Maintaining and Extending MOOC Clickstream Curation in the MOOC-Learner-Project

Austin Liew, Erik Hemberg, Una-May O’Reilly, MIT CSAIL

Development and career advancement with MOOCs (Massive Open Online Course) partly depends on our ability to understand learners’ behavior in terms of how they navigate content. A primary challenge for this analysis is to work from a massive, time-oriented stream of click events that a MOOC platform logs as multiple students work simultaneously. To support sharing analytics the MOOCdb project\(^1\) provides MOOCDB\_translation\_software\(^2\) that curates such streams into a new format that makes it easier for each learner, type of learning activity, or a learner trajectory to be retrieved [4]. As MOOC learning design improves and platforms become more advanced, maintaining this and other MOOC software is a significant task [1]. “Code rot” sets in [3]. Software becomes unstable (e.g. hard to change without introducing bugs), test cases are neglected, and/or documentation become out of step with enhancements. For example, the moocRP [5] software has out of date install instructions and the software libraries and MOOCDB\_translation\_software does not support new platform logs, nor does it support more recent Coursera releases.

We analyze the code rot and related code quality measures of the existing MOOCDB\_translation\_software of the MOOCdb project. Results using industry standard software tools allow us to confirm intuition that rot has set in and to confirm what users report: it is hard to understand or change. We then describe how we have subsequently maintained and extended it, culminating in the release of a new version called MOOC-Learner-Curated (MLC). Understandability and stability have been addressed by creating a short software test path that enables a Test Driven Design (TDD) methodology that relies upon the quick and frequent client/user feedback this software project could expect. For reproducibility, a metadata table is added that contains the commit hash of the translation software used to generate the MySQL data. This provides metadata for developers to help debug issues with MLC. The commit hash also indicates the necessary versioning information to reproduce the same instance with the MLC. Extendability is demonstrated with VisMOOC [6] extensions for modules that implement specific functionality. The extensions allow innovations for learner analysis while they still separate and maintain the goal of creating a sufficiently generic MOOC schema to encapsulate the most important kinds of data. To achieve these upgrades, we used the LCCA (Legacy Code Change Algorithm) [2]. Our evaluation of improvement is based on the rate at which issues were resolved during different stages of LCCA. We report metrics associated with: 1. Understandability 2. Stability 3. Extendability, based on resistance to change 4. Reproducibility. MLC saw much more frequent changes in response to discovered issues once the project was significantly refactored and restructured. Stability is greatly increased with the addition of a test harness to the core modules.

References


\(^1\)http://moocdb.csail.mit.edu/
\(^2\)https://github.com/MOOCdb/Translation\_software