UpGrade: An Open Source Tool to Support A/B Testing in Educational Software

Steven Ritter, April Murphy, Stephen E. Fancsali
Carnegie Learning, Inc.
Pittsburgh, PA, USA
{sritter, amurphy, sfancsali}
@carnegielearning.com

Vivek Fitkariwala, Nirmal Patel
PlayPower Labs
Ahmedabad, India
{vivek, nirmal}
@playpowerlabs.com

J. Derek Lomas
Delft University of Technology &
PlayPower Labs
Delft, The Netherlands
dereklomas@gmail.com

ABSTRACT
This paper describes a new, open source tool for A/B testing in educational software called UpGrade. We motivate UpGrade's approach, describe development goals and UpGrade's software architecture, and provide a brief overview of working within UpGrade to define and monitor experiments. We conclude with some avenues for future research and development.

Author Keywords
Field experimentation; A/B testing; Educational technology.

INTRODUCTION
Simon's 1967 address at the Presidents Institute at Princeton University [4] described "learning engineers" as "professionals in the design of learning environments" who work (with university faculty in the context of Simon's address) to "design and redesign learning experiences" and encouraged a collaborative, experimental approach to improving learning outcomes and "increasing learning effectiveness." IEEE's Industry Connections Industry Consortium on Learning Engineering (ICICLE) describes the emerging discipline of learning engineering as "a process and practice that applies the learning sciences using human-centered engineering design methodologies and data-informed decision making to support learners and their development" [2]. Researchers and practitioners of the emerging discipline tend to have expertise at the intersection of computer, learning, and data science(s), and the rise of large-scale educational technology platforms provides a variety of means by which both human-centered design methodologies and data-informed decision making can be applied to improve learning.

An important set of methodologies for making data-informed decisions involves field testing instructional improvements, which can be done at large scale on widely deployed educational technology platforms. Specifically, the ability to field test instructional improvement via randomized experiments or A/B tests represents an especially important opportunity for learning engineers to use rigorous, evidence-based approaches to improve outcomes more rapidly than by way of a traditional research cycle. Building on emerging needs and requirements for deploying large-scale A/B tests within their own learning platforms, Carnegie Learning partnered with PlayPower Labs to develop UpGrade, a free and open-source A/B testing framework designed to support randomized field testing in educational software. This framework provides a way for learning engineers to engage with improving learning experiences at scale, and takes into account a variety of constraints and requirements imposed upon educational platform developers, especially, but not limited to, those imposed by the realities of school-based learning in K-12 and other institutional settings.

UpGrade is unique from widely-available A/B testing systems, including general platforms like Optimizely, Launch Darkly, PlanOut, and others, which represent a mix of commercial and open source solutions, and A/B testing systems deployed in some educational contexts (e.g., the E-TRIALS Testbed on the ASSISTments platform [1]). Specifically, UpGrade addresses several concerns that are not well satisfied by current off-the-shelf products but which are important to educational software, particularly when such software is used in institutional settings. One such concern is the ability to assign groups of students to condition as a block (e.g., classes, schools, districts), to ensure, for instance, that students in the same class receive consistent experimental features, especially when experiments are intended to test often substantively different approaches to providing instruction and practice on the same topics. Issues concerning group random assignment are addressed by Ritter et. al. [3].

In what follows, we begin by briefly describing an example experiment that manifests some of the key experimental design issues that UpGrade was designed to address, though UpGrade is capable of handling more and less complex designs. Next, we consider a set of development goals for UpGrade for its initial implementation as well as its overall architecture. Then we describe experimental conditions and

L@S '20, August 12–14, 2020, Virtual Event, USA.

the lifecycle of an experiment before turning to the practicalities of defining and monitoring A/B tests with UpGrade. We conclude with some directions for future research and development as well as a call for collaborators and platform developers seeking to build learning engineering capacity by integrating UpGrade into their platforms.

AN ILLUSTRATION: ALTERNATIVE ACTIVITY EXPERIMENTS

UpGrade (and this paper) focuses on features that are of importance in educational software, but there are no particular barriers to using UpGrade to support A/B testing within other kinds of software. UpGrade can be used to A/B test user interface elements, layout, messaging and any other aspects of the user's experience that is controlled by software.To start, we describe a hypothetical experiment that manifests several key issues that UpGrade is designed to address.

