We’ll Make Modelers Out of 'Em Yet: Introducing Modeling into a Curriculum

Eric J. Rapos
Department of Computer Science & Software Engineering
Miami University
Oxford, Ohio, USA
rapose@miamioh.edu

ABSTRACT

The prevalence of model-driven software engineering in industry combined with a personal interest in the topic led to the conception of a new course aimed at introducing students to topics related to modeling. This paper presents a retrospective examination of this course, including: a course overview, intra-student grade comparisons on topic-centered assessments, and student feedback regarding course topics and implementation. The paper provides sufficient detail of the course offering such that, if desired, readers could offer a course with similar goals, outcomes, and structure. Finally, specific lessons learned are presented in hopes of enabling future improvements to the course and as a warning to other academics should they begin to offer similar courses.

CCS CONCEPTS

• **Social and professional topics → Software engineering education**; • **Software and its engineering → System modeling languages**; *Software notations and tools*; Unified Modeling Language (UML);

ACM Reference Format:
Eric J. Rapos. 2018. We’ll Make Modelers Out of 'Em Yet: Introducing Modeling into a Curriculum. In *ACM/IEEE 21th International Conference on Model Driven Engineering Languages and Systems (MODELS ’18 Companion), October 14–19, 2018, Copenhagen, Denmark.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3270112.3270113

1 INTRODUCTION

In the previous academic year (2017-2018) I offered a course on model-driven software engineering in my department for the first time. This paper describes the motivations for this course, the content and outcomes of the course, specific data from the course (student grades & feedback), and conclusions about the offering.

1.1 Motivation

As a member of the model-driven software engineering (MDSE) community, I have an interest in the expansion of, and buy-in to, MDSE activities. The importance of teaching modeling has been presented by Hamou-Lhadj et. al. [7], indicating that the recent paradigm shift in development has, and must continue to be reflected in teaching. As a new faculty member at an institution with minimal exposure to modeling it became evident that to attract students to my research group it would first be necessary to expose them to the topics of software modeling, and MDSE. To this end, I sought departmental approval to offer a course on modeling and was given approval to introduce it as a *special topics course* for a trial run of the material. I felt it was worthwhile to share my experiences in introducing modeling to an undergraduate curriculum with others who may be interested in doing the same. The goals of this paper are to provide a retrospective look at what went well, what did not, and some lessons learned along the way.

2 BACKGROUND & RELATED WORK

2.1 Model-Driven Software Engineering

Model-driven software engineering (MDSE) - sometimes referred to as model-driven engineering (MDE), model-driven software development (MDSD), model-driven development (MDD), and other variants - is the process of developing systems where the primary artifacts are usually models of the system or where modeling and abstraction play a substantial role in the engineering and development of the system. Kent [8] succcinctly describes early views of MDE and the relation to the Object Management Group’s model-driven architecture (MDA) [13].

2.2 Teaching MDSE

The instruction of MDSE has increased in recent years and as a result numerous academics have shared their experiences and in papers similar to this one. While MDSE has gained popularity, its instruction is lacking as students are not adequately prepared for the tools and skills required, and the challenges they will face.

Kuzniarz and Staron present their best practices for teaching UML based software development [9]. Having first discovered this work after completing my offering of an MDSE course it is interesting to see both the overlap as well as what was missing from my course. Clarke et. al. present their experiences with teaching MDSE in a software design course [3]. Their experiences differ as the course had prerequisites of a graduate course in software engineering which indicates more experience than the students in my course. The other difference is that Clarke et. al. detail the inclusion of MDSE into a course on traditional software design, whereas this paper focuses on an entire course specifically focused on MDSE.

Stephan presents findings on the challenges of incorporating modeling into Agile software engineering courses [14]. Specifically, Stephan relates the challenges to the best practices presented by Kuzniarz and Staron [9]. While Agile is not the focus of the course discussed in the paper a lecture was included which discussed the principles of agile modeling [1]. This inclusion was made based on recent focus on Agile methods in software development and may feature more prominently in future offerings.

