [11].

Summative quizzes for credit present security issues when students print questions, feedback, and correct answers; have others take quizzes for them; use outside resources; or work with friends. Self-check quizzes avoid the need to address these security issues, since the self-check quizzes are formative rather than summative. Students are encouraged to use outside resources, such as the hypertextbook, and work with other students, both of which can foster further opportunities to enhance student learning.

INITIAL DESIGN CONSIDERATIONS AND REQUIREMENTS

In developing the quiz delivery system, we wanted to make it respond quickly to user requests. Ajax (Aynchronous JavaScript and XML) is a way to enhance user experiences on the Internet by making interactive web applications more responsive, so they feel more like traditional desktop applications [5], [6]. When the user interacts with a traditional, non-Ajax, web application, he has to wait for the server to respond—a pause resulting in the dreaded blank screen, progress bar, or hourglass. Ajax enables the server to do its work behind the scenes, so the user can continue to interact with the web page.

We used Ajax to develop the self-check quizzes because it enhances user experiences and relies on standards-based technologies, including XHTML (Extensible Hypertext Markup Language), CSS (Cascading Style Sheets) the Document Object Model, XML (Extensible Markup Language), and the JavaScript programming language. Another key design issue was to enable students to take the quiz as many times as they like and to provide them feedback on their answers, both of which enhance student learning [1].

The requirements we used to develop the first version of the biofilms hypertextbook quiz delivery system are in Figure 1.

Figure 1: Functional Requirements for Self-Check Quiz System Prototype

Name: FR-1: Quiz Link Page

Summary: Entry into the application will be through a web page that contains links to each quiz and a link to the instruction page.

Rationale: Each section of the hypertextbook, in which a quiz is associated, will link to the quiz application's main page. This will ease maintenance issues as developers move and reorder parts of the hypertextbook.

Requirements: The quiz application's introduction displays links to each quiz that is available, as well as a link to the Quiz Introduction Page.

Name: FR-2: Quiz Introduction Page

Summary: The introduction page opens in a new window (or tab) and explains how to use the quiz application.

Rationale: New quiz users need instruction in using the quiz system before they continue on to the actual quiz. Return users may need the introduction page to refresh their knowledge before taking the quiz.

Requirements: The introduction screen is a static HTML page that includes instructions about taking the quiz and a button to close the window (or tab) when they are finished reviewing the instructions.

Name: FR-3: Interactive Quiz Screen – Question Only

Summary: The interactive quiz screen displays one question at a time in multiple-choice format. Users select check boxes to choose answers.

Rationale: When a user has only one question in front of him, he is less distracted and can better concentrate on the question content. This is particularly important when the user has answered the question, and the screen dynamically displays the corrected version of the question.

Requirements:

- There will be a single HTML page that reads XML formatted documents from the server to retrieve questions that appear on the question screen.
- The initial page will include the question, with the multiple-choice answers beneath. Each answer has a check box associated with it.
- Each HTML page has a "Check Answer" button.
- When the user clicks the "Check Answer" button, the application moves to the Interactive Quiz Screen – Corrected Question state.

Name: FR-4: Interactive Quiz Screen – Corrected Question

Summary: When the user has finished answering the question, he clicks the "Check Answer" button, and the screen dynamically displays the corrected content of that particular question.

Rationale: By using dynamic content, the quiz pages provide the user with a richer experience while utilizing the quiz system. He will see feedback to his answers (right or wrong) as well as feedback about the other possible answers. The feedback explains why a particular answer was a correct choice or not.

Requirements:

- Ajax and HTML provide the means to display the corrected answer content dynamically when the user clicks on the "Check Answer" button.
- A script will parse the XML document to display the correct answers and feedback content. The feedback for each answer will appear beside each choice.
- Correct answers will turn green and incorrect answers will turn red.
- A "Next Question" button will provide a way for the user to move to the next question.

Name: FR-5: Display Quiz Results

Summary: When the user reviews his answers to the last question, the application will display a "View Results" button rather than the "Next Question" button. Clicking the "View Results" button will take the user to a page that displays the results of all the questions. From the results page the user has the option of closing the quiz or printing the results.

Rationale: The quiz results page gives the user a comprehensive list of her quiz results. The option to print the results allows the student to print a hardcopy for future reference or study.

Requirements:

- All individual question results will appear on one screen.
- "Close" and "Print" buttons will be supplied.

