A curriculum feedback collection tool for UC-WISE courses on the Moodle Virtual Learning Environment

Aditya Kashyap

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2011-148
http://www.eecs.berkeley.edu/Pubs/TechRpts/2011/EECS-2011-148.html

December 16, 2011

Acknowledgement

The work described in this paper was supported by the UC-WISE Project. I’d also like to acknowledge the support of Dr. Nathaniel Titterton in helping bring the feedback tool to live production as well as my Masters thesis advisor, Michael Clancy

A curriculum feedback collection tool for UC-WISE courses on the Moodle Virtual Learning Environment

Aditya Kashyap, Masters Candidate in Computer Science, Computer Science Division, EECS Department, University of California, Berkeley, CA 94720-1776 adityakashyap@berkeley.edu

ABSTRACT: In this paper we present a page-integrated tool for collecting feedback for online UC-WISE course content on the Moodle Virtual Learning Environment. First, we discuss previously employed methods that include a end-of-semester in-browser survey and in-person student surveys, each of which has significant limitations. The feedback tool's defining advantage is: 1) its availability on all course elements within an online curriculum (yielding feedback data with greater granularity); and 2) a deep integration with the web portal to deliver analytical tools to instructors. Next, we present the design goals we set for this tool and the features that were developed to realize those goals. Following this is an explanation of the entire development cycle, the structure of the code base, and subsequent improvements on each iteration. Lastly, we map out steps for improving the tool further.

I. Introduction

Engineering and science students do a great deal of their learning in laboratory settings, where they are able to experiment, observe, and evaluate through practical applications of the knowledge they are given. Recent education research indicates that students learn and absorb a great deal more by applying knowledge in a learner-centered environment than they do by passively listening to an instructor lecture for several hours a week (Collins, Brown, & Holum, 1991; Titterton and Clancy 2007). Unfortunately, this lecture-based format that still dominates present-day higher education. Having said that, there is a growing movement in tertiary education focused on promoting the active learning process with a focus on the student. These student-centered teaching strategies feature peer instruction, inquiry-based learning, and cooperative learning (Bransford, Brown, & Cocking, R.R, National Research Council (U.S.); Dillenbourg, 1999). A student-centered approach must incorporate active content that involves the student in a variety of educational activities carefully selected and skillfully matched to complement one another to enhance the student's overall grasp of a subject. Each of these activities has its own particular goals such as: exploration in a sandbox environment; answering questions with instant scripted feedback; student self-assessment and reflection of progress, and guided collaboration requiring mandatory student contribution through question gates (Carle, Canny, & Clancy 2006).

A. The UC-WISE Project

It is in this spirit that the UC-WISE project (University of California Web-based Instruction for Science and Engineering) sets out to establish a framework for a laboratory-based medium of higher education courses through an offering of curricula and technology. Such courses help facilitate online collaboration and assessment. The goal is two-fold: discovering the most effective applications of this technology into the University of California's science courses as well as providing educators with a means to modify and experiment with courses and receive feedback from other course developers (Clancy 2006).

Several UC Berkeley engineering courses have been developed into UC-WISE versions and many of these have been repeatedly taught after successful revisions on earlier iterations. Some examples include introductory undergraduate computer science courses CS3 and CS61B. The former ran from Spring 2003 to Fall 2010. The latter was first taught in Fall 2004, Fall 2005, Spring 2008, Spring 2010, and then again in Spring 2011. The Fall 2007 semester saw the pilot of a lab-based CS61C.

The utilization of JHAVÉ, a program visualization facility, is one example of how UC-WISE has incorporated new technology and resources into courses to improve the learning experience. This program can be used to piece together a sequence of images that illustrate the flow of subsequent stages of a computer program's execution, all while prompting the student with engaging questions (Clancy 2006). Another example is a program like WebJava (or BlueJ compiler's built-in tool) which allows instructors to develop exercises which take in real Java code from students and relay compile-time and run-time output of the code back to the instructor. The instructor may gain useful insights as to where confusion or misconceptions specifically lie and addresses those areas more proactively. In the case of BlueJ, instructors can hard-wire programs to return specific feedback case by case for common, anticipated student mistakes.

B. PACT and Moodle

PACT, or "Pattern-Annotated Course Tool," is a tool utilized by the UC-WISE system. Initially prototyped by Andy Carle, PACT allows for the application of pedagogical patterns to course activities, making them instances of these and also presenting the usage of these patterns in a visualized form (Carle, Canny, & Clancy 2006). Because a UC-WISE course's curriculum is comprised of smaller "steps", or learning activities, it is an ideal candidate to be annotated by pedagogical patterns, which provide excellent insights into how educators may reuse elements of a given curriculum. As existing UC-WISE curricula are annotated, pattern inconsistencies will come to light, highlighting areas for potential improvement of the curriculum.

Current UC-WISE courses employ the Moodle platform (*moodle.org*), an Open Source Course Management System (CMS) or Virtual Learning Environment (VLE). Moodle must be installed on a web server, and in the case of the UC-WISE project, it is hosted at *sage.cs.berkeley.edu*. One of the key strengths of Moodle is its flexibility to scale to an instantiation supporting hundreds of thousands of students while still being nimble enough to be used for a few dozen students in only one class.

While some institutions prefer to use Moodle simply as a means to deliver course curriculum to students, many (including UC-WISE) seek to take full advantage of its activities modules (e.g. databases, wikis, and forums) to build highly collaborative learning communities centered around a specific discipline or topic. The platform also features valuable tools such as assignments and quizzes, which instructors can use to assess the efficacy of the learning that is taking place in their online course. Moodle's greatest asset, however, is its active open-source development community, which is constantly contributing new modules and tools for educators to deploy (stock or customized) in their course builds.

C. Challenges of Feedback Analysis

The theoretical efficacy of UC-WISE's approach to education can be argued endlessly with convincing citations from education and psychology papers, but in the end the only way to truly assess and improve the educational benefits of lab-based instruction is to conduct an in-depth study that compares traditional lecture courses to their UC-WISE versions.

Formulating the approach for a rigorous data-driven evaluation of UC-WISE is only one challenge the UC-WISE team faces. What information should be collected? And how should it be collected? These same questions are also relevant to another challenge: how to constantly improve upon each course's curriculum with every semester that the course is taught. Until the development of the tool described in this paper, there were only two real means of gathering student feedback: 1) an end-of-semester survey that would then be analyzed for further areas of investigation and 2) manually surveying UC-WISE students each semester to get feedback on the usefulness of different tools and activities in facilitating their learning.

Each of the aforementioned approaches to collecting feedback had (and still has) significant limitations. The end-of-semester survey attempts to capture an entire course worth's of feedback in the last week of instructions. It was discovered that many students do not take the survey seriously as they are anxious to be done with the semester, and thus fill in facetious input and random ratings in various fields (Clancy 2011). Of those students that sincerely attempt to complete the survey in good faith, many have trouble remembering their specific frustrations or confusion with materials / exercises presented in the earlier parts of the course (Manaris 2002). Furthermore, students have a tendency to downplay or even forget initial frustrations or confusions with a specific exercise or subject matter if they grasp the concept later on in the course (Trochim 2002). While this may be fine on an individual level, it hurts the quality of analysis and subsequent course improvement that can be done. This is because some course content may be poorly explained, but it may still escape reevaluation if a good portion of students later grasped the related topic through other material or on their own. The survey results may indicate that only a minority of students had trouble with a specific topic, indicating a false positive that the problem for those students must be somewhere other than in the wording or explanation in an activity. Often only when a good-sized portion of students continue to struggle with a concept until end of term do instructors investigate curriculum wording. Also, an end-of-term survey cannot give a temporal sense of the progression of a student through the weeks of the course, other than whatever self-opinion the student may give in the survey (which can hardly be considered objective). Instructors will be hard pressed to understand when exactly a student became confused or when he or she began to finally gain grasp of a given concept. Lastly, an end-of-semester survey lends itself to higher, broad level questions about the course and overall self-evaluation of performance rather than requests for feedback on specific activities or exercises.

