Cloud-Based Content Synchronization Method for Collaborative Learning Environment

http://dx.doi.org/10.3991/ijac.v8i1.4410

Geunsik Lim, Donghwa Lee, and Sang-bum Suh
Samsung Electronics, Suwon, Republic of Korea

***Abstract*—Existing online education has moved from desktop environments to mobile environments due to the proliferation of smart devices. Online educational systems running on smart devices have the advantage of allowing users to learn online regardless of the location of the users. In particular, data synchronization enables users to cooperate on contents in real time anywhere by sharing their files via cloud storage. However, users cannot collaborate by simultaneously modifying files that are shared with one another. In this paper, we propose a method of content collaboration and a technique of history management that are based on distributed system structure. Moreover, our novel locking scheme automatically synchronizes data shared in the cloud for multiple users and multiple devices. Our proposed system helps users performing work on the collaborative content by automatically merging updated contents of cloud files among users.**

***Index Terms*—cloud computing, educational technology, mobile learning, revision control system.**

I. INTRODUCTION

Recently, many users have been able to engage with online education systems [1], [2] where they can collaborate with one another over long distances through their smart devices. *Smart device* refers to the mobile device that can download and install an application from an application store that corresponds to the specific smart device in use. Figure 1 shows that cloud-based online education enables students to learn using on-line materials, even though students do not physically attend school. However, existing cloud services [3] do not have a suitable synchronization facility for the case in which the system requires the synchronization of shared data among the users and when multiple users try to access the shared data at the same time. Also, existing cloud storage [4] is not aware of multiple users using multiple devices.

In other words, current systems cannot ensure the consistency of the shared data since only one user can modify the shared file. For example, Figure 2 shows that *Dropbox* produces a transaction error, [5] called a *file conflict*, when the two users try to save a modified version of the same shared file. Moreover, existing cloud services focus on the case of a single user with multiple devices rather than the case where more than users are collaborating. Therefore, the research of next generation cloud storage techniques become crucial to support consistent data synchronization [6] for content collaboration among users within an internet-of-things (IoT) environment. IoT refers to the interconnection of uniquely identifiable embedded computing devices connected to the existing internet infrastructure. IoT is expected to offer advanced connectivity to devices, systems, and services that goes beyond machine-to-machine communication.

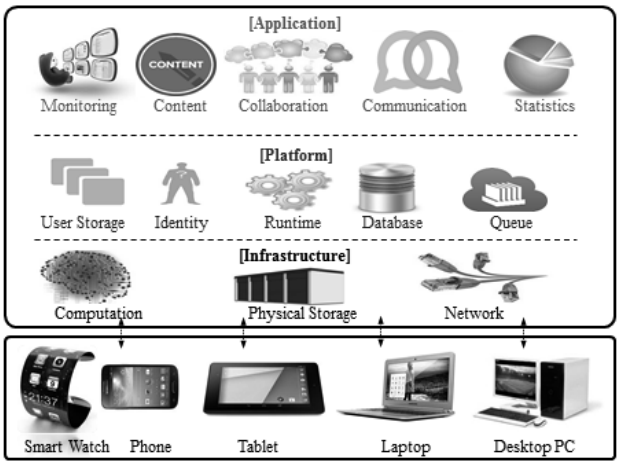

Figure 1. Application, platform, and infrastructure in cloud computing environment to provide resources of server to the various devices.

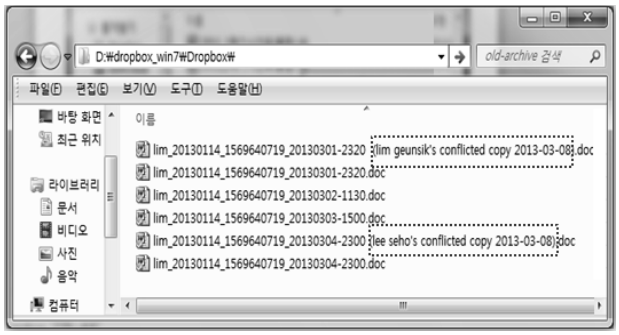

Figure 2. A conflict status of the shared cloud files (e.g. conflicted copy) due to the multiple access among the users.

