Dysgu: A Tool to Keep Students Engaged Outside the Classroom*

Muztaba Fuad
Department of Computer Science
Winston-Salem State University
Winston-Salem, NC, USA
https://orcid.org/0000-0002-1942-1193

***Abstract*— This paper presents Dysgu, a mobile software to facilitate skill generating activities outside the classroom. Dysgu presents an innovative approach to such out-of-class activities by combining multiple dimensions of best practices from different spectrum of student learning into a coherent idea and delivering such activities with personalization and adaptation. The goal of the Dysgu system is to study the impact of allowing students to perform frequent and interactive activities outside the classroom on their learning and engagement, given that students can compare their progress with the rest of the class and the activities are smaller (in scope) with scaffolding support, and delivered via a mobile platform. Initial usability tests and software engineering quality matrices show that the software is easy to use, manage and extend.**

Keywords—Mobile learning, homework, out-of-class activity.

I. Introduction

It is fairly accepted that [1]-[2] current students are not as prepared for college work as their predecessors. This lack of preparation is worsened by the lack of practice outside the classroom. College advising guideline [3] states that students should study a specific number of hours at home for each hour of credit per week. However, it is found [2] that students seldom follow such guidelines. A possible reason [3] for the lack of study outside the class is students' increasing working hours on jobs for rising tuition cost, rising living expenses, and other socio-cultural challenges. Working more hours inevitably leads to students spending less time studying outside the class, which have detrimental effects on student's learning. Thus, it is essential to develop methods that will allow effective usage of student's available time for studying outside the classroom in such socio-economic situations.

To address this challenge, this paper presents an active learning environment, Dysgu, that enables teachers to create interactive out-of-class activities that are delivered in a structured way to facilitate greater student engagement, better practice of learned concepts, and improved student learning outside the classroom. This is achieved by providing students with personalized and active learning activities that they can do during their most convenient times using a widely-used platform (i.e., mobile), thus utilizing their time and maximizing their engagement with the course content.

II. Rational For Mobile Out-of-class Activities

Activities that faculty assigns to students to do at home are an integral part of a traditional face-to-face classroom. However, those activities have been scrutinized [4] for the lack of considerations for the individual learner's learning needs, style of learning, students' personal situations, etc. Online and distance learning provides some personalization. However, missing interactions between students and faculty in such online environment makes it challenging for improved student engagements. Therefore, a more blended approach where face-to-face classroom teaching is accompanied by interactive and personalized activities, where students can get real-time progress updates, has the potential to support students learning and engagement outside the classroom. Since mobile learning environments can deliver content anytime and anywhere, by having a mobile-based active learning environment, which adapts to student's needs, can assist faculty in guiding student's knowledge acquisition; well after students leave the classroom.

III. Dysgu

Dysgu is designed as a cloud-oriented client-server software as shown in Fig. 1. The two entities of the system-client and server; do not have any direct communication between them. All interactions between the server and client in Dysgu is delegated through cloud services. The major differences of Dysgu with a similar system are the mode of interaction, type of activities, ways it provides personalization and ability to adapt to students' needs. More on that will be discussed in section IV.

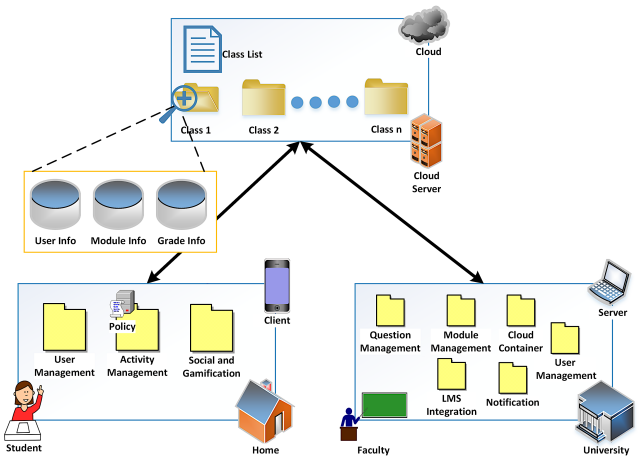

Fig. 1 Software architecture of Dysgu.

In Dysgu's software architecture, the cloud stores information about different classes. There is a collection of repositories for each of the classes. These are stored and maintained in the cloud and updated asynchronously by both entities of the software. Dysgu provides transparent cloud operations, and the users of the software (both students and faculty) do not have to worry about day-to-day management of these services. The cloud repository is used as an intermediary for the client and the server to communicate

* This research is supported by National Science Foundation grants # 1712030

with each other in complete synchronicity and transparency. Users of the system do not have to pay for any cloud service as Dysgu uses free facilities offered by the providers.

The faculty uses the server (Fig. 2) to manage the system, which includes creating and deploying a class, managing students and their credentials, creating interactive exercises, creating and deploying problem-solving modules, creating different reports, monitoring either individual student or class progress and status, etc. A client application (Fig. 3) in student device allows students to log in to the class, work on a problem-solving session, monitor progress etc. Both entities of the software are designed to execute asynchronous to each other.