In educational software, it is common for activities to be presented sequentially (as in Figure 1). These activities could be chapters in an ebook, interactive exercises, videos, test questions, among other possibilities. In some systems, all students would receive the same sequence; in others, the sequence might vary for different students. In some systems, students pick the activity they are to work on; in others, the system will pick the activity for the student.

Figure 1. A sequence of five educational activities.

An "alternative activity" experiment looks like Figure 2: randomly-selected students are assigned to receive Activity 2 with probability *p*, and students are assigned to receive Activity 2a with probability *1 - p*. Activity 2 and Activity 2a are intended to serve the same educational function. For example, in an eBook application, Activity 2a might be an alternative book chapter. The alternative might be completely different or it may differ in only some details (such as one particular diagram). The goal of the experiment is to see whether students learn better from the original or the revised chapter.

Educational content sometimes builds on earlier-presented content. For example, Activity 4 (Chapter 4) in our eBook might reference the material presented in Chapter 2 in a way that requires the student to have received the corresponding version of that chapter. Figure 2 illustrates this situation. A student randomly assigned to see Activity 2a must also see the coordinated Activity 4a. This type of coordination is managed by UpGrade via what we call "experiment sites."

Many students will complete Activity 2 (e.g., reading Chapter 2 in our hypothetical eBook) and never revisit it. However, some students might go back and reread Chapter 2 or redo the activity, perhaps months later in preparation for a test. Suppose that an experiment is running when the student initially reads the chapter, but the experiment has concluded by the time the student goes back to review. Should the student receive the same version of the chapter or activity that they saw the first time? The answer will likely vary depending on both the type of changes and the extent of changes to the chapter or activity. But, at least in some cases, we would want the chapter to be consistent upon every encounter. To achieve this, the experimental condition must persist, even after the experiment is over. We call the goal to provide individual students with a consistent experience over time *individual consistency*.

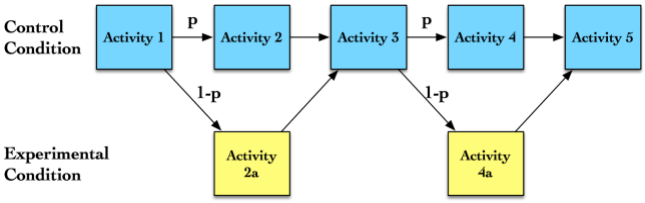

Figure 2. An alternative activity experiment.

In a classroom setting, if the teacher wishes to provide whole-group instruction to the class, and the control and experimental versions of the eBook are fundamentally different, then it would be disruptive to both teachers and students if half the class got the control version of the eBook and half got the experimental version. Since teachers commonly teach multiple classes on the same topic, it would be most convenient for them if the students in all of their classes received the same version of the eBook. The goal to provide common experiences to groups of students is called *group consistency*, and UpGrade supports group random assignment (see [3]).

There are cases where it is impossible to satisfy both individual and group consistency. For example, if students in Class A are assigned the control eBook and those in Class B are assigned the experimental textbook, which version of Activity 2 should a student receive if they start in Class A and then transfer to Class B? Consistent resolution of these kinds of anomalies and flexibility for researchers to determine what resolution is appropriate are major functions of UpGrade.

DEVELOPMENT GOALS & ARCHITECTURE

We set out to develop UpGrade according to a set of requirements and goals that we think will maximize the usefulness of the platform for educational technology developers working in school-based instructional and similar settings while still maintaining flexibility for broader use-cases. UpGrade runs as a web service separate from the application with which it is integrated (see Figure 3). Developers may host their own UpGrade server, or companies may host servers as a service for educational software developers.

Since educational data can be sensitive, UpGrade stores only anonymized identifiers for students, teachers, classes, schools and other sensitive information.