3 COURSE OVERVIEW OF CSE 470E / 570E

Wanting to provide sufficient knowledge to the students I focused on areas that were most familiar to me and tools that I have had experience with in the past. The following were chosen for the course: meta-modeling, the Eclipse Modeling Framework (EMF), modeling as a primary artifact, behavioral modeling, Papyrus-RT, Simulink, and model-based testing. The course was then rounded out with guest talks on MDSE in Academia, MDSE in Industry, and some administrative lectures for projects and presentations.

3.1 Course Learning Outcomes

For all courses at my institution, instructors and/or the department must provide *Course Learning Outcomes*. These outcomes describe to students the main goals of the instructor and the topics they will come out of the course knowing. The following are the top level *Course Learning Outcomes* for CSE 470E / 570E (sub learning outcomes exist but were ommitted due to space constraints):

(1) Explain key differences between standard software engineering and model-driven engineering practices.
(2) Explain and be able to create meta-models and resulting instance models for typical software systems.
(3) Explain the code generation process.
(4) Explain dynamic/real-time modeling.
(5) Explain validation using models as primary artifacts.
(6) Demonstrate proficiency in creating simple models in current modeling technologies.

3.2 Deliverables

The deliverables were a collection of labs (small introductory assignments which begin in class), assignments (individual work larger in scale than labs), a group project (four phases of work that expand upon assignments), online reading quizzes (open book online quizzes), and exams (a midterm and a final). There was also a project presentation for all students and a research paper for the graduate students in the course. Table 1 provides a breakdown of the deliverables for the course and the weighting of each for both undergraduate and graduate students.

Table 1: Course Deliverables and Weights

Deliverable	Undergraduate %	Graduate %
Quizzes (5)	5%	5%
Labs (8)	8%	8%
Assignments (4)	24%	20%
Project (4 Phases)	34 %	28%
Project Presentation	4 %	4%
Research Paper	-	10%
Exams (Midterm & Final)	25%	25%
Total	100%	100%

It is important to note that the course generally followed the formula of a topic being introduced during a series of lectures, followed up on in one or two labs, presented again in an assignment, and one more time in the course project. As such, it is possible to track a particular topic through the course from lab, to assignment, to project. These groupings are presented in further detail as part of the Case Study in Section 4. In order to understand the linking of labs to assignments to projects, as previously described, it is first important to identify the topics for each of these particular deliverables. The following is a breakdown of these categories:

- 8 Labs: Meta-Modeling, EMF, Models as Primary Artifacts, Behavioral Modeling, Papyrus-RT, Simulink, Model-Based Testing, & Advance Model Implementation
- 4 Assignments: Meta-Modeling & EMF, Models as Primary Artifacts, Behavioral Modeling, & Model-Based Testing
- 4 Course Project Phases: Meta-Models, Behavioral Models, Papyrus-RT Implementation, & Model-Based Testing

3.3 Course Textbook

The text chosen for the course (based on colleague recommendations) was Model-Driven Software Engineering in Practice (Second Edition) [2], published in 2017, written by Marco Brambilla, Jordi Cabot, and Manuel Wimmer.

3.4 Technologies Used

- **Draw.io** [4]: A simple web-based drawing tool useful for initial modeling of systems.
- **Modelio** [11]: A desktop modeling tool used to create UML models in the course.
- **Eclipse Modeling Framework (EMF)** [5]: EMF played a large role in the introduction of meta-modeling through the use of its internal meta-modeling language Ecore.
- **Papyrus-RT** [6]: Papyrus-RT is an open-source implementation of UML-RT [12].
- **Simulink** [10]: The final tool introduced to students is MATLAB's modeling language Simulink. Simulink, which is a graphical modeling tool that is data-driven and is used heavily in embedded systems due to its simulation support.

4 CASE STUDY

My department offers undergraduate degree programs in CS as well as SE, and a Masters Degree in CS. The course was open to students meeting prerequisites from any of our students as well as students from a Computer Technology degree at our satellite campus. 26 students in the course consented to take part in the research for this paper: 20 undergraduate (16 CS, 2 SE, 2 CT) and 6 graduate. This case study includes the grades, as well as qualitative data in the form of anonymous, optional, weekly, feedback from students to highlight student perspectives on content and topics.

4.1 Dataset

For the purposes of this paper, student grades from the 26 students were collected for the 8 labs, 4 assignments, 4 project phases, a project presentation, midterm and final exams, and the final course grade. The data in its raw form is presented in Table 2.