To address this last limitation of a lack of granularity in the end-of-term survey, graduate student instructors (GSIs) survey students during the semester to get a sense of overall feedback on different tools or activities. The main issue here is that the results are from a highly scattered sample and are far from thorough; only some students are surveyed and only on certain days about certain activities pertaining to that day. Some students who have valuable input may never get asked at all. Other students are shy, or feel conscious about giving course criticism to the instructor (in private or otherwise). Also, such surveying consumes valuable GSI time that could be spent helping students or the class as a whole. Lastly, even though some specific feedback is collected, there is no easy way to organize it or quantitatively analyze and compare specific activities. The approach for analysis here is a manual one –instructors must go through the surveys and essentially "get a feel" for which activities work and which do not. There is no efficient way of tabulating or quantifying results.

In this paper we present a Moodle feedback collection tool designed to address the shortcomings of the two aforementioned feedback surveys as well as fill in any functional gaps. The feedback collection tool's key value contribution is its focus on bringing full feedback functionality to each "atom" of a UC-WISE course, resulting in an unprecedented level of granularity.This will allow instructors to pinpoint problematic areas (as well as recognize highly effective and positively received portions) for any given course, any given day, any given activity- all down to the specific sentence if the situation calls for it. Furthermore, the feedback tool presents the data it collects in an effective tabular format integrated into the course portal so instructors can quickly and quantitatively identify areas of the curricula that demand attention.

II. Related Work

As one can imagine, there has been great variation in different approaches to analyzing the efficacy of lab-centric courses, both in terms of differing processes as well as a deployment of custom tools and methodologies to execute on those proposed analyses.

Moodle 2.1 features a rudimentary, native Feedback module that allows for the creation of custom surveys, unlike the *Survey* tool ("Feedback Module" 2011). The Feedback module supports the *template* tool, allowing users to create and re-use sets of Feedback questions. The template tool is a bit cumbersome because a new Feedback activity must be created for every element of the course that requires specific feedback. Furthermore, the template tool does not directly link or associate a course element with its appropriate Feedback activity. Because of the bare-bones nature of this module, many educational institutions have built upon this with their own custom solutions to meet their specific needs, but there have not been any contributions to Moodle's open-source communities for public distribution yet (Clancy 2011).

Researchers at the IRTC (Instructional Technology Resource Center) at Idaho State University have developed their own custom Feedback tool (IRTC, et al 2010) However, unlike the tool in this paper, it is implemented as an *activity* that instructors can use to create a custom survey for their students. It features a timing option which allows the instructor to limit the accessibility of the activity to a given time period. The tool has the option to record user names, share survey results with students, send e-mail notification, allow for resubmits to specific answers, as well as a "group mode", which supports feedback for group activity. "For example, if the Feedback tool is used to gather opinions from a small group activity, using those same groups will allow you to relate responses to the particular group dynamics. For a whole-class or individual activity, such as thoughts about a guest speaker, "no groups" should be selected" (IRTC 2010). The tool also features different question types, instantiated as Moodle *blocks*, templates for different types of questions that need to be posed (included Captcha, Label, Long Text Answer, Multiple Choice, Numeric Answer, and Short Text Answer). The tool simply dumps all results into a raw text file, however, and does not support any visualization or tables to sort through or easily access the data. There is also no option to link a question to a specific activity within the course; each activity will need its own feedback tool activity to appear directly after any content that instructors desire feedback on.

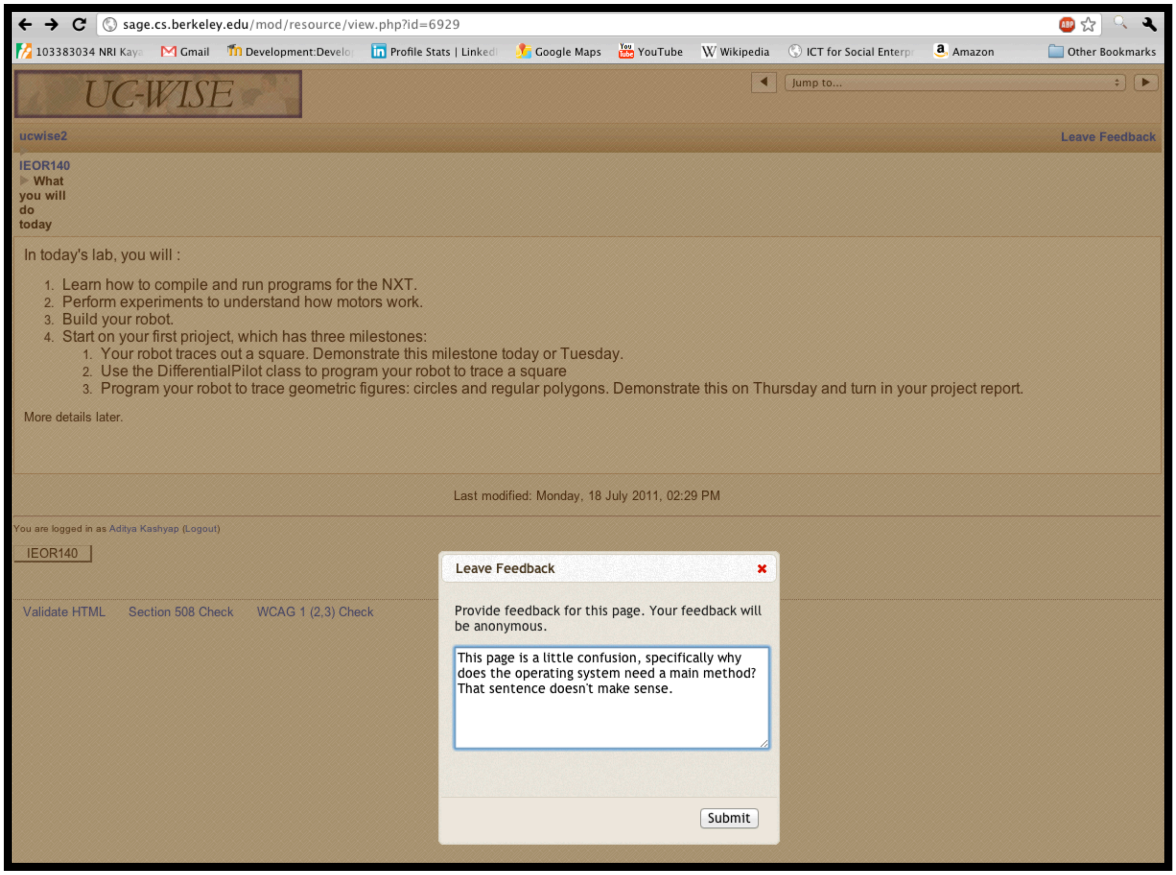
III. Design Goals and Features

The UC-WISE2 feedback tool was designed first and foremost to yield better curriculum revisions through better analysis. Better analysis comes from improving data and feedback collection, and that was the primary goal sought with the development of this tool. The design goals revolved around addressing the shortcomings of the original two methods of feedback collection: end-of-semester online surveys and GSI-student interviews. We will now discuss these design goals and their corresponding features:

