Teaching Software Construction at Scale with Mastery Learning: A Case Study

Elisa Baniassad, Alice Campbell, Tiara Allidina, Asrai Ord
Department of Computer Science
University of British Columbia
Vancouver, Canada
ebani@cs.ubc.ca, alicecam@cs.ubc.ca, tiara.allidina@alumni.ubc.ca, spring.asrai@alumni.ubc.ca

***Abstract*—Mastery Learning involves delineating learning units and assessing each unit individually and repeatedly until a student obtains success. Mastery Learning has been shown to help students better identify and grasp fundamental concepts. We applied Mastery Learning in a second-year software construction course with roughly 450 students. We delineated 23 topics, and administered either a written or verbal quiz to assess each topic. We built a quiz auto-grading, analysis, visualization, and feedback system to help cope with the scale of the class. By the end of the semester we had administered over 12K quizzes. We found evidence that students grasped both fundamental concepts and advanced concepts better than in prior semesters. Because we made two changes at once (introducing videos and Mastery Learning) it is difficult to isolate whether the Mastery Learning Approach was solely responsible, but assessment results suggest that the repeatable micro-quizzes were instrumental in these gains. Auto-grading and extensive data collection allowed a depth of analysis that afforded us invaluable and lasting pedagogical insights.**

I. Introduction

Software Construction concepts are seemingly cumulative: to show proficiency on design patterns, surely you must understand inheritance! But when grading a design patterns question on a final exam, a substitution error would likely result in a penalty, but not a failing score. A student might get 50% of the points on the exam, or across the range of assessments in the course, but are they understanding the fundamental 50% of the course material? Should they obtain a passing grade if they know that the Observer implements the update method but don't really understand polymorphism? In theory, one could design an exam that makes the fundamental concepts 50% of the points, but in practice, that is tricky to achieve.

We teach a large Software Construction (SC) course to approximately 450 students (split over 3 sections) each semester. The course covers the basics of Java and Object-Oriented design. In follow-on courses for which the SC course is a prerequisite, students were showing troubling gaps in knowledge, to the point that instructors had to re-teach material to keep the bulk of the class on track. This might have been understandable for more advanced topics (it would make sense that not all students would grasp the Liskov Substitution Principle or one of the design patterns), but this was also the case for basic concepts, such as whether you could substitute an object with its supertype.[1] We noticed in downstream courses that basic clarification questions were coming even from "good" students that had shown proficiency in the final exam on the more advanced topics.

Mastery Learning [2] is a teaching approach that focuses on delineating learning outcomes such that they can be individually acquired by students. It aims to help students have a better account of what they have learned, and what they still need to work on. It can also help identify which topics are more central or important than others, by making those topics a requirement of passing the course, or a requirement for progressing to subsequent concepts. Students are quizzed on, and effectively held accountable for, each of the important details in the course. Rather than having comprehensive quiz, midterm, or exam questions that perhaps obfuscate or combine concepts, these micro-assessments test each small concept in an isolated way, so that students could identify precise pieces of information they have not yet grasped.

We had heard of Mastery Learning being applied by Wrigstad *et al.*[6] in an introductory software engineering course. In their model, they identified 100 achievements (worth 1 point each) that were assessed in verbal quizzes administered by a TA. Students were quizzed in pairs. They had an electronic sign up/scheduling system whereby students could pace their own tests, taking quizzes only when they felt prepared. Students could retake quizzes until they passed them. TAs immediately informed the students of whether or not they had passed their mastery test. The results of applying Mastery Learning were impressive: students were more aware of what they knew, and what they didn't know. Because certain topics were required to pass the course, instructors for follow-on classes could rely on prerequisite student proficiency. Students were also able to set their own pace, scheduling quizzes for when they felt they were ready.

We wanted to apply Mastery Learning in our SC course, but were concerned that the size of the class would pose challenges. We have fewer TA hours than were available in [6], so having verbal presentations, even in pairs, would either require a reduction in the number of topics, or would necessitate such short grading encounters as to be overly superficial.

[1]For instance with `Subtype t = new Supertype()`

We decided to try a version with 23 topics (listed in Section XIV), 14 of which were identified as so fundamental that students were required to pass them to pass the course. Students could retake those 14 quizzes as many times as needed. We created a quiz auto-grading and data visualization infrastructure. The auto-grading was absolutely essential, given that by the end of the semester we had administered over 12K quizzes.

Students expressed satisfaction with the course overall. Student performance was strong on a standard-style (not multiple choice) midterm and final exam, and while it's difficult to compare different cohorts of students, we believe that the approach had the hoped-for benefits: students kept on pace and they had a better grasp of fundamental concepts.[2]

We garnered significant pedagogical insight because of the data-centric nature of the approach: because we performed very fine-grained assessments, we were able to perform fast, in-depth analysis into student progress both individually and as a group.

Based on our experience, we believe that with auto-graded micro-quizzes combined with data visualization, Mastery Learning can be successfully administered and achieve the desired learning benefits, even in large software design courses.

In this paper we describe our Mastery Learning setup, and examine our experience in terms of student outcomes (student experience and pedagogical outcomes) and instructor outcomes (pedagogical insights and organizational concerns).

II. Related Experiences with Mastery Learning

Recently, Mastery Learning has been applied in large programming and software development courses. University of Toronto has applied a mastery approach in their 2000-student first year introduction to programming course [7]. McCane *et al.* [5] looked at applying Mastery Learning in a large introduction to Python course (roughly 250 students). Engle *et al.* [3] report on using Mastery Learning combined with code review to deliver a course in software development. Others have applied Mastery Learning to introductory programming courses ([4] for instance.). The most similar to our own course was that offered by Wrigstad *et al.* [6] to 160 students.

In all cases, students were found to obtain learning benefits—especially students in the lower grade range. These offerings have differed slightly in their organization and style, to suit the curriculum and course infrastructure, but they all hold in common fine-grained assessment and a differentiation between levels of assessment from required to advanced.

