EduGit: Toward a Platform for Publishing and Adopting Course Content

Michael C. Stewart(✉), Jason Forsyth, and Zamua O. Nasrawt

James Madison University, Harrisonburg, VA 22807, USA
{stewarmc,forsy2jb,nasrawzo}@jmu.edu
http://hcientist.com
http://www.jasonforsyth.net/

Abstract. In light of the continued dearth of an online Community of Practice for the collaborative development and distribution of educational materials for Human Computer Interaction, we propose EduGit. EduGit is a web application that serves as a platform for publishing, adopting, and collaboratively authoring educational resources. The system facilitates and incentivizes publishing and adoption of course materials through interactions with existing learning management systems and estimates impact through aggregate material adoption and usage statistics.

Keywords: Education · Sharing · Online community · Open educational resources · Community of practice

1 Introduction

There has been a sustained interest in electronically sharing educational materials since there have been electronic means to do so (e.g. [19]). Across various disciplines there have been periodic efforts addressing the need to share resources. This sustained interest is due to the lack of a satisfactory approach. Currently, authors of course content have little support for systematic publishing and updating of their materials. Instead these authors develop their material and then often share it with their current students via their institution's adopted Learning Management System (see: Fig. 1). When a contact wishes to use their materials, they communicate via some backchannel such as email and distribute the materials (e.g. [13]). This form of sharing is problematic for several reasons:

1. The sharing instructor may accidentally share more (e.g. student grades or identifying information).
2. The sharing instructor may accidentally omit content they intended to share.
3. The sharing instructor would have to manually record those people who are potentially adopting their materials.
4. The adopting instructor may forget to give proper attribution for the adopted materials.

G. Meiselwitz (Ed.): HCII 2019, LNCS 11579, pp. 217–226, 2019.
https://doi.org/10.1007/978-3-030-21905-5_17

5. The sharing and adopting instructors may not have established a protocol for the adopter to offer critique or other feedback to the sharer.
6. The adopting instructor may incorrectly assume they have permission to share the resources with other instructors.
7. Instructors who receive the materials from someone other than the author may make incorrect attribution for the resources.
8. Would-be adopters may be unable to access the materials due to a lack of sufficient technical skills.
9. Would-be adopters may be unable to access the materials due to limitations on accessing the publishing platform.
10. Materials being published in many different places that different institutions may not support equally could result in an information indexing/retrieval problem.