- **Capture highly specific curriculum feedback, as granular as possible on any and all course elements, however atomic**

A. Universal Interface

The Feedback Tool allows students to submit feedback on any course element. As long as the student is on a course-related page, the "Leave Feedback" link at the top-right of the screen in the header never disappears. Because of the integration of the feedback tool into the header, the student is assured he or she will be able to leave feedback for *whatever* course element is desired. The tool is programmed with logic to automatically associate any feedback submitted from a given page with the course element that is present on that page. Not only does this allow for a clean, automated way to manage feedback data in curriculum-relevant fashion, but it also adds a deeper layer of specificity to the feedback submitted (as compared to the end-of-semester / GSI surveys) as a student can single out a specific activity or even a sentence on a specific page that is particularly confusing. In the prior survey methods feedback on specific elements was rare because students would not remember the particular element / problem with the element while giving their survey answers. With the feedback tool, the student can submit feedback on an element instantly and element-specific feedback will finally be collected regularly.



In today's lab, you will :

1. Learn how to compile and run programs for the NXT.
2. Perform experiments to understand how motors work.
3. Build your robot.
4. Start on your first prioject, which has three milestones:
    1. Your robot traces out a square. Demonstrate this milestone today or Tuesday.
    2. Use the DifferentialPilot class to program your robot to trace a square
    3. Program your robot to trace geometric figures: circles and regular polygons. Demonstrate this on Thursday and turn in your project report.

More details later.

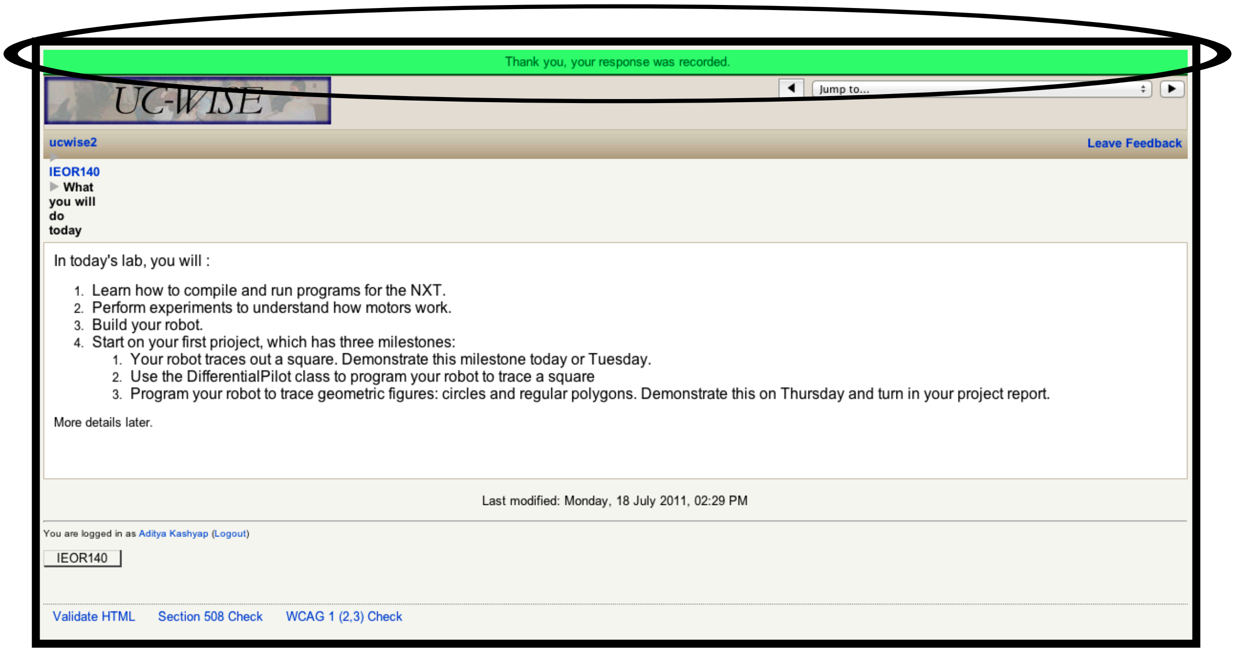
Last modified: Monday, 18 July 2011, 02:29 PM

You are logged in as Aditya Kashyap (Logout)

Provide feedback for this page. Your feedback will be anonymous.

This page is a little confusion, specifically why does the operating system need a main method? That sentence doesn't make sense.

Figure 1. Student View: Leave Course Element Feedback Dialog


Thank you, your response was recorded.

In today's lab, you will :

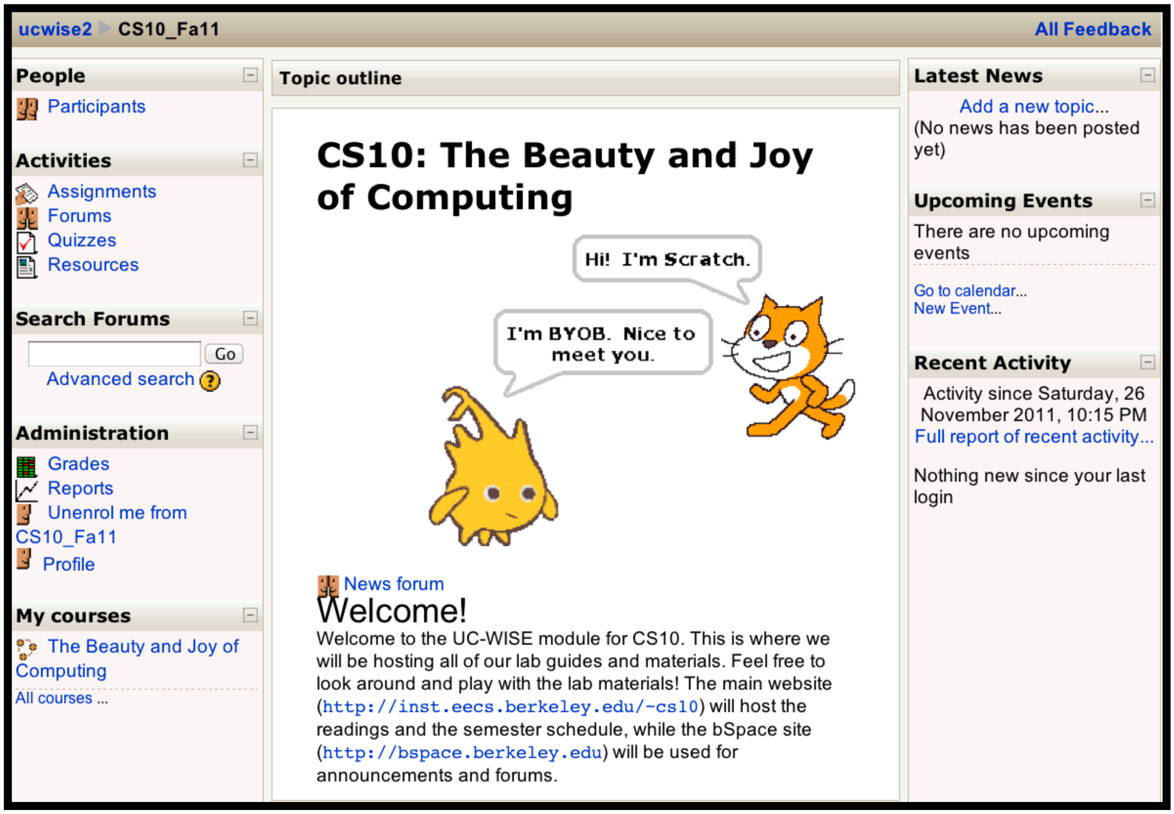
1. Learn how to compile and run programs for the NXT.
2. Perform experiments to understand how motors work.
3. Build your robot.
4. Start on your first prioject, which has three milestones:
    1. Your robot traces out a square. Demonstrate this milestone today or Tuesday.
    2. Use the DifferentialPilot class to program your robot to trace a square
    3. Program your robot to trace geometric figures: circles and regular polygons. Demonstrate this on Thursday and turn in your project report.