The remainder of this paper is organized as follows. Section II gives an overview of related work. Section III addresses the design and the implementation of the proposed techniques. Section IV shows the experimental results. Finally, Section V provides the conclusion for the paper.

II. RELATED WORK

Dropbox [6] supports file sharing and network file synchronization in order to easily share files over the cloud anywhere. Cloud computing [8] helps users to share data and to synchronize such data among the user's devices. However, current cloud computing do not support cases where users modify shared files simultaneously. Moreo-

ver, users cannot edit shared files at the same time without *file conflicts* for cases where there are multiple users with multiple devices.

GoogleDrive [9], [10] has completely supported for different operating systems, such as Windows, Linux, and Android, and provides a document-based communication network to modify shared documents using a web browser. Moreover, many users can simultaneously modify shared files through the transaction mechanism provided by Google docs. However, *GoogleDrive* focuses on providing file sharing services using a public cloud rather than a private cloud. Also, this system does not automatically integrate the history management of shared files and the cloud storage. In other words, there is no history about the particular changes of the file.

This paper addresses the use case where a multi-user-aware file sharing technique and history management technique that are implemented to share files used within collaborative learning and teaching environment. Our proposed system describes an advanced private cloud computing [11] that can provide cloud storage for users.

III. PROPOSAL: CLOUD-BASED CONTENT COOPERATION SYSTEM

Our proposed system manages the changes in the content using a modified content unit without access to the time unit of the file in order to synchronize the shared files. Moreover, the proposed system supports a novel synchronization mechanism [12] that is based on a locking algorithm using the serialized queue for the case where many users try to access the shared file. These approaches overcome the problem when the consistency of the shared file is not guaranteed due to the simultaneous access of the shared file for the environments with multiple users and multiple devices.

A. Design

We describe how to interconnect distributed revision controller and the shared data via cloud computing in order for many users to be able to edit the same content in cloud storage. This paper focuses on a cloud-based content collaboration technique [13] that can automatically merge and track shared content in real time.

The proposed system only requires a cloud server [14] without a source code management (SCM) server since our system is based on the synchronization mechanism of distributed cloud files. In other words, we do not need a source code management server to manage the modified content of a shared file. We can synchronize the data that has changed by using the cloud computing infrastructure and implementing the history management of the shared file with distributed clients. This technique has been named the "cloud-based content collaborative system to assist in a collaborative learning and teaching network environment" (COCO).

Figure 3 shows the entire process flow of the proposed system. The proposed system consists of server-side and client-side. The server-side synchronizes and archives the user data in the cloud server, and the server-side consists of three components as follows to aware multiple users and multiple devices in the collaborative learning and teaching in mobile environments:

1) Service daemon: synchronizes the shared data when the users modify the data held in the cloud server with their devices.

2) Cloud storage: saves the content generated by the users [14].

3) Web application: supports file upload and download via a web browser. Web interface should be made with HTML5 and JavaScript to support most of the operating systems such as Windows, Android mobile platform, and Ubuntu. In real environment, we can avoid the unnecessary engineering cost which is generated by the different software platform architectures by selecting the compatible web interface without the platform specific user interface.

The client-side transfers the access status and the content of the data to server-side when the users modify the contents of the file. It consists of three components as follows:

a) Content sync agent: synchronizes the changed content between the server and the client in real time.

b) Revision controller: manages the changed file content using a distributed revision control method to update the data change of the shared file from the distributed clients.

c) Cloud client: manages the access time of the content and information of the client.

