Future Internet **2015**, *7*, 484-499; doi:10.3390/fi7040484

OPEN ACCESS

future internet

ISSN 1999-5903

www.mdpi.com/journal/futureinternet

Article

Improving Teacher Effectiveness: Designing Better Assessment Tools in Learning Management Systems

Dov Kruger [1,*], Sarah Inman [2], Zhiyu Ding [1], Yijin Kang [1], Poornima Kuna [1], Yujie Liu [1], Xiakun Lu [1], Stephen Oro [1] and Yingzhu Wang [1]

[1] Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ 07030, USA; E-Mails: tim.zy.ding@gmail.com (Z.D.); yijinkang@gmail.com (Y.K.); poornimakuna@gmail.com (P.K.); yliu104@stevens.edu (Y.L.); luxiakun@gmail.com (X.L.); stephenjoro@gmail.com (S.O.); banboonwang@gmail.com (Y.W.)

[2] Instructional Technology, Stevens Institute of Technology, Hoboken, NJ 07030, USA; E-Mail: sinman@stevens.edu

* Author to whom correspondence should be addressed; E-Mail: dov.kruger@stevens.edu; Tel.: +1-201-798-1590.

Academic Editor: Liz Bacon

Received: 1 September 2015 / Accepted: 1 December 2015 / Published: 18 December 2015

Abstract: Current-generation assessment tools used in K-12 and post-secondary education are limited in the type of questions they support; this limitation makes it difficult for instructors to navigate their assessment engines. Furthermore, the question types tend to score low on Bloom's Taxonomy. Dedicated learning management systems (LMS) such as Blackboard, Moodle and Canvas are somewhat better than informal tools as they offer more question types and some randomization. Still, question types in all the major LMS assessment engines are limited. Additionally, LMSs place a heavy burden on teachers to generate online assessments. In this study we analyzed the top three LMS providers to identify inefficiencies. These inefficiencies in LMS design, point us to ways to ask better questions. Our findings show that teachers have not adopted current tools because they do not offer definitive improvements in productivity. Therefore, we developed LiquiZ, a design for a next-generation assessment engine that reduces user effort and provides more advanced question types that allow teachers to ask questions that can currently only be asked in one-on-one demonstration. The initial LiquiZ project is targeted toward STEM subjects, so the question types are particularly advantageous in math or science subjects.

Keywords: Learning Management Systems; web-based Assessment; LMS; CMS; HOTS; Bloom's taxonomy; STEM

1. Introduction

Over the last decade, LMSs have made huge gains in the classroom, with educational institutions leading the LMS market at 21% [1]. Furthermore, usage is predicted to grow by about 23% in the next few years [2]. Despite this progress, reported use of assessments is far lower; according to user research, 26% of people are dissatisfied or very dissatisfied with their current LMS [1]. Given that an LMS is designed to augment teaching and learning, why is the use of the assessment engine so low?

The advent of LMSs has made it possible for teachers to be more effective with a higher student-teacher ratio by opening new channels of communication. This is, in part, due to automating time-consuming tasks such as grading, question randomization, and the iterative cycle of retesting for mastery. Unfortunately, current generation learning management systems are cumbersome and unwieldy. While they increase productivity by automating the above functions, they increase effort in other areas.

This paper consists of three main sections: First, we analyze current systems and identify problems in the current design. Second, we explore how these shortcomings can be overcome. Third, we show some of the new question types under development and explore how to ask fewer questions while measuring the desired metric more accurately.

This paper describes the limitations of current LMS assessment systems, and describes some of the design for LiquiZ, a next-generation assessment engine. This paper attempts to answer the following:

- What are some the design limitations in the current LMS options?
- How can learning technology support a metacognitive approach to assessment design?
- How can we reduce the effort of deploying assessment through the online assessment system so that it as advantageous to teachers?

2. Literature Review

2.1. Metacognition and Learning

Much of good instruction comes from blending the technology we have available with the learning environment—including changing the culture of student engagement from a traditional lecture style class to an interactive, reflective experience. There are a number of tools to facilitate this type of interaction, but it can be daunting for faculty to implement. Flavell [3] defines the term, metacognition, as the knowledge about and regulation of one's cognitive activities in learning processes. While metacognitive knowledge can be subject to students misunderstanding themselves, metacognitive skills have built-in feedback mechanisms [4].

As such, assessments are critical when designing for metacognition. The systems of old have assessment engines that operate as knowledge checks rather than tutorial devices that can facilitate reflection among learners. This reflection allows the learner to construct knowledge because the learner guides the knowledge-acquisition process [5].

2.2. Analysis of Current LMS Leaders: Blackboard, Moodle, Canvas

Our selection criteria were to take the three leaders in the field. By coincidence, Stevens has used two of these (Moodle and Canvas). Blackboard, the first system to market, is still the market leader in percentage of universities covered. As of Fall 2015, Blackboard's market share is 34.5 percent, almost double its closest competitor, Moodle. Blackboard has been losing market share for years, perhaps naturally since the sector is growing fast with many competitors. Moodle is the second largest in the LMS market, the largest of the open source solutions and still growing. Canvas has the third largest user base and is also the fastest growing LMS.

However, for eLearning and the LMS to be successful, it must be rooted in pedagogical practices. Govindasamy [6] lists assessment as among the five most important parameters for making eLearning implementation successful. As such, it is no surprise that LMSs tout online assessment as a timesaving benefit to using the system. However, according to our research, this is one of the least-used features of LMS today. Why is this feature not being used if it is in the teachers' interest to do so?