III. Course Setup Overview

Our infrastructure for delivering Mastery Learning is shown in Figure 1.

[2]Our technique differs from a traditional Mastery Learning application in that we administered a midterm and final exam. The final exam is compulsory at our institution, and we offered the midterm to provide benchmark validation against the Mastery Learning approach.

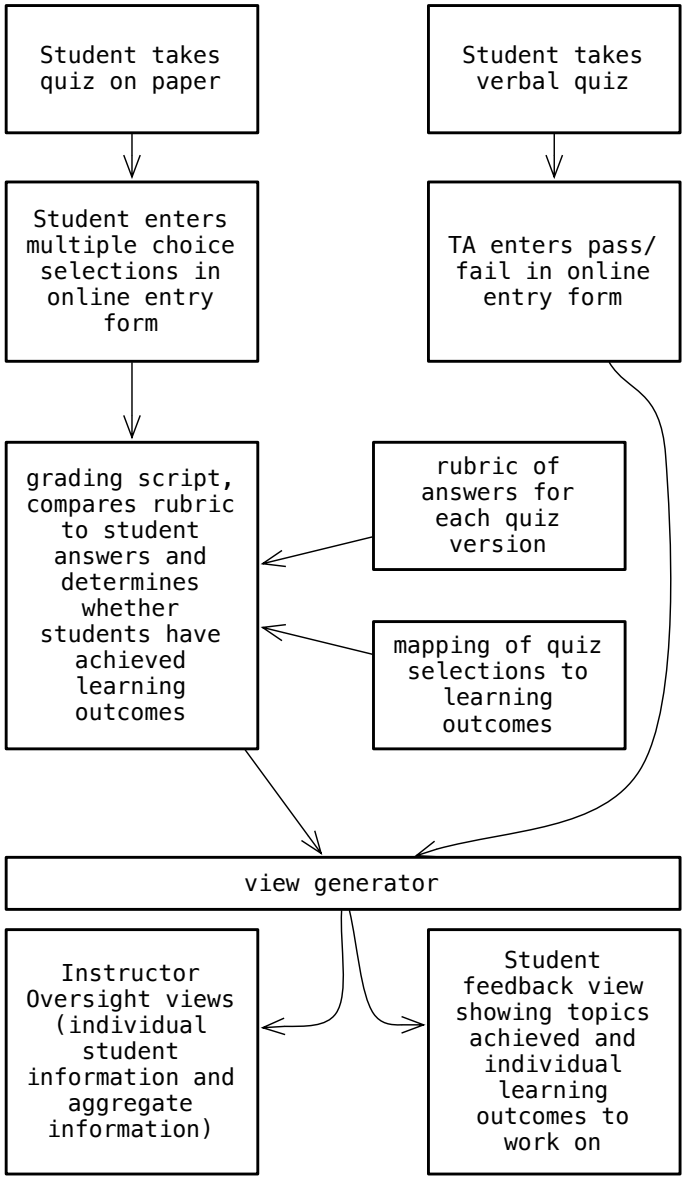

Fig. 1. Quiz delivery setup

The course offering had three sections, each with roughly 160 students. There were a total of 18 2-hour lab sections, ranging in size from 11-28 students (most near 28), each staffed with 4 TAs. We had 26 TAs (2 graduate TAs, and 24 undergraduate TAs) with a 12-hour weekly cap each, though the weekly average for each TA was 8.3 hours. We also had a half-time course coordinator who provided administrative support for the quizzes, ensured that grades were accurately entered into the gradebook, scheduled all TA hours, and arranged accommodations for those students who needed them.

A survey of the TAs revealed that they would be hesitant to fail students, except in obvious cases. These are undergraduate TAs who, in many cases, are socially acquainted with students in their labs. We believed that it was likely, also, that if students were graded in pairs, then TAs would not be able to enforce

the rule that there be equal contribution by the students. We did, however, want to require students to discuss their code. This was something that had not been substantially present in the course. Rather than holding only written quizzes, we decided that verbal assessments would be used for 1/3 of the topics, and that we would try to assess each of the verbally assessed topics on short written quizzes.

We created short multiple choice quizzes for 2/3 of the topics, with four quiz variants per topic. Students were quizzed on a topic two weeks after it was covered in lecture. In the intervening week the student worked on an assignment related to a cluster of topics. Quizzes were 5 minutes long, whether written or verbal, and were given in labs that ran throughout the week.

Quizzes were distributed on paper, but answers were entered electronically to facilitate instantaneous auto-grading and statistical analysis. Quizzes were auto-graded and feedback was generated to help guide the student's future study. Because of the size of the class and the on-paper quiz delivery, we could not allow students to self-schedule quizzes. Instead we pre-scheduled three quizzes per week (two written, one verbal) with additional retakes offered as needed.

Infinite retakes for required quizzes allowed us to provide some self-pacing, though students would only get the full 1 point for a quiz if they passed it the first week it was offered. If they passed the quiz in a subsequent week, their passing grade would be .5.

Many Mastery Learning offerings take a staged approach, meaning that you cannot progress to a later stage until you have passed all the assessments in your current stage. For logistical reasons we did not enforce anything like this—we did not want TAs to have to gate-keep who could take which quiz, and we did not want students feeling badly if they were told they were not allowed to take an assessment in front of their peers. Instead, students just took quizzes as they came up, and it was indeed the case that some students were re-taking basic required quizzes in the same lab sitting as they took (and often passed) more advanced ones.

To make the setup work without being able to individually sit with students to discuss their progress, we needed to provide rich grading information back to students. We also needed instructor oversight views to keep track of the progress of the students on aggregate. Each of these will be described further below.

IV. Quiz Delivery

We constructed a quiz, either verbal or written, for each of the 23 topics in the course. Written quizzes were given in the first 15 minutes of the 2-hour lab, with individual verbal quizzes taking the rest of the first hour. The second hour was open to drop-in questions from students from other labs.

Verbal quizzes consisted of a simple question that asked the student to draw on their recent experience working on their assignment. One such question was "show me how your code uses exceptions." TAs determined whether the student satisfactorily answered the question.