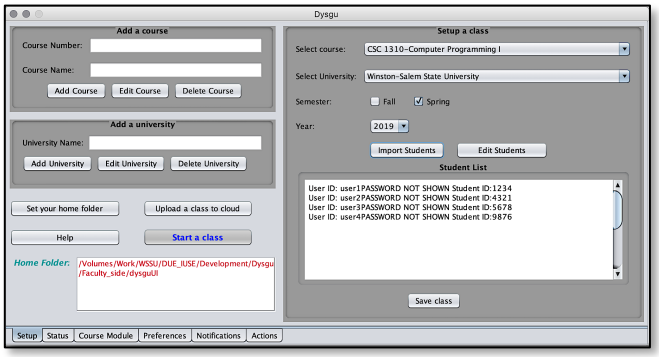

Fig. 2 Server interface.

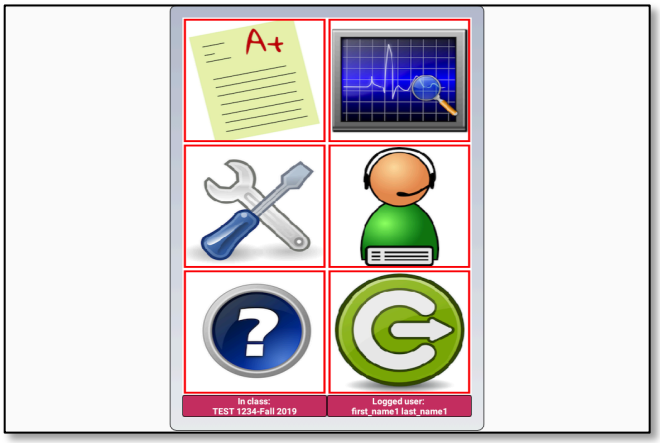

Fig. 3 Client interface.

Because Dysgu is designed to facilitate interactive problem solving, it should be used in classes which is skill generating. Dysgu comes with a set of interactive activities (which are highly parameterized), however, anyone can build and incorporate new activities in Dysgu using published API. Although Dysgu can be used with any level of college students, we envision it to be used with freshman and sophomore students, as these students usually struggle with transitioning from high school to college. Dysgu can positively affect these types of student's learning and engagement outside the classroom as junior and senior students are usually more matured.

A. Interactive Exercise

Dysgu supports interactive problem-solving activities. Interactive problems [5] allow students to interact with different steps or parts of a visual representation of the problem by selecting or touching different components of that visualization. Most such interactive problem-solving session is comprised of multiple interactive screens and students are allowed to navigate those screens. This allows students to see the effect of their interaction in real-time. The properties of an interactive activity in Dysgu are as follow:

a. There must be interactive elements (such as buttons, lists, tables, dropdown boxes, etc.) where students have to physically interact with those elements (by dragging, dropping, rearranging, etc.) to come up with the answer. Fig. 4, shows one such example where students are given a list of elements, which they have to classify into different beans. Students can drag and drop any of the items to any of the beans or rearrange them as they like. Some of the items can also have images and will zoom open when students touch them or started dragging them.

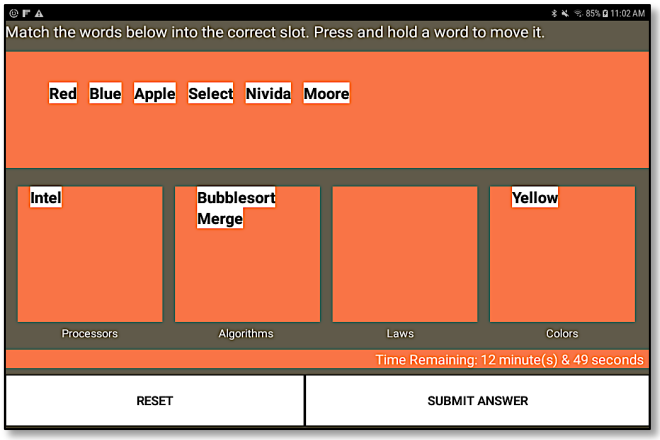

Fig. 4 Single screen interactive activity

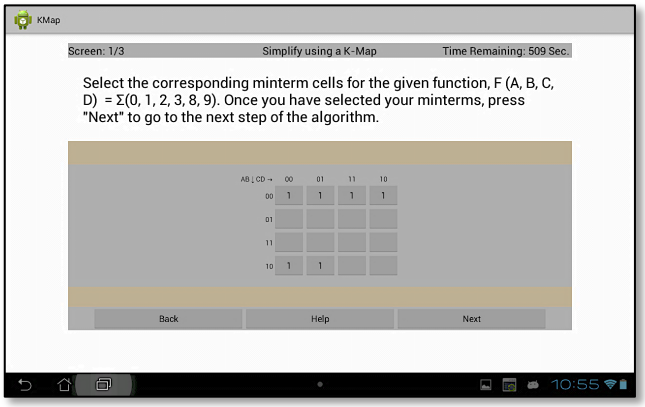

Fig. 5 Multiple screen interactive activity.

b. There can be multiple screens in the problem-solving activity, each of the screens having the same property as listed in (a) above. Additionally, each screen should have a way to traverse to the next (or the previous) screen, which allows the students to see the effects of their interaction in one screen to another. Fig. 5 shows one such interactive problem-solving activity, where students have to traverse through multiple screens to solve the problem. At any time, students can traverse any direction, change their choices, see effects of their choices and modify their choices if the traversing caused them to change their mind.