More details later.

Last modified: Monday, 18 July 2011, 02:29 PM

You are logged in as Aditya Kashyap (Logout)

Figure 2. Student View: Feedback Submission Confirmation


ucwise2 ▸ CS10_Fa11

All Feedback

People

Participants

Activities

Assignments
Forums
Quizzes
Resources

Search Forums

Go

Advanced search

Administration

Grades
Reports
Unenrol me from CS10_Fa11
Profile

My courses

The Beauty and Joy of Computing

All courses ...

Topic outline

CS10: The Beauty and Joy of Computing

News forum

Welcome!

Welcome to the UC-WISE module for CS10. This is where we will be hosting all of our lab guides and materials. Feel free to look around and play with the lab materials! The main website (`http://inst.eecs.berkeley.edu/~cs10`) will host the readings and the semester schedule, while the bSpace site (`http://bspace.berkeley.edu`) will be used for announcements and forums.

Latest News

Add a new topic...
(No news has been posted yet)

Upcoming Events

There are no upcoming events

Go to calendar...
New Event...

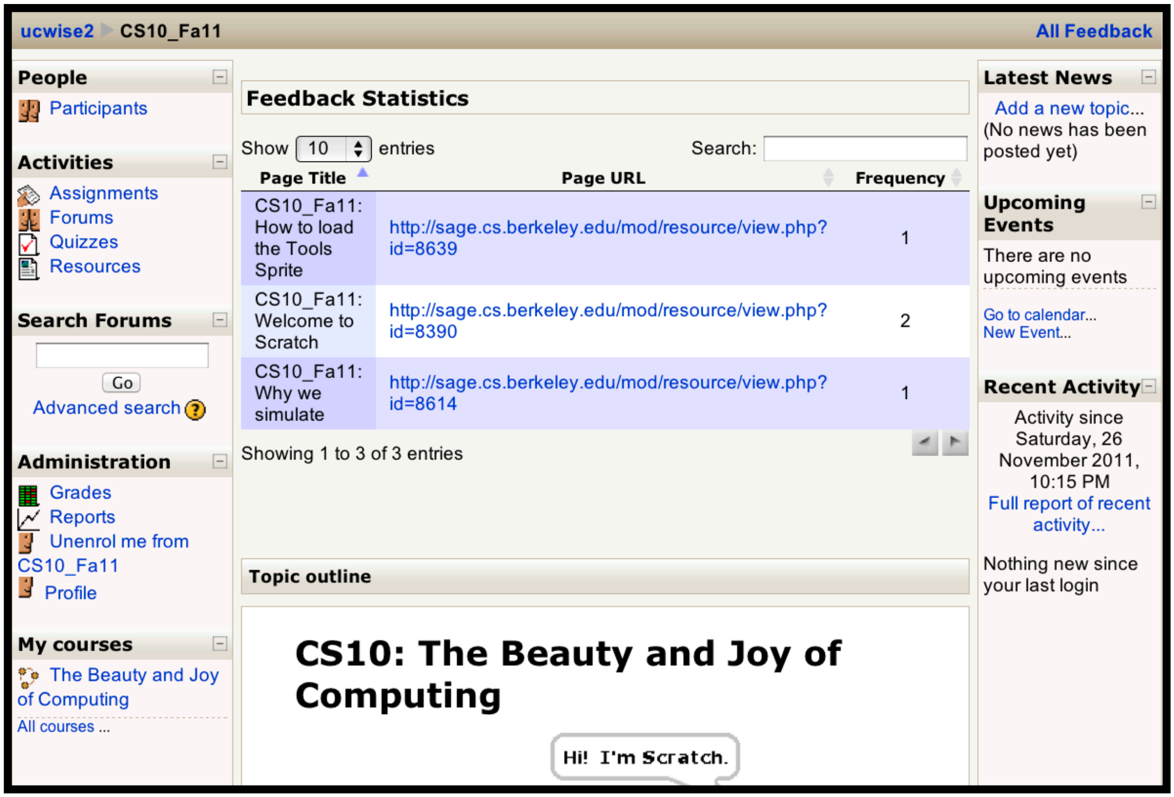
Recent Activity

Activity since Saturday, 26 November 2011, 10:15 PM

Full report of recent activity...

Nothing new since your last login

Figure 3. Teacher View: Course Homepage (View All Feedback)


ucwise2 ▸ CS10_Fa11

All Feedback

People

Participants

Activities

Assignments
Forums
Quizzes
Resources

Search Forums

Go

Advanced search

Administration

Grades
Reports
Unenrol me from CS10_Fa11
Profile

My courses

The Beauty and Joy of Computing

All courses ...

Feedback Statistics

Show 10 entries

Search:

Page Title	Page URL	Frequency
CS10_Fa11: How to load the Tools Sprite	http://sage.cs.berkeley.edu/mod/resource/view.php?id=8639	1
CS10_Fa11: Welcome to Scratch	http://sage.cs.berkeley.edu/mod/resource/view.php?id=8390	2
CS10_Fa11: Why we simulate	http://sage.cs.berkeley.edu/mod/resource/view.php?id=8614	1

Showing 1 to 3 of 3 entries

Topic outline

CS10: The Beauty and Joy of Computing

Latest News

Add a new topic...
(No news has been posted yet)

Upcoming Events

There are no upcoming events

Go to calendar...
New Event...

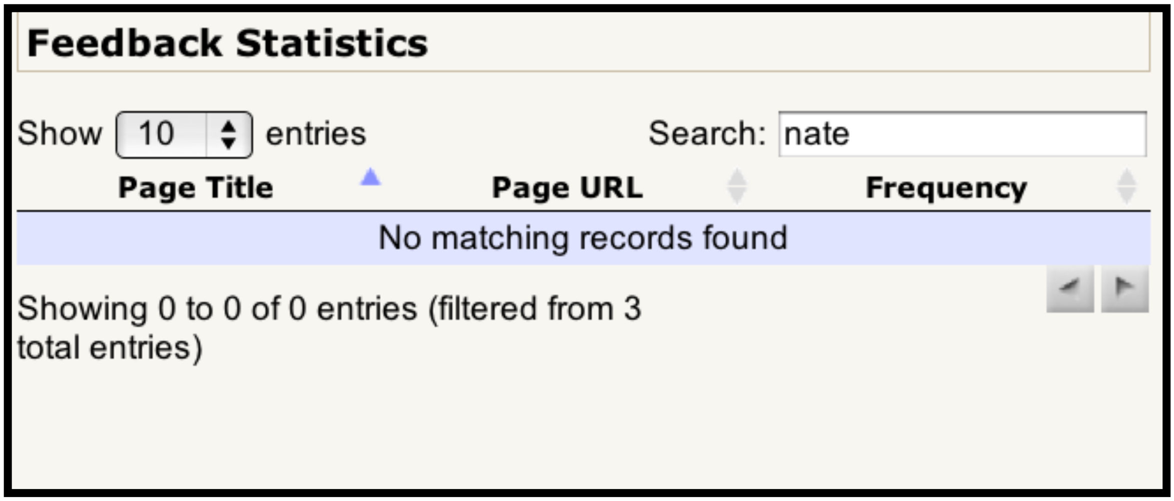
Recent Activity

Activity since Saturday, 26 November 2011, 10:15 PM

Full report of recent activity...