Written quizzes were designed for reuse, both throughout the week and across semesters. For this reason, we did not distribute the quiz questions electronically: we did not want students taking screenshots of questions and distributing them to their peers.

To avoid manually grading or scanning answers, students used an online version of a generic scantron entry form. An example question is shown in Figure 2.

```
Which of these statements will compile:
1. Supertype s1 = new Subtype();
2. Subtype s2 = new Supertype();
```

Fig. 2. Quiz question sample

The paper quizzes were meticulously collected at the end of the five-minute period. Students wrote their names and student numbers on the papers so we could (in theory) track who had kept their copy. To discourage cheating, we created four versions of each quiz. The version identification was only visible on the back side of the quiz—the student entered that quiz code prior to turning over their sheet. This made it more difficult to tell if a student sitting next to them had the same quiz as they did. The quiz variations looked very similar, with very bland and generic naming such that it would be tricky to tell how the quizzes differed.

V. Quiz Auto-Grading and Auto-Generated Feedback

Quiz submissions were auto-graded using a Python script that additionally collected and analysed statistical data from the results [1]. A rubric for each quiz was compared to the student's selections. If the student made too many wrong selections (because the quizzes were short, typically we only allowed two wrong selections), then they did not pass the quiz. Students were not informed how many wrong selections they made—just whether or not they passed.

Students needed speedy feedback about quiz results so they could know whether to prepare to retake a quiz. This meant that Monday-Tuesday lab students would receive their quiz grades prior to those in Wednesday-Friday labs having taken the same quiz. For this reason, and because we wanted to reuse quizzes, we could not give students back their original graded quiz. Instead, we mapped each quiz question to a learning outcome. For instance, the first of the statements in the quiz question above would have been linked to the learning outcome *a subtype can be substituted for its supertype*, and the second would have been *a supertype can not be substituted for its subtype*. If the student did not select option 1 above, they would receive a grading report that included the message *you need to study that a subtype can be substituted for its supertype*. If they answered correctly, they would receive the message *you have shown understanding that...* and then the rest of the learning outcome. The intent was that given this information about which concepts they needed to continue working on, students could study for their retake quiz if one

was needed. An example of one of these reports is shown in Figure 3.

Feedback for your latest B3 quiz:

You have shown understanding of how many fields a class has

You still need to work on whether the original object reference shows changes if a duplicate reference alters an object

You still need to work on how many objects are active

You have shown understanding of how many methods a class has

Fig. 3. Quiz feedback report for a student who needs to retake B3: Classes, Objects and Variables, a week 1 topic

If students needed more in-depth feedback, they could attend office hours and review a paper copy of the quiz they wrote with an instructor.

VI. Instructor Oversight Views

Because of the size of the class, we needed technical support to stay on top of how students were performing. To facilitate this, we automatically constructed views from the grading data to provide us oversight and spot students who were in trouble.

A. Topic results summary

To let instructors get a quick overview of how many students needed to retake a quiz, we provided a view that showed a simple count of how many students were passing each topic.

topics:	B1	B3	B4	A1	A3	A4	A7	C2	C3
failing:	17	133	57	59	0	0	0	0	0
passing:	513	362	189	188	0	0	0	0	0
missing:	20	55	304	303	550	550	550	550	550

Fig. 4. Passing, Failing and Missing student counts for every topic (the high missing numbers for B4 and beyond are because this snapshot was taken half way through the week in which B4 and A1 were given in labs, and one week prior to the distribution of the A3 quiz)

We wanted all the students to pass all the required quizzes, so we watched those numbers as the course progressed and reached out to students who were not passing required quizzes from earlier in the course. We scheduled additional tutorials for them and tried to work with them to overcome any conceptual barriers.

B. Averages for every answer for every quiz

To assess the fairness of each quiz question (with relation to the other variants of the quiz), and additionally to quickly spot any errors in the grading rubric, we constructed a view showing the average grade for each quiz question. Averages for each question were colour-coded to highlight areas of difficulty. If one quiz had a single question in which students were performing badly, we could look at that question and decide whether to exclude it from consideration. The pattern evident in successful quiz variations is the one shown in Figure 5, where each quiz has similar (though typically not identical) patterns of success. The 3 lower quizzes in the diagram (c3tst1,2,3) were the quizzes given in the first week of this topic, with a similar pattern of success. Questions 13 and 14 were difficult for all students, and each quiz had 1-2 other questions that had lower scores. The top quiz was the quiz given as a retake—this meant students would have already taken a variant of this quiz, and so we don't see the low points in specific questions other than the difficult pair of questions that were moved to 27-28. This shows that students who did the retake had on average improved over the first time they took the quiz—the slightly low scoring questions were now scoring better, and the very low pair of questions (27-28) had better averages than the original (13-14). The pink "36" on the left column indicates that out of 40 selections, students had made on average 4 incorrect selections.

C. Graded quiz answers for an individual student

Students who came to see us for more information needed to know exactly which questions they answered incorrectly. A special view showed every answer to every quiz for a particular student, and colour-coded them to indicate whether their answer was correct.

D. Learning outcomes achievement

We wanted access to a finer-grained view of students' attainment of learning outcomes. We mapped learning outcomes to particular quiz questions, and constructed a view that identified how many students had correctly and incorrectly answered questions associated with each learning outcome. This helped us straightforwardly identify which learning outcomes students easily achieved and which they found more challenging.

VII. Analysis Overview

We used various methods to reflect on and assess the experience of applying Mastery Learning:

Surveys: At the start of the semester we gave students an opt-in survey with ethics approval on their expectations for the course and their learning mindset. At the end of the semester we gave students a second opt-in survey asking them about the effectiveness of, and their satisfaction with, the delivery mechanisms in the course.

Focus Group: Approximately mid-way through the semester, we held a focus group asking students to comment on their satisfaction with the course, how well the delivery mechanisms were working, and for any free-form feedback about the quiz experience. Students specifically commented on whether they felt the quizzes helped them to stay on pace and whether the quiz format and feedback were working for them.