4.2 Intra-Student Performance

In order to determine the effectiveness of the chosen method of presenting topics in 3 successive deliverables (lab, assignment, project) it was helpful to perform an intra-student analysis. The idea being that in general there should be an upward trend as the student gains further exposure to a particular topic. It was expected that a student would show initial understanding (a middling grade) of a topic in the lab, a deeper understanding (a higher grade) on the assignments, and mastery (highest grade) on the project. As such, this section tracks grades for 7 topics (based on the first 7 labs) through the remainder of the course for each student and plots these lines on graphs to examine the trends.

Below are the progressions of each deliverable:

- Lab 1 → Assignment 1 → Phase 1
- Lab 2 → Assignment 1 → Phase 1
- Lab 3 → Assignment 2 → Phase 2
- Lab 4 → Assignment 3 → Phase 2

Table 2: Student Grades on Primary Deliverables

Labs								Assignments				Project					Exams			
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**P**	**M**	**F**		**Grade**
90.00	90.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	91.43	94.29	117.86	100.00	99.63	92.00	91.00		100.89
95.00	90.00	100.00	90.00	100.00	100.00	100.00	100.00	91.67	96.67	100.00	100.00	95.29	94.71	98.82	97.65	91.38	100.00	91.00		99.36
100.00	80.00	65.00	85.00	100.00	100.00	100.00	100.00	97.00	100.00	100.00	100.00	91.43	94.29	117.86	100.00	99.63	87.38	97.33		98.59
80.00	100.00	75.00	80.00	100.00	100.00	100.00	100.00	100.00	100.00	97.00	100.00	91.43	94.29	117.86	100.00	99.63	84.88	78.33		96.42
90.00	70.00	75.00	95.00	100.00	70.00	80.00	100.00	97.00	84.00	74.00	100.00	71.43	83.57	87.86	97.14	95.50	90.88	97.00		91.86
45.00	90.00	15.00	65.00	100.00	85.00	100.00	100.00	100.00	100.00	98.00	100.00	71.43	83.57	87.86	97.14	95.50	90.00	77.67		90.17
80.00	90.00	65.00	75.00	55.00	50.00	90.00	100.00	100.00	86.67	85.00	81.67	95.29	94.71	98.82	97.65	91.38	76.88	73.33		89.13
55.00	90.00	90.00	85.00	90.00	100.00	100.00	100.00	100.00	98.00	95.00	83.00	71.43	83.57	87.86	97.14	95.50	63.00	76.33		87.27
65.00	65.00	25.00	75.00	60.00	0.00	70.00	100.00	91.67	93.33	80.83	100.00	77.65	66.47	78.24	85.88	87.50	87.50	76.00		83.35
95.00	80.00	25.00	85.00	75.00	85.00	100.00	90.00	100.00	98.33	86.67	79.17	80.00	75.29	70.59	90.59	87.63	60.38	62.67		81.44
