The Beginning of a Beautiful Friendship? Intelligent Tutoring Systems and MOOCs

Vincent Aleven[1(✉)], Jonathan Sewall[1], Octav Popescu[1], Franceska Xhakaj[2], Dhruv Chand[3], Ryan Baker[4], Yuan Wang[4], George Siemens[5], Carolyn Rosé[1,6], and Dragan Gasevic[7]

[1] Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, United States
{aleven,sewall,cprose}@cs.cmu.edu, octav@cmu.edu
[2] Computer Science Department, Lafayette College, Easton, United States
xhakajf@gmail.com
[3] Department of Mechanical Engineering, National Institute of Technology Karnataka, Mangaluru, India
dhruvchand@live.com
[4] Teacher's College, Columbia University, New York, United States
ryanshaunbaker@gmail.com, elle.wang@columbia.edu
[5] LINK Lab, The University of Texas at Arlington, Arlington, United States
gsiemens@gmail.com
[6] Language Technologies Institute, Carnegie Mellon University, Pittsburgh, United States
[7] Schools of Education and Informatics, University of Edinburgh, Edinburgh, United Kingdom
dgasevic@acm.org

Abstract. A key challenge in ITS research and development is to support tutoring at scale, for example by embedding tutors in MOOCs. An obstacle to at-scale deployment is that ITS architectures tend to be complex, not easily deployed in browsers without significant server-side processing, and not easily embedded in a learning management system (LMS). We present a case study in which a widely used ITS authoring tool suite, CTAT/TutorShop, was modified so that tutors can be embedded in MOOCs. Specifically, the inner loop (the example-tracing tutor engine) was moved to the client by reimplementing it in JavaScript, and the tutors were made compatible with the LTI e-learning standard. The feasibility of this general approach to ITS/MOOC integration was demonstrated with simple tutors in an edX MOOC "Data Analytics and Learning."

1 Introduction

MOOCs and online courses are by now very widespread and popular [7]. At their best, they succeed at offering open and free opportunities to complete courses offered by some of the best universities in the world and at creating large-scale social participation, Although they are perhaps best known for the use of video lectures, they also support learning by doing, offering either simple activities with automated feedback (e.g., multiple choice questions to test your understanding) or complex activities with peer help, peer discussion, and peer grading. Although these solutions have been quite

C. Conati et al. (Eds.): AIED 2015, LNAI 9112, pp. 525–528, 2015.
DOI: 10.1007/978-3-319-19773-9_53

successful at scale, they have their drawbacks. A single question with feedback on the final answer is a minimal way of scaffolding an elaborate reasoning process [8][9]. Peer discussion and feedback are not always timely; peers may not know the right answer or may disagree, and many learners may be reluctant to post questions and concerns to a large audience [2]. MOOCs sometimes have limited capabilities to support individual learning [4] or personalizing instruction.

Intelligent tutoring systems (ITSs) address some of these limitations. Their effectiveness in helping students learn has been well-documented [5][9]. They provide step-by-step guidance during (moderately) complex problem solving. They can track learners' skill growth and select problems on an individual basis. They can adaptively respond to student strategies and errors. On the other hand, MOOCs support learning in ways that ITSs do not, for example with video lectures, discussions forums, and so forth. Hence, we propose integrating ITS-style learning-by-doing into MOOCs.

To achieve this integration, we see two main challenges: ITSs tend to be technologically complex and not always compatible with browser technology, at least without substantial server-side processing. Also, ITSs are often not interoperable with existing MOOC platforms or other learning management systems. In the current paper, we address these challenges. We present a case study in which a widely used set of ITS authoring tools, the Cognitive Tutor Authoring Tools [1] (CTAT, http://ctat.pact.cs.cmu.edu) was extended so that tutors built with these tools can run in browsers in a way that is compatible with e-learning platforms. We demonstrated the technical feasibility of this approach in an edX MOOC during the Fall of 2014.

2 CTAT/TutorShop and Example-Tracing Tutors

CTAT supports the authoring, without programming, of example-tracing tutors, a type of tutoring system that provides step-by-step guidance in complex problem-solving activities [1]. Example-tracing tutors have been widely used in ITS research and development projects and have been shown to support student learning in a range of domains. CTAT is integrated with TutorShop, a module that provides course management and learning content management services for CTAT-built ITSs. .

For the discussion that follows, it is important to explain how key tutoring functionality is separated and distributed in the CTAT/TutorShop architecture. In this architecture, the tutor interface, the tutor's inner loop functionality, and its outer loop are all strictly separate. By the inner loop, VanLehn [9] means the tutor's within-problem guidance. In CTAT/TutorShop's architecture, the tutor engine (which implements the example-tracing algorithm) takes care of the inner loop. Prior to the changes described in this paper, it ran on the server and was implemented in Java. TutorShop takes care of the outer loop; it personalizes the selection of problems based on a student model [3]. This student model is computed in the inner loop and communicated to the outer loop at the end of each problem, where it is stored between sessions. The tutor interface is separated from the tutor back-end. This is referred to as the tool-tutor separation, with a well-specified API [6]. The interface is launched from the TutorShop at the start of each problem, but after that communicates only with the inner loop (the example tracer) until the student finishes the problem, when the interface updates TutorShop with the revised student model and requests the next problem.

3 Technical Integration and Pilot Test

Given the factored architecture described above, our approach to tutor/MOOC integration involved two key technical changes. First, we moved the inner loop (the example-tracing tutor engine) to the client, to reduce client-server traffic and server load; we did so by reimplementing it in JavaScript. The second change was to make the TutorShop LTI-compliant so that tutors embedded in the host MOOC platform (here, edX) could be launched from the TutorShop and communicate with both the edX course management facilities and the TutorShop (e.g., for storing the student model and other analytics). We implemented the Leaning Tools Interoperability (http://www.imsglobal.org/lti/) tool provider interface in the TutorShop. The availability of Ruby packages for OAuth and LTI greatly simplified our task.