Nothing new since your last login

Figure 4. Teacher View: Feedback Statistics Tool



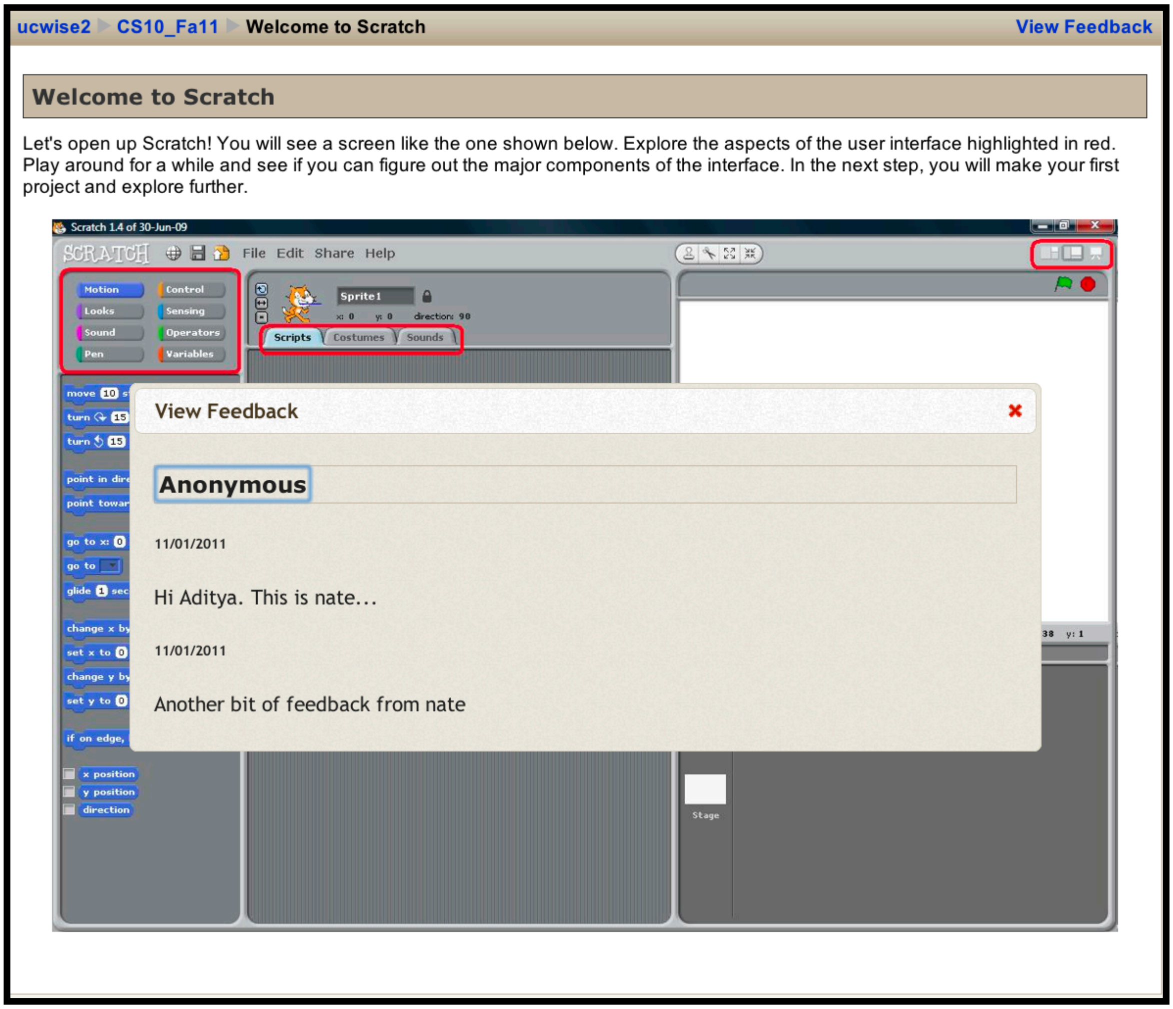
Feedback Statistics

Show 10 entries Search: nate

Page Title	Page URL	Frequency
No matching records found		

Showing 0 to 0 of 0 entries (filtered from 3 total entries)

Teacher View: Feedback Statistics Tool – Search Functionality



Welcome to Scratch

Let's open up Scratch! You will see a screen like the one shown below. Explore the aspects of the user interface highlighted in red. Play around for a while and see if you can figure out the major components of the interface. In the next step, you will make your first project and explore further.

View Feedback

Anonymous

11/01/2011

Hi Aditya. This is nate...

11/01/2011

Another bit of feedback from nate

Figure 5. Teacher View: View Page Feedback

B. Automatic Tagging of Feedback to Course Elements in Associate Table

Even if a student did go back several weeks in the curriculum just to dig up the problematic course element, the end-of-semester survey has no way of automatically associating that piece of feedback with the problematic course element. The instructor could either "eyeball" the data to get a feel for problematic areas of the curriculum (which is not very precise or rigorous), or would have to manually tag each feedback note with the associated course element. The Feedback Tool eliminates these concerns by automatically storing every piece of student feedback with its associated course element in the database, with no input effort on part of the user.

- **Identify the exact time and source of student confusion**

C. Timestamp and Source Page Data Collection

As discussed in-depth earlier, one of the major deficiencies of a survey approach is that students often forget / are not motivated to give feedback on course elements that gave them problems in earlier portions of the course. In addition, students have a tendency to downplay or overlook challenges they may have initially faced with a piece of the curriculum if they have since overcome the confusion and gained a command of the concept. This can lead to an underreporting of problematic course material, but fortunately the Feedback Tool addresses both of these problems; all user feedback is logged for time and page source of origin. This information is invaluable as instructors can analyze the data to observe trends and spikes of confusion in as the course timeline as well as observe if and when a student or group of students overcomes an initial confusion (and possibly discern the source of the confusion reversal). This is also logged in a SQL table.

- **Improve veracity of feedback**

D. Voluntary Feedback

The end-of-semester survey which is given has been given as a mandatory daily activity/ assignment for many UC-WISE courses in the past lends itself to disingenuous feedback; many students are annoyed by being required to fill out such surveys and subsequently fill in bogus and inaccurate feedback to quickly be done with the task. Submitting feedback with the Feedback Tool is a voluntary option, thus instructors are much more likely to get genuine feedback from students that truly care.