60.00	100.00	85.00	80.00	55.00	0.00	100.00	80.00	100.00	100.00	50.83	95.83	90.59	61.76	68.82	62.94	83.75	72.25	83.00		81.38
90.00	40.00	25.00	0.00	0.00	55.00	80.00	100.00	100.00	91.67	92.50	87.50	77.65	66.47	78.24	85.88	87.50	67.25	72.00		80.02
80.00	40.00	15.00	45.00	40.00	95.00	90.00	90.00	100.00	96.67	88.33	79.17	80.00	75.29	70.59	90.59	87.63	44.00	59.67		77.56
75.00	70.00	65.00	80.00	0.00	75.00	65.00	70.00	91.67	74.17	80.00	69.17	90.59	61.76	79.41	58.82	86.38	78.13	78.67		77.44
85.00	85.00	25.00	75.00	25.00	80.00	80.00	100.00	87.50	91.67	92.50	75.00	47.06	72.94	68.82	62.94	83.75	85.75	73.67		77.23
100.00	75.00	55.00	55.00	55.00	55.00	100.00	80.00	95.83	81.67	86.67	75.00	90.59	61.76	79.41	58.82	86.38	70.25	62.67		76.96
80.00	50.00	25.00	65.00	65.00	45.00	90.00	40.00	95.83	95.00	80.00	83.33	90.59	61.76	79.41	58.82	86.38	58.13	67.33		75.87
70.00	20.00	15.00	75.00	55.00	15.00	25.00	60.00	58.33	96.67	86.67	83.33	57.65	81.18	82.35	83.53	82.13	54.50	72.00		74.99
10.00	70.00	25.00	0.00	60.00	45.00	65.00	100.00	87.50	70.83	90.00	51.67	77.65	66.47	78.24	85.88	87.50	60.75	70.00		74.94
70.00	55.00	25.00	70.00	40.00	25.00	30.00	80.00	95.83	81.67	58.33	75.00	77.65	66.47	78.24	85.88	87.50	57.00	63.33		74.40
35.00	50.00	15.00	50.00	55.00	95.00	90.00	30.00	79.17	91.67	94.17	83.33	80.00	75.29	70.59	90.59	87.63	26.00	46.33		72.02
70.00	65.00	15.00	75.00	65.00	45.00	60.00	90.00	95.83	98.33	24.17	87.50	57.65	81.18	82.35	83.53	82.13	82.13	29.67		71.62
80.00	80.00	45.00	70.00	60.00	30.00	45.00	60.00	58.33	96.67	80.83	47.50	80.00	75.29	70.59	90.59	87.63	44.50	52.00		70.93
65.00	40.00	15.00	65.00	50.00	30.00	0.00	0.00	66.67	55.00	56.67	0.00	57.65	81.18	82.35	83.53	82.13	41.75	56.00		62.89
55.00	40.00	15.00	60.00	0.00	0.00	70.00	0.00	58.33	0.00	35.00	0.00	95.29	94.71	98.82	97.65	91.38	30.25	40.33		61.06
0.00	20.00	0.00	25.00	0.00	0.00	0.00	0.00	87.50	41.67	12.50	41.67	90.59	61.76	79.41	58.82	86.38	37.00	48.33		55.31
70.00	**67.12**	**42.31**	**66.35**	**57.88**	**56.92**	**74.23**	**75.77**	**89.83**	**85.40**	**77.91**	**76.11**	**79.92**	**77.31**	**85.05**	**84.69**	**89.21**	**67.02**	**69.06**		**80.12**
(24.8)	(23.1)	(30.2)	(24.8)	(32.4)	(35.3)	(30.2)	(33.3)	(13.6)	(22.2)	(23.5)	(27.1)	(13.1)	(11.9)	(14.6)	(14.5)	(5.3)	(20.4)	(16.6)		(11.48)