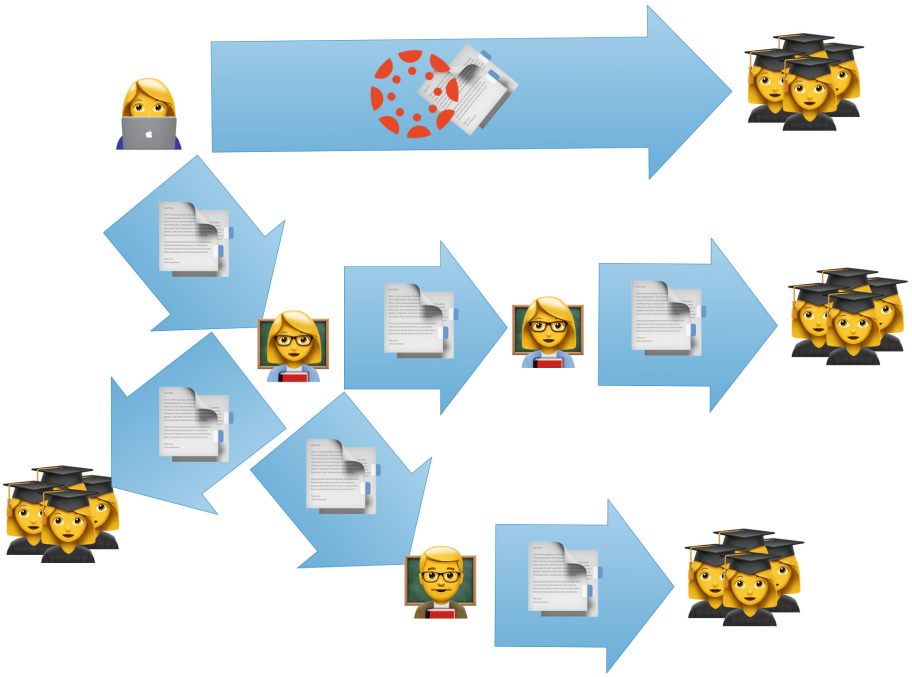

Fig. 1. Currently, authors share their curricular materials with students via an LMS, and then separately send files to other instructors.

In the Human Computer Interaction (HCI) community alone, there have been numerous efforts to define, share, and critique curricula and to build a community of practice for HCI Education [4–6,12,22]. These efforts have tackled various of the enumerated issues, and others beyond this list as well.

1.1 Design Space for Sharing Educational Materials

Through our on-going literature review, we will contribute a taxonomy that can help interested researchers see at a high-level the approaches that have been taken previously, and which parts of the sharing problem have been previously studied. So far we have identified several dimensions of the design space:

1. target audience
   (a) academic level
   (b) academic discipline
2. granularity of sharing: syllabi, large assignments (projects), quizzes, exams, in-class activities (e.g. slides), entire courses
3. ease of use for publisher
4. ease of use for adopter
5. license of published content
6. support for attribution
7. incentives

In the Discussion (Sect. 3.3), we will discuss where our approach falls in these dimensions.

We observe that in the field of software development there seem to be successful communities of practice engaged in sharing and reuse of software artifacts (e.g. [10]). Through (1) tracking the provenance of the source code, (2) nudging developers toward open licenses, and (3) facilitating the incorporation of improvements from third-parties into the original code bases, GitHub has fostered a vibrant community of collaborative software development. Following our participation in [22], we are developing EduGit [23] to support the sharing (publishing and adoption) of course materials, with lessons gleaned from GitHub and past education resource sharing efforts (e.g. those above as well as [2,3,8,9,20,21].

2 EduGit

EduGit is first a catalogued repository of educational materials. Interested authors of educational materials can easily publish their materials to the catalog. Interested educators can easily adopt those published materials for use in their own course. However, EduGit is more than a simple catalog of published educational materials.

2.1 EduGit Goals

EduGit aims to help educators establish a consequential practice of citing sources. The system tracks the adoptions of published materials to provide evidence of use. Similar to citation counts or the "h-index" for estimating researcher impact, statistics about actual adoption of an author's educational materials by other educators can help establish their educational impact. HCI educators may have differential need for such measures depending on their professional role [16]. EduGit has two primary Goals:

1. Facilitate
   - publication of educational resources.
   - adoption of educational resources.
2. Incentivize
   - publication of educational resources.
   - collaboration on educational resources.

In addition to the primary goals related to sharing educational resources, we are carefully designing EduGit to create a space for specific people to have specific interactions [11] that we believe are not well supported otherwise. We are designing EduGit to facilitate and incentivize critique of educational resources and pedagogical approaches.

2.2 System

EduGit is in the Proof-of-Concept development phase. The system is being developed as a web application (see: Fig. 2). It will consist of a relatively thick web client implemented as a ReactJS web application. The web application will

