Versioning System as a Framework for Bidirectional Content Synchronization Among Learning Management Systems

Royyana Ijtiadie (1), Diana Purwitasari (1), Achmad Affandi (1), Yoshifumi Chisaki (2), Tsuyoshi Usagawa (2)
(1) Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
(2) Kumamoto University, Kumamoto, Japan
email: roy@its.ac.id

ABSTRACT

The condition of educational institutions in the urban area in Indonesia is quite different compared to the rural area. The problem lies mainly on demography factors and limited accessibility. This problem can be reduced by the usage of e-learning system. It shares learning contents between an institution in the urban area and its partner in the rural area. A license-free learning framework should be used by educational institutions in Indonesia, where there is no costs for licensing. A protocol framework covering content synchronization, unidirectionally and bidirectionally, has been proposed to support learning contents sharing over limited infrastructure. Beside this protocol framework, a versioning framework is also necessary. This paper discusses both frameworks.

Keywords: GIT, versioning framework, content synchronization, learning management systems

1 INTRODUCTION

The usage of Moodle as Learning Management System (LMS) becomes popular in educational institutions. However, it does not suit the situation of educational institutions in remote or rural area in Indonesia. The limitation includes the diversity of communication support, less qualified resources available, and difficulty to support sustainable long life learning. Discrepancy between areas is caused by demography factor and limited accessibility problems for developing infrastructure in rural area. To overcome it, the Ministry of Communication and Informatics with the Directorate General of Post and Telecommunication, has launched a Universal Service Obligation (USO) project that provides a better communication infrastructure. With the support of this infrastructure, e-learning is expected to help reducing burden works for developing and maintaining course materials and reducing digital gap among educational institutions. USO is one of ICT Strategy of Indonesia until 2015 including national movement for smart community development by providing nationwide infrastructure and facilitating the development of basic application and excellence software like e-Learning, e-Library, e-Health, e-Payment models.

E-learning in each institution has various materials which have potential to be complemented among them. Implementation of unidirectional synchronization systems with LMS Moodle [1][2][3] will synchronize learning objects from table that saves learning materials. Synchronization performed routinely at the time the network load is not busy. Performance observation of the implementation is focused on various available bandwidths in the networks [2]. The bandwidth availability varies from 56 kbps until 2 Mbps to transfer 50 Mbytes learning content, configured with and without other traffic in the networks. The results are the usage of wireless networks will drop significantly compared to the wired one, but variation of bandwidth capacity will not vary with the increase of available bandwidth.

The project of building e-learning protocol framework is initiated by JICA Technical Cooperation Project for Research and Education Development on Information and Communication Technology (PREDICT) in period 2006-2010 (Phase 1) and 2012-2015 (Phase 2). The goal is to overcome resources limitation in Eastern Part Indonesia (EPI) universities which are mostly in rural area.

The direction of unidirectional synchronization is “server to client(s)” only. Two main processes are to synchronize on server and to synchronize on client [1]. In the server, there will be (i) create backed-up database, (ii) create sub database for comparison, (iii) create synchronized database for client, and (iv) compress synchronized sql file of database. While in the client there will be (i) compare database on client with sub database on server, (ii) load compressed sql file from server if
necessary, (iii) decompress sql file, and (iv) copy sql file into database on client.

In order to deliver efficiently in synchronization among LMS over limited bandwidth network, hash values of learning content are generated using popular MD5 algorithm [4]. This could reduce bandwidth consumption and make the client receiving learning contents in smaller size.

Messages exchanged between LMSs to determine the updates are not efficient because extra messages from client to server should be sent first [1]. The updates usually come at small part rather than large part like creating certain groups of quiz questions, reviewing and revising last term questions, making new quiz, adding reference book links, etc. E-mail technology has been used widely as one of the popular communication tool. The synchronization of contents between only two LMSs had been examined using e-mail system [5][6]. Instead of sending a request e-mail, client only needs to check new updates in its mailbox and after that it does the synchronization routines.

Unidirectional synchronization implementation [7] considers an integrated service platform architecture that consists of service extension of such a point in rural area. Such architecture is constructed by considering choice of low power devices and equipments [8]. The architecture uses minimal required power consumption of equipment and systems and also distributes service area by considering bandwidth requirements.

The direction of unidirectional synchronization is one-way while the concept of bidirectional could communicate between server to client or vice versa. In regard of maintaining the version of learning contents, some versioning methods are needed. This paper proposed a framework that utilizes free mechanisms of learning system and versioning system. Therefore the free framework could be used by educational institutions in Indonesia without any additional costs for licensing. The next sections will discuss about free versioning framework and how to combine the versioning framework with learning management framework.

---

**Figure 1.** Proposed architecture of GIT Framework for LMS synchronization.

**Figure 2.** Processes in the proposed architecture of LMS synchronization.
2 GIT OVERVIEW

Git [9] is a version control software that has responsibility to record changes of files over time. Git is a software initially developed by Linux creator, Linus Torsvald to carry out the maintenance of linux kernel project. The creation of is intended to have a speedy process, simple design and strong support for non linear development which may contain thousands of parallel branches.

Git works by thinking of its data more like a set of snapshots of a mini filesystem. In addition, employing file system structure providing generic repository for any kind of files.

The commit process (saving the state of project/set of files) basically takes a picture of the actual whole set of files in that moment and stores a reference to such snapshot. To indicate changes over time, everything in git is check-summed before it is stored and later referred to by such check-sum (in this paper, it will be referred to as version number).