- Bottom Row is the course average (with the standard deviation in parentheses).
- Right Column is the student's final course grade percentage.
- P - Presentation, M - Midterm, F - Final Exam

- Lab 5 ⟶ Assignment 3 ⟶ Phase 3
- Lab 6 ⟶ Assignment 3
- Lab 7 ⟶ Assignment 4 ⟶ Phase 4

Figure 1 presents student grades for each of the 7 sections. Each sub-figure represents students' grades (%) on the lab (left), assignment (center), and project (right); the one exception being Lab 6 (Simulink) as it was not incorporated in the project.

Another interesting question is whether or not a student's performance on exams is an indicator of overall performance in the course. Figure 2 shows the students grades on the course, midterm exam, and final exam. Grades are sorted (left to right) by descending final grades, which is discussed in Section 5.

4.3 Overall Course Performance

In this section, results from the overall course performance are presented, including averages, medians, grade distributions and other interesting findings. The high-level view of the course can be captured by looking at the mean and median grades over all students in the course. These course statistics are as follows: **Mean:** 80.12% (with a standard deviation of 11.48), **Median:** 77.50%. These fall close to the initial target of 80% chosen at the onset of the course. The final course grade distributions are shown in Table 2.

5 DISCUSSION

5.1 Analysis of Findings

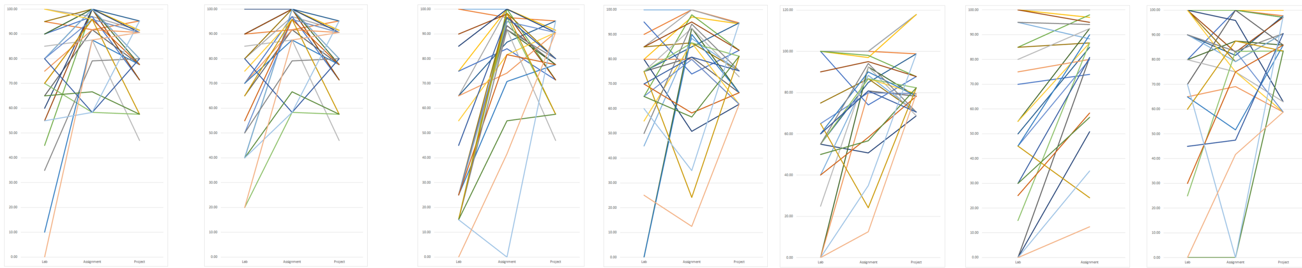
In this section, the findings presented in the previous section are discussed in detail with particular focus on the following questions:

- Is there an improvement between labs, assignments, & projects?
- Are exams a good indicator of final grades?
- What was learned from lab grades? Assignment grades? Project Grades? Exam Grades?

Figure 1 is useful in discussing whether or not the methodology of 3-fold assessment of each topic has any merit. The desired result of this methodology was that the grades would consistently increase from lab to assignment to project; in almost all cases this was not the result. Student grades did improve greatly from lab to assignment but the project grades typically declined from the assignment. Success in this method was limited only by the students' performance with the course project which is discussed in detail later.

The next area of discussion asks if exams are good indicators of overall performance in the course. While it is reasonable to expect a correlation between each of the exams and the final grade, the focus is on the strength of the correlation. Furthermore, low weighting of exams means their impact is not solely based on them being a factor of the final grade. The midterm has a correlation coefficient of **0.8043** with the final grade and the final exam has a correlation coefficient of **0.8238** with the final grade. While neither of these are extremely strong it can be reasonably stated that a student's performance on their midterm is a decent indicator of their expected final performance (given the specifics of this course).

In this course, the lab grades were less than desirable. While some students excelled, many were unsuccessful in the labs which were the initial assessment tools for each topic in the course. The mean grade across all students and all labs was **63.82%** with a median of **70.00%**. It is clear that the understanding of the labs could be improved in future offerings. While the labs introduced topics in a low stakes environment (each lab being worth 1%) the students still had difficulty grasping material. Assignment grades seem to be as a whole much better in terms of overall performance. Out of 104 (26*4) assignments, only 17 received a failing grade. Throughout the course students earned a mean grade of **82.31%** on assignments



(a) Meta-Modeling (b) Eclipse Modeling Framework (c) Model Creation (d) Behavioral Models (e) Papyrus-RT (f) Simulink (g) Model-Based Testing

Figure 1: Course Grades (%) on Labs, Assignments, and Projects for Various Topics

Table 3: Course Grade Distributions

Letter Grade	A+	A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F
Percent Range	>97%	93 - 96.99%	90 - 92.99%	87 - 89.99%	83 - 86.99%	80 - 82.99%	77 - 79.99%	73 - 76.99%	70 - 72.99%	67 - 69.99%	63 - 66.99%	60 - 62.99%	<60%
# of Students	3	1	2	2	1	3	3	5	3	0	0	2	1

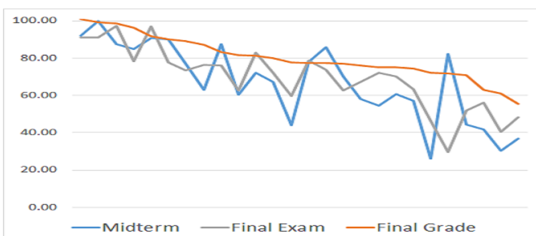


Figure 2: Midterm, Final Exam, and Course Grades

(standard deviation of 22.9); the median grade was **90.83%**. This shows that given more time to work on the assignments, the ability to discuss problems during office hours, and time to check over work, the students were able to improve understanding and performance. Project grades present a real mystery in this course. It was always the intent that the project, as the final opportunity to work on a particular skill, would be the strongest grade for each student. The mean grade across all students on the 4 phases of the project was **81.74%** (standard deviation: 13.9) with the median almost identical at **81.76%**. If the goal of the lab-assignment-project progression of grades is to be successful in the future there needs to be an adjustment of difficulty in some (or all) of the three different assessment mediums. This is discussed later in the section on lessons learned from this offering.