Benchmarks against the previous semester: We gave similar questions on the midterm and final exams to those given

topic: C3		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
c3tstrt1	36	1.0	0.99	0.82	0.8	0.96	0.84	1.0	0.93	0.92	0.92	1.0	1.0	0.88	0.88	1.0	1.0	0.92	0.93	0.8	0.91	0.96	0.97	0.87	0.85	0.99	1.0	0.43	0.43
c3tst3	36	0.85	0.85	1.0	0.99	0.93	0.87	0.9	0.68	0.99	1.0	0.86	0.87	0.31	0.31	0.88	0.9	0.99	0.99	0.89	0.96	0.77	0.94	1.0	0.99	0.9	0.89	0.97	0.97
c3tst2	36	0.99	0.99	0.93	0.94	0.66	0.94	0.89	0.85	0.91	0.89	0.99	1.0	0.25	0.26	1.0	0.98	0.89	0.87	0.91	0.67	0.96	0.8	0.88	0.89	0.99	1.0	0.95	0.93
c3tst1	36	0.89	0.89	0.99	1.0	0.86	0.87	0.63	0.9	1.0	0.99	0.92	0.91	0.39	0.39	0.9	0.89	0.99	1.0	0.95	0.8	0.95	0.7	0.99	1.0	0.92	0.94	0.94	0.94

Fig. 5. Averages for every selection option on every quiz. Topic C3 is *Testing Exceptions*

quiz	score	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	1
b3cov2	40		1					1			1						1	1		
a1suda1	39			1		1	1			1	1									
b1ps22	40	1	1	1	1							1				1		1		
b4df2	40			1								1					1			

Fig. 6. Graded answers for every quiz taken by a student

B3: Classes, Objects, Variables	**wrong**	**right**
how many methods does a class have	19	404
how many variables are in scope	114	307
whether the original object reference shows changes if a duplicate reference alters an object	80	343
how many objects are active	18	405
how many fields does a class have	8	415

Fig. 7. How many students are passing/failing a certain learning outcome

the previous semester to see whether the quizzes benefited or detracted from student learning.

Course Evaluations: We looked at students' qualitative feedback on course evaluations administered by the Faculty of Science, scanning through for positive, negative, and constructive comments that would help us address our questions.

Quiz Data: We analysed quiz answers to better understand students' achievement of learning outcomes. We then examined our experience in terms of:

- **Student Outcomes:**
  - Student experience
  - Pedagogical success
- **Instructor Outcomes:**
  - Organizational concerns
  - Pedagogical insights

VIII. Student Outcomes—Student Experience

We looked specifically at whether students remained on track with the course material, and whether they were satisfied with the non-traditional feedback for quizzes. We also looked at how many quizzes students took.

A. Students kept on track

Survey Question	Agree	Neutral	Disagree
Weekly quizzes provided me with useful feedback about how I was doing in the course (n=256)	66%	18%	17%

According to survey results, students mostly felt that quizzes helped them determine how they were doing in the course.

In our end of semester survey, we asked the students whether or not they agreed with the following statement: *Weekly quizzes encouraged me to study more consistently than I might otherwise have done*. Out of 256 respondents, 75% agreed, 12% were neutral, and 13% disagreed. Course evaluation comments echoed this assessment.

Students in the focus group also noted that the quizzes helped them stay on track with the topics:

Student 3: The quizzes were helpful. They forced us to stay on top of everything every week for each quiz, instead of studying at one time for the midterm.

Student 2: I thought they were really helpful as well. When I went to study for the midterm, I found that having the quiz structure helped me maintain a good grade in this course. I was taking 6 courses this semester. The quizzes forced me to think about the material all the time.

Later they were asked: *did the quizzes make you study more consistently?*

Student 2: It made me work with the material more than it made me actually study.

Student 1: Not really.

Student 3: I would go over some of the practice questions beforehand to see what kind of questions they could ask.

Student 4: I would review before the quizzes, which is something I would not normally have done. The quizzes motivated be to revisit the material and make sure I understood it.

Student 5: If you keep up the quizzes are helpful, if you don't keep up, they were frustrating.

Because the course materials are in video format, we were able to track, for part of the course, when students watched videos. We expected to see that students watched videos somewhat prior to quizzes, but may have had to revisit them prior to the midterm and final. We only have viewing data for up to the midterm; it is depicted in Figure 8. Two bright arcs and one bright vertical bar are visible. The first arc's dates align with views prior to lecture. The second arc's dates align with views prior to quizzes. The bar aligns with views prior to the midterm. It's difficult to see in black and white, but in the full colour version of the chart, we can see that the brightness of the pre-midterm bar is not as bright as the spike of first video viewings at the beginning of the class, suggesting that while everyone was watching the first videos at the same time, there was no similar spike in pre-midterm viewing. Instead students spread out their viewing prior to lecture and quizzes.

We also have anecdotal evidence that students did not overwhelm the office hours prior to the midterm, and we did not see an undue spike in questions on the discussion forum

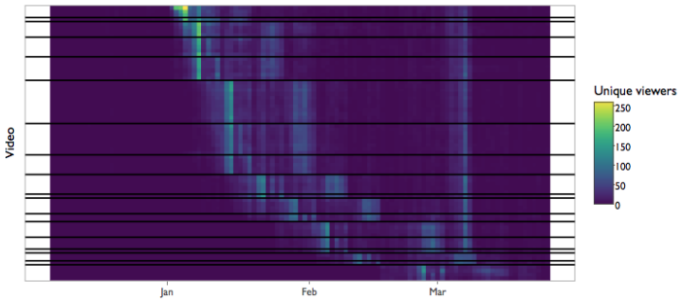

Fig. 8. Videos viewed over time for the first half of the course. Horizontal lines represent topics in the course, with rows in between representing videos within the topic. The horizontal axis is time, from the start of the course in January to just after the midterm in March. A bright spot on the row indicates a watched video—the brighter the mark, the more people watching at that time.

prior to the midterm.