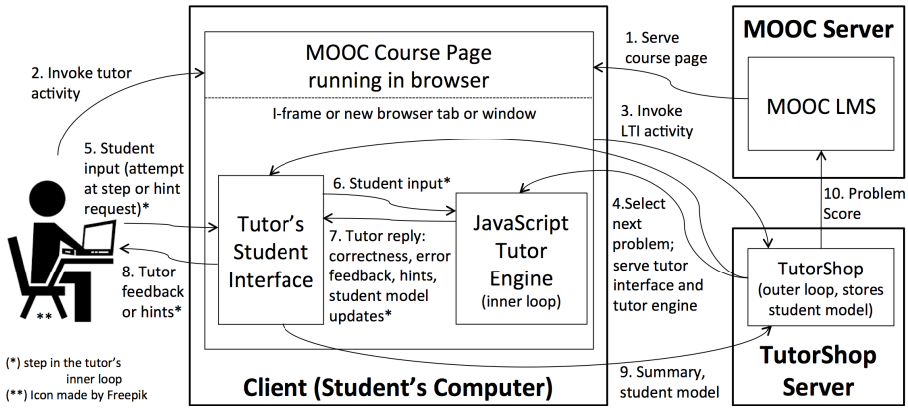

Fig. 1. Overview of the CTAT/TutorShop/edX integration

Figure 1 summarizes the data flow in a MOOC with CTAT/TutorShop as an LTI tool. After the student (1) sees a course page and (2) invokes the tutor activity, the page (3) invokes TutorShop's tool provider URL; TutorShop (4) replies by downloading HTML with the student interface and the JavaScript tutor engine into the page's iframe. Thereafter, whenever the student (5) attempts a step in the problem, the user interface (6) passes the action to the JavaScript tutor engine, which (7) replies with correctness feedback, possibly an error feedback message, and student model updates; the UI (8) displays the feedback, and steps 5-8 repeat until the student has finished the problem. At that point, the UI (9) sends a performance summary with revised student model to TutorShop, and TutorShop (10) updates the LTI score. TutorShop chooses the next problem (adaptively, based on the student model) and returns to (4).

To test the technical integration of CTAT/TutorShop and edX, we tried out simple CTAT tutors in the edX Data Analytics and Learning MOOC. Specifically, CTAT tutors were included in 2 of the 8 weeks as part of the weekly activities/assignments Since this was the first attempt to incorporate CTAT tutors in MOOC assignments, the tutor activities were not required for students seeking to receive a course certificate. Students' performance on these two activities did not influence their final grades. As a result, only a small number of learners completed the tutor activities.

4 Discussion and Conclusion

Our pilot study demonstrates the feasibility of the MOOC/ITS integration between edX and CTAT/TutorShop. Although the pilot study involved a very simple tutor, the integration makes it possible to embed any CTAT tutor in an LTI-compliant MOOC or online course platform. To the best of our knowledge, it was the first technical demonstration of embedding an ITS in a MOOC, an important first step towards tutoring at scale. Testing with very large numbers of participants remains for future work. This technology integration may benefit other ITSs or ITS authoring tools, as some of the same steps might apply. Key is the separation of tutor interface, inner loop, and outer loop, so interface and inner loop can run on the client, while the outer loop is its own server-based web application. MOOC/ITS integration is attractive from a practical and from a research perspective. Tutors could enhance MOOCs by supporting some forms of learning by doing with detailed feedback and adaptive problem selection. The integration may enable MOOC researchers to address research questions about how learning by doing might best supplement other forms of learning in MOOCs and may open up opportunities for ITS researchers to do research at scale.

Acknowledgments. The work reported in this paper was supported by grants DRL-1418378 and SBE-0836012 and funding from Google.

References

1. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.R.: A new paradigm for intelligent tutoring systems: Example-Tracing tutors. International Journal of Artificial Intelligence in Education **19**, 105–154 (2009)
2. Baxter, J.A., Haycock, J.: Roles and student identities in online large course forums: Implications for practice. The International Review of Research in Open and Distance Learning **15** (2014)
3. Corbett, A., McLaughlin, M., Scarpinatto, K.C.: Modeling student knowledge: Cognitive tutors in high school and college. User Modeling and User-Adapted Interaction **10**, 81–108 (2000)
4. Mackness, J., Mak, S., Williams, R.: The ideals and reality of participating in a MOOC. In: Dirckinck-Holmfeld, L., Hodgson, V., Jones, C., De Laat, M., McConnell, D., Ryberg, T. (eds.) Proceedings of the 7th International Conference on Networked Learning 2010, pp. 266–275. University of Lancaster, Lancaster (2010)
5. Pane J.F., Griffin B.A., McCaffrey D.F., Karam R.: Effectiveness of cognitive tutor algebra I at scale. Educational Evaluation and Policy Analysis :0162373713507480 (2013)
6. Ritter, S., Koedinger, K.R.: An architecture for plug-in tutor agents. International Journal of Artificial Intelligence in Education **7**, 315–347 (1996)
7. Siemens, G.: Massive Open Online Courses: Innovation in education? Open Educational Resources: Innovation, Research and Practice **5** (2013)
8. VanLehn, K.: The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. Educational Psychologist **46**, 197–221 (2011)
9. VanLehn, K.: The behavior of tutoring systems. International Journal of Artificial Intelligence in Education **16**, 227–265 (2006)