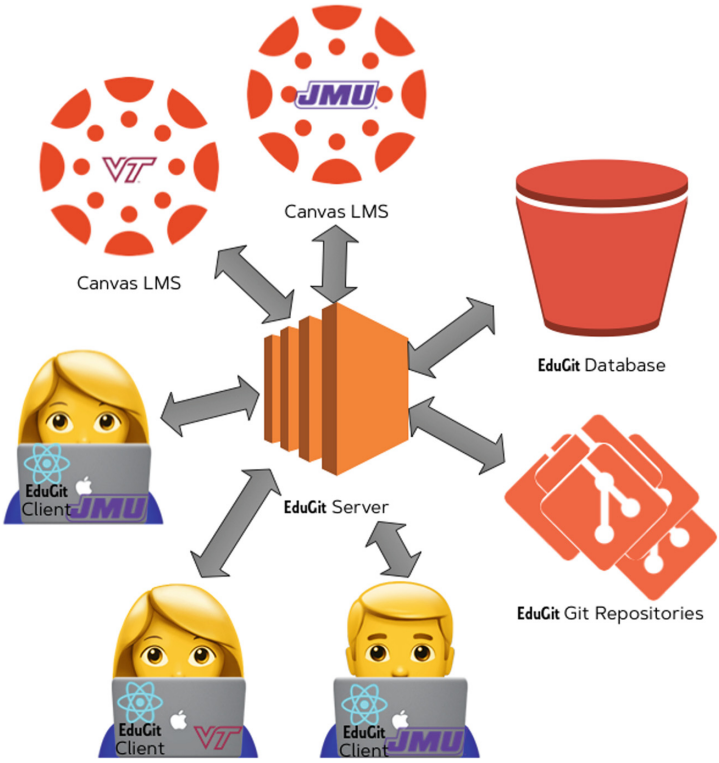

Fig. 2. Instructors at multiple institutions can authorize EduGit to access their Learning Management System to publish their courses, or to import published courses from EduGit for adoption.

communicate via RESTful API with the Django server. The server in turn, will communicate with (1) users' existing Learning Management Systems, (2) EduGit's own database, and (3) git repositories on the EduGit server that will facilitate versioning of the published materials.

2.3 Features

In the first version of EduGit, educational resources will only be able to be shared in units of a course (i.e. all content for a whole term of a course). While this design decision conveniently scopes the problem, it also reduces the administrative and cognitive efforts of publication and adoption because the syllabus, assignments, and other resources are all packaged together. This results in an adopter having more context than they might have through piecemeal sharing of individual resources.

A second simplification of the problem space that we employ in the first version of EduGit is limiting publication and adoption of courses to those available or to be offered in the Canvas Learning Management System (LMS) [14].

Publishing. An educator wishing to publish their course to the EduGit catalog will simply click a "Publish my course(s)" button, which will prompt them to sign in to their institution's Canvas instance. Having authenticated and granted EduGit access to their institutional Canvas account, the educator will see a detailed presentation of all the courses they have taught, each with a "Publish" button. Adding their course to the EduGit catalog simply requires clicking the corresponding "Publish" button.

Adopting. Similarly, an educator who finds a course they wish to adopt in the EduGit catalog can simply click the course's corresponding "Adopt" button. This will redirect the educator to their institution's Canvas authentication process. Having successfully authenticated and authorized EduGit to access their Canvas account, EduGit will present the adopting educator with a detailed list of their courses. Simply clicking one of these courses' corresponding "Select" button will import the published Canvas course contents into the adopter's course.

Collaboration and Updates. EduGit requires access to educators' existing institutional Canvas accounts to function. This access enables EduGit to support the educators in tracking changes to courses they have published or adopted (see: Fig. 3).

When an educator makes a change to a course that has been adopted by others, the educator can be prompted to indicate whether the changes might be beneficial (downstream) to the adopters. Likewise when an adopter makes changes to a Canvas course that they have adopted from another educator, EduGit can prompt the adopter to indicate whether it would be appropriate to propose the change (upstream) to the course from which they adopted the content.

