Orchestration Graphs: Enabling Rich Social Pedagogical Scenarios in MOOCs

Stian Håklev
Ecole Polytechnique Fédérale de Lausanne
stian.haklev@epfl.ch

Louis Faucon
Ecole Polytechnique Fédérale de Lausanne
louis.faucon@epfl.ch

Thanasis Hadzilacos
Open University of Cyprus and Ecole Polytechnique Fédérale de Lausanne
thanasis.hadzilacos@ouc.ac.cy

Pierre Dillenbourg
Ecole Polytechnique Fédérale de Lausanne
pierre.dillenbourg@epfl.ch

ABSTRACT
One of the initial promises of MOOCs was to enable participants from around the world to learn and build knowledge together, however existing MOOC platforms are very limited in their collaborative functionality. Using a recent educational modeling language which can express a broad diversity of educational scenarios, we present a technical infrastructure design and prototype which enables instructors to design and run pedagogically rich and therefore complex scenarios. We present this as a theoretical and technical contribution to support a broad program of research and innovation related to collaborative learning at scale.

Author Keywords
MOOCs; Scripting; Orchestration.

INTRODUCTION
Globally distributed MOOCs are characterized by large numbers of learners, and significant student diversity, in terms of geographical location, life experience, culture and language. Ideally, we would treat the number of learners, and their diversity, as an astounding opportunity for rich learning and communal knowledge building, rather than as an obstacle to be overcome.

However, several studies show that simply asking students to solve a task or discuss a problem is unlikely to engage them in a productive process of collaborative knowledge construction [1]. A key question, which has guided much of the research in the field of Learning Sciences, is how this process can best be supported and guided by an instructor or learning designer, through the careful design of the physical and digital learning context and its affordances.

L@S 2017, April 20 - 21, 2017, Cambridge, MA, USA

ISBN 978-1-4503-4450-0/17/04…$15.00

DOI: http://dx.doi.org/10.1145/3051457.3054000

To help support such learning, Fischer and his colleagues [4] proposed a form of scaffold, which they call an "external collaboration script", that specifies the roles and goals of the participants, allocates material and tasks, and coordinates student grouping and learning activities. Once scripts have been designed, they need to be implemented or enacted. Dillenbourg, et al. call this process "orchestration", defined as "how a teacher manages, in real time, multi-layered activities in a multi-constraints context" [3].

CHALLENGE
Despite a long history of research on pedagogical scripts for small groups and school classes, there has been very little research on appropriate pedagogical scripts that take advantage of very large numbers of heterogeneous learners. One example is Håklev, who has suggested design principles for pedagogical scripts for collaborative MOOCs that use a matrix of weekly thematic scripts which feed into a set of interdependent longer running scripts [5].

We need more creativity and sharing in this area, and to enable this, a common notation, or educational modeling language, as well as tools for editing and sharing educational scenarios, could help. However, our largest challenge is how to implement these scripts in very large settings. A school teacher can fall back to using sticky notes and "talk to your neighbour" strategies, but currently implementing even a very simple script on a MOOC platform (for example wanting to set up a peer-review session, but in a different way than the built-in approach) requires significant software development resources not available to most groups.

This paper will present a suggested common educational modeling language called Orchestration Graphs; our prototype framework for authoring and sharing Orchestration Graphs; as well as the design of an ecosystem of activities and operators which can be "run" by an Orchestration Engine. We posit that this is an innovation which has the potential to promote development and research around rich collaborative activities and scenarios at scale.

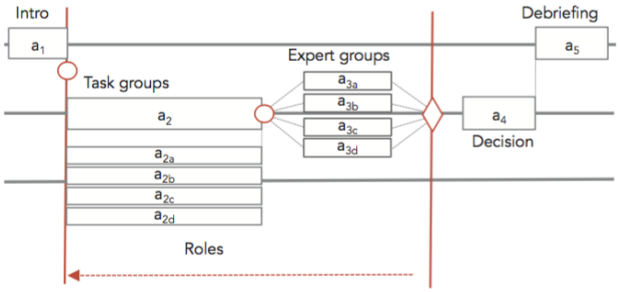

Figure 1. Example of an Orchestration Graph. From Orchestration Graphs by Pierre Dillenbourg, Lausanne: EPFL Press. Copyright 2015 by Pierre Dillenbourg. Reprinted with permission.