In this paper, we contribute three achievements that are useful for an online collaborative education system in real time as follows:

- Synchronization mechanism of shared file based on changed data: manages the changes in the file content without using a file access unit to support fine-grained data synchronization.
- Distributed content tracking model: supports the distributed system mechanism [15] using a peer-to-peer communication technique without a centralized model. Therefore, an additional SCM server by utilizing storage mechanism of the cloud server and computing resource of the client is not required in the proposed content tracking model.
- Automatic revision control technique: considers an environment including multiple users and multiple devices in order to solve the *file conflicts* that happen in a case where many users try to modify shared content simultaneously.

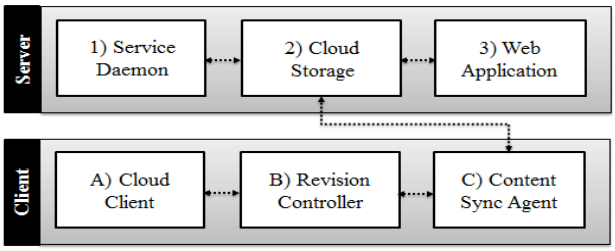

Figure 3. Operation flow of the proposed system (COCO). The COCO refers to the COloud-based COtent Collavotravie System.

B. Changed Data-Based Content Synchronization

The existing synchronization techniques check the access status of the file periodically against different access time of the shared files. In this case, when two users try to simultaneously save the shared file, the users have to manually manage the conflicts in the shared files due to absence of a synchronization operation.

To solve this problem, we have to avoid opening of the shared file in case that a user opens a shared file that has already been opened by another user. When the user tries to close the opened file after modifying it, the client-side must operate an unlocking mechanism due to the locked status of the shared file. Next, the client has to guarantee the access status of the file to enter the locking operation when the user of another device tries to modify the shared file.

At this time, the server-side preserves the information of the modified content in case that the shared file is modified by another user. It is possible for us to easily manage collaboration of the share content by using the history of the file content against the preserved information. Figure 4 shows how we can control the history management of files using the modified file content as a minimal unit of the transaction, as compared to that of the existing system. The existing system cannot handle modifications of the same file in order to avoid file conflicts because it performs a transaction [11] for the file modification by using the access time to synchronize the shared file in the cloud without using any locking mechanism for the cloud file. Our proposed system solves the problem of *file conflicts* of the existing system with the modified content based fine-grained locking mechanism is called object-based transaction when many users try to modify the content of the shared file.

C. Distributed Content Tracking Technique

Figure 5 shows the distributed content control structure that can manage information of the changed file content when users modify the shared cloud file at the client-side [16]. The cloud based distributed history management technique does not need additional management system that manage the changed history of the clients.

The history information of modified contents from the client can be replaced with cloud server based network synchronization technique. The mechanism helps that each client does not need SCM server by saving the history information of the updated files into the user storage of cloud server in real time. As a result of that, the history data of the changed files are automatically synchronized thanks to the network synchronization mechanism of the cloud computing infrastructure.

D. Automatic Revision Control Method

When the users try to modify the shared files at the same time, the cloud server has to equip the serialization facility to handle the simultaneous access from the clients. Figure 6 shows how the proposed system synchronizes data between the server-side and the client-side in the case where a user creates new content using a mobile device so that the system can recognize a multi-user and multi-device environment [16].

The proposed system synchronizes the data of the shared files by processing the *check-in* and the *check-out* commands [16] according to the access status and the access time of the user's shared file. The *revision controller* of client-side automatically merges the user data when another client sends the *check-in* commands to the cloud server. With current systems, if changes are automatically merged with *auto merging (experimental)* option and *avoid file conflict* option whenever the user tries to save modified content of a shared file, too much network overhead can be generated. Therefore, if the users try to close the modified file when the client manually sends the *check-in* command to the cloud server, we can minimize the network overhead for frequent *send/receive* operations when the content of the shared file is changed. At this time,