Any interactive activity is not tied to any single problem or exercise. It can be populated with any number of similar problems by feeding it with the new problem. Therefore,

faculty can utilize one such activity to test students with different problems with a different degree of difficulties. Since the client grades the student's answer and provides a set of feedback, faculty workload is not increased. These interactive activities can be developed by anyone and incorporated with Dysgu following published APIs. Additionally, to adapt to student's asynchronous lifestyle and schedule, such activities in Dysgu can be paused and resumed before the due date corresponding to the learning path and course module's policy. This allows the students to work on a problem in their own time and pace.

B. *Learning Paths*

Course modules in Dysgu is a collection of interactive problem-solving activities, organized into a list of learning paths, where each activity in a path have certain relationships. Normally, problems in a learning path have a different degree of difficulties which gradually increase from easy to hard for examining student's comprehension of certain concepts. Fig. 6 shows a typical Dysgu module. General properties of a module are as follows:

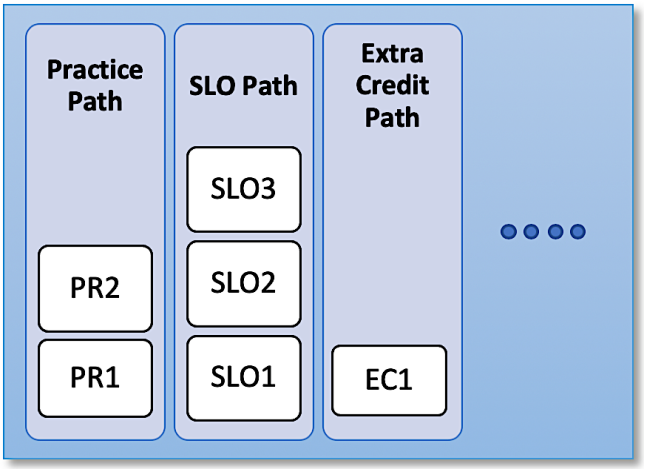

Fig. 6 Course modules in Dysgu

a. A module can have a total of five paths, each path can have five activities in it. The reason for these sizes is set considering student workload, allowable time and learning outcomes.
b. Some of the path or paths in a module can be designated as a practice path (s) to help students practice the concept with different problems. Therefore, such a practice path is not graded, however, students can see whether their answer is correct or not and can earn points.
c. Other paths can be designated as student learning outcome (SLO) paths, where students will be assessed of their learning skills. Therefore, activities in these path(s) have scores and no points (Section III.C) and the score is used to grade the student as part of the course.
d. Additionally, faculty can designate some paths as extra credit, which have problems that require additional effort from students. These paths are also not graded, but students might be assigned points depending on their performance. This lets students to try-and-fail without penalizing their grades. It is found [6] that, in such occasions, students do better in solving difficult problems and that they utilize higher order thinking.

C. *Scoring and comparisons*

Dysgu has two separate grading schemes: scores, and points. A score is assigned to any activity in a SLO path, which can only be positive and the total of all scores in SLO path (s) should add up to 100 (to make calculations easier). For successful completion of an activity, the assigned score will be given to that student, which is later used to calculate the student's overall grade in the class. Points (can be negative) on the other hand are assigned as reward of finishing an activity. For example, when a student finishes an activity first, that student will get some points assigned by the faculty. Additionally, if a student does not finish an activity in a SLO path, that student will incur negative points. Scores are utilized to calculate student's grade, whereas, points are used as currency in the system. Points acts as currency and students can use it to extend the time to solve a module or buy chances to solve difficult activities. Students are also assigned badges depending on different faculty set policy on the amount and type of score, points, times and tries. To boost student enthusiasm, Dysgu provides aggregate and statistical information about peers. For example, the number of students already completed an activity or a path or a module, the number of peers progressing toward stage *t* of a path, etc. Furthermore, Dysgu presents a progress indicator for a student in comparison to the rest of the students in the class. It shows the student score, where the student stands compared to the class, how many points the student has and what are the badges the student earned. Additionally, students can review any completed activity and compare themselves with the rest of the class. Dysgu provides statistical and summative information about the class without disclosing individual student grades to follow federal privacy laws.

D. *User Experience*

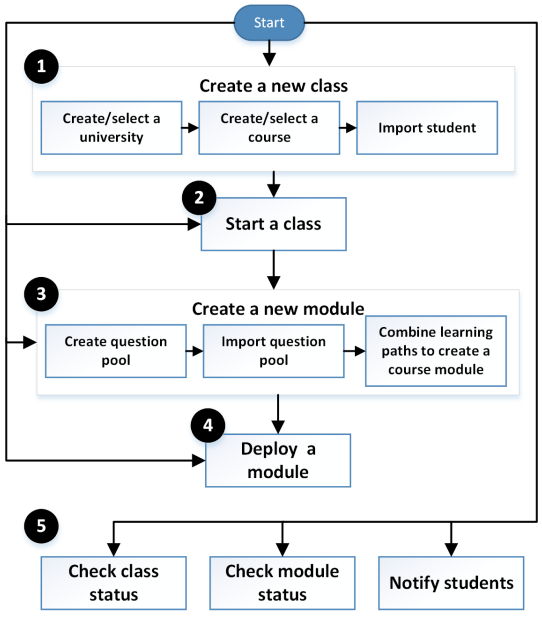

Fig. 7 Workflows in Dysgu.