5.2 Student Feedback

As an initial offering it was important to solicit feedback from students as frequently as possible. A weekly survey was administered to students in the course allowing them the opportunity to provide anonymous feedback. The survey was entirely optional but students were given time each week to fill them out. There were 3 standard questions asked every week, and varying questions asked that related to the particular point in the course. The quantitative results are presented first, followed by the qualitative.

The first weekly question ("Rate your overall satisfaction with the lectures this week. (on a scale of 1-5)"); can be seen in Table 4.

Table 4: Weekly Responses to "Rate your overall satisfaction with the lectures this week." (on a scale of 1-5)

Week (Topics)	Average Score
1 (Introduction)	4.4
2 (Meta-Models)	4.0
3 (Eclipse Modeling Framework)	2.0
4 (Models as Primary Artifacts)	3.9
5 (Models as Primary Artifacts)	3.3
6 (Behavioral Modeling)	3.7
7 (Behavioral Modeling/Midterm Review)	3.5
8 (Midterm / Papyrus-RT)	3.3
9 (Papyrus-RT / Simulink)	4.1
10 (Simulink / MDE in Industry)	Not Administered
11 (Model-Based Testing)	3.6
12 (Model-Based Testing / Project)	4.3
13 (Project / MDE in Academia)	3.8
14 (Presentations / Exam Review)	Not Administered

The second quantitative result is to the question "How would you rate the effectiveness of the assignments so far in this course in helping you understand and apply the content?" (on a scale of 1-5). This question was asked in the 6th week of the course and the average response was 3.1 and likely reflected the students' dissatisfaction with the lab grades on the first several labs.

The remainder of questions are all qualitative questions, some weekly and some asked only once. These can be found in Table 5.

5.3 Lessons Learned

This section presents a retrospective look at the course taking into consideration student feedback and instructor observations. Essentially these are the lessons learned and how they can/will be addressed for future offerings of this course. For any prospective MDSE instructors keep these lessons in mind.

Lab Deadlines and Examples: One of the most common points of student feedback was the time allowed for labs and the inability to complete them on time. Initially students were given 80 minutes but this led to many students not completing labs. After reviewing the weekly feedback and student grades, I decided to allow until midnight the day of the lab to complete the work. This change mitigated many of the issues and student feedback was positive. Another concern was not being exactly sure what was expected. A

Table 5: Qualitative Question Responses

Question (Week Asked)	Selected Responses
Identify any concepts you had difficulty understanding this week in class. (All Weeks)	"The book is difficult to understand", "How to use EMF", "Statecharts", "Just small things with installing matlab", "Some concepts in Model based testing"
Please provide any additional feedback (things you liked, things you didn't like, things you want more/less of, etc.) (All Weeks).	**"The lab is too long to finish in one hour." (many similar responses)**, "I like the in-class examples and activities. I find them very helpful", "Group / class activities were helpful... it is useful to discuss concepts with classmates to solidify understanding of lecture materials.", "I expect to move very deep on one tool rather than just scratch the surface of many tools."
What are your thoughts on the course material so far? (interesting? complex? etc.) (Week 2)	"pretty straightforward", "It is interesting and applicable in today's environment", "Interesting but difficult to grasp"
What is the MOST important/valuable thing you have learned in the course so far? (Week 8)	"EMF", "thinking with more abstraction", "The concept of behavioral modelling", "How useful models can be for code generation"
What skills have you learned in this course that you find useful for your future? (Week 12)	"Modelling tools", "Reinforcing testing coverage criteria", "Generating code from models and testing using models, which saves a ton of time and money.", "Increasing level of abstraction to address and solve problems. With different levels of abstraction, communication to different sets of people (different fields) become easier."

suggestion that came up later in the course was to spend the first 10 minutes of the lab session demonstrating a solution to a similar problem so the students know the format of what is expected. This suggestion was not incorporated in this offering due to time constraints but will be in the future.