To understand this, we considered a number of use-cases in a teacher's workflow. First, the instructor must construct problem sets. If the instructor uses problem sets provided by textbook publishers, they often will be forced to purchase access to the publishers' proprietary website and LMS. The result is that teachers can either use textbook questions that are not integrated into the LMS, or enter questions manually.

Next, because the LMS grades the questions, answers must be provided in advance. As such, the teacher must design for all possible answers. Regardless of adaptive design in various LMSs, this form of upfront work is not required in synchronous, face-to-face evaluations. With paper, in-person quizzes, teachers can give an exercise to students and examine results once the students submit. It is vital that teachers be able to change answers on the quiz and reassess when discovering a problem. However, flaws in Canvas and other LMSs make this problematic. Answers can theoretically be changed while the quiz is being taken but the feature is unreliable. For this reason, students who input correct scores will often receive incorrect scores. This untenable situation requires the teacher make manual adjustments to scores, nullifying the benefit of automatic grading for the quiz. When we combine poor user interface, bugs, and up-front time and resource investment, utilizing the LMS hardly reduces teacher effort in assessment design. The real savings would come in the second or third time the course is offered. In order to improve adoption rates, we must reduce the initial cost so that using online quizzes is immediately beneficial to the teacher.

A third use-case is changes in scheduling. We undertook this study during the winter semester. As we are based in the New York area and were hit with heavy snows, classes were cancelled multiple times. We had entered forty quizzes into the course in advance in order to sort in the correct order; however, due to the class cancelations, we had to shift every quiz back to account for the later date. Worse yet, Canvas requires the teacher provide three dates: open, due, and close. Both due and close have to be updated; without a close date, students will still be able to access and submit quiz responses. This case shows that technology can be a two-edged sword. Because assignments can be viewed and students want to know, there is a strong incentive to post them early. But doing so then adds a huge burden when schedules change. Changing thirty-five quiz dates requires hundreds of clicks and a large time investment.

A fourth use-case challenge is the discrepancy between the features in LMSs and the pedagogical explanations for said features [7]. This has led LMS providers to predominantly view themselves as mere content and technology providers. Though assessment and evaluation is one of the key components to creating a learning environment, instructors are often already burdened with learning and navigating the eLearning setting.

Last, but by no means least, is the fact that every few years, schools change LMS providers. Changes in technology periodically force IT directors to upgrade systems; however, this requires additional effort from faculty when systems change. In our case at Stevens, we switched from WebCT to Moodle to Canvas within a span of six years. This rapid change of systems has caused teachers to view new systems as potentially transient; it is therefore understandable that they are resistant to invest the time into learning new systems.

Furthermore, the IMS Global Learning Consortium [8] defines a standard format for representation of assessment content called Question and Test Interoperability specification (QTI). QTI is designed to support the translation of material from authoring to delivery systems as well as LMSs. However, despite the promise of an import feature in our current version of Canvas, every quiz ported by the author has had to be manually changed. In one case, a quiz containing seventeen questions had an erroneous correct answer "6" added to every question. A majority of the questions required hand editing, often re-entering answers and question types. If vendors do not provide reliable imports, again, the incentives for teachers are not to fully use their systems. Note that in this case, the fault does not lie in the standard, but in the Canvas implementation; this difference is immaterial to the hapless professors who simply wants their course online in a new system.

Taken together, it is evident that the obstacles to utilizing the LMS to its full capacity are steep, and that teachers are not being irrational in declining to take full advantage of them. The next question, therefore, is whether it is possible to design software that improves the situation.

3. LiquiZ Improvements to Teacher and Administrator Workflow

3.1. LiquiZ Architecture Overview

LiquiZ uses advanced software architecture to achieve high speeds and fast response for a web application. First of all, the server side is running in Java Servlets using a MySQL database to store the data. Java servlets is one of the fastest server technologies, because once the program loads, it stays in memory. Java also tends to be significantly faster than php and Ruby on Rails, the technologies used in open source options such as Moodle and Canvas.

Every time a request is made to current-generation systems, they have to interpret their programs, execute a database query which goes to another server, wait for the answer, and then format the response before sending it to the client. By contrast, LiquiZ takes advantage of the large amounts of RAM on servers to keep the entire server data in memory, sending that data immediately to the client. The only time the database must be accessed is to write data that is critical and changing. In other words, when students take a quiz, the quiz is already in memory, but when they submit answers those answers are logged to the database. The result is much less work (fewer, smaller transactions) for the server, and thus much higher number of users per server.

The LMS problem is naturally easy to parallelize. If the servers become slow because the load is too high, we split the users into another server. There is a natural segmentation by department and by class. Students of one class do not need any information from others.

On the client side, we preload all the libraries used to draw the pages, and each time the user clicks, the program either immediately updates the screen if all data is available, or requests the data it needs from the server. In either case, the entire client uses a single page design—that is, it is a single web page that keeps redrawing itself. Our one-page design is unique in that it is in many cases completely self-contained, with no round trip to the server at all. It is possible to edit a quiz and save data locally, so that even with an internet connection down or slow, quizzes can be edited at full speed.

The combination of these two technological details have yielded extremely high performance, with the result that editing a quiz on LiquiZ feels like using a word processor, rather than a web page. In an extreme example, a quiz that took three hours to edit in Canvas took only ten minutes on LiquiZ. An architectural block diagram of the LiquiZ system is shown in Figure 1.