- **Encourage greater rates of voluntary feedback submission from students**

E. Anonymize Feedback Option

Voluntary feedback rates increase when students have the option of anonymizing their submissions (O'Malley 2000). There will always be some students who are too shy to directly give feedback to their GSIs or course instructors for one reason or another, and the Feedback Tool presents an outlet for this group in a level of anonymity that was not previous available. GSI interviews with students are face-to-face, and thus effectively cannot be anonymized, aside from sending anonymous emails or leaving anonymous notes. The end-of-the-semester survey is only pseudo-anonymous, and a student's identity can effectively be traced back through the server backend. The code for the Feedback Tool, on the other hand, manually wipes all user-identifying data from its database table immediately upon feedback submission, including "student_id", "login", and "IP" fields, replacing it with a string of 0 bits. Feedback is currently anonymous by default, but it can easily be changed by deleting one line of code that specifies the 0-bit overwrite. Currently, a user cannot turn the anonymize option on or off; it must be applied course-wide on the server. The student is explicitly notified that all feedback will be anonymous when they Feedback popup appears upon clicking "Leave Feedback."

F. Feedback Submission Confirmation

The issue with submitted anonymous physical or electronic mail to GSIs is that often the student receives no confirmation that his or her feedback was received. After a student clicks 'Submit' in the Feedback Tool popup and the JSON handler successfully posts a feedback form to the server, a JQuery animation drops down from the top of the page view with a message reading "Thank you, your response was recorded."

G. Web 2.0 UI: Move-able, Resize-able Lightbox Popup

A basic principle in consumer web is that if one wants to encourage a specific customer action on the website, the experience of completing that action must be made as frictionless, intuitive, and enjoyable as possible. The Feedback Tool makes feedback submission fast, easy, and fun. Clicking 'Leave Feedback' in the page navigation header brings up a Feedback Input popup. The pop-up is a lightbox, a custom jQuery object that grays out the rest of the screen upon appearing. This grabs the attention of the student and ensures there is no confusion on whether the Feedback Tool has been selected. All other web page elements become inactive until the student clicks 'Cancel' or 'Submit', eliminating any chances for distraction or confusion.

The Feedback popup itself can be positioned anywhere within the browser window, which is often critical if a student needs to refer to specific material on the course page while giving feedback. All too often, websites make the mistake of not making their popups adjustable, which creates unnecessary user pain if the popup appears on top of feedback-related page content. For the same reason, the Feedback popup is also resizable; users can make it as large or small as they wish depending on the length of their text input.

- **Provide instructors with a useful framework to view collected feedback and tools to easily identify course areas that demand attention**

The Feedback Tool has logic to determine whether the current user that is logged in is an instructor or a student. If a student is logged in, he or she will only see a 'Leave Feedback' option on course activity pages and that will be the only functionality provided. If an instructor is logged into the course portal, the Feedback Tool detects this and a 'View Feedback' option appears instead on *all* course pages.

H. 'All Feedback' View and Feedback Statistics Tool

Instructors need an easy way to view all feedback for a given course and sift through the data in a meaningful way. The Feedback tool provides this functionality through the course homepage. When a instructor visits the homepage, instead of the usual 'View Feedback' option in the header, an 'All Feedback' link will appear. Clicking this brings up the Feedback Statistics tool, a sortable table that displays all feedback for the course, organized by source element. The tool shows Page Title, Page URL, and Frequency (total number) of feedback entries for each respective element. Clicking on the Page Title or Page URL immediately takes the user to the respective page in the course. All of these columns are sortable, which is especially useful for the Frequency column –allowing an instructor to quickly discern which course activities / pages are receiving the most feedback and quickly address those first. There is functionality for additional columns to be added later.

Instructors may also search and filter through all feedback entries using the integrated search bar functionality, which employs AJAX so that all search / filter results can be displayed without refreshing the page. Lastly, instructors can set the number of entries to be displayed in the table according to their liking.

I. Page-Specific Feedback View

The page-specific feedback view is similar to the student view on any course feedback page, except that the 'Leave Feedback' link the header is now replaced with the 'View Feedback' link. This provides for a fast and intuitive way for instructors to quickly see the feedback for a course element as they are perusing through a course day's activities. The Feedback tool directly associates all feedback with its subject page automatically. With the student survey approaches of the past, instructors could never be sure which specific course element a student was referring to unless the student posted a link or word-for-word title to the activity in the feedback that was submitted. Even if the name was specified, the instructor would have to go back through the course's table of contents and find the course page. With the

Feedback tool, automatic feedback and page linking makes feedback analysis painless.

IV. Developing the Feedback Tool

A. Overview

In a vanilla module, it is similar to a system like Ruby on Rails, where migrations are possible. One can setup a module through a standardized procedure and then upgrade it incrementally. This is done though PHP files, where the developer specifies version number, what the update obsoletes, etc. For the purposes of my feedback tool, I was interested in setting up a database where the feedback that is collected may be recorded. To implement this tool as a module, first I needed to specify setup information, such as version numbers, what it provides, and also the database schema, which is in Excel format.

Originally, the tool was implemented as a module, and the database was created as a result of Moodle automatically importing the module which actually provided it. The problem was that there were some side effects as a result of implementing the tool as a module. Undesired HTML and CSS artifacts appeared on the home page of courses. A Moodle module is not designed for the purposes that I was trying to achieve – it's meant to be delivered as a useful add-on, not as a side feature, or a more transparent feature as I wanted. To remedy this, in the next revision I integrated the tool's functionality more into the Moodle core directly.

The desire was to have a feedback module that appears on Moodle course pages that display curriculum. There are two main parts to such pages: the navigation bar with the 'Leave Feedback' link in the header and the actual course content below on which feedback will be left. Determining what type of pages the feedback option is available for is determined on the server-side. When a page is being rendered, there is some code on the server that determines what type of page it is, and whether or not it should render the feedback option.

A link is inserted into the default header code that generates the header, and it is placed on the site that is visible to the user. The PHP also writew some Javascript onto the page itself that allows for the page to submit information back to the PHP module. Essentially, the PHP will render the page with the feedback link visible and also embed some Javascript. When a user clicks on that link, the Jquery (a Javascript library) UI is invoked, and the dialog (popup lightbox that grays out the rest of the page) appears. After a user fills in feedback and clicks 'Submit', a HTTP POST request gets fired back to the server. There is a central place where all requests are received (ManageFeedback file). Given a received message, this central sorter then delegates the message using the appropriate logic. The POST request has several fields, and one of the fields dictates whether it is adding a message or retrieving a message. If it is adding a message, the central sorter sends the request to a function which will create a new database entry that will appropriately map the feedback to a student ID, course content page ID, etc. For our purposes we wanted feedback to remain anonymous, so the student ID is automatically written as a '0'. If desired in the future, however, the database scheme does provide StudentID as a field in the database. Along with this, it stores the feedback message, the page link, and other information to put the feedback in context and make useful information out of it.

There are three basic modes of utilizing the feedback tool. As a student, a user can leave feedback, and that is all he or she can do. As a teacher, one can view all feedback of a given page, or view aggregate statistics for all feedback for a given course. Viewing an individual page's feedback works as follows: if the user makes a request that they want to see all feedback, the central feedback handler (ManageFeedback) will check to see if the user is an instructor first. Once it validates the user is an instructor, it looks in the database for all entries that have the same CourseID as the course that instructor is currently viewing and returns all that data back. All this communication is being done in JSON encapsulation. Then, it will dump all the feedback back as a JSON data structure and then the code parses through all that and renders it to a jQuery UI widget, where the instructor can see a list of feedback to all the students.