Focus group comments, survey responses, and the video data show that students were kept on track during the semester. If they fell behind, their quiz performance let them know what they needed to catch up on.

B. Students wanted lecture and quizzes to be closer together

Quizzes were given two weeks after each topic was covered in lecture. In the week in between, students worked independently on an assignment that reinforced a cluster of topics. In our post-semester survey, we asked students whether the learning sequence was too long, too short, or just right. 252 respondents answered this question. Only 4% of respondents felt it was too short. While the most common response was just right (54%), a substantial percentage of respondents felt it was too long (43%). Students in the focus group also mentioned the sequence was too long. One student in the anonymous course evaluations offered the explanation that the distance between the lecture and the quiz made them feel like they were working simultaneously on too many topics (one in lecture, one in practice, and one on quizzes). A member of the focus group agreed: *...the quizzes and the videos are too spaced out. You are basically keeping track of 3 different topics at once*. In the subsequent offering we adjusted the course such that we gave quizzes the week after lecture, and the students work on the related assignment during the same week as the lecture.

C. How students felt about feedback instead of returned quizzes

We were particularly interested in whether students were satisfied with receiving back grading messages as opposed to their actual graded quiz. It was abundantly clear from student in-person comments that their favorite option was to get their own quiz back, graded, to use as a study guide for their upcoming quiz and future assessments. But since we felt we needed to keep the quizzes from being distributed, we asked students how they would feel about other options.

In the student focus group, all students agreed with this student's statement: *the feedback we did get was better than nothing, but it fell short of what we would have had if we got the actual quiz back*. This shows that there is still work to be done on getting specific feedback to students. We made the change of giving a realistic practice quiz to students. This allowed students to self-test and identify specific questions about which they had misunderstandings as either preparation for the initial quiz or retakes.

D. Some students took a lot of quizzes!

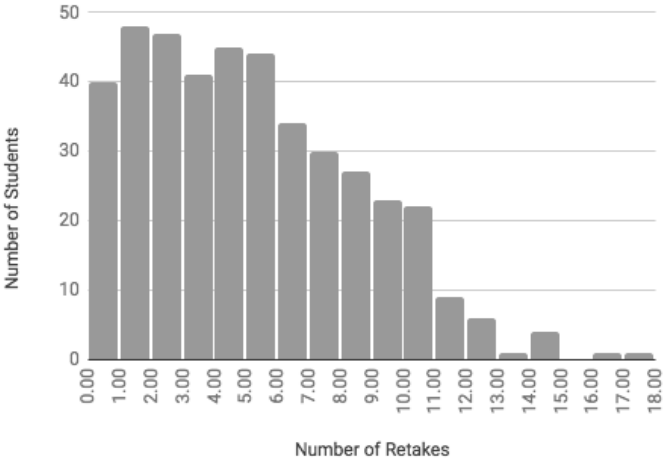

Fig. 9. Number of retakes taken by number of students. The total number of quizzes was 12173 at the end of the semester.

Roughly 40 students took every one of the 23 quizzes exactly once and passed it. At the other end of the range, some students made over 50 attempts to pass the quizzes, meaning they took many quizzes at least twice. The distribution of quizzes taken is shown in Figure 9. By the end of the semester the students had collectively taken over 12K quizzes.

Our experience contrasted with other offerings of Mastery Learning in that, in other examples, a big stated gain was that students *got to choose their own grade*, meaning that they could stop at some point if they felt they had achieved all they wanted to, or reached their own grade goal for the course (notably [4]). In other setups, it seemed that students would at some point choose not to take optional assessments. In our model, however, students overwhelmingly took every quiz at least once. Only 16 students missed a single optional quiz!

A large difference between our approach and other approaches (notably [6]) was that we still had a midterm and final exam, and passing the final exam was required for passing the course. Students knew that they could perhaps pass the final knowing only the required quiz topics, but did not know how the remaining points would be distributed. Survey responses on grade expectations (shown in Figure 14) clarify the picture: in this group of primarily Computer Science majors, roughly 80% of the students wanted to achieve over 80% in the course, and roughly a third of the students wanted an A+. The best bet for a student to get a high grade would be to get as many points as possible at whatever time they were offered, hedging their bets in case of a poor showing on the final.

IX. Student Outcomes—Pedagogical Success

We compared students' performance to that of the previous semester along various dimensions. Our findings are affected by several threats to validity:

- We often see small shifts in averages and exam performance from semester to semester so within a few points, differences are potentially just due to chance.
- We introduced videos for every topic in the course. This meant that students who did not attend class could still get classroom-like instruction. Class-time was then somewhat inverted, with us presenting some instruction, but also working through deeper problems in lecture time.

Comparing how well students learned Type Substitution rules: We asked the same type substitution question from the prior semester's midterm on this semester's midterm. We slightly changed the question wording and the names of the types, but kept the task the same. These changes were made so that those who had seen the prior midterm would not recognize the question. This topic is one that students were quizzed on (type hierarchies). At the time of writing the midterm, most students had passed the quiz, either on the first or subsequent attempt. In the prior semester, the average grade for that question was 49%, whereas in this semester the average grade was 65%. This jump is outside the range of the typical variation in performance. It is difficult to isolate whether the difference was due to the quizzes or the videos, but we believe the quizzes were important: students who came to us to review failed quizzes said they had watched the videos and felt confident in their knowledge, even though they had come away with a basic misunderstanding. Not passing the quiz forced them to learn the material correctly.

Comparing how well students learned to draw sequence diagrams: In prior semesters, sequence diagrams had notoriously been one of the most difficult topics for students, with students achieving low average grades on sequence diagram exam questions. We had moved the topic to the third year Software Engineering course, and performance was still poor. Because there was a video for teaching sequence diagrams, we reintroduced it in this semester in our SC course, and assessed it with a non-required quiz. 14 students opted not to take the quiz, and 72 had not yet passed the quiz by the final exam (the cut-off for retakes).