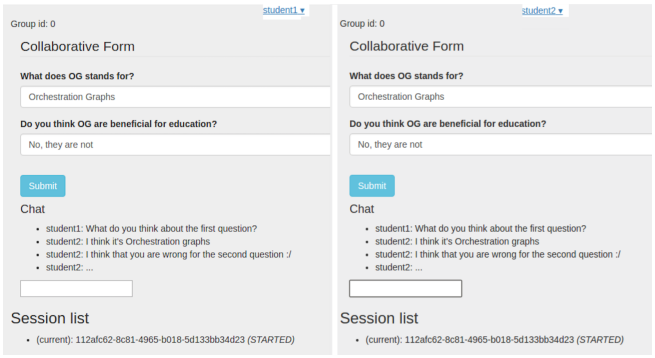

Figure 2. Two students participating in a collaborative learning activity.

ORCHESTRATION GRAPHS

An Orchestration Graph (such as the example in Figure 1) is a structured view of a learning scenario, consisting of learning activities (nodes) [2]. These nodes have a start and an end-time. The X-axis of the graph denotes time, and the placement and width of activities on it denote start and duration. In short synchronous scripts, these time cut-offs might be absolute, to enable students to "sync up" (e.g., after five minutes, everyone has to switch groups), whereas in long-running asynchronous scripts, activities might be open for several days, and students can choose when to engage with them.

Planes

Activities take place at different social planes (Y axis), the three key ones being individual activities, team activities, and whole class activities.

Activities

Activities range from reading a text or watching a video, to contributing ideas in a brainstorm, or experimenting with a simulation. They can take input data and social structures from previous activities and operators, and produce student products (such as students answers and designs), as well as detailed student trace data. See an example of a running activity in Figure 2.

Edges

Activities (nodes) are connected through edges. These can contain pedagogical justifications (e.g., activity 1 is an advanced organiser for activity 2), learning analytics information (e.g., student success is activity 1 is 34% correlated with/predictive of success in activity 2), and operators.

Operators

Operators receive data from student activities (products), and generate input for subsequent activities. Operations can include aggregation, disaggregation, assignment, translation and transformation of the student product. Operators are also used to generate social structures based on input data, such as organizing students into groups based on their previous answers.

PRIOR ART

The idea of an educational modelling language (EML) has a long history. Several of these languages also come with their own software for editing, and sometimes executing the graphs. LAMS [9] offers a graphical editor of learning workflows, which can be executed by the system. Collage [7] proposes richer design patterns, which can be executed within GLUE!-PS [10].

Two recent platforms, ILDE [6] and Learning Designer[8] build upon a community of designers. The latter provides feedback to designers, for instance the proportion of various learning activity types (investigation, practice, etc.). However, the designed scenarios are not runnable on the platform.

EXAMPLE

Orchestration Graphs can describe orchestrated live interactions, such as a "jigsaw script" centred around earthquake mitigation in San Francisco (depicted in Figure 1). Each team comprises four roles: the mayor of San Francisco (a_{2a}), a seismology expert (a_{2b}), a security officer (a_{2c}), and an insurance agent (a_{2d}). Students begin by discussing the task in groups composed of all four roles, but are then interrupted by "expert group meetings", where all mayors meet with each other separately. This cycle can be repeated several times, before the teams conclude (a_4) and meet as a whole class.

Scripts can also move across much larger granularities and time frames. For example, Håklev [5] describes a script which begins by thousands of students (in-service teachers) contributing individual educational resources that they find useful. Students are then presented with relevant resources to their own teaching (based on semantic tags), and tag, vote and add comments. The curated and annotated list of resources feed into the small (3-6 person) groups which work on developing a lesson plan using technology. Their work in progress is cycled back to the community for constructive peer-review on a weekly basis. Thus we see overlapping scripts that last over several weeks, and which distribute input from thousands of students, to small teams, and back to larger "communities of interest".

Repair/orchestration

All of the individual activities, and the graph as a whole, have a standardised interface for live-streaming learning traces (log data) from learners, and a well-defined API for the teacher to interface with the execution (re-ordering groups, modifying the configuration or order of activities, etc). This can allow