REVISED VERSION DESIGN CONSIDERATIONS AND REQUIREMENTS

After testing, deploying, and getting feedback from the biofilms hypertextbook development team and stakeholders, we decided to make the following changes to improve the quiz delivery system:

- Eliminate the problem of some buttons working differently with different web browsers.
- Eliminate the limitation of exactly five choices per question by allowing an arbitrary number of choices.
- Improve the appearance of the user interface.
- Eliminate the need to update aspects of the HTML and JavaScript code each time an administrator adds a new quiz or reorders chapter or section numbers.
- Add screen shots to the instructions page to make the instructions clearer.
- Enable the ability to have images as well as text in questions.
- Enable two entry points to the quiz application. One entry point loads a specific quiz contained as parameters appended to the hyperlink that directs the user to the quiz application. The other entry point opens the quiz application with no quiz loaded initially.

In response to these issues, we redesigned the quiz delivery system and implemented a new user interface. The requirements for the revised version of the quiz delivery system, based on the feedback, are in Figure 2.

Figure 2: Functional Requirements for Self-Check Quiz System Revised Version

Name: FR-1: Quiz Page

Summary: The quiz application will consist of one main HTML page, with a link to a static instructions page.

Rationale: Links from appropriate locations within the hypertextbook will direct users to the quiz application's main page. The links may contain parameters that specify a specific quiz to load or may be parameterless, in which case the application opens with no quiz loaded initially. Each section of the hypertextbook in which a quiz is associated will link to the quiz application's main page. This will ease maintenance issues as the administrator moves and renumbers parts of the hypertextbook.

Requirements: The Quiz Page will contain a link to an Instructions Page and a means of selecting and taking quizzes.

Name: FR-2: Quiz Instructions Page

Summary: The introduction page opens in a new window (or tab) and explains how to use the quiz application.

Rationale: New quiz users need instruction in using the quiz system before they continue on to the actual quiz. Return users may need the introduction page to refresh their knowledge before taking the quiz.

Requirements: The instruction page is a static HTML page that includes instructions and screen shots explaining how to take a quiz.

Name: FR-3: Quiz Selection Panel

Summary: The application will contain a panel that will enable the user to select the chapter, section, and level of the quiz to load.

Rationale: The user can easily choose any available quiz.

Requirements: The application will construct a tree in the quiz selection panel when the application loads by parsing an XML file that contains information about the chapter, section, and level of all existing quizzes.

Name: FR-4: Quiz Panel—Question Presentation State

Summary: The interactive quiz panel displays one quiz with tabs for all of the questions in that quiz. Each tab contains one question, with the text of the question followed by an optional image, followed by an arbitrary number of choices. Each choice contains a corresponding checkbox that the user

can select to indicate that he believes that choice is correct

Rationale: When a user has only one question in front of him at a time, he is less distracted and can better concentrate on the question content. This is particularly important when the user has answered the question, and the screen dynamically displays the corrected version of the question.

Requirements:

- The single HTML page will change dynamically in response to XML documents retrieved from the server.
- Each tab will include one of the questions, an optional image, and multiple choice distractors beneath. Each choice has a check box associated with it.
- Each tab has a "Check Answer" button.
- When the user clicks the "Check Answer" button, the current tab moves into the Quiz Panel—Corrected Question state.

Name: FR-5: Quiz Panel—Corrected Question State

Summary: When the user has finished answering the question, he clicks the "Check Answer" button, and the tab dynamically displays the corrected content for that particular question.

Rationale: By using dynamic content, the quiz pages provide the user with a richer experience. He will see feedback for all the choices (whether he answered the choice right or wrong). The feedback will include whether the user answered the question correctly and an explanation for each choice.

Requirements:

- Ajax and HTML provide the means to display the corrected answer content dynamically when the user clicks on the "Check Answer" button.
- An XML document will contain the correct answers and feedback content. The feedback for each answer will appear beside each choice.
- Correct answers will turn green and incorrect answers will turn red.
- The application will deactivate the check boxes and remove the "Check Answer" button in the Corrected Question State.

One major drawback of Ajax applications is that, without special programming considerations, the browsers “Back” and “Forward” buttons may not enable the user to navigate through information in the way she is accustomed in other web situations [5]. To help mitigate this pitfall and for other advantages explained below, we chose to implement the advanced version of the self-check quizzes web application using the Google Web Toolkit (GWT) [7].

GWT enables programmers to write web applications using the Java programming language. GWT has widgets that enabled us to spruce up the user interface. GWT cross-compile the Java code into optimized JavaScript that works with the major browsers [7] [8]. In developing the self-check quiz application, I used the GWT’s plug-in for the Eclipse integrated development environment [4]. Advantages of using Eclipse include code highlighting, code suggestion and completion, and incremental compiling. Writing code in Java has additional benefits in that Java is strongly typed, object-oriented, has an extensive API, and has support for unit testing.