Performance on Sequence Diagram (SD) Final Exam Question...	Average
...overall	85%
...if they passed the SD quiz on first try	90%
...if they passed the SD quiz on subsequent try	85%
...if they failed the SD quiz	75%
...if they never took the SD quiz	54%

Fig. 10. Performance of students on the Sequence Diagram question on the final exam, grouped by their earlier performance on the Sequence Diagram quiz.

Student outcomes on this question (shown in Figure 10) show us that those who attempted but did not pass the quiz outperformed those who never took the quiz. This could be due to several factors, including that students who didn't take the quiz may have checked out of the course at this point. But the difference between students who eventually passed the quiz and those who did not at least tells us that persistence pays off.

Comparing how well students learned to implement the Iterator pattern: The last two topics in the course (Basic Iterator, where students make a class iterable, and Advanced Iterator, in which students make their own iterator) were not quizzed because of timing issues. In the final exam for both semesters, students were asked to implement an iterator that iterated over two lists in a similar way. The average for this question in the prior semester was 50%, and the average for this semester was 57%. This jump is within the range of variance between semesters.

What we see overall is that students fared comparatively better on those topics they were quizzed on, whereas for the final topic upon which they were not quizzed, there was no clear difference in student performance.

X. Instructor Outcomes—Pedagogical Insights

We reflected on what we were able to learn, given the quiz data we had collected. To assess the efficacy of the quizzes, we looked at the relationship between quiz performance and exam performance. To see whether we could glean insights into teaching that we could apply in subsequent semesters, we looked at the quiz outcomes topic-by-topic and also within-topic outcomes.

A. Were quizzes predictive of final exam scores?

In addition to comparing how specific quizzes related to specific exam questions (shown in Figure 10), we compared the number of incorrect selections on quizzes to final exam scores, to see whether there was any predictive power in quiz results—did students who struggled on the quizzes also struggle on the final exam? We also wanted to look at the efficacy of using multiple choice questions to assess student learning of software design topics. The comparison is shown in Figure 11.

Those with few incorrect selections also achieved high final exam scores. That's not to say students who performed nearly perfectly on the final also did perfectly on the quizzes: there is considerable variation at the top. Further down the scale, the results are more diffuse with a few notable outliers[3]. This chart clearly tells only part of the story and a deeper analysis is needed to understand the different learning patterns for students at the top and at the bottom of the range. The fact that the relationship is not perfectly linear is somewhat encouraging—it suggests that students who do not pass some quizzes can still do well on the final exam!

[3]The outlier in the lower left quadrant was a student who took no non-required quizzes and so had fewer incorrect selections.

We were concerned that the multiple choice nature of the quizzes would not accurately measure students' understanding of concepts. Multiple choice questions pre-chunk the problem. To mitigate that, we tried where possible to design quiz questions that involved performing a task (such as drawing a flowchart), and that had response options that refer to features that would be in their solutions (such as a connection from line 1 to line 3). The correlation between incorrect selections and the final exam was -.69, suggesting that multiple choice quizzes were an acceptable proxy for long-form written questions.

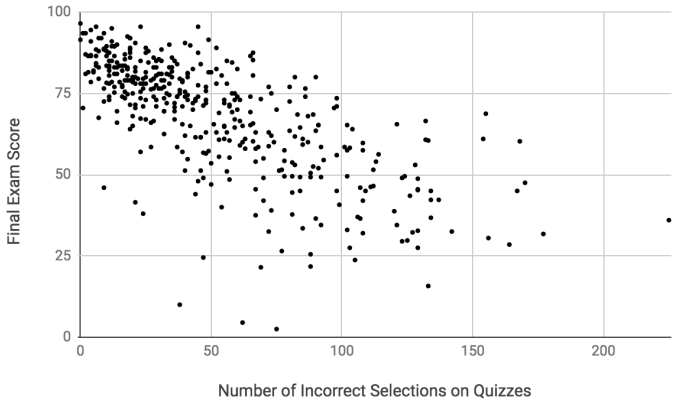

Fig. 11. Final exam scores versus number of incorrect quiz answers.

B. We were able to spot fine-grained points of difficulty to drive future course improvements

The rule that students had to retake the required quizzes until they passed meant that we were able to track in depth which core topics our students found most difficult. In the prior model, we would have seen students get a question right or wrong on a midterm, and may have looked at the averages for certain questions to reflect on whether the topics were easy or difficult to grasp. With the repeating quizzes, however, we were able to see how many students were successful on the first try, then how many were successful on the second try, and third, etc.

This data allowed us to see the relative difficulty of each topic. Looking at the maximum and average number of tries for each topic gives us a cursory sense that B3 (Classes, Objects and Variables) and A7 (Abstract Classes and Overriding) are topics that students needed many tries to pass (Figure 12). This makes sense conceptually, since most students in this course have a functional programming background with little to no former experience with imperative code or Object-Orientation.

Using the learning outcomes count view (Figure 7), we were able to look at the specific learning outcomes tested on each quiz to see where students were having trouble. Students could pass a quiz without getting every selection correct (they were allowed two incorrect selections). Each concept covered more than two selections, so students had to get each concept at least somewhat right. But we can look back and see which concepts students continued to get somewhat wrong. For B3, 114 students out of roughly 500 students were not perfectly able to distinguish which variables are in scope, and 80 were unable to correctly identify that an object changed if it was altered via a second reference. For A7, 111 students were unclear that you were able to call private methods within a supertype when a call is made to the subtype but delegated back to the supertype (for instance, if method `foo` is called on an object of type `SubClass`, but `foo` is defined in `SuperClass`, is `SuperClass.foo` allowed to make calls to other private methods inside of SuperClass). 71 were not clear whether you were allowed to override private methods. Given this data, we may, in future offerings, dedicate more resources to clarifying these concepts, and perhaps isolate those concepts in a separate quiz.