Figure 3. The teacher's analytics dashboard and orchestration controls

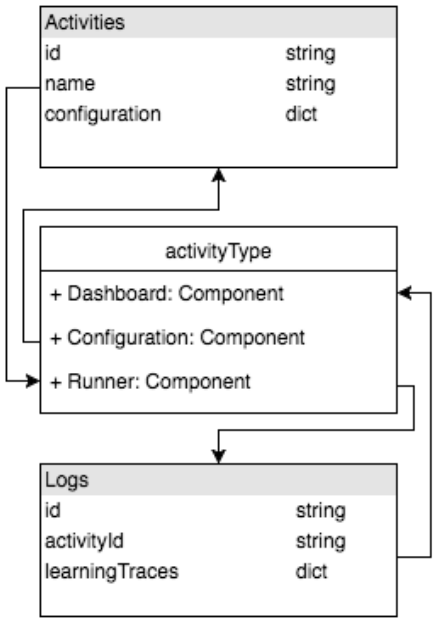

Figure 4. API of activity types

an instructional team to monitor the progress of thousands of students through innovative visualisations and alerting systems, and intervene where groups are not working well, or the community is not moving forwards as planned (see an example of a dashboard in Figure 3).

Given this standardised interface, it would also be feasible to construct intelligent agents that use machine learning to "act on the teacher's behalf" (perhaps even trained on a history of teacher actions), automatically monitoring group and individual student progress, and intervening when necessary.

ECOSYSTEM OF ACTIVITIES AND OPERATORS

The online education space has a long history of plugins, widgets and protocols for integrating external learning activities into Learning Management Systems, and now MOOCs. However, existing standards do not satisfy several of our requirements: the ability to have an artefact generated in one activity used as an input in another activity, the access to a live stream of learning analytics data, class reorganisation into teams, and the ability for a teacher to "orchestrate"/intervene in an ongoing activity.

We have defined an API (shown in Figure 4) to enable rich integration with highly modular activities. Activities are currently distributed as separate packages; in the future we will allow for activities which run on separate servers. Activities provide their own metadata, configuration function (e.g., a video player requires a URL, a quiz requires questions and alternatives), and a "runner", which receives the configuration data, the input data from any previous activities, and a social structure. Activities also implement a live analytics dashboard using the learning traces sent as logs by the "runner".

Operators are located on the edge between two nodes, and receive the student products from the source node, and either transform the product, or generate new social structures based on the student products. Operators are also separate packages (and will be able to run on remote servers as APIs), providing a configuration function, and a "runner", which receives it's own configuration, as well the data to process, returning the output data or social structure.

INTEGRATION IN MOOC PLATFORMS

The first version of the Orchestration Engine is being developed as a stand-alone tool, but there are several possible ways in which it can be integrated into traditional MOOC platforms. One possibility is to offer several synchronous sessions within a MOOC. Students could be asked to log on at a certain time and date, and visit a specific section in any platform supporting the Learning Tools Integration (LTI) API[11].

This activity would function as a "window" to the Orchestration Engine, and within this window, an entire learning scenario could play out. While the script were running, the instructional staff would be monitoring the live learning analytics, and have the ability to intervene to repair or improve the script.

We could also imagine that each separate activity in an Orchestration Graph could be given a unique LTI URL. A course designer working within the EdX platform (as an example) who wanted to implement something similar to the macro script described above [5], could add an "Individual survey" activity in Week 1 of the MOOC. In the graph, this activity would be connected to an operator that would generate group structures based on the responses, and further to a group activity that would rely on this group structure. The designer could then insert the group activity as a separate LTI component in Week 2 of the EdX course, and the Engine would take care of all the data flow and social structure behind the scenes (see an example in Figure 5).

FUTURE POSSIBILITIES

Our work on the educational modelling language Orchestration Graphs, and the technical infrastructure/architecture for

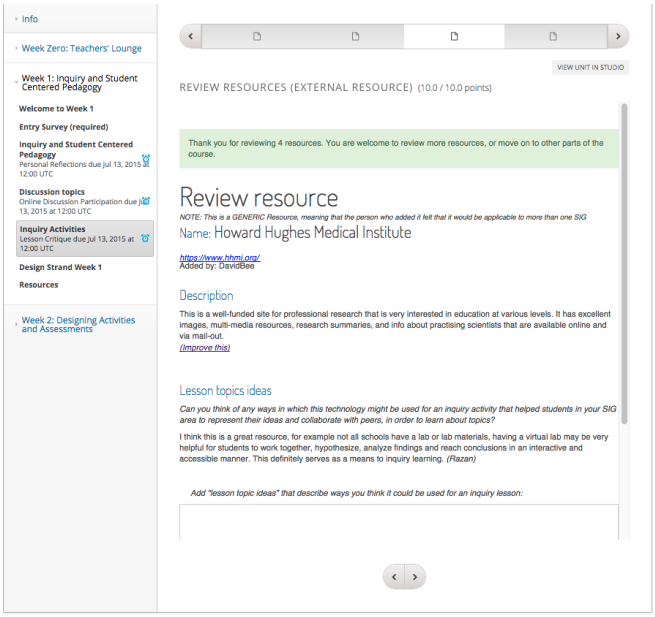

Figure 5. Example of an external activity from the Orchestration Graph seamlessly embedded in an EdX MOOC