USER GUIDE

Instructions for taking the quizzes are included on the “Instructions” link in the quiz application. The instructions contain detailed information for the user, including screen shots. The remainder of this section provides a guide for the hypertextbook administrator.

I sought to build the quiz delivery system as a modular component of the biofilm hypertextbook, so the code for the quiz application would not require refactoring to work with future changes to the structure of the hypertextbook. Communication between the quiz delivery system and the rest of the hypertextbook is limited to the links that direct users to the HTML page that loads the quiz application.

The link to the quiz application can take two forms:

1. A link to the quiz application with no specific quiz loaded initially
2. A link to the quiz application with a specific quiz loaded initially

A link to the quiz application with no specific quiz loaded takes no parameters. For example the URL might look like, <http://biofilmbook.com/quizzes>. Figure 3 shows how the application would initially appear if a link with no parameters directs the user to the quiz application.

Figure 3. Appearance of the Biofilm Hypertextbook Quiz Delivery System With No Specific Quiz Loaded Initially.



Self-Check Quizzes

▶ Select Quiz

⊕ Chapter 1

⊕ Chapter 2

⊕ Chapter 3

⊕ Chapter 4

⊕ Chapter 5

A link to the quiz application with a specific quiz loaded takes parameters for the chapter, section, and level. For example the URL might look like,

<http://biofilmbook.com/quizzes?chapter=001§ion=001&level=green> . The

parameters are appended to the general URL and follow the “?” character. Note that the parameters are in lower case, appear in key-value pairs, and that the chapter and section numbers must contain three digits (including leading zeros if needed). Figure 4 shows how the application would initially appear if a link with parameters directs the user to a specific quiz chapter, section, and level.

In both scenarios the user can employ the quiz selector (left side of the window) to select or change the chapter, section, and level of the quiz any time after the application loads.

XML files contain the content of the quizzes, so there is no need to change or modify the code of the quiz delivery system when the hypertextbook administrator wants to add, delete, reorder, or modify quizzes. The quiz delivery system uses two types of XML files:

1. Exactly one file (Quizzes.xml) that specifies the quizzes that are available (by chapter, section, and level)
2. Files that specifying the content of the quizzes, which exactly one XML file for each quiz

Figure 4. Appearance of the Biofilm Hypertextbook Quiz Delivery System With a Specific Quiz Loaded Initially.



Self-Check Quizzes

► Select Quiz

⊕ Chapter 1

⊕ Chapter 2

Chapter 1, Section 1, Level green

1 2 3 4 5

Biofilms require which of the following:

☐ A surface

☐ High temperatures

☐ Nutrients

☐ A host organism

☐ Moisture

Check Answer

Figure 5 shows an example of the Quizzes.xml file. It contains a <chapter> element for each chapter that contains at least one quiz. The <chapter> element has an “id” attribute, which takes an integer value (without leading zeros). Nested within each <chapter> element, are one or more <section> elements. Each <section> element has an integer “id” attribute, as well.

Nested within each <section> element are one or more <level> elements, which can have exactly one of the following values: green, blue, or black. The XML file in

Figure 5. An Example of the File (Quizzes.xml) That Specifies the Available Quizzes.

```
<?xml version="1.0" encoding="UTF-8"?>
<quizzes>
  <chapter id="1">
    <section id="1">
      <level>green</level>
      <level>blue</level>
      <level>black</level>
    </section>
    <section id="2">
      <level>green</level>
    </section>
    <section id="3">
      <level>green</level>
    </section>
  </chapter>

  <chapter id="2">
    <section id="1">
      <level>blue</level>
    </section>
  </chapter>
</quizzes>
```

Figure 5 indicates that the following quizzes are available.

Chapter 1, section 1, green level

Chapter 1, section 1, blue level

Chapter 1, section 1, black level

Chapter 1, section 2, green level

Chapter 1, section 3, green level

Chapter 2, section 1, blue level

Names for the files that specify individual quizzes take on the following format:

“XML_” [three digit chapter number] “_” [three digit section number] “_” [level] “.xml”

Below is the name of the file for the quiz for chapter 1, section 1, green level:

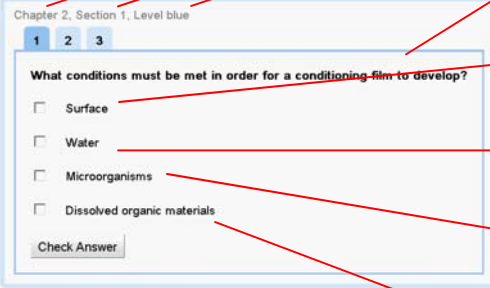

XML_001_001_green.xml


Note that the leading XML must be in upper case, while the other characters are all lower case. Also, note that chapter and section numbers must contain three digits. Thus, they may contain leading zeros.