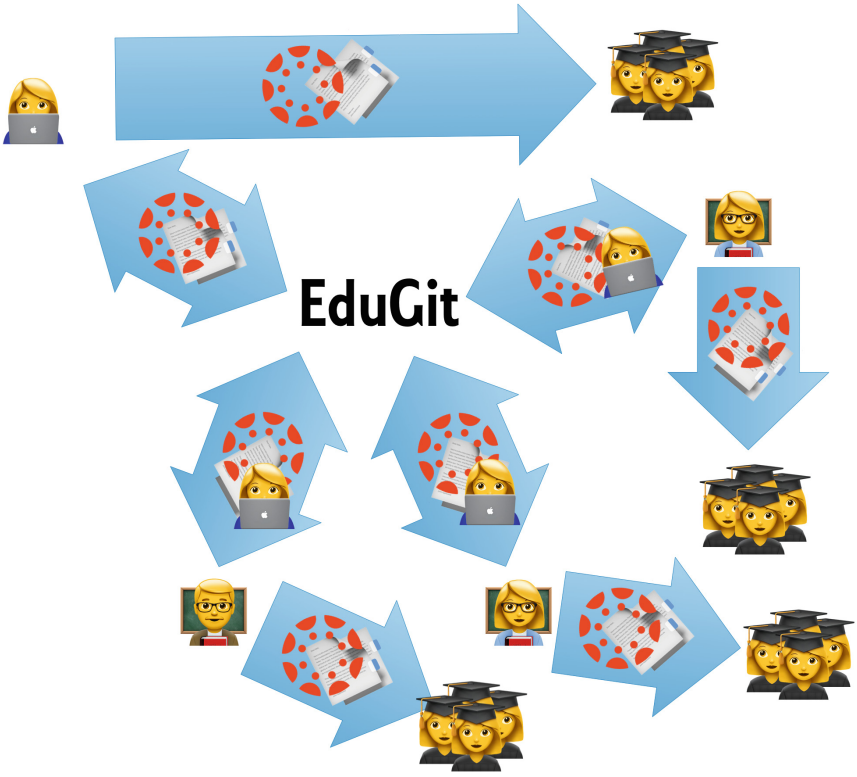

Fig. 3. Authors publish- and track the adoption of- their curricular materials through EduGit's access to their institution's adopted Learning Management System.

In choosing whether to share these updates up- or downstream, educators may wish to describe the changes and perhaps their rationale. Like git's commit messages, and GitHub's pull requests, these discussions of rationale for changes create a space for contextualized, detailed discussion of pedagogical choices that currently lack a designated arena.

2.4 User Experience

EduGit should be simple to use. Educators of any academic level and from any discipline should be able to publish and adopt courses with ease. Despite the "git" [24] suffix, publishers and adopters of content should have a simple graphical user interface in the EduGit web application for publishing and adopting content. Given the simplifications in the first version (see: Sect. 2.3), the process of publishing and adopting materials is simplified to publishing and adopting courses. This reduction in scope results in a simpler user experience for both (1) the actions of publishing and adopting, and (2) the pedagogical practice of comprehending and incorporating another educator's resources.

It is worth discussing why we mention restricting our problem scope to the Canvas LMS. The same features mentioned previously are available in Github and similar applications, and there even exist movements to re-purpose these applications for education without the need for any new tooling [7]. Initiatives of this nature provide standards for modeling course materials in these version control applications such that they can be viewed, shared, and tracked easily. The issue lies not with the initiative in any mechanical sense, but with the target audience. Interfacing with any current version control system is frankly not accessible to the broader population of educators. Git's primary interface is on a command line, and even the graphical interfaces are designed with code in mind. EduGit aims to wrap all of Git's features into a friendlier package that is easily accessible to a lay audience.

Other motivations for developing a new application have to do with institutional policies and practices. Universities cannot license LMS applications based solely on feature set. They must also proactively protect user privacy as mandated by law (e.g. FERPA in the United States). This limits the set of applications available to an educator. Current LMS applications only lack the sharing and provenance features; it would be unreasonable for a new application to re-implement all of the existing features. EduGit has two key advantages by integrating with existing LMS applications (initially only Canvas):

1. Standard features such as course and student management can still be handled by the LMS. This means EduGit will have a dramatically reduced development work load.
2. Sensitive data about students such as demographics and grades are not needed for our target features. This means institutional privacy policies won't block educators from using EduGit.