Branch is a movable pointer that points to a particular commit version. Branching can be used for diverging from main development line which can be intended to distinguish the harmful trial-and-effort zone from the primary production zone.

3 CONTENT SYNCHRONIZATION AMONG MULTIPLE LEARNING MANAGEMENT SYSTEMS

At educational institutions in which e-Learning has been used, creating course materials (hereinafter will be referred to as content) requires some extensive efforts at some extent, which causes additional burden in terms of teacher’s load of work. In addition, experience is highly necessary to create a better content. Employing content from a more experienced institution on particular course subject could be a way to reduce such burden. In the origin, the content is a subject to change over time assuming the author actively modified the content for their teaching purpose. In order to keep the shared content “fresh” against the origin, a particular procedure called content synchronization is necessary.

In the concept of unidirectional content synchronization among Learning Management Systems (LMS), two parties of LMSs are involved, namely Master and Slave. Master LMS provides the content to be shared. Slave LMSs will be the user of such content to be further being used in teaching and anything else. The content synchronization is carried out to make both contents at both sides consistent by sending the updates each time modification occurs at the origin at Master LMS.

Bidirectional synchronization is a concept that the content at Slave LMS is allowed to make modification. Furthermore, the newly improved content could be proposed to the Master LMS to be

Figure 3. Initial version of the content in both Slave LMS and Master LMS.

Figure 4. Different modification takes place at both sides.

Figure 5. The history of the snapshot after fetch operation has been done.
the current version. The synchronization in bidirectional way will give benefit in degree of content improvement since the user of the content (teacher) can actively involved in content development.

4 GIT AS A BIDIRECTIONAL CONTENT SYNCHRONIZATION FRAMEWORK

In regard to maintain the version of set of files related to the content, the use of Git as a versioning system is very potential. Distributed version control of Git is employed in order to meet the demand of content exchange between multiple LMSs on different hosts. In the rest of this section, we assume that the content comprises set of related files in a particular directory which is converted to and from using an LMS module. Figure 1 shows the general architecture of this approach. The version of the current content will be maintained by means of Git framework.

In the context of content synchronization, the workflow can be defined as the following (Figure 2). Initially, Master LMS side will have the content and maintain it in versioning system in release repository. Release repository is a repository containing several branches for holding the development line. The production branches is used to generate the content accessible to local LMS, while shared branch is aimed for providing sharing to other LMS.

At the Master LMS, the contributor may have their own repository while they need to push their content including the changes into the release repository. A module in LMS will handle the contribution lists as well as changes to be applied to production branch which further will be displayed in the local LMS, while at the same time, it is also saved into shared branch.

At the Slave LMS, the workflow works similarly, except that Slave LMS will be fetching from Master LMS remote repository. The contributed changes can be pushed to proposal branch. The shared branch will be subject to conflict management process. The conflict is likely to take place due to multiple contributors. The LMS module at Slave LMS has role of mergin the approved changes into shared branch.

Git allows multiple remote repositories as well as multiple branches allowing to have workflow in which each content contributor can have their own local repository and separate development line and is able to access to everyone else’s. Figure 2 shows the possible branches and repositories to be used in content synchronization.

To show how it is carried out, we provide a study case as initially commenced in Figure 3. Slave LMS performs clone operation to get the view of content at Master LMS as well as its history. A1, A2, A3 are the version number of the snapshot with A3 is the most recent version. As such, the Slave LMS will have the copy of the Master LMS’s content. The branch named ‘Production/Master’ which is previously on the Master LMS is carried to Slave LMS pointing to the similar version while in the local (Slave LMS) as well while branch ‘Production’ does the same.

Over period of time, content at Master LMS may change to certain modifications as well as at the Slave LMS, leading to a different pointing at branch pointer as shown in Figure 4.

In order to maintain the content at Slave LMS fresh with the origin, periodically, the fetch operation needs to be carried out. Figure 5 show the history of the snapshot after fetch operation has been done. The fetch operation will retrieve the version which does not exist in Slave LMS (A4, A5). Afterward, the branch ‘Production/Master’ will point to the most recent version.

Up until this point, different branching pointer can be used to identify which content are to be appeared to the users through LMS. Decision needs to be made by the teacher whether to use content fully from Master LMS. This case complies with unidirectional synchronization scheme.
Similarly, since the teacher at the Slave LMS is allowed to make content modification, a different version of content may take place, indicated by different branching pointer. Figure 1 has shown that ‘Production’ pointer at Slave LMS points to S2 version.

Bidirectional synchronization allows such branch to be submitted to Master LMS to be considered as a content modification proposal. Figure 6 show the Master LMS can fetch the versions which they do not have, and make branch based on it.

Having done such operation, the Master LMS will have content development line of Slave LMS indicated by its branch ‘Production/Slave’.

The next process, content modification proposal should be considered by either administrator or the origin author to be merged into the current version of the content.

The appearance of the content can be picked up from a particular branch. The administrator could show to the original author both branches and let him/her decide which one needs to be approved.

5 CONCLUSION

This paper discussed the content synchronization among multiple Learning Management Systems (LMS) by means of Git framework. The Git framework allows the author of the content or content developer to work like a software developer. Using Git terminology, the content can be viewed as a set of files which is tracked for modification occurrences over the time. In addition, using this framework, the content can be better maintained. This paper covers the early stage of the development. In the future the feature of rolling back the version needs to be added to give the author a freedom of editing without worry about losing the recent content.

REFERENCE