Dysgu uses a workflow-based approach to give faculty an intuitive step-oriented process to use the software. A workflow usually comprises of a sequence of steps through which the faculty is guided by the software. Fig. 7 shows some of the major workflows in the server-side of the Dysgu system. For example, an initial workflow is to create a class.

To do that, faculty needs to first create (or select from existing) a university information and create (or select from existing) a course information (Fig. 2). Once those are selected, faculty needs to import student information (from Learning Management System) and select the year and semester of that class and then can save the class in the home folder for it to be deployed later. Starting a class means uploading the class files in the cloud so that students can start logging in and also to let all the client devices know that there is a new class been started.

Another important workflow for faculty is to create exercise modules for students to work on and then deploy the module to the class. As described earlier, A module is comprised of multiple smaller questions arranged in learning paths of different degree of difficulty. To formulate a module, the faculty first need to create the questions by utilizing the corresponding question generator. Fig. 8 shows a sample question generator for a specific type of problem-solving activity. Dysgu currently supports three different types of interactive problem-solving activities. However, Dysgu is developed in a way that it can be easily extended to support any new type of problem-solving activities as long as the designers of those activities follow some standards and interfaces. Using the question generator, faculty saves a pool of questions in a file, which is later used to design a module using the module builder interface (Fig. 9).

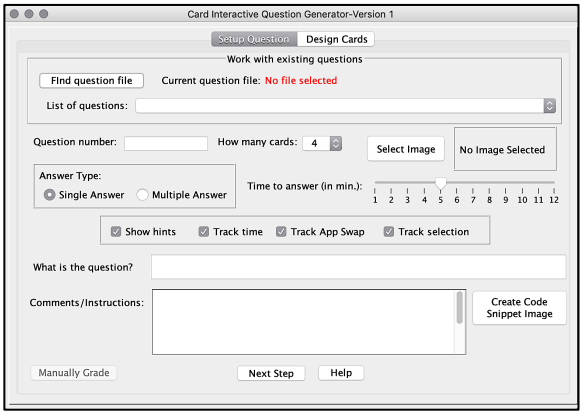

Fig. 8 Question generator.

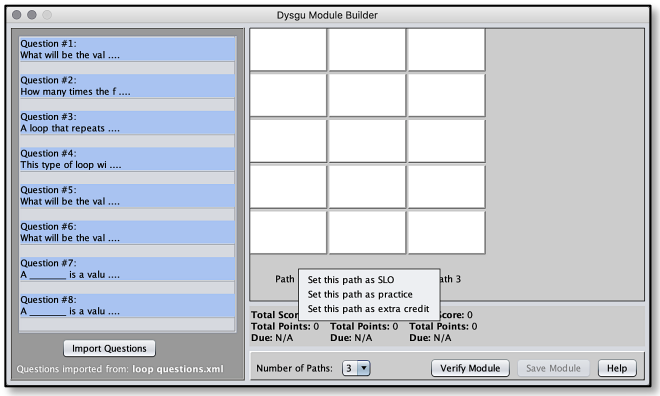

Fig. 9 Module maker interface.

The module builder (Fig. 9) helps faculty to formulate a problem-solving module, as described in previous sections. Once faculty import a question pool file, they are allowed to drag and drop the questions into the corresponding path. Problems in a path are provided to students as bottom-up and therefore the expectation is that faculty will put easier problems at the beginning of the paths and will make them harder as they progress along that learning path to facilitate instructional scaffolding.

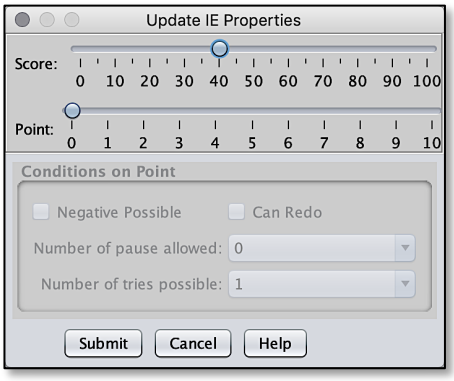

Fig. 10 Path properties.

Each path can have a deadline and faculty can either individually or across the modules assign a deadline by clicking on that path. Faculty needs to fine-tune each problem on a path by setting different properties of that problem (Fig. 10). If a point is assigned to a problem, the faculty needs to set different policies (such as whether the point can also be negative, whether students can redo a problem etc.).

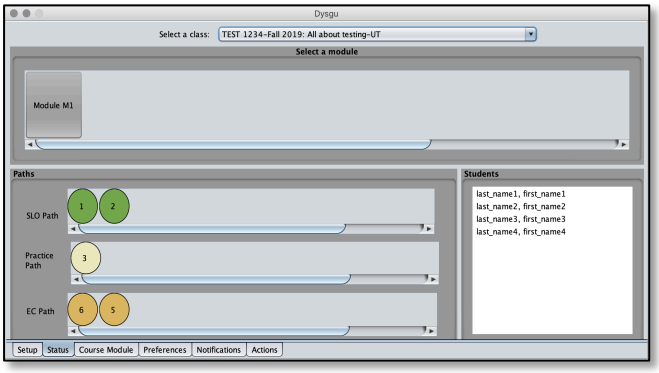

Fig.11 Class status interface.