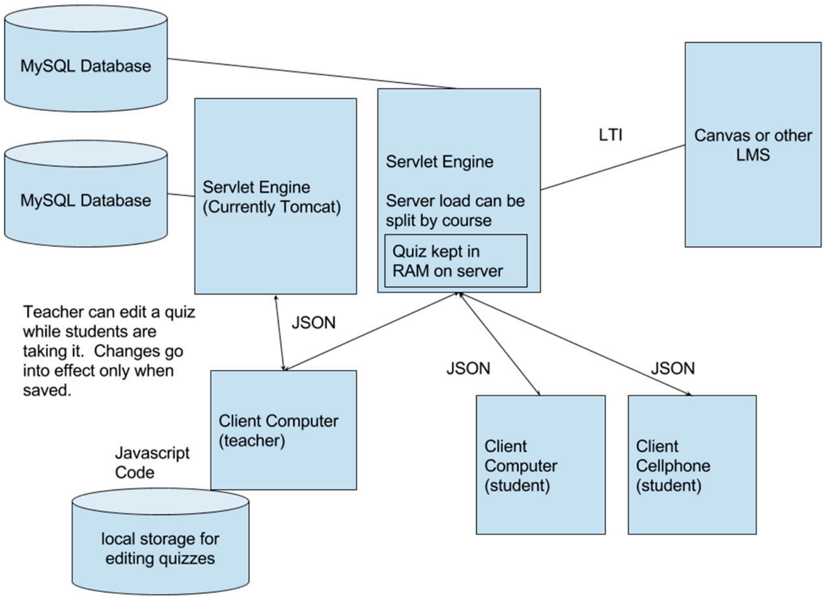

MySQL Database
MySQL Database
Servlet Engine (Currently Tomcat)
Servlet Engine
Server load can be split by course
Quiz kept in RAM on server
LTI
Canvas or other LMS
Teacher can edit a quiz while students are taking it. Changes go into effect only when saved.
JSON
JSON
JSON
Client Computer (teacher)
Javascript Code
local storage for editing quizzes
Client Computer (student)
Client Cellphone (student)

Figure 1. System Architecture of LiquiZ.

3.2. Optimizing User Interface and Page Traversal

The first step in improving an LMS is to reimagine how to interact with it. First, we minimize clicks by putting forms on a single page, having the sequence of fields so that the focus is usually where the user wants it without clicking. In other words, we used standard good practice in user interface design. The following screen shot shows a typical form in which pressing enter or tab automatically moves the cursor to the next field.

Current generation LMSs have more advanced question types, including pattern matching to catch a wider range of correct answers with less effort, formula questions for generating random variation of numerical problems, and numerical answers that accept a range of solutions. However, we have tested Blackboard, Moodle, and Canvas—three of the leading LMS with the biggest populations of support—and found that they are limited and inconvenient to use. Table 1 shows the number of clicks within page, and clicks transitioning to a new page required to create the same quiz in each of the three systems.

Table 1. Comparing Costs to Enter Data in Three LMS.

LMS	Est. page transitions	On-page clicks
Blackboard	14	59
Moodle	50	0
Canvas	5	50

3.3. Re-envisioning Workflow

Another important aspect to our assessment design is to examine and re-envision teacher workflow. There are three possible scheduling options in on Canvas. In our LiquiZ system, there is an additional scheduling option for flexibility. They are shown below in Table 2.

Table 2. Assignment Dates in LiquiZ.

Dates	Purpose
Visible	The assignment is invisible to students before this date.
Open	Assignment cannot be entered before this date.
Due	The student should ideally submit the assignment before this date/time.
Close	After this date, no student can submit the assignment.

To attend to the need for flexibility, the system must be capable of entering all these dates. However, most of the time these dates will be in a fixed relationship. The author, for example, typically gives out four small assignments per week in the beginning of a programming class. Each is due the following week, with the final closing date one week after that. Rather than force the user to enter all four dates for every assignment, LiquiZ allows the teacher to specify the date in relative terms. Since most of these attributes are always the same for a given teacher (seven days in this example), they are specified in a separate entity called a "policy", which controls many aspects of the quiz.

The LiquiZ policy includes many commonly required parameters of assessments, shown in Table 3.

Table 3. LiquiZ Policies.

Attribute	Purpose
VisibleBeforeOpen	Number of days before the open date that the quiz can be seen by the student
DaysUntilDue	The number of days after the open date that the assignment is due (days is stored in decimal, so an exact second can be defined)
DaysUntilClose	The number of days after the due date that the assignment remains open
PenaltyPerDayLate	The percentage cost for each day late (defaults to zero)
LatePenalty	Flat penalty for late submission
EarlyPerDayBonus	The opposite of a late penalty, with a bonus per day early
EarlyBonus	Flat bonus for being early
ShowCorrect	Either allow users to see the correct answer right away, or make them wait
ShowOwnAnswers	May users see and review their answers or do they have to wait until the close date when everyone has taken the quiz already

There are other elements to a policy, but the key concept is that there is a default. The teacher can either accept the default policy for the system, define a default for themselves, circumventing the need to enter any additional information, or they can create multiple policies and give each one a unique name. In this way, for almost any assignment, all the information (except for a single date) may be specified with a single policy name.

We took a similar approach toward surveys. In all three of the studied learning management systems, a survey is just a multiple-choice question with no correct answer. When creating a survey, the assessment designer will have to enter in the standard choices such as the Likert scale (strongly agree, agree, neutral, disagree, strongly disagree). On Moodle, the simplest way to generate a second question is to copy the first one and change the question. Canvas is far clumsier, requiring the user to re-enter the same answer set every time.

By contrast, in LiquiZ we select the predefined entity "Likert5" and refer to it. Commonly used choices are available with a single mouse click and selection. Again, teachers are free to define standard choices of their own, but the most popular standardized survey questions will already be loaded in the system, ready for use.