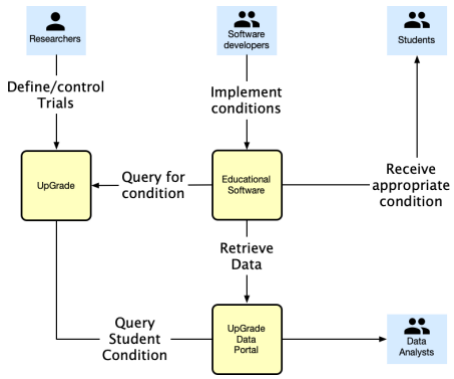

Figure 3. UpGrade architecture. Upgrade operates as a separate web service from the educational application and the data portal. Researchers work directly with UpGrade to define experiment parameters. Educational software queries UpGrade to determine conditions for individual students. UpGrade allows simple data analysis sufficient for monitoring experiment progress, researchers can also use the UpGrade Data Portal to export data for more detailed analysis.

Educational software, particularly that used in schools, responds to yearly (or semester-long) educational cycles. Content and features that might be highly relevant at the beginning of the year may be much less relevant as the school year proceeds. For this reason, experiment designs to be targeted at a particular classroom or school need to be closely coordinated with that classroom or school's curriculum progress. With UpGrade, an experiment can be targeted at a larger set of classes or schools, with the expectation that, due to variation in sequencing and pacing, some students will "see" the experiment while others will not.

In order to promote the development of a learning engineering community, UpGrade is free and open source.[1]

CONTROLLING EXPERIMENTAL CONDITIONS

Within UpGrade, every experiment is given a unit of assignment. If the unit of assignment is *individual*, then group membership information is irrelevant. If the unit of assignment is *group*, then the researcher can specify the group type (e.g., class, teacher, school, district), and UpGrade will assign conditions on a group basis.

For group-randomized designs, researchers can direct UpGrade to make decisions about conflicts between individual and group consistency through the use of the "Consistency Rule." This rule can take on three values:

Group: Under this rule, the priority is that all students in a group have a common experience, even if it means that some students may have an inconsistent experience. UpGrade may exclude a group from the experiment, if it is not possible to keep the group consistent, potentially sacrificing statistical power.

Individual: Under this rule, the priority is that individual students have a consistent experience, even if that experience differs from the rest of the class (or group).

Experiment: Under this rule, the student's experience is guided by whether the experiment is running. During the period where the experiment is active, students will receive the experience appropriate to the group assignment, even if it violates individual consistency. This rule is most appropriate when the manipulation may not be evident to students and when the researcher believes that prior experience will not affect the student's experience in an experimental condition.

An additional consideration with respect to consistency is what to do when the experiment is over. The "Post Experiment Rule" determines this. Researchers can choose to "continue" the student's experience (keep them in condition), "revert" to the default behavior or transfer all students to a single condition (presumably the "winning" condition with the best learning outcomes).

EXPERIMENT LIFECYCLE

To run an experiment, at least two conditions must be made available (the control is often the existing behavior). Generally, this involves software development, web programming and/or design of media. Running an experiment also requires someone to design the characteristics of the experiment: the unit of assignment, the eligible users to participate in the experiment, the number of participants required, start and end dates, etc. In many organizations, these two functions will be performed by different individuals (or even different departments).

UpGrade allows these two functions to proceed independently. UpGrade enables design of the experiment independent of implementing the experiment conditions (and either can proceed the other), but the experiment cannot be delivered until both tasks are completed.

To manage the experiment lifecycle, experiments progress through (at most) seven states:

Inactive: The experiment has not yet started. Students are not assigned to experiment conditions, and students using the software will not see any experimental variants. Experiment design and/or development may still be underway.

Preview: UpGrade provides a way to manually assign "demo" students to condition. This is intended to be used to test or preview an experiment without assigning real students. Note that these students can persist, even after the experiment is running. This status simply enables previewing assignment before the experiment officially starts.

Scheduled: The experiment is designed and developed and is set to automatically launch at a specified time and date but has not yet started.

[1] Under https://opensource.org/licenses/BSD-3-Clause

Enrolling: The experiment is running. Students are assigned to and continue working according to condition. We use the term "enrolling" instead of "running" since students may continue to experience the experimental conditions after the experiment stops enrolling.