As students start solving problems, faculty can monitor (Fig. 11) class progress on a module, on an individual question or even on a particular student. In this screen, when faculty clicks any of the module, the paths and activities in that module are shown. Then, when any of the activities are clicked, another dialog box (Fig. 12) shows details about that particular activity and a pie chart on the grades about how the class did on that activity. Therefore, faculty gets a glimpse of how the class are doing on a specific problem, which problems the students are solving successfully and where they might be having problems. Additionally, faculty can click on any student in the class and another dialog box will open (Fig. 13) that lists grade statistics of that particular student. Faculty can analyze different aspect of the student's performance with regards to module, learning paths or activity.

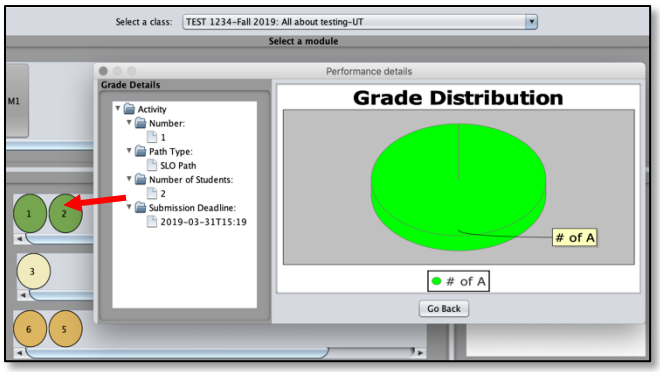

Fig. 12 Activity grade distribution.

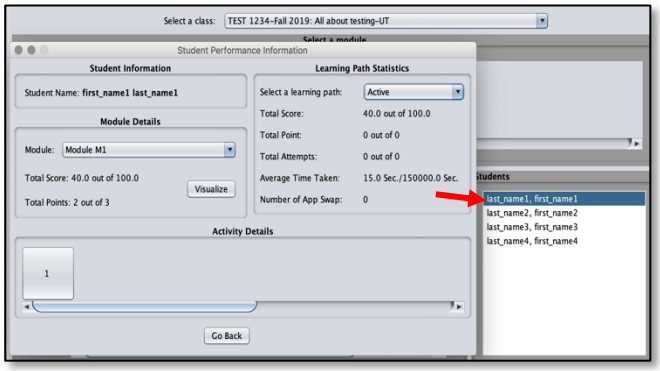

Fig. 13 Student performance detail.

On the student mobile devices, students first have to log in to a class by using their credentials. Once successfully logged in (Fig. 3), students can find current course modules, check on their status, setup schedule or other settings, see any existing or new text messages from faculty or can send a text message to the faculty.

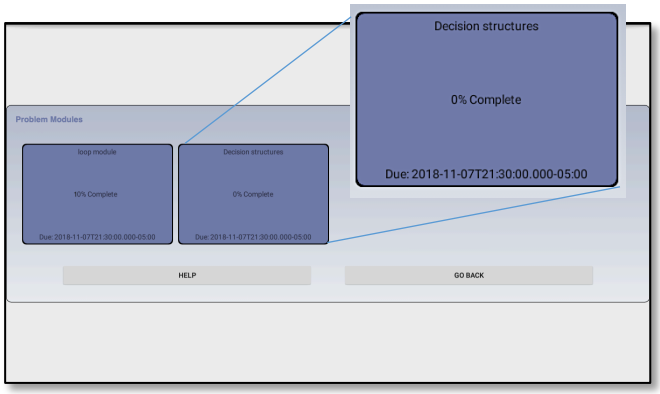

Fig. 14 Course module screen.

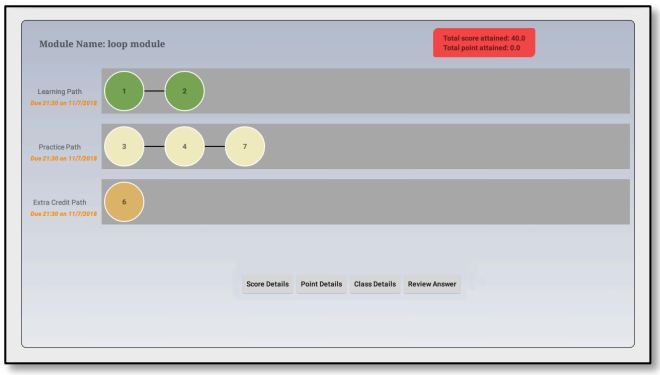

Fig. 15 Individual module screen.

A student can check available problem modules and the deadline to finish a module (Fig. 14). Even if a module's due date is over, it will still be available in this list for the student to review the answer. When one of the modules is selected, another window opens (Fig. 15), that shows the learning paths in that module and individual problems in each path, which they can select to solve. If a problem is selected, a corresponding interactive exercise activity app will run (Section III.A) that allows students to solve the problem.

A student can check their progress (Fig. 16) on different modules and can compare their progress with their classmates. Dysgu shows the student score, their placement (module, path, problem specific) compared to the class, timing information about problem-solving activities, and points and badges the student earned. Students can also review the finished activity along with comparing their performance with the rest of the class. Additionally, students are awarded badges depending on different conditions (fastest in the class to answer, highest score, etc.) and they can see what badges that they won and compare them with the rest of the class.