3 Discussion

EduGit will contribute to rich discussions about the development of online communities of practice for educational resources [15,17,18,25]. In addition to questions of the establishment and maintenance of the online community of practice, interesting questions of user experience and pedagogical practice confront EduGit.

3.1 Attribution

Currently, there is little meaningful citation of another educator's materials. While many of us include acknowledgements in our syllabi or even in specific project or assignment specifications, these citations are not indexed or tabulated. Perhaps this lack of tracking promotes less consistent attribution of authorship. Educators often participate in a "free-information" style of sharing course materials that encourages others to be more relaxed about citing sources, or asking for permission. EduGit at its core does not want to change this model. In fact

it wants to formalize this ideology by automating the tedious task of attribution. As a result of automatic, accurate, and persistent attribution, educators are incentivized to offer their course materials. Furthermore, contributions "up stream" from adopters to original authors are also attributed automatically. This incentivizes educators to share what has worked for them when they pilot a newly adopted course on their own students.

As EduGit aims to support attribution, there may be an issue in soliciting original content. That is, having lacked a meaningful outcome of attributing educational resources to the original authors, experienced educators may have a significant amount of unattributed content in "their" courses. These educators may feel uncomfortable publishing "their" course materials as they recognize their inability to properly cite their sources so long after they originally adopted them. An important research question as we develop EduGit will be how to help establish the new practice of more consistent attribution and how to transition from the status quo of less careful citing. Perhaps a statement on published courses that authors are sharing the courses as they teach them, but do not guarantee the content is theirs alone would help ease the concerns of publishing courses. Alternatively, similar to the "take-down" model on other user-contributed content web sites, EduGit could provide a feature to report content as belonging to the reporter. Initially, EduGit could rely on the publisher to agree that the reporter is the original source for the materials so that proper attribution can be made.

3.2 New Patterns of Collaboration

There is currently little professional space for developing and sharing quality educational materials. While some conferences (e.g. SIGCSE [1]) do have Experience Reports, there remain too few opportunities for educators to seek criticism of and collaboration on their educational materials. In EduGit, we can provide the ability to attribute to the contributor the change a publisher incorporates into their course. We expect this to help improve the (likelihood and) quality of the collaboration.

3.3 Locating EduGit

We have so far identified seven dimensions that help characterize the Design Space for Sharing Educational Materials (Sect. 1.1). EduGit aims to support instructors in any discipline. EduGit will not assume any specific academic level. EduGit will limit the granularity of sharing to the unit of a whole course (syllabus, assignments, exams, slides and other in-class content). EduGit will have extreme ease of use for the publisher and adopter alike, publishing or adopting the course in a just a few clicks. EduGit will initially support only open licenses (such as Creative Commons's share-a-like licenses). EduGit will automatically record attribution records when one instructor adopts a course from another, but EduGit will also support the ability of someone to indicate that an attribution may have been accidentally omitted to help establish new practices around

proper attribution in the educational communities. EduGit will thereby incentivize sharing and even critical feedback by offering evidence-based reports of users' contributions to published courses.

4 Future Work

In future versions of EduGit, we will eliminate the version 1 simplifications. We will support other learning management systems (beyond Canvas). We will support the ability to adopt portions of a course. We will support cross-LMS adoption. As the first implementation of tracking provenance will be implemented using git [24], and will internally model the relationships between courses as "forks" of repositories (e.g. on GitHub), the first version of EduGit will not provide the ability to compose multiple published courses into a single adopted course. We will remove this limitation in future versions to support a richer relationship between existing and newly adopted courses.