3.4. Sharing Between Users, Departments, and Schools

Development of assessment requires a great deal of content reuse. No matter how well-designed a user interface is, it will always be more difficult to create than to reuse. The most efficient way for teachers to build up assessments is to share a pool of questions and/or quizzes in their field. Rather than relying on proprietary content from publishers, teachers should have more control over how they use and re-use their course content. While publishers would like teachers to believe that they are the answer, in fact they are the problem. As such, one of the key requirements to any learning technology is that it be compliant with established standards. In our case, we required SCORM compliance as well as compliance with the QTI standards.

For LiquiZ we need to build a fine-grained system that will let users share entities from a question to a test, a book containing many tests, or even whole sets of courses. To do such sharing we will have to tag material with ownership, and users will have to state the license under which they are releasing the material. As LiquiZ grows and incorporates these concepts, the reader may be sure that we will make this sharing simple and efficient.

4. New Question Types

A key feature in assessment is that teachers utilize information to inform teaching; this involves accurate feedback regarding student success. However, this information is only useful feedback when it is used to meet the needs of the student [9]. If the information is merely recorded, "passed to a third party who lacks either the knowledge or the power to change the outcome, or is too deeply coded to leader to appropriate action, the control loop cannot be closed and "dangling data" substitute for effective feedback" [10]. Therefore, a key component of designing an assessment engine is ensuring teacher access to data in a meaningful way.

Another issue particular to assessment in online learning is that questions be designed to elicit metacognitive functions in learners. Metacognitive decision-making is a process of dynamic self-assessment and reflection that is generated through the interplay of inductive reasoning (specific to general) and deductive reasoning (general to specific) before arriving at a decision to act [11]. It is a process that individuals engage in to reason out ill-structured problems and/or situations, and it is a process that can be learned. Fore designing learning interactions within online environments, we need a model that is grounded on metacognitive principles and practice.

Standalone tools only offer basic question types with limited customizability such as multiple-choice questions, fill in the blank, true/false questions, and matching, which do not take advantage of computer-supported learning. Furthermore, these question types tend to score low on Bloom's taxonomy [12] but are useful for gauging students' knowledge of the basic facts. However, this type of questioning merely replicates older standardized testing, offering only the advantage of automatic grading. On the contrary, current technology allows for asking more pedagogically sound questions.

The main features we found necessary in designing an assessment engine are as follows: useful feedback data, variety of question types, and greater ease of use for faculty.

4.1. Question Types

All three studied LMS implementations supported the concept of formula questions. We can define an equation to be exercised. For example, in physics, students should demonstrate that they can solve problems using equations such as:

$$F = ma \tag{1}$$

Rather than generate a single problem where the student is given mass 5 kg and acceleration 2 $\mathrm{m/s^2}$, we can generate random problems using this equation. In Canvas, for example, the equation is written and the user can generate up to 200 randomized questions:

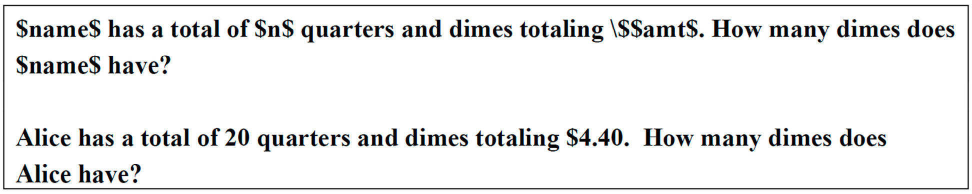
$$[F] = [m][a] \tag{2}$$

LiquiZ generalizes this notion to include specific value lists, because in some cases teachers want to restrict the values of variables for a pedagogic reason. We may define that m may range from 10 to 1000 in steps of 5, and a can vary from 0 to 10 in steps of 0.1. Or, we provide the ability to list acceptable values such as $a = [2, 3, 5, 10, 20]$.

Current generation system will only allow solving the equation forward. Thus the teacher can write $F = [m][a]$ and put in values for m and a, but reversing the equation would require writing another question. LiquiZ represents the equation and can solve it for any variable, so with a single specification of m and a, students can be asked for any of the three variables given the other two.

LiquiZ makes two other important generalizations. First, random values include strings (sequences of letters) and names, so LiquiZ is capable of generating families of word problems. For example, Figure 2 shows how random variables, delimited by dollar signs, may be used to specify a problem that is different every time it is generated.

LiquiZ questions can combine any question types in any order. For example, a random program may be entered where random elements are inserted and multiple fill-in questions ask for the results of the program.

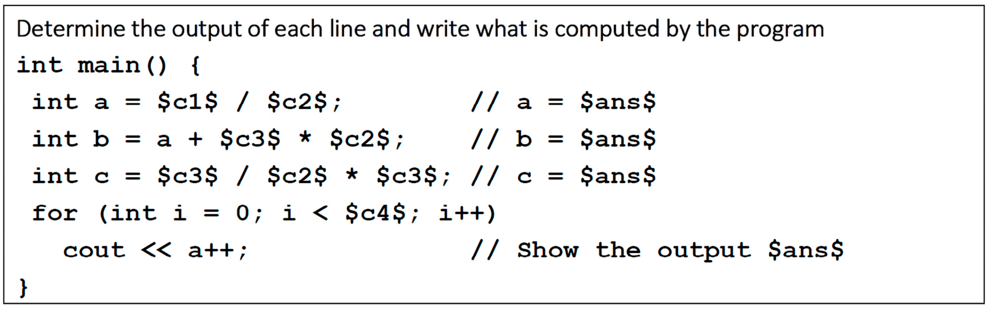


$name$ has a total of n quarters and dimes totaling \$$amt$. How many dimes does $name$ have?

Alice has a total of 20 quarters and dimes totaling $4.40. How many dimes does Alice have?

Figure 2. LiquiZ randomized word problem.