Fig. 16 Grade detail screen.

Fig. 17 View badge details.

Depending on the student's schedule (which students can set in the software), Dysgu reminds the students of module's deadline and other encouraging or cautioning notifications (Fig. 18). Dysgu recognizes the geophysical location of the student depending on the privacy settings and provides customized and contextual alerts to the students. For example, if Dysgu recognizes that the student is not at home during a set study time, it reminds the student frequently about incomplete work. As students continue solving problems, they will earn badges depending on different criteria (such as time taken, highest in a module or in the class, most score or points, etc.) and they can not only see their own badges, but also can see other's in the class and compare their performance relative to the rest of the class. By providing such information about peers and by having higher

expectations (in terms of rate of competition, progressive deadlines, problems with different degrees of difficulty, etc.) than traditional pen-and-paper activities, the expectation is to promote such self-fulfilling prophecy that seems to work [7].

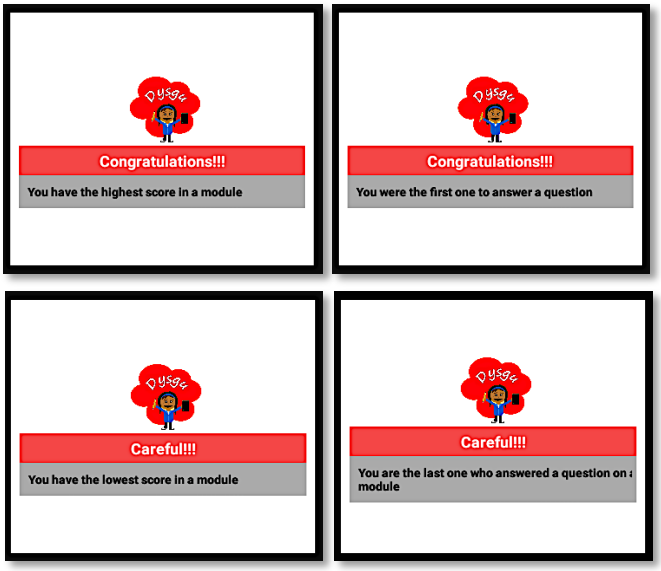

Fig. 18 Sample Dysgu notifications.

IV. Similar Systems

A plenty of research have been made to improve student's in-class learning and engagement, however, in comparison there is not much work done to improve traditional out-of-class student activities, explicitly, which combines mobile technology. A prototype mobile learning environment [8] is developed to reinforce student learning for out-of-class activities in a programming course. This system supports multiple-choice questions only and allows a student to answer a specific question per day, even if the student finishes it up right away. [9] have presented a collaborative online learning environment to link in and out of class activities and to allow student to collaborate using social networking and gamification components. In this system, all questions are narrative or discussions oriented and tied to a specific reading material set by the faculty. This system is browser-based, which lacks the flexibility and interactivity provided by mobile apps. Work presented by [10] combined both social networking and gamification to instruction and showed that it improves student's academic performance. De-Marcos's work has similar goal as the proposed study, however the delivery, type of interaction, kinds of activity and nature of engagement are different in Dysgu. Additionally, some commercial software such as Piazza [11], Prulu [12], Quizlet [13], and Socrative [14] are similar to Dysgu, however; they are different in design philosophy, scope of activity, activity type, pedagogical interventions, technique used, deployment platform or target audiences. The design goals of Dysgu addressed several critical dimensions necessary for creating an engaging learning platform. One of those goals were to have the ability to provide features that contributes to student learning, such as personalization through instructional scaffolding and frequent feedback. Secondly, to support the ability to deliver easily manageable and quality content for students, such as parameterized problems and problems with different degree of difficulties. Next, to have interactivity in such content so that students are more engaged in them, for instance, multiple screen problem-solving activity where students can traverse and see effects of their interactivity. Dysgu also provides adaptability by the way of sensing student's situational context (such as geo-location, personal schedule etc.) and provides contextual feedback and suggestions to acclimate to student's needs. Dysgu promotes motivating factors that brings students back and engage them to solve problems, such as game-centric currency that allows students to buy extra time or hints to solve a harder problem. Finally, Dysgu provides a dimension to compare student's own performance to the rest of the class, which allow students to feel an integral part of the big class and not alone solving the problems. Flipped classes [15] became popular recently because of its perceived benefits in student learning and engagement. However, there are concerns [16]-[17] related to flipping classes, especially that it might not work for students who come from socially disadvantaged background. Dysgu can address those concerns by its nature of pervasive access, interactive nature and motivation-oriented environment to augment flipped classes in order for it to become more productive for a wide variety of students.

V. Evaluation and Current Status

The current version of the software is developed using Java and Android. Although the software hasn't been used in a class yet, a small group of undergraduate students helped to pilot the system and provided a lot of useful feedback that was utilized to update the design and which showed that the system is useful to them. A sample of the comments from the students are as follow:

> "I can start working on the activity any time and see what my classmates are doing and compare my progress"

> "Dysgu is intuitive to use. Although I want it to look more professional, it does the job and easy to use"

The pilot study also provided a lot of inputs about the design of the user interface and by combining several mobile learning requirement catalogs [18]-[19], Dysgu was designed to be more responsive, attractive and motivating.