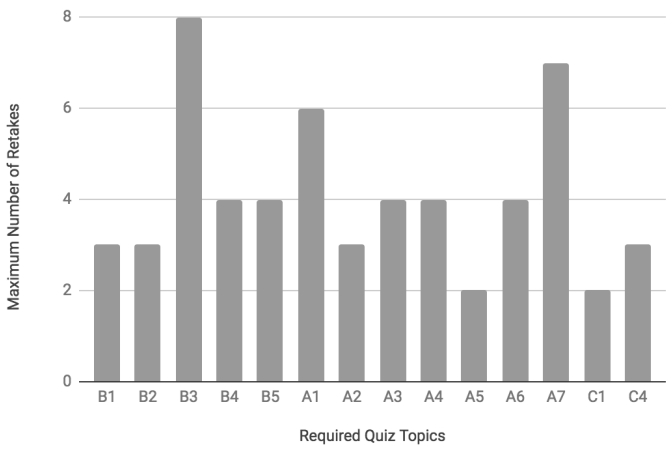

Fig. 12. Maximum number of retake tries for each required topic

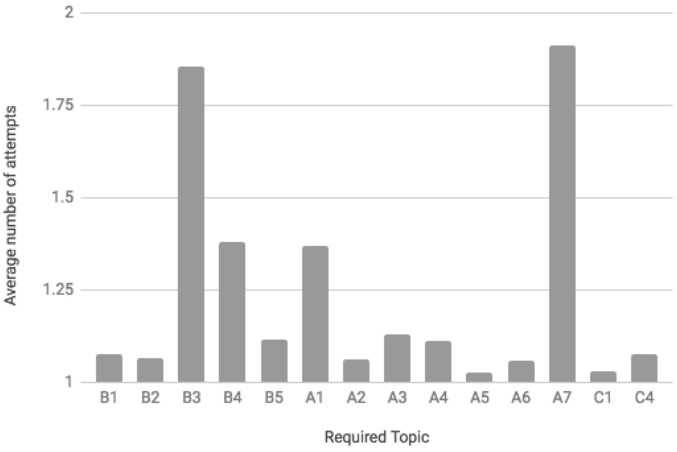

Fig. 13. Average number of retake tries for each required topic

We were able to use this insight to immediate effect: using fine-grained quiz information, we focused exam review material on those concepts that students found challenging. The focus group reflected that the review session was very

useful. We also looked for longer term pedagogical lessons: the peaks in Figures 12 and 13 for the B3 and A7 retake counts suggest that the class time spent on those topics could be expanded, and perhaps the concepts on those quizzes could be broken up into finer-grained topics.

Expected range	Term-start (n=212)	Term-end (n=212)
90-100	37%	30%
80-89	46%	49%
70-79	15%	14%
60-69	1%	4%
50-59	0%	3%
< 50	0%	2%

Fig. 14. Percentage of students by hoped for grade range

C. Topic Burndown: How Quickly were Topics Acquired?

Using the data from the quizzes, we are able to reflect on how quickly each of the topics was acquired by students on aggregate (shown in Figure 15).

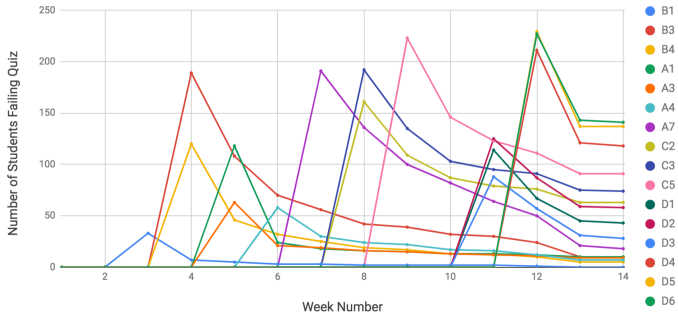

Fig. 15. Topic Burndown: Number of students failing quizzes each week. Initial peak for each topic indicates the first week in which that topic was offered.

This data allowed us to see which quizzes people initially did not pass, and which remained troublesome in subsequent weeks. It gives a longitudinal view to the quiz retake averages overall. For instance, for B3 (Classes, Objects, Variables) and A7 (Abstract Classes and Overriding), we can see that the slopes are much shallower than that of A1 (specifying methods), suggesting that correcting misunderstandings about A1 was faster than correcting the cognitive model for B3 and A7. This makes sense intuitively: A1 is a convention for specification, which could apply to any language, including the functional language students had seen in the prerequisite for this course. However B3 and A7 required understanding new computational models. The quiz feedback for the topics with shallower recovery may need to be richer, to better help students correct their misconceptions.

XI. Instructor Outcomes—Organizational Issues

We wanted to examine the organizational issues related to applying Mastery Learning to a large, multi-sectioned course. We looked at startup costs, TA grading practices, and cheating.

A. Startup costs were high

Development of the initial infrastructure required significant effort. The grading script and visualisation generator took a few iterations to get right. We created over 90 quizzes and their associated learning outcome messages, which was a large time investment that required careful attention to detail: because students were not seeing their original quiz, they could not dispute their grades in the typical way—so we had to be very sure we did not have typos in the rubric or in the messages they received. If we did, students could be told that their answers were incorrect when they were correct, or the reverse.

We have now offered the course a second time. Because the quizzes and infrastructure were in place, we found that, in terms of instructor time, the course is comparable to other, more traditional, courses.

B. TAs did not like failing people

We compared the number of quiz failures between written and verbal quizzes, and saw that there was a large difference: there were hardly any failed verbal quizzes. This suggested that unless a student truly had no idea what they were talking about, or had not done any of the assignment upon which verbal discussions were based, the TA would give the student a passing grade. This was regardless of the complexity of the topic. This reassured us that we made the right choice to slightly cover verbally quizzed topics in the written quizzes on related topics. This gave us the best of both worlds: students got practice talking about their code, and received formal feedback about whether they solidly grasped the mechanics of a concept.