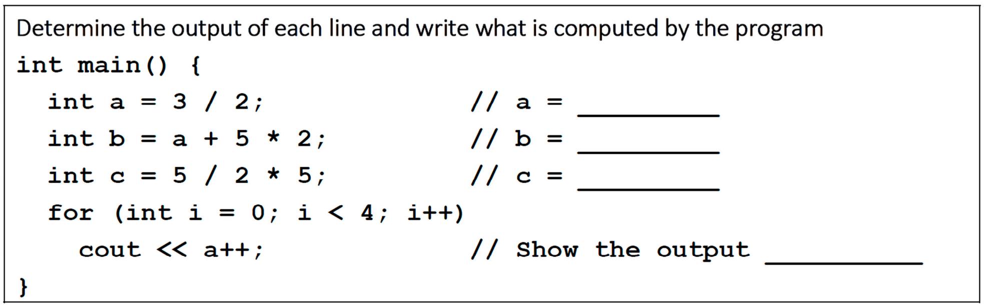
For example, Figure 3 shows a specification of multiple random variables and answers in a question. The random variables embedded in code would turn into a specific example as shown in Figure 4.



Determine the output of each line and write what is computed by the program

```
int main() {
 int a = $c1$ / $c2$;        // a = $ans$
 int b = a + $c3$ * $c2$;    // b = $ans$
 int c = $c3$ / $c2$ * $c3$; // c = $ans$
 for (int i = 0; i < $c4$; i++)
   cout << a++;              // Show the output $ans$
}
```

Figure 3. LiquiZ code problem specification containing random variables.



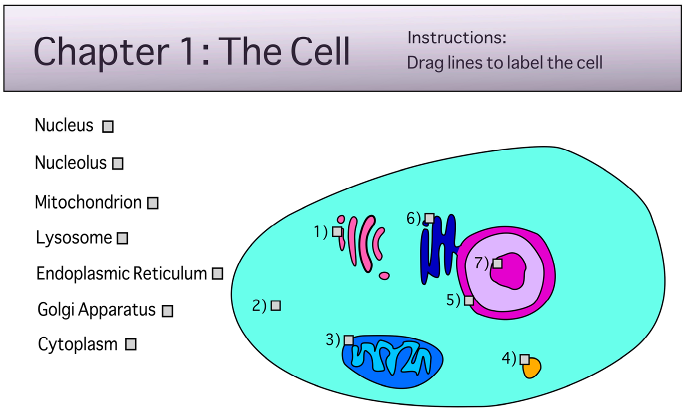
Determine the output of each line and write what is computed by the program

```
int main() {
  int a = 3 / 2;             // a = _________
  int b = a + 5 * 2;         // b = _________
  int c = 5 / 2 * 5;         // c = _________
  for (int i = 0; i < 4; i++)
    cout << a++;             // Show the output __________
}
```

Figure 4. LiquiZ randomized code output.

4.1.1. Visual Diagram Entry

Graphing questions are still in the design stages, but LiquiZ will allow random generation of graphs for students to analyze, or interactive entry of graphs and diagrams. This allows a more visual approach to assessing function and graph knowledge. Studies [13] have shown that visualization is key to metacognition. Peterson [14] outlines four categories for the relationship visualization has with thinking: reasoning, learning a physical skill, comprehending verbal descriptions, and creativity. Visual diagrams not only present concepts in an alternate visual format, but also present an avenue of options that cannot be expressed in current systems like Blackboard and Canvas. By giving the student the ability to draw or write an answer, assessments can engage a variety of learning styles; not only are students engaging visually but they are also creating the diagrams by drawing it themselves. With the ability to enter grade educational material based on diagramming, images, and point-to-point connection comes the ability for teachers to create their own online exercises to test student knowledge of processes which multiple choice and fill-in-the-blank questions simply cannot adequately cover. Visual diagram questions allow a student to answer by dragging and dropping in answers, connecting answers by lines, and/or moving answers into different locations. Visual diagrams also have the capability to be combined with any of the other types of questions. For example, a teacher is able to import an image of a cell and ask students to drag cell organelles into proper position and to name the organelles using a drop down menu or by connecting the organelles by dragging lines to link organelles with their names. Figure 5 shows an example drawn from biology.


Chapter 1: The Cell

Instructions:
Drag lines to label the cell

Figure 5. An image-labeling question.

Similarly, in electrical engineering, we can ask low-level questions about the equations, or simply show a circuit and ask the student to fix it, assuming we have a drawing interface that allows questions to be posed in that manner. Figure 6 shows a simulation of a breadboard and asks the student to "fix" the broken board. Contrast this kind of assessment with a typical standardized exam. In this case, students can demonstrate the skill, which is the outcome, or they cannot. There is no way to "prep for the test" short of knowing the subject.