To examine the quality and maintainability of the code, we use a static code analysis tool named CodeMR [20], which analyzes the code structure, measures different quality matrices of the code, allows visualizing the design, and reports design flaws. CodeMR provides a set of software quality metrics, suitable to cover the most important aspects of structural quality. It is the belief of the developer that, any software designed for educational purposes not only have to be effective in student learning and engagement but also should have good software quality properties to make it easily manageable, maintainable, testable and extensible. Fig. 19 shows the object-oriented software quality metrics [21] (% of the code) generated by CodeMR using static code analysis. These measures are used to determine the code's structural complexity. Studies [22]-[23] show a correlation between a program's quality metrics and code's maintainability and extensibility. In that regard, it is evident from Fig. 19 that the

values for Dysgu falls within the expectable threshold [24] for software to be easy to maintain and to extend. However, as shown in Table I, some of the metrics' value for Dysgu has some room for improvement and will be investigated more in the future to achieve that.

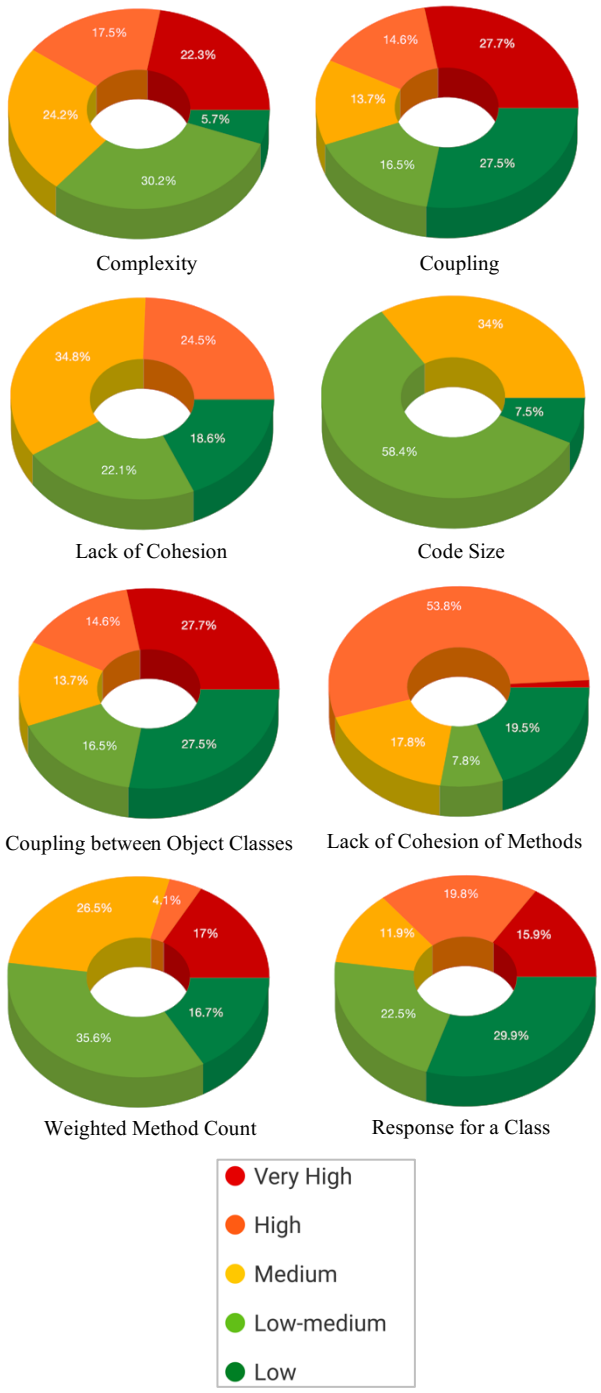

Fig. 19. Distribution of software quality attributes.

Once Dysgu is used in classes, the project will execute a Mixed Method Convergent design [25] that will incorporate both quantitative and qualitative approaches and use results from both to address evaluation questions. Student performance and course data gathered with the new intervention in the target courses will be compared with the prior offerings of the courses taught in a traditional way. Pre-post survey will include questions on course modules and learning environment software, student confidence and satisfaction and their perception of learning with technology.

TABLE I. QUALITY METRICES RANGE FOR DYSGU.

Matrix	Goal	Status
Complexity	Low to Medium	58.3% is within the goal
Coupling	Low to Medium	57.3% is within the goal
Lack of Cohesion	Medium to Very High	59.3% is within the goal
Code Size	Low	65.9% is within the goal
Coupling Between Objects	Low to Medium	57.7% is within the goal
LOC Methods	Medium to Very High	72.7% is within the goal
Weighted Method Count	Low to Medium	78.8% is within the goal
Response for a Class	Low to Medium	64.3% is within the goal

VI. CONCLUSION

This paper presents a new mobile-based out-of-class learning environment called Dysgu, where students actively participate to solve interactive problems with scaffolding, personalization and engage in self-reflections and game-centric interactions. Dysgu utilizes the transformative power of mobile technology for out-of-class activities, student engagement in active learning outside the classroom, and the possibility of providing early intervention.

REFERENCES

[1] L. Musu-Gillette, J. Robinson, J. McFarland, A. KewalRamani, A. Zhang and S. Wilkinson-Flicker, "Status and Trends in the Education of Racial and Ethnic Groups 2016", National Center for Education Statistics, 2016.