Enrollment Complete: The experiment has collected enough data to be analyzed (at least by the original design). Students who have been assigned to a condition will continue to be presented experiences corresponding to their condition (so the experiment continues to collect data).

Cancelled: This is an abnormal termination state, to be used in cases where the experimental condition(s) are buggy or clearly educationally ineffective. Under this status, students will not be receive an experimental condition, even if they are in a group that has been assigned a condition or if they have previously experienced the experimental condition themselves. Cancelling an experiment can violate both individual or group consistency.

Archived: An archived experiment is one that is completed and no longer supported. In archived status, the code in the user-facing app that supported the experimental conditions should not be expected to be present. Archived status is permanent; experiments cannot transition to any other status.

DEFINING EXPERIMENTS

Consistent with the idea that the researcher defining experiment parameters may not be the programmer or web developer implementing experimental conditions, UpGrade provides an easy to use web-based interface for defining experiments (Figure 4).

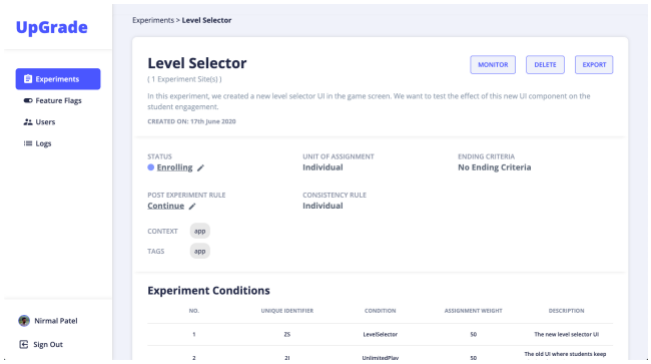

Figure 4. User interface for defining major experiment conditions and parameters. The researcher can set status, unit of assignment, consistency and post-experiment rules and ending criteria.

MONITORING RESULTS

Educational applications can provide UpGrade with metadata specifying the metrics they collect on students. This allows researchers to link these metrics to specific experiments. Metrics can be simple (e.g., "total time spent using the app") or grouped. Grouped metrics allow multiple measure for different components of the app (e.g., "time for Activity 1" separate from "time for Activity 2"). Monitored metrics provide a real-time view of experiment progress. These metrics might be sufficient to determine the effectiveness of an intervention, but data export functionality allows student condition data to be linked with application data outside of UpGrade for more sophisticated analyses.

CONCLUSION

Having described design goals and features of UpGrade, we look forward to expanding the learning engineering community of UpGrade users and contributors. We are considering, for future releases, providing APIs so that educational applications can provide user-defined experiments (e.g., teachers could compare variants with their students), ways to use UpGrade's knowledge of experimental design to facilitate analysis and publication (e.g., integration with e2icoach,[2] automatic preregistration of experiments,[3] output of experimental design data for more sophisticated statistical models), expanded experiment designs (e.g., within-subject and factorial designs) and related functionality like feature flagging.

Source code for UpGrade is available,[4] and additional information about the platform will be available at upgradeplatform.org. If you are interested in using or contributing to UpGrade, please contact upgradeplatform@carnegielearning.com.

ACKNOWLEDGEMENTS

This work was supported by grants from the Bill & Melinda Gates Foundation and Schmidt Futures.

REFERENCES

[1] Neil T. Heffernan and Cristina L. Heffernan. 2014. The ASSISTments Ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *Int. J. Artif. Intell. Educ.* 24, 470-497.

[2] Institute of Electrical and Electronics Engineers (IEEE). 2020. Home – ICICLE. Retrieved July 31, 2020 from https://sagroups.ieee.org/icicle/

[3] Steven Ritter, April Murphy, and Stephen E. Fancsali. 2020. Managing group random assignment in UpGrade. Submitted to *Proceedings of the First Workshop on Educational A/B Testing at Scale (at Learning @ Scale 2020)*.

[4] Hebert A. Simon. 1967. The job of a college president. *Educational Record* 48, 68-78.

2 https://e2icoach.org/

3 See, for example, the Center for Open Science's preregistration platform: https://www.cos.io/our-services/prereg

4 https://github.com/CarnegieLearningWeb/educational-experiment-client