Technology Issues: Another highly reported issue with the course was the technology issues faced by many students. As the course was a late addition to the department's offerings, the software (EMF, Papyrus-RT, Simulink, etc.) was not installed on lab machines and the students needed to install them on personal devices. While in-depth tutorials were provided, students faced sufficient issues installing and configuring tools (primarily Papyrus-RT). For the next offering of the course these tools have already been installed on the lab image. Several students noted concern with using a wide variety of tools for only short periods of times rather than learning one tool in depth. While this is more of a high-level issue, it does raise a distinct lack of one single tool that implements MDSE topics in an end-to-end manner rather than stringing together tools. A solution to this may come in the form of a new domain-specific modeling language aimed specifically at teaching model-driven software engineering.

In Class Demonstrations: There was concern raised about the speed of in class demonstrations. Given an 80-minute window I opted to describe the theory behind tools and demonstrate them in one session. Based on initial feedback I began creating videos of the demos for students to follow later and this was well received. Even with the videos, students raised concerns with the speed as they wanted to be able to follow along in real time and ask questions. The proposed response to this issue is that in future offerings I plan to divide the theory and practice into two sessions: one to cover the ideas and concepts and one to work with tools and complete a small example (possibly outside of class as a tutorial).

Project & Assignment Difficulty: There is a clear disparity in the difficulty of the assignments and project in the course that needs to be addressed. Based on the 3 tiered system discussed previously, the intent is to improve student understanding from labs to assignments to the project. The decrease observed from assignments to projects could be due to the assignments not being difficult enough, the project being too difficult, or some combination of these (which is most likely). This problem can be addressed by creating assignments that are more on par with what is expected in the project, and a project that is manageable by a group of students with non-expert levels of experience.

6 CONCLUSION

For a first foray into teaching model-driven software engineering in this department, the experience was a success. By the end of the course students showed an interest in the topic and its applications to software engineering. The method of lab → assignment → project worked with relative success, and given an adjustment of difficulty, will likely show further improvement. While there is no easy fix to the 'many tools in little depth' issue presented it could be solved through the introduction of a teaching focused modeling language. I have a strong desire to continue working towards improving the course with the ultimate goal of incorporating it into our department's curriculum as a permanent course in our SE program. Considering the prevalence of MDSE in industry we want to produce successful students upon graduation. With a solid foundation, it is clear that we'll make modelers out of 'em yet!

REFERENCES

[1] Scott Ambler. 2002. *Agile modeling: effective practices for extreme programming and the unified process.* John Wiley & Sons.

[2] Marco Brambilla, Jordi Cabot, and Manuel Wimmer. 2017. Model-driven software engineering in practice. *Synthesis Lectures on Software Engineering* 3, 1 (2017), 1–207.

[3] Peter J Clarke, Yali Wu, Andrew A Allen, and Tariq M King. 2009. Experiences of teaching model-driven engineering in a software design course. In *Online Proceedings of the 5th EducatorsâĂŹ Symposium of the MODELS Conference.* 6–14.

[4] Draw.io. 2018. Draw.io - About Us. https://about.draw.io/about-us/

[5] Eclipse Foundation. 2018. Eclipse Modeling Framework (EMF). https://www.eclipse.org/modeling/emf/

[6] Eclipse Foundation. 2018. Papyrus-RT. https://www.eclipse.org/papyrus-rt/

[7] Abdelwahab Hamou-Lhadj, Abdelouahed Gherbi, and Jagadeesh Nandigam. 2009. The impact of the model-driven approach to software engineering on software engineering education. In *Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference on.* IEEE, 719–724.

[8] Stuart Kent. 2002. Model driven engineering. In *International Conference on Integrated Formal Methods.* Springer, 286–298.

[9] Ludwik Kuzniarz and Miroslaw Staron. 2005. Best practices for teaching UML based software development. In *International Conference on Model Driven Engineering Languages and Systems.* Springer, 320–332.

[10] Mathworks. 2018. Simulink. https://www.mathworks.com/products/simulink.html

[11] Modeliosoft. 2018. Modelio Home Page. https://www.modelio.org/

[12] Bran Selic. 1998. Using UML for modeling complex real-time systems. In *Languages, compilers, and tools for embedded systems.* Springer, 250–260.

[13] Richard Soley et al. 2000. Model driven architecture. *OMG white paper* 308, 308 (2000), 5.

[14] Matthew Stephan. 2017. Challenges in Teaching Modeling in Agile Software Engineering Courses. In *International Conference on Model Driven Engineering Languages and Systems - Educators Symposium at MoDELS.* 4pp.