[2] P. Babcock and M. Marks, "The Falling Time Cost of College: Evidence from Half a Century of Time Use Data", Review of Economics and Statistics, vol. 93, no. 2, pp. 468-478, 2011. Available: 10.1162/rest_a_00093.

[3] S. Nonis and G. Hudson, "Academic Performance of College Students: Influence of Time Spent Studying and Working", Journal of Education for Business, vol. 81, no. 3, pp. 151-159, 2006. Available: 10.3200/joeb.81.3.151-159.

[4] H. Cooper, J. Robinson and E. Patall, "Does Homework Improve Academic Achievement? A Synthesis of Research, 1987–2003", Review of Educational Research, vol. 76, no. 1, pp. 1-62, 2006. Available: 10.3102/00346543076001001.

[5] M. Fuad, D. Deb, J. Etim and C. Gloster, "Mobile response system: a novel approach to interactive and hands-on activity in the classroom", Educational Technology Research and Development, vol. 66, no. 2, pp. 493-514, 2018. Available: 10.1007/s11423-018-9570-5.

[6] M. Muztaba Fuad and E. J. Jones, "Using Extra Credit to Facilitate Extra Learning in Students", International Journal of Modern Education and Computer Science, vol. 4, no. 6, pp. 35-42, 2012. Available: 10.5815/ijmecs.2012.06.05.

[7] J. Hattie, Visible learning. London: Routledge, 2010.

[8] R. Pinter and S. Cisar, ""One question per day" a mobile learning project", in IEEE 12th International Symposium on Intelligent Systems and Informatics (SISY), Subotica, Serbia, 2014.

[9] C. Beal, J. Strohm, L. Schwindy and P. Cohen, "Teach ourselves: A peer-to-peer learning community linking in- and out-of-class activity", Bulletin of the Technical Committee on Learning Technology, vol. 15, no. 1, pp. 13-16, 2013.

[10] L. de-Marcos, A. Domínguez, J. Saenz-de-Navarrete and C. Pagés, "An empirical study comparing gamification and social networking on e-

learning", Computers & Education, vol. 75, pp. 82-91, 2014. Available: 10.1016/j.compedu.2014.01.012.
[11] "Piazza • Ask. Answer. Explore. Whenever.", Piazza.com, 2019. [Online]. Available: https://piazza.com. [Accessed: 04- Oct- 2019].
[12] "Prulu - The most intuitive way to manage your class' Q&A.", Prulu.com, 2019. [Online]. Available: https://prulu.com. [Accessed: 04- Oct- 2019].
[13] "Quizlet-Learning Tools and Flashcards"., Quizlet.com, 2019. [online] Available at: https://quizlet.com/ [Accessed 4 Oct. 2019].
[14] "Home - Socrative", Socrative, 2019. [Online]. Available: http://www.socrative.com. [Accessed: 04- Oct- 2019].
[15] J. Bishop and M. Verleger, "The Flipped Classroom: A Survey of the Research", in 120th American Society of Engineering Education Annual Conference & Exposition, Atlanta, 2013.
[16] Flipped Classroom Trends: A Survey of College Faculty. (2014). [online] MAGNA. Available at: https://www.facultyfocus.com/free-reports/flipped-classroom-trends-a-survey-of-college-faculty/ [Accessed 4 Oct. 2019].
[17] M. Weimer, "A Few Concerns about the Rush to Flip", Faculty Focus, 2014 [Online]. Available: http://info.magnapubs.com/blog/articles/teaching-professor-blog/flipped-courses-concerns-rush-flip/. [Accessed: 04- Oct- 2019]
[18] A. Economides, "Requirements of Mobile Learning Applications", International Journal of Innovation and Learning, vol. 5, no. 5, p. 457, 2008.
[19] G. Soad, M. Fioravanti, V. Falvo, A. Marcolino, N. Filho and E. Barbosa, "ReqML-catalog: The road to a requirement catalog for mobile learning applications", in IEEE Frontiers in Education Conference (FIE), 2019 [Online]. Available: https://doi.org/10.1109/FIE.2017.8190718. [Accessed: 04- Oct- 2019]
[20] "CodeMR | Better code, better quality!", CodeMR, 2019. [Online]. Available: https://www.codemr.co.uk/. [Accessed: 04- Oct- 2019]
[21] T. McCabe, "A Complexity Measure", IEEE Transactions on Software Engineering, vol. -2, no. 4, pp. 308-320, 1976.
[22] D Wallace, A H. Watson, T J. Mccabe, "Structured testing: A testing methodology using the cyclomatic complexity metric", US Department of Commerce, Technology Administration, National Institute of Standards and Technology, vol. 500, no. 235, 1996.
[23] B. Henderson-Sellers and D. Tegarden, "A Critical Re-examination of Cyclomatic Complexity Measures", Software Quality and Productivity, pp. 328-335, 1995.
[24] R. Shatnawi, "A Quantitative Investigation of the Acceptable Risk Levels of Object-Oriented Metrics in Open-Source Systems", IEEE Transactions on Software Engineering, vol. 36, no. 2, pp. 216-225, 2010.
[25] J. Creswell and V. Plano Clark, Designing and conducting mixed methods research. SAGE, 2017.