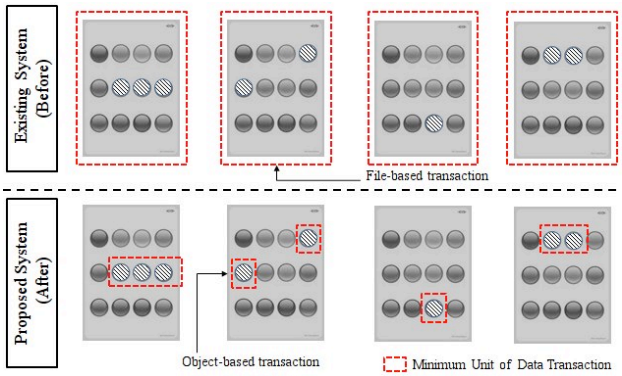

Figure 4. Operation comparison of data synchronization between the existing system and the proposed system. The circle symbol refers to the changed data as a minimal object. The red rectangle refers to the minimum transaction unit to synchronize the changed data.

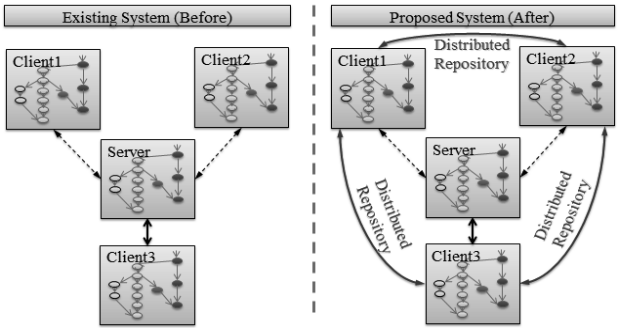

Figure 5. The difference with respect to content history management between the existing system and the proposed system. The proposed system does not need to have single central place (e.g. *server*) where users commit their changes. Each user can have its own repository is called distributed repository, and he can pull from each other repositories.

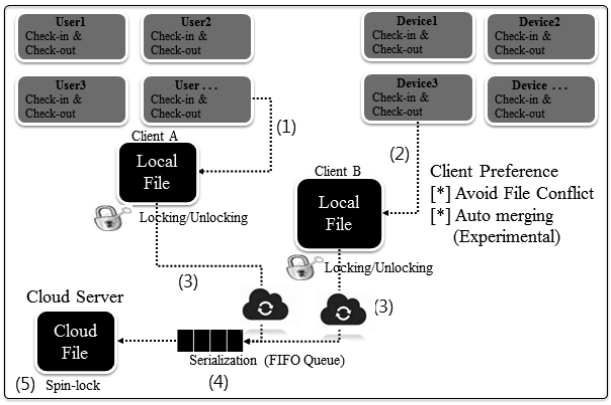

Figure 6. Flow chart of the file locking operation to collaborate the shared files among the users

the cloud server serializes the requests from the clients using a *First in, First out* (FIFO) behavior [17]. This technique is effective and practical because some users have to limit network usage over 3G networks in the real environment [18].

Figure 7 explains how the proposed system shares the data in the cloud storage while managing the revision control system when the user modifies the shared file. At this time, a history management repository to integrate the modified shared data [19] is a client without a server. Each client is allocated a unique directory with a different name. When the file content is modified by using a directory that exists with a different name [20], the client automatically synchronizes the modified content to cloud server. Next, the shared directory of all clients keeps the same file and same history information by synchronizing the shared data same as with a file in the cloud of the different device using the most up-to-date data standard that is updated in the cloud server. The *Directory Sharing* operation of the client-side shows how the history management of the shared contents will be synchronized by file update mechanism of cloud computing infrastructure. In this case, we need the new data synchronization solution to resolve inconsistency problem of synchronized data files because the cloud computing infrastructure does not guarantee the consistent transaction of the shared files among the users. We can simply resolve the inconsistent status of history data happened, according to the distributed revision control method [16] by managing the user name uniquely with *./member{1|2|3|...}* directory structure in the client. The unique directory name is sequentially generated by attaching an increased number. Although the users will gradually possess more than two devices, the method can simply avoid the inconsistency of the shared files according to the multi-users and the multi-devices environments. Finally, the cloud server executes a serialization task such as Figure 6 when the user saves the content of the file in the cloud according to the time of the cloud server.