editing and running these graphs, form part of a larger research program to empirically investigate and improve our understanding of large-scale collaborative learning scripts. Typical learning analytics research with MOOCs has generated large amounts of data points, but the analysis has rarely taken into account the instructional design or the pedagogical role of each activity.

If we begin with a computer legible description of the pedagogical design, and can automatically map rich learning analytics (both general, and activity-specific) onto the graphs, we can begin to develop models of student learning in the form of transition matrices (predicting the success of a student on an activity from his observed behaviour, his performance on the previous activity and the performance of other students on the current activity).

CURRENT STATUS AND CONCLUSION

We currently have a working prototype system, which allows users to edit scripts by adding and configuring activities and operators. We have a set of activities and operators, which are all completely isolated code-bases connected to the Engine through the public API, and we are able to run scripts generated by the editor, with learning analytics and teacher orchestration capabilities.

We are in the process of refining the design of the API, and expanding the collection of activities and operators, as well as rebuilding the architecture to be more scaleable (distributed across multiple servers). We plan a number of smaller experiments in the spring of 2017 aimed at testing UI/UX with instructional designers/teachers, and running scripts with actual students. The platform will then become an integral part of an EdX MOOC in fall 2017, when students will not only participate in scripts designed by the instructor, but will also get the opportunity to themselves design and run scripts as part of the learning process.

REFERENCES

1. Elizabeth G. Cohen. 1994. Restructuring the Classroom: Conditions for Productive Small Groups. *Review of Educational Research* 64, 1 (1994), pp. 1–35.
2. Pierre Dillenbourg. 2015. *Orchestration Graphs: Modeling scalable education*. EPFL Press.
3. Pierre Dillenbourg, Miguel Nussbaum, Yannis Dimitriadis, and Jeremy Roschelle. 2012. Design for Classroom Orchestration. *Computers & Education* 69 (2012), 485–492. DOI: `http://dx.doi.org/10.1016/j.compedu.2012.10.026`
4. Frank Fischer, Ingo Kollar, Karsten Stegmann, and Christof Wecker. 2012. Toward a Script Theory of Guidance in Computer-Supported Collaborative Learning. *Educational Technologist* 48, 1 (2012), 56–66.
5. Stian Håklev. 2016. *From seminar to lecture to MOOC: Scripting and orchestration at scale*. Ph.D. Dissertation. University of Toronto.
6. Davinia Hernández-Leo, Juan I Asensio-Pérez, Michael Derntl, Luis P Prieto, and Jonathan Chacón. 2014. ILDE: community environment for conceptualizing, authoring and deploying learning activities. In *European Conference on Technology Enhanced Learning*. Springer, 490–493.
7. Davinia Hernández-Leo, Eloy D. Villasclaras-Fernández, Juan I. Asensio-Pérez, Yannis Dimitriadis, Inés Ruiz-Requies, Bartolomé Rubia-Avi, and Iván Jorrín-Abellán. 2006. COLLAGE: A collaborative Learning Design editor based on patterns. *Journal of Educational Technology & Society* 9, 1 (2006), 58.
8. Diana Laurillard. 2013. *Teaching as a Design Science: Building Pedagogical Patterns for Learning and Technology.* Routledge.
9. Patrick McAndrew, Peter Goodyear, and James Dalziel. 2006. Patterns, designs and activities: unifying descriptions of learning structures. *International Journal of Learning Technology* 2, 2-3 (2006), 216–242.
10. Luis Pablo Prieto, Juan Alberto Muñoz-Cristóbal, Juan Ignacio Asensio-Pérez, and Yannis Dimitriadis. 2012. Making learning designs happen in distributed learning environments with GLUE!-PS. In *21st Century Learning for 21st Century Skills*. Springer, 489–494.
11. Charles Severance, Ted Hanss, and Josepth Hardin. 2010. Ims learning tools interoperability: Enabling a mash-up approach to teaching and learning tools. *Technology, Instruction, Cognition and Learning* 7, 3-4 (2010), 245–262.