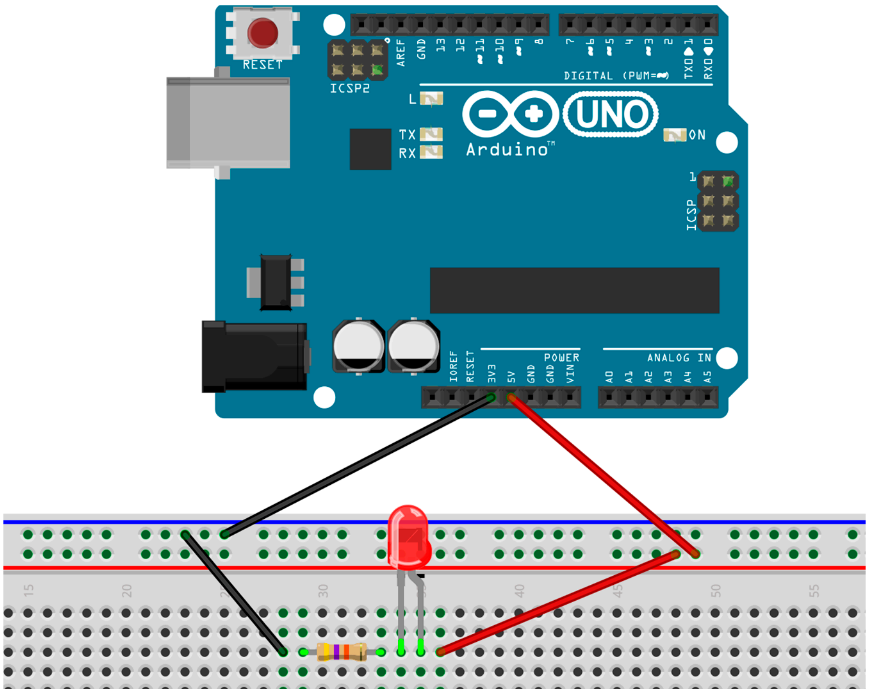
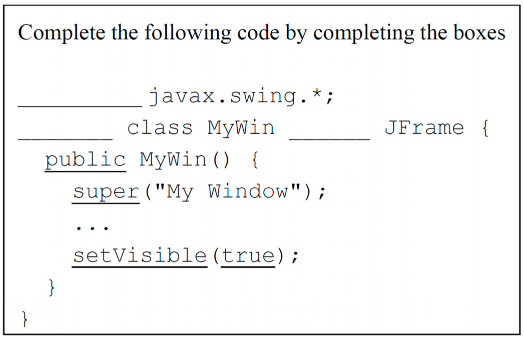


Figure 6. Diagrams as answers: fix the circuit.

4.2.2. Programming Questions

In computer programming, the definition of a successful outcome is clear—students must be able to write code on their own. This is not traditionally used in assessments, and therefore questions on a test about programming often are not full programs because it is error-prone and difficult (and time consuming) for humans to grade such questions. However, full programming with a compiler to build the program is not the only kind of assessment that is reasonable. Just as English language students learn to write paragraphs but also drill to build vocabulary, programming classes need to build lower level skills as well.

In some cases, having students determine what to type at key locations of a fixed program is more effective than free-form code because allowing the student to write an arbitrary program permits them to use the language features with which they are most familiar. Using specific questions, on the other hand, it is possible to demonstrate mastery of a more complete range of language features. Figure 7 shows a typical program with underlines showing the critical sections the student must complete:



Complete the following code by completing the boxes

```
__________ javax.swing.*;
________ class MyWin ______ JFrame {
  public MyWin() {
    super("My Window");
    ...
    setVisible(true);
  }
}
```

Figure 7. Complete the code: understanding keywords in context (CLOZE).

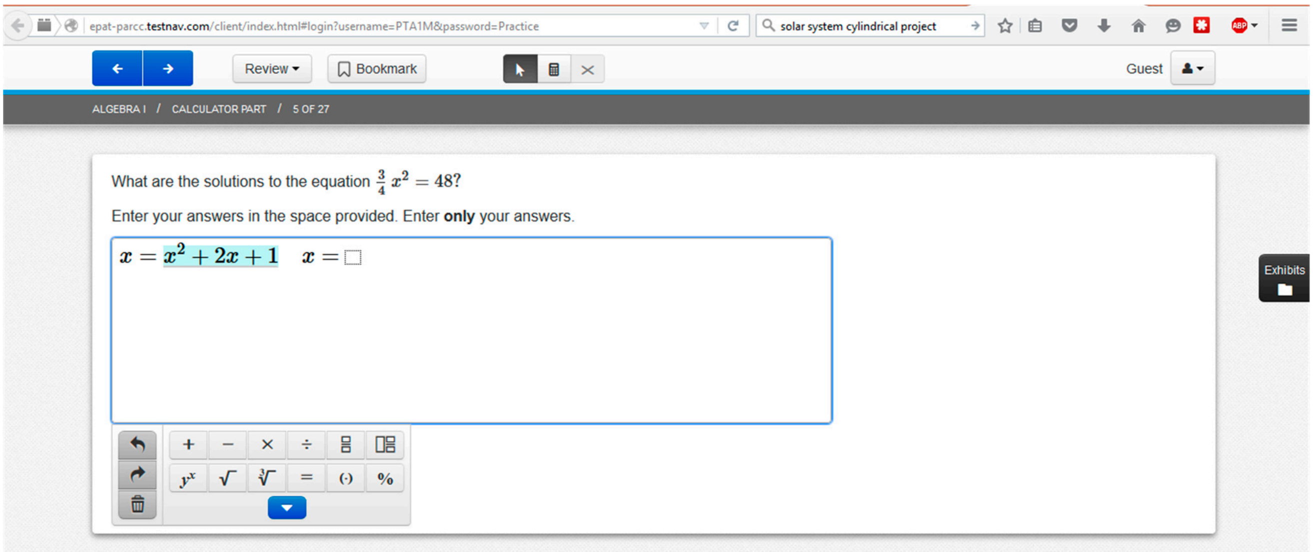
4.2.3. Equation Questions

Only a few engines such as the PARCC (Partnership for Assessment of Readiness for College and Careers) exam appear to support allowing an instructor to ask a question whose answer is an equation. The PARCC mathematics exam has a number of interesting features. It allows the student to interactively edit graphs in response to questions. The graph editor, however, is clumsy and non-intuitive. We have not yet built a graph editor into LiquiZ, but certainly the PARCC test shows both the promise of being able to define graphs, and the dangers of a poorly conceived interface being used in high-stakes testing. See Figure 8 for the PARCC interface.