Figure 6 shows a side-by-side view of a three-question quiz as it appears in the hypertextbook quiz application and the xml file that defines the quiz.

The top-level element in the XML file is `<quiz>`. The elements that are direct children of the `<quiz>` element include: `<type>`, `<chapter>`, `<section>`, `<level>`, and `<question>`. Exactly one of each of the first four kinds of elements appears, while one or more `<question>` elements can appear. The `<type>` element provides for possible future expansion to indicate different types of quizzes. The `<chapter>`, `<section>`, and `<level>` elements, respectively, specify the chapter, section, and level of the quiz. The `<question>` elements each specify one question in the quiz.

Figure 6 A Sample Three-Question Quiz As It Appears in the Hypertextbook Quiz Application (Left) and the Corresponding XML File (Right).

	<pre> <?xml version="1.0" encoding="UTF-8"?> <quiz> <type>standard</type> <chapter>2</chapter> <section>1</section> <level>blue</level> <question> <question_text>What conditions must be met in order for a conditioning film to develop?</question_text> <image>none</image> <choice> <correctness>true</correctness> <choice_text>Surface</choice_text> <explanation>Yes. Surface is needed</explanation> </choice> <choice> <correctness>true</correctness> <choice_text>Water</choice_text> <explanation>Yes. Water is needed</explanation> </choice> <choice> <correctness>true</correctness> <choice_text>Microorganisms</choice_text> <explanation>Yes. Microorgansims</explanation> </choice> <choice> <correctness>true</correctness> <choice_text>Dissolved organic materials</choice_text> <explanation>Yes. DOM needed</explanation> </choice> </question> </pre>
	<pre> <question> <question_text>In which of these growth curve stages (shown below) are the planktonic cells most similar to cells in a biofilm?</question_text> <image>growth_curve.jpg</image> <choice> <correctness>false</correctness> <choice_text>Lag phase</choice_text> <explanation>No. Not lag phase.</explanation> </choice> <choice> <correctness>false</correctness> <choice_text>Log or exponential phase</choice_text> <explanation>No. Not log phase.</explanation> </choice> <choice> <correctness>true</correctness> <choice_text>Stationary phase</choice_text> <explanation>Yes. Stationary.</explanation> </choice> <choice> <correctness>false</correctness> <choice_text>Death Phase</choice_text> <explanation>No. Note death.</explanation> </choice> </question> </pre>

	<pre> <question> <question_text>Which of the following strategies would you consider to have the best chance of preventing or eliminating biofilm infections in patients who have received an artificial implant such as this prosthetic hip (below).</question_text> <image>hip_replacement.jpg</image> <choice> <correctness>>false</correctness> <choice_text>Heavy metal disinfectant treatment of the implant prior to surgery.</choice_text> <explanation>No. Not heavy metals.</explanation> </choice> <choice> <correctness>>true</correctness> <choice_text>New antibiotics.</choice_text> <explanation>Yes. antibiotics.</explanation> </choice> <choice> <correctness>>true</correctness> <choice_text>Quorum sensing inhibitors that prevent attachment.</choice_text> <explanation>Yes. Inhibitors.</explanation> </choice> <choice> <correctness>>true</correctness> <choice_text>Compounds that encourage premature detachment of the biofilm.</choice_text> <explanation>Yes. Detachers.</explanation> </choice> </question> </quiz> </pre>
---	---

Each <question> element contains three types of direct child elements:

<question_text>, <image>, and <choice>. There is exactly one <question_text> element, which specifies the text of the question.

There is exactly one <image> element. The <image> element can contain the value “none,” if there is no image for the question. (Note that each character in the word “none” is lower case.) If an image is part of the question, the exact name of the image file, including the extension, appears within the <image> element. In Figure 6, the first question has no associated image, while the second and third questions do have images associated with them. Note that the hypertextbook administrator must place the actual image files in the “images” folder in the quiz application, in order for the application to load the images into the quiz.

There may be one or more <choice> elements. Each choice element represents one of the distractors for the question. Each choice element has three direct child elements: <correctness>, <choice_text>, and <explanation>. There is exactly one of each of the three child nodes within each choice element. The <correctness> element can have a value of “true” or “false,” and indicates whether the distractor is correct. The <choice_text> element contains the text for the distractor. The <explanation> element begins with the phrase “Yes.” or with the phrase “No.” “Yes” means that this distractor is correct. “No” means that the distractor is incorrect.