References

1. ACM. SIGCSE Technical Symposium 2019 (2019)
2. Akbar, M., et al.: How educators find educational resources online. In: Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education, ITiCSE 2011, New York, NY, USA, p. 367. ACM (2011)
3. Akbar, M., et al.: Digital library 2.0 for educational resources. In: Gradmann, S., Borri, F., Meghini, C., Schuldt, H. (eds.) TPDL 2011. LNCS, vol. 6966, pp. 89–100. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24469-8_11
4. Churchill, E., Preece, J., Bowser, A.: Developing a living HCI curriculum to support a global community. In: Proceedings of the Extended Abstracts of the 32nd Annual ACM Conference on Human Factors in Computing Systems - CHI EA 2014, New York, New York, USA, pp. 135–138. ACM Press (2014)
5. Churchill, E.F., Bowser, A., Preece, J.: Teaching and learning human-computer interaction. Interactions **20**(2), 44 (2013)
6. Churchill, E.F., Bowser, A., Preece, J.: The future of HCI education. Interactions **23**(2), 70–73 (2016)
7. Conrad, P., Kharitonova, Y.: UCSB CS Course Repos (2017)
8. Fouh, E., Sun, M., Shaffer, C.: OpenDSA: a creative commons active-ebook (abstract only). In: Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, SIGCSE 2012, New York, NY, USA, p. 721. ACM (2012)
9. Fox, E.A., et al.: Ensemble PDP-8: eight principles for distributed portals. In: Proceedings of the 10th Annual Joint Conference on Digital Libraries, JCDL 2010, New York, NY, USA, pp. 341–344. ACM (2010)
10. GitHub. GitHub (2019)
11. Harrison, S., Tatar, D.: Places: people, events, loci-the relation of semantic frames in the construction of place. Comput. Support. Coop. Work (CSCW) **17**(2–3), 97–133 (2008)
12. Hewett, T.T., et al.: ACM SIGCHI Curricula for Human-Computer Interaction. ACM, New York (1992)
13. Hu, H.H.: CS 1 POGIL Activities (2017)

14. Instructure. Canvas Learning Management System
15. Lampe, C., Wash, R., Velasquez, A., Ozkaya, E.: Motivations to participate in online communities. In: Proceedings of the 28th International Conference on Human Factors in Computing Systems - CHI 2010, New York, New York, USA, p. 1927. ACM Press (2010)
16. Peck, E.M., Smith, M.E., Stewart, M.C.: HCI for PUI: human-computer interaction for primarily-undergraduate institutions. In: WORKSHOP: Developing a Community of Practice to Support Global HCI Education of the 2018 CHI Conference on Human Factors in Computing Systems, vol. 2018 (2018)
17. Pitt, R.: Mainstreaming open textbooks: educator perspectives on the impact of OpenStax college open textbooks. Int. Rev. Res. Open Distrib. Learn. **16**(4) (2015)
18. Preece, J.: Online Communities: Designing Usability and Supporting Socialbilty. Wiley, New York (2000)
19. Rubincam, D.P.: Electronic book (1977)
20. Shaffer, C.A., Akbar, M., Alon, A.J.D., Stewart, M., Edwards, S.H.: Getting algorithm visualizations into the classroom. In: Proceedings of the 42nd ACM Technical Symposium on Computer Science Education, SIGCSE 2011 (accpetance rate: 34%), New York, NY, USA, pp. 129–134. ACM (2011)
21. Shaffer, C.A., et al.: Algorithm visualization: the state of the field. Trans. Comput. Educ. **10**(3), 9:1–9:22 (2010)
22. St-Cyr, O., MacDonald, C.M., Churchill, E.F., Preece, J.J., Bowser, A.: Developing a community of practice to support global HCI education. In: Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems - CHI 2018, New York, New York, USA, pp. 1–7. ACM Press (2018)
23. Stewart, M.C., Nasrawt, Z.O.: EduGit (2018)
24. Torvalds, L.: GIT
25. Wenger, E., McDermott, R.A., Snyder, W.: Cultivating Communities of Practice: A Guide to Managing Knowledge. Harvard Business Press, Boston (2002)