The PARCC equation editor is intuitive and clean; however, in the sample test questions we reviewed it was completely unused. This reflects the difficulty of grading answers in equations. The following figure shows a typical case in which the equation editor is displayed.

According to the test design, students should be allowed to enter a constant for x; however, the equation editor allowed for erroneous equation inputs. No actual use of their equation editor was observed.

The key to make such questions practical for the average instructor is to understand how to define the set of answers that are correct. This is extremely labor-intensive, and can result in good answers not being accepted. The equation question type supports entering equations as answers, and can automatically compute potential correct answers.



epat-parcc.testnav.com/client/index.html#login?username=PTA1M&password=Practice

solar system cylindrical project

ALGEBRA I / CALCULATOR PART / 5 OF 27

What are the solutions to the equation $\frac{3}{4}x^2 = 48$?

Enter your answers in the space provided. Enter **only** your answers.

$x = x^2 + 2x + 1 \quad x = \square$

Figure 8. PARCC Equation Editor.

For example if the question is to evaluate the formula $(x-y)^2$, correct answers would include:

$$x^2 + y^2 - 2xy \tag{3}$$

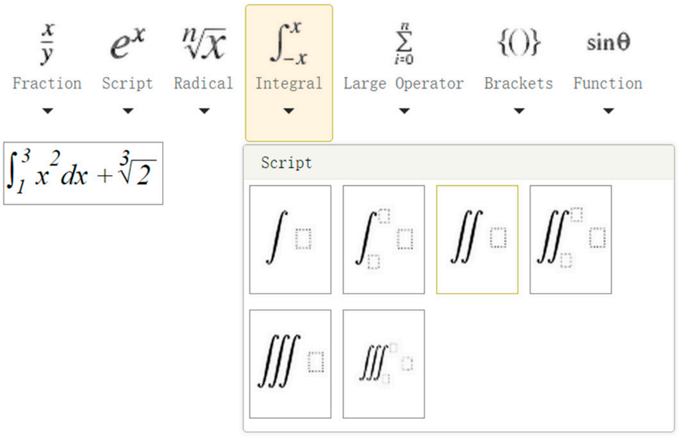
$$y^2 + x^2 - 2xy \tag{4}$$

$$x^2 - 2xy + y^2 \tag{5}$$

$$y^2 - 2xy + x^2 \tag{6}$$

$$xx - 2xy + y^2 \tag{7}$$

The answers above include permutation of terms and rewriting squaring as multiplication. The current problem the team is struggling with is to help the teacher by accepting all correct answers without accepting answers which are “cheats” or “tricky”. For example, an engine which accepts any answer equivalent to the question above will accept the question itself as an answer. The student could therefore type in $(x-y)^2$. If we explicitly disallow the identical answer, a clever student can still cheat the system by entering something algebraically equivalent like $(-y+x)^2$, $1(x-y)^2$ or $(x-y)^2(x/x)$. See Figure 9 for an example of the simplified user interface.



$\int_1^3 x^2 dx + \sqrt[3]{2}$

Figure 9. Entering equations: simplified visual interface

4.3. Adaptive Quizzing and Real-Time Questions

The LiquiZ system is currently designed to allow rapid off-line editing. One mode in development is an interactive quiz system that would enable teachers to dynamically create in-class questions, or create them in advance. The teacher can pose a question in class and ask students to answer via smart phone, tablet, or laptop. The grading and reporting is semi-anonymous—students do not know each other's answers, reducing the fear of shy or weak students, but the teacher gets a report and can insist that everyone participate. The system grades submissions and elicits a short report on number of submissions and solution accuracy. Some noticeable impacts of conducting in-class quizzes with the system are: reviewing important concepts or questions effectively, identifying difficult or problematic topics in time to review them, and encouraging individual learning and class participation. This data-driven, real-time feedback methodology addresses the deficiencies educators encounter with conventional off-line quizzes. The increased interactivity has been shown to augment class attendance [15]; this effect should only increase given that students know their record of participation objectively impacts their grade. This data-driven real time feedback methodology addresses the deficiencies educators encounter with conventional off-line quizzes, which being incapable to increase class attendance or to get timely grading. It is also the competitive advantage of the system among other online quiz systems.

5. Experimental Section

Since Stevens Institute of Technology uses Canvas, our experiments largely compare LiquiZ against the Canvas LMS. We also have some test results gathered in testing Blackboard, Moodle and Canvas against each other.

The LiquiZ prototype is not yet fully functional, but our initial tests show delays well under 0.1 s when we access a remote server leased in New York (off campus). By comparison, bringing up the Canvas list of quizzes during the semester often exceeds 10 s, with similar times to access a quiz. While it remains to be seen how LiquiZ scales, we believe we can maintain this factor of 100-fold speed improvement.

The number of clicks needed to enter a question is dramatically reduced, and all those clicks are intra-page, incurring no delay. Table 4 shows the relative costs of typical operations.

Table 4. Operations count: Canvas *vs.* LiquiZ prototype.

Action	Canvas Clicks/Typing	LiquiZ Clicks/Typing
Create a new quiz with my standard options	8/2	3/1
Create a new multiple choice question	8/6	3/5
Create a new fill-in question	9/2	3/2
Create 10 multiple choice survey questions	80/60	4/14

The numbers in the above table do not show timings, but this is because some of the Canvas clicks involve lag in getting data from the server.

Equally important, all the intra-page clicks reduce the number of hits to the server, which means that server load per student is dramatically lower. This means that for the same server resources, far more students can be supported without the corresponding slowdown that would be observed in Canvas.