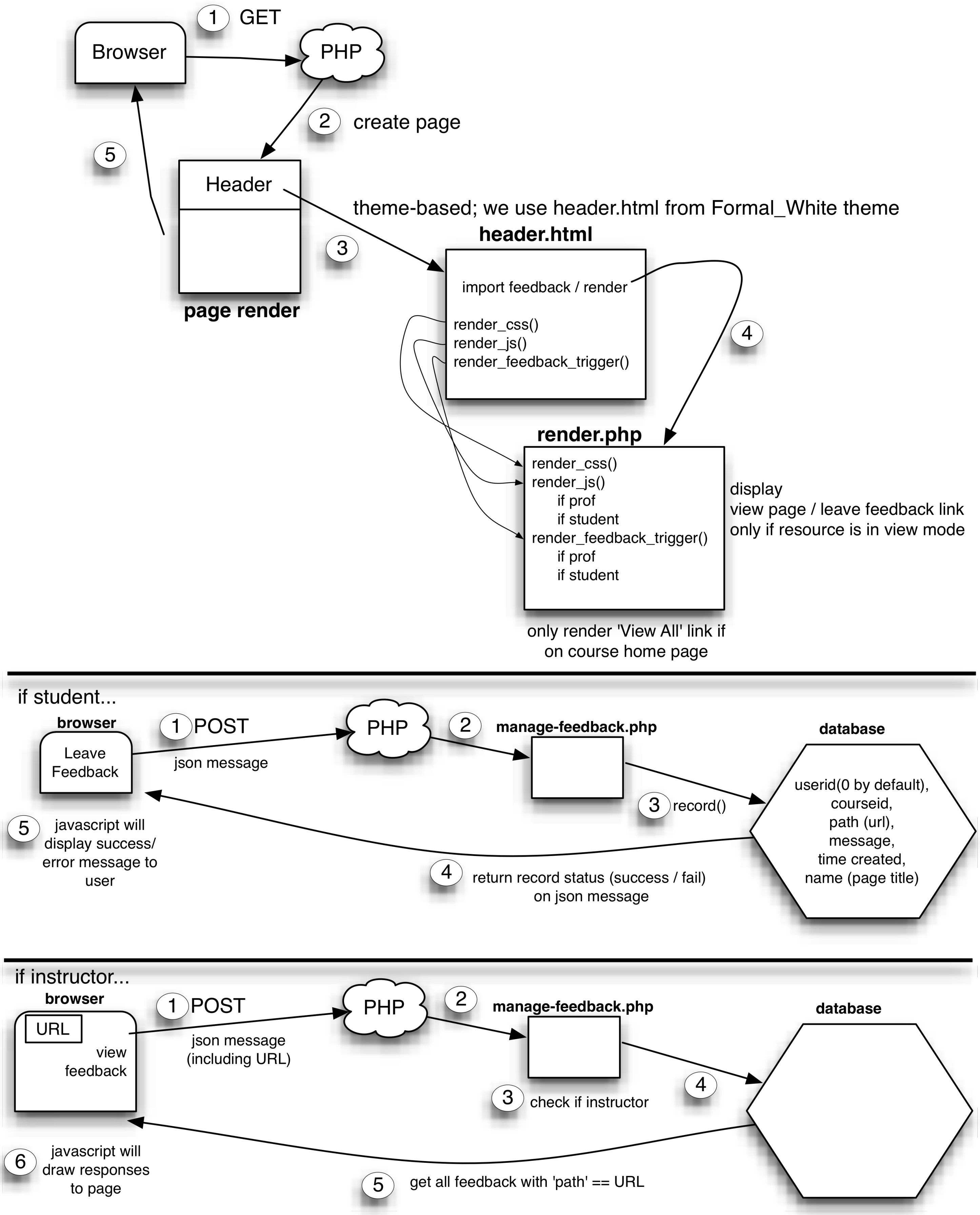


Figure 6. Feedback Tool Work Flow

The final mode of interaction involves the table view when an instructor views all feedback for a course. The table view is another Javascript library. For the methods mentioned previously, the URL is a unique determinant of the resource that the user is currently viewing. When viewing all feedback for a given page, the code looks for all feedback associated with that particular URL. In the course level view of *all feedback for a course* (the TableView), the code looks for all database rows corresponding to the course ID instead. All this information is also received back in a JSON format and its aggregated within a jQuery table library that I used.

B. Iterations

The initial implementation was a module. On the Moodle forums, there is a template that shows how to start off a module ("Moodle Module Template" 2011. This was the vanilla module that I began to modify. To start off, I set up the module so it would create the database when it was imported. I also set up the central feedback point of communication (ManageFeedback) that was mentioned earlier.

The module implementation was producing undesired HTML artifacts on course pages. In order to fix this, I realized I could no longer use a module and instead shifted the code into the core library folder ('lib'). Now, the tool was not a module, and instead it was integrated into the core Moodle itself. The problem with this approach was that one no longer has the convenient feature of getting a database automatically setup when the module is imported (since there was no longer a module). Because of this, I had to manually create a table in the database. I exported the table schema from the previous database created by the module and using that I could recreate the table whenever I wished. The rest of the code remained the same in this first revision. I could still use the older module helper functions to communicate with the database.

The second revision consisted of feature additions, styling, and some code re-factoring. In the first revision, there were some core Moodle files I was modifying to provide support for the tool. The UC-WISE team suggested that it was not ideal to modify core Moodle files as it made installing the tool into new courses cumbersome (with several core files needing modification with manual code inserts in specific locations). This approach also reduced the tool's portability for future release to the Moodle community. The goal was to make this tool a plug-and-play solution. In an effort to make the tool as decoupled as possible, I factored out all of the changes in the core files and incorporated them into separate files. I managed to splice all together by modifying the styling Theme files we were using so that they would call out to these additions that I made. Now, all the user has to do to install this tool is: 1) add the tool's new stand-alone functionality files, which does not overwrite any existing files and 2) update a Theme file (use a custom theme that which was not responsible for incorporating all the elements). In the previous version, the tool's code was more centralized and was unnecessarily being loaded whether it was needed or not. For example, the Javascript was incorporated as a part of 'javascript.php', which was the core Javascript file for *all* of Moodle's operations. In the newly revised way, Javascript is inserted only-as needed on the client-side rather than being loaded regardless of need on every single page.

V. Future Work

The eventual goal is to release the Feedback Tool to the Moodle community so others can use it, but first, this will require rewriting the tool as a block. Currently, the Feedback Tool is implemented as a Moodle *module* as opposed to a *block*. As a module, the Feedback Tool is a customized solution for the UC-WISE project and works adequately, but it requires manual modification of several files, including header.html and navigation theme files. Because every course, even within UC-WISE, has its own theme, code must be manually added to each course's specific theme file to include the Feedback Tool. This is not scalable for open-source release.

A block is unlike a module in that it appears on the left or side vertical navigation bar. The advantage of a block is that it does not require any custom modification of files because then the Feedback Tool would not sit in the page navigation header. Moodle community members would simply have to add one line of code that adds the 'Feedback Tool' block into their build. The current issue blocking this rewrite is that for now, the UC-WISE courses are hosted on the Sage server, which uses Moodle 1.9. In Moodle 1.9, blocks only appear on course homepages but not on specific course activity pages. Thus, it would not be possible to display the Feedback Tool on specific course activity pages –only on the course home page, which would be useless. Sage needs to be upgraded to Moodle 2.0, which allows for blocks to be displayed on course resources.