C. We did not notice much cheating

As far as we know, no paper versions of the quizzes were lost. Because quizzes were administered in labs staffed with 4 TAs, students were under close scrutiny, and were never caught taking photos of their quizzes or, except in one case, communicating using their phones. In that one case, the TA identified and stopped the behaviour, and the student was given a fresh (different) quiz to write. In one case TAs identified two students who were sitting suspiciously close to one another, and the TA reported that one student had been copying the answers from the other's computer screen. We looked at the results for these students, and they had indeed entered the same answers, but since they were taking different versions of the quiz, one student passed and the other (the copier) did not.

We also looked for a learning effect across the week. Since we were using the same 3-4 versions of the quizzes throughout the week, we were concerned that by the Friday sitting, students may have memorized the answers to every version of the quiz. We noticed that some students, upon completing the quiz, immediately wrote down the code from the question. They would then bring the code to the TA to ask them to go over what would have been the correct answers. This practice caused us concern that, since quizzes for a topic varied only slightly, we would see a learning effect across the week. But in fact, we saw no pattern of improved averages or pass/fail

ratios throughout the week. This could be because the code on the quiz variants were subtly changed so that re-reading one's own quiz code was needed, and even if the code was identical, students couldn't be sure that the options were presented in the same order. Thus, to cheat on the quizzes, students probably had to actually learn the learning outcome.

As a result, we felt comfortable giving realistic sample quizzes so students could practice against the kinds of questions they would see on the quiz. Given what we saw, we don't believe that seeing a similar question helps them memorize the answer without learning the material.

XII. Conclusions

We applied Mastery Learning with a heavy reliance on auto-grading and data visualization, in a class with roughly 450 students. Compared to the prior semester's offering of the course, we found that students performed better in questions on fundamental learning outcomes on the midterm and final exams. They also kept up to date with the material and expressed satisfaction with the learning approach. It also benefited instructors: the data-heavy approach enabled us to track student competencies in detail and identify students who were struggling early in the semester. It also afforded us insights for future course improvements, helping us identify topics that students found complex and that may need to be teased apart more in the future. Auto-grading and data visualization tooling was essential. We would not attempt micro-assessments at this level of granularity without that support. Using a quiz feedback system as opposed to handing quizzes back seemed to curtail cheating and afforded crucial quiz reuse.

We believe that a software engineering course at any undergraduate level could use this technique, regardless of size. We believe that this approach scales indefinitely, though extent of individualized feedback is dependent upon TA availability. We have a number of findings that we believe are generalizable to other courses:

- The traditional Mastery Learning benefits can be obtained at scale using the data-centric and auto-graded quiz approach. Our quiz and skill-acquisition tracking framework is course agnostic, taking as inputs a class list, quiz rubrics, and feedback messages. As such, after the initial work of creating those elements, it is a feasible, low-cost, sustainable pedagogical approach.
- Even courses that provide individualised feedback could benefit from this approach: TA hours can be shifted to in-person assessments—they no longer need to hand-grade submissions.
- Not returning quizzes and instead giving more general feedback was a satisfactory approach to afford reuse of quizzes throughout the semester and across offerings. However, wording the feedback to best support student learning requires care and likely extensive iteration. Feedback combined with a realistic practice version of the quiz can mitigate student confusion about which questions they got wrong.
- For those concerned that multiple choice quizzes would be an inadequate proxy for traditional midterm and final exams, our findings suggest that there is actually a strong linear relationship between performance in the quizzes and performance in the final.
- Granular data on student performance provided by the multiple choice micro-quiz format affords analysis of how quickly skills are acquired, and also provides the potential for profound pedagogical insights.

XIII. Acknowledgments

We would like to thank Paul Carter and Ali Madooei, who co-taught the course during the semester in which the Mastery Learning approach was first employed, and who provided valuable feedback on its application in this context. We would also like to highlight the contributions of Trey Schiefelbein, who worked out many of the logistics of deploying the physical quizzes.

XIV. Appendix A: Topics

BASICS (all required)
B1. Program Structure
B2. Methods and Calls
B3. Classes, Objects and Variables
B4. Data Flow
B5. Conditions, Loops, Execution Flow and Debugging
ABSTRACTION (all required)
A1. Specifying and Using a Data Abstraction
A2. Testing a Data Abstraction
A3. Implementing a Data Abstraction
A4. Types, Substitution, Interfaces
A5. Multiple Interfaces
A6. Extends, Override, Super
A7. Abstract classes, Overloading
CONSTRUCTION (C1, C4 required)
C1. Throwing exceptions
C2. Unchecked exceptions and exception hierarchies, assertions
C3. Testing exceptions
C4. Extract design hierarchy, associations
C5. Extract sequences
DESIGN
D1. Coupling and Cohesion
D2. Refactoring
D3. Liskov Substitution Principle
D4. Composite Design Pattern
D5. Observer Design Pattern
D6. Java Observer Pattern
D7. Not quizzed: Iterable and Iterator

References

[1] http://www.cs.ubc.ca/~ebani/masterylearningframework.html
[2] Bloom BS. Learning for Mastery. Instruction and Curriculum. Regional Education Laboratory for the Carolinas and Virginia, Topical Papers and Reprints, Number 1. Evaluation comment. 1968 May;1(2):n2.
[3] Sophie Engle and Sami Rollins. 2013. Expert code review and mastery learning in a software development course. J. Comput. Sci. Coll. 28, 4 (April 2013), 139-147.
[4] Noel LeJeune. 2010. Contract grading with mastery learning in CS 1. J. Comput. Sci. Coll. 26, 2 (December 2010), 149-156.
[5] McCane, B, Ott, C, Meek, N. and Robins, A., "Mastery Learning in Introductory Programming," Proceedings of the Nineteenth Australasian Computing Education Conference, ACE '17, 2017.
[6] Wrigstad, T., Castegren, E. "Mastery Learning-Like Teaching with Achievements". SPLASH-E 2017, October 22-27, 2017, Vancouver
[7] New pilot initiative could change how U of T's Introduction to Computer Programming is taught https://www.utoronto.ca/news/new-pilot-initiative-could-change-how-u-t-s-introduction-computer-programming-taught