No experiments have been conducted on question types yet, but as LiquiZ becomes mature enough, it will be tested in a number of partner schools and classes at Stevens.

6. Conclusions

Existing assessment tools are inadequate for designing for metacognition in two ways. First, the design of these tools limits the type of questions instructors can ask, which cripples the performance of online assessments. The ad hoc tools external to the LMS are the most limiting (typically just multiple choice and fill in the blank) but also suffer because it is up to the teacher to respond to the results and collect the grades.

Second, the LMSs we studied—Blackboard, Moodle, and Canvas—are poorly designed from a user interface perspective, requiring far more time to construct quizzes than is necessary. We found that these systems were more intuitive for the students but less flexible for faculty. Although one of the biggest advantages of an LMS is that a series of assessments/assignments can be designed for mastery learning, the fundamental hurdles built into the current generation LMS prevent most teachers from using them. We believe that by making assessment design easier for faculty, students will benefit from a more robust quizzing process.

A better assessment system should enable teachers to quickly create quizzes with question types that allow for reflection and feedback, create randomized families of questions that can be reused and shared, and quickly assign them as homework to students, requiring students to review and retake quizzes if they need additional reflection on their misunderstanding. Additionally, teachers should be able to interactively ask questions in class to determine class comprehension as well as involve more students in active learning. In this way, the assessment engine should not allow only for metacognitive skill development within the learner, but also for reflection and re-design for the instructor.

We are committed to making sure that a high quality, flexible LMS assessment engine is available to every student, for daily use, free of charge. Despite development of open source educational tools and content made available via the Internet, organizations such as College Board and ETS have continued to raise the price of standardized tests. We hope that LiquiZ will be the beginning of a movement to democratize testing, and let a truly free, open market of ideas shared by millions of educators determine the best questions and best methods of assessment.

The LiquiZ system, currently being implemented is attempting to address these issues. We hope to complete a first version by Spring 2016 and start testing with students in a computer-engineering course at Stevens Institute of Technology.

Acknowledgments

The authors would like to acknowledge the help of Asher Davidson and Yubin Shen in implementing some of the key components of LiquiZ during the summer 2015 Electrical and Computer Engineering team projects. We would also like to thank Michael Scalero and Kim Gabelmann for support in using Canvas, and in helping get developer support for interfacing with Canvas. Thanks to Youcef Oubraham for help using Moodle. Last, thanks for Ann Oro for reviewing an early draft of this paper.

Author Contributions

Dov Kruger is the lead investigator of the LiquiZ project and designed the software and architecture. Sarah Inman is the Senior Instructional Designer at Stevens Institute of Technology. Prior to this appointment, she researched learning theory and design. She evaluated LiquiZ, reviewed the major LMSs, researched assessment, and wrote and co-edited sections of the paper. The remaining authors are students who all contributed by developing components of LiquiZ and writing sections of this paper.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Medved, J.P. LMS Industry User Research Report. Available online: http://www.capterra.com/learning-management-system-software/user-research (accessed on 8 April 2015).
2. E-Learning Market Trends and Forecast 2014–2016 Report. A report by Docebo. Available online: http://www.docebo.com/landing/contactform/elearning-market-trends-and-forecast-2014-2016-docebo-report.pdf (accessed on 8 October 2015).
3. Flavell, J.H. Metacognition and cognitive monitoring. *Am. Psychol.* **1979**, *34*, 906–911.
4. Veenman, M.V.J.; Bernadette, H.A.M.; Hout-Wolters, V.; Afflerbach, P. Metacognition and learning: Conceptual and methodological considerations. *Metacognit. Learn.* **2006**, *1*, 3–14.
5. Schwartz, N.H.; Anderson, C.; Hong, N.; Howard, B.; McGee, S. The influence of metacognitive skills on learners' memory of information in a hypermedia environment. *J. Educ. Comput. Res.* **2004**, *31*, 77–93.
6. Govindasamy, T. Successful implementation of e-learning pedagogical considerations. *Internet High. Educ.* **2002**, *4*, 287–299.
7. Firdiyiyek, Y. Web-based courseware tools: Where is the pedagogy? *Educ. Technol.* **1999**, *39*, 29–34.
8. IMS Question & Test Interoperability Specification Overview. (n.d.). Available online: https://www.imsglobal.org/question/index.html (accessed on 8 October 2015).
9. Yorke, M. Formative assessment in higher education: Moves towards theory and the enhancement of pedagogic practice. *High. Educ.* **2003**, *45*, 477–501.
10. Sadler, D.R. Formative assessment and the design of instructional systems. *Instr. Sci.* **1989**, *18*, 119–144.
11. Kaniel, S. A metacognitive decision-making model for dynamic assessment and intervention. In *Advances in Cognition and Educational Practice: Dynamic Assessment: Prevailing Models and Applications*; Lidz, C.S., Elliot, J.G., Eds.; Elsevier Science: New York, NY, USA, 2000; Volume 6, pp. 643–680.
12. Krathwohl, D. A revision of bloom's taxonomy: An overview. *Theory Pract.* **2002**, *41*, doi:10.1207/s15430421tip4104_2.
13. Gilbert, J.K. Visualization: A metacognitive skill in science and science education. In *Visualization in Science Education*; Springer: Drodrecht, The Netherlands, 2005; pp. 9–27.

14. Peterson, M.P. Cognitive issues in cartographic visualization. In *Visualization in Modern Cartography*; MacEchren, A.M., Taylor, D.R.F., Eds.; Pergammon Press Ltd.: Oxford, UK, 1994.
15. Kay, R.H.; LeSage, A. A strategic assessment of audience response systems used in higher education. *Australas. J. Educ. Technol.* **2009**, *25*, 235–249.