Also, there is always room for added functionality. Some of the features desired for future versions include a visualization tool for feedback statistics (similar to the graphical style of PACT), and additional cohorts for the Feedback Statistics table such as "date and time of last feedback" and "average collective grade of all feedback submitters for a specific course element".

It would also be useful to add tagging functionality to the tool so that course pages could be identified by key concepts or common keywords frequently mentioned in student feedback. A 'hottest keywords' trend box could visually display the most common feedback topics. The lowest-hanging fruit is adding the ability for students to view all the feedback they have submitted for a course as well as for a specific page. It may also be interesting to explore if the UC-WISE team would want to make *all* course feedback available not only to instructors but to students as well. Lastly, integration of the tool with PACT could prove to be a powerful resource, if implemented correctly.

The Feedback Tool went live in late October of 2011 to two UC-WISE courses: CS 10 and IEOR 140. Because the tool was launched so late in the semester, very few students are aware of it and there has been a relatively low volume of feedback. Emails were sent out to the GSIs of the two courses, urging them to highlight the new tool in their next class session, but the effects were hardly felt. In the future, it will be critical to reach out to professors and GSIs prior to the beginning of the term and ensure that they are both educated in and willing to discuss the tool during the first week of laboratory instruction.

VI. Evaluation Plan

Had we been able to release the tool according to the original schedule, I would have evaluated it through several methods.

First, I would have compared the percentage of unique students in a course who contributed feedback at least once by using the feedback tool with the percentage of *genuine* survey responses that we would traditional end-of-the-semester course survey. Perhaps the tool motivated a student to submit genuine feedback who otherwise who would have not. Comparing these rates will help shed light on the matter.

Second, I would specifically focus on analyzing the feedback tool's ability to help flag problematic course content on a broader, class level. Going back to previous semesters of a course, I would identify topics or concepts that students had the greatest issues with on the exams as a whole. If current students were now using the feedback tool to raise attention to course content related to those topics, then I would know there is some value in the tool. Then, if instructors took action on those course items and rephrases / redesigned them and the students performed better on exam problems related to those topics (again, as an aggregate), then I would know that they clearly had an advantage over previous years' students: the ability to call the instructor's attention to course elements that need to be better explained.

Third, it may be valuable to explore if there is any correlation between the amounts of feedback students submit in a class to their performance in the class. One hypothesis is that the feedback tool increases student engagement in the curriculum as they feel it is easier to voice their frustration / confusion, and thus they are more likely to revisit old course material that they expect or hope to be updated due to their feedback.

Lastly, with the permission of students who choose to submit feedback non-anonymously, I could track the feedback they left on specific course elements on a timeline. On the same timeline I would note any instructor or GSI action taken to address those issues, either directly through interaction with the student or indirectly by fixing or rewriting confusing course content. This data could indicate how effective the tool is in addressing specific needs of students by looking at response rates from instructors. Furthermore, the instructors could look at the students' tests and discern whether or not problematic areas raised in the feedback for properly addressed depending on their performance on certain test problems.

VII. Conclusions

Although end-of-semester and GSI surveys serve as good starting points for feedback collection, in order to perform a robust analysis that informs instructors on curriculum improvement, the feedback data collected must be more granular, trustworthy, and abundant than what these processes yield. Furthermore, analysis of this data is limited due to lack of specificity, concerns of anonymity, and the manual sifting required to organize and present the data in meaningful ways. In response to these shortcomings, we created the Feedback Tool, a Moodle module which automates and truly anonymizes the feedback submission processes, all with an unprecedented level of granularity so that both students and instructors can pinpoint and address very specific problematic areas within the course curriculum. We presented the reasoning behind the design decision for the tools, the product features that we employed to bring those design goals to fruition, and the underlying technology and implementation effort in building the tool. In our opinion the Feedback Tool shows great promise in yielding rich data from which curriculum best practices may be discerned to improve course curriculum not only more rapidly, but more effectively as well.

Acknowledgements
The work described in this paper was supported by the UC-WISE Project. I'd also like to acknowledge the support of Dr. Nathaniel Titterton in helping bring the feedback tool to live production as well as my Masters thesis advisor, Michael Clancy.

References

Bransford, J., Brown, A. L., & Cocking, R.R, National Research Council (U.S.). *Committee on Developments in the Science of Learning*

Carle, A., Canny, J., & Clancy, M., (2006). PACT: A pattern-annotated course tool. *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, 2006 (1), 2054-2060

Clancy, Michael. "The UC-WISE Project." *Computer Science Division | EECS at UC Berkeley*. Department of Computer Science, UC Berkeley, 30 Aug. 2006. Web. 14 Nov. 2011. <http://www.cs.berkeley.edu/~clancy/web/ucwise.html>.

Clancy, M., Titterton, N., Ryan, C., Slotta, J., & Linn, M. (2003). *New roles for students, instructors, and computers in a lab-based introductory programming course. SIGCSE '03: Proceedings of the 34th SIGCSE technical symposium on Computer science education*, Reno, Navada, USA, 132-136.

Collins, A., Brown, J. S., & Holum, A. (1991). Cognitive apprenticeship: Making thinking visible. *American Educator*, 15, 6-46.

Dillenbourg, P. (1999). *Collaborative Learning: Cognitive and Computational Approaches* Oxford, UK: Elsevier Science Ltd.

"Feedback Module." *Moodle Docs*. Moodle Community, 30 Sept. 2011. Web. Autumn 2011.<http://docs.moodle.org/21/en/Feedback_module>.

IRTC Guide: Using the Feedback Tool in Moodle. Issue brief. IRTC, Idaho State University, 22 Mar. 2010. Web. 10 Nov. 2011.<http://www.isu.edu/itrc/resources/feedbacktool.pdf>

Manaris, Bill and McCauley, Renée.(2002) "Computer Science Education at the Start of
the 21st Century - A Survey of Accredited Programs," in *The Proceedings of the 32th ASEE/IEEE Frontiers in Education Conference* (FIE'2002), Boston, November 2002, F2G10 - F2G15

McCauley, R., Pharr, W., W., Pothering, G., & Starr, C. (2004). "A Proposal to Evaluate the Effectiveness of Closed Laboratories in the Computer Science Curriculum," *Journal* *of Computer Science in Colleges,* 19 (3), 191-198.

"Moodle Module Template." *Moodle Forums*. Moodle Community, 26 Nov. 2011. Web. Autumn 2011. <http://download.moodle.org/download.php/plugins/mod/NEWMODULE.zip>.

O'Malley, Patrick M., Lloyd D. Johnston, Jerald G. Bachman, and John Schulenberg. "A Comparison of Confidential versus Anonymous Survey Procedures: Effects on Reporting of Drug Use and Related Attitudes and Beliefs in a National Study of Students." *Journal of Drug Issues* 30.1 (2000): 35-54. Print.

Titterton, N. and M.J. Clancy. “Adding some lab time is good, adding more must be better: The benefits and barriers to labcentric courses.” *International Conference on Frontiers in Education: Computer Science & Computer Engineering* (FECS 2007). 2007. Athens, GA: CSREA Press. p. 363-367.

Trochim, William M. The Research Methods Knowledge Base, 2nd Edition. Internet. 2002. <http://trochim.human.cornell.edu/kb/index.htm>