The quiz application uses the <correctness> and <explanation> elements to provide feedback to the user after she clicks the “Check Answer” button for each question. Figure 7, shows the results of a user who selected just the first two distractors in the first quiz question in Figure 6. Since the correct answer for the question included all of the distractors, the application marked the first two check boxes correct with a green color, and the second two check boxes incorrect with a red color. The explanation for each of the distractors also appears.

The website administrator must place the Quizzes.xml file and the xml files for the individual quizzes in the application’s xml folder in order for the application to function as expected.

Figure 7 A Sample of the Feedback Generated by the Quiz Application for the First Question in Figure 6, in Which the User Selected Only the First Two Distractors.

Chapter 2, Section 1, Level blue

1

2

3

What conditions must be met in order for a conditioning film to develop?

<input checked="" type="checkbox"/>	Surface	Yes.Surface is needed
<input checked="" type="checkbox"/>	Water	Yes. Water is needed
<input type="checkbox"/>	Microorganisms	Yes.Microorgansims
<input type="checkbox"/>	Dissolved organic materials	Yes. DOM needed

FUTURE WORK

The prototype version originally interacted with a database on the server to verify usernames and passwords, access quizzes, and store quiz results. I removed the database interaction due to time constraints and the nature of the hosting service where the biofilms hypertextbook currently resides. A future project could involve implementing the database and server-side code for the quiz delivery system.

I also began creating a web-based widget that instructors or other authorized users could use to write quizzes. The widget would verify usernames and passwords and then provide an interface for entering questions into the hypertextbook. The widget would work by converting the information the user entered into a properly formatted XML file and then storing the XML file in the quiz delivery system's database. I abandoned work on this part of the application for the same reasons I mentioned above—time constraints and the nature of the hosting service.

Another interesting extension to the project would be the creation of a secure environment in which instructors could administer for-credit quizzes. As mentioned in the Background section of this paper, for-credit quizzes pose more security issues than self-check quizzes.

REFERENCES

- [1] Cole, J. and Foster, H. 2007. Chapter 5: Quizzes in *Using Moodle: Teaching with the Popular Open Source Software*, 2nd edition. O'Reilly Community Press.
http://docs.moodle.org/en/Using_Moodle_book.
- [2] Cunningham, A.B., Lennox, J.E., and Ross, R.J. Eds. 2001-2008 *Biofilms: The Hypertextbook*. <http://www.biofilmbook.com>
- [3] Cunningham, A.B. and Ross, R.J. Project Overview Information Page. *Biofilms: The Hypertextbook*. Center for Biofilm Engineering.
http://www.erc.montana.edu/biofilmbook/PREFACE_MATL/Overview.htm
- [4] The Eclipse Foundation. <http://www.eclipse.org/>
- [5] Flanagan, D. 2006. *JavaScript: The Definitive Guide*, 5th ed. O'Reilly & Associates. [ISBN 0-596-10199-6](http://www.amazon.com/dp/0596101996).
- [6] Garret, J.J. 2005. Ajax: a new approach to web applications. *Adaptive Path*.
<http://adaptivepath.com/ideas/essays/archives/000385.php>
- [7] Google Web Toolkit. 2009. Google.com. <http://code.google.com/webtoolkit/>
- [8] Google Web Toolkit. Product Overview.
<http://code.google.com/webtoolkit/overview.html>
- [9] Rößling, G., Naps, T., Hall, M., Karavirta, V., Kerren, A., Leska, C., Moreno, A., Oechsle, R., Rodger, S., Velázquez-Iturbide, J. A. and Urquiza-Fuentes, J. Merging interactive visualizations with hypertextbooks and course management. *ACM SIGCSE Bulletin*, 38 (4), 123–138, December 2006.
- [10] Ross, R.J. and Grinder, M.T. 2002. Hypertextbooks: Animated, Active Learning, Comprehensive Teaching and Learning Resources for the Web. In *Software Visualization*, number 2269 in Lecture Notes in Computer Science, S. Diehl, ed. Springer, 269–284.
- [11] Sosnovsky, S., Shcherbinina, O., and Brusilovsky, P. Web-based parameterized questions as a tool for learning. In: Rossett, A. (ed.) Proc. of World Conference on E-Learning, E-Learn 2003, (Phoenix, AZ, USA, November 7-11, 2003), AACE, 309-316

- [12] Woit, D. and Mason, D. 2001. Effectiveness of Online Assessment, Inroads, *The SIGCSE Bulletin*, 35.
- [13] Wiktionary, the free dictionary. 2009. Biofilm Entry Page.
<http://en.wiktionary.org/wiki/biofilm>