IV. EVALUATION

To verify the effects of the proposed system, we established a server environment with an Intel i7 Quad core CPU, 8GiB of DDR3 RAM, and 512GiB of SSD running Community Enterprise Operating System (CENTOS) 6.6 with Linux 3.10. The server consists of an Apache webserver, PHP5 scripting language, and MySQL database. The server-side supports the web interface and the web storage with HTML5 and PHP script language for the users that are using the different operating systems. Figure 8 shows how the client-side consists of three clients, such as a server-based web client, desktop client, and a mobile client. We set up the cloud computing infrastructure with the network protocol of the open source project, called *Owncloud*, to maintain the openness.

A. Scenario for evaluation

We evaluated how the proposed system automatically merges the content without causing any *file conflict* for the shared file when users modify the content of the shared cloud file at the same time;

- Cloud server: manages the cloud storage. [21] When the user changes the content of a file, the service daemon of the server is in charge of executing the transaction of the data with a spin lock [22] mechanism and a FIFO queue.
- Desktop software: is client software for desktop computer of a user. When the user modifies the shared cloud file and saves the modified file, the client requests the data synchronization of the cloud file to the cloud server. The *content sync agent* is responsible for the distributed revision control to synchronize the updated contents among the clients.
- Mobile software: allows mobile users to modify the shared file with their mobile devices. We implemented the HTML-based web application to minimize the redundant development cost because of a lot of mobile devices such as Android Phone, Windows Phone, and iPhone. At this time, the internal operation flow of the *content sync agent* is similar to that of the desktop software.

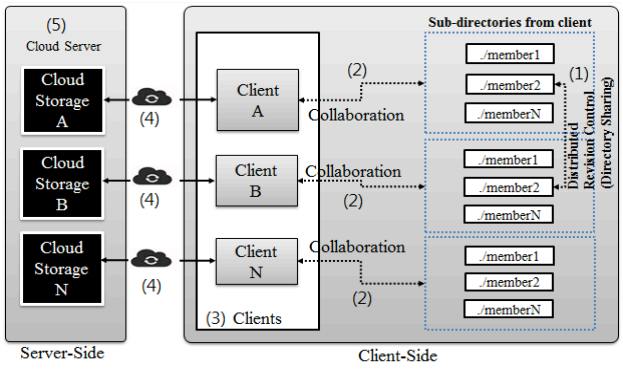

Figure 7. The merging flow of the distributed cloud files for collaborative content history management form the clients

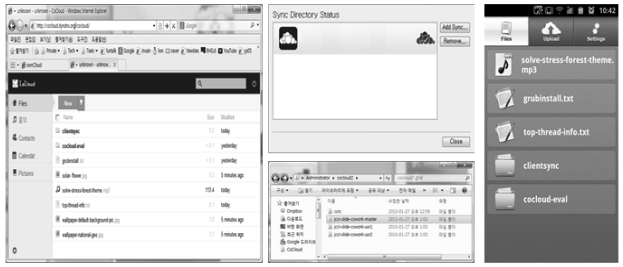

Figure 8. Server, Desktop, and Mobile for contents cooperation among users: web interface and web storage for server, client program for desktop computer, web-based GUI application for different mobile systems.

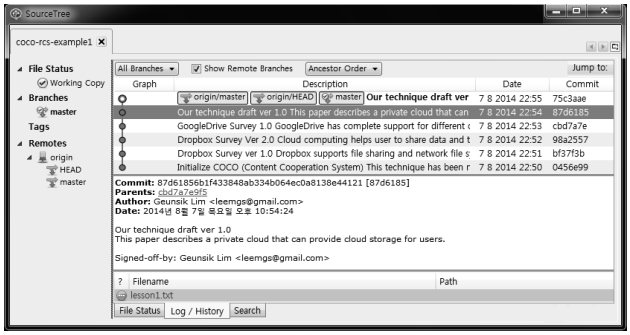

Figure 9. The content revision history of the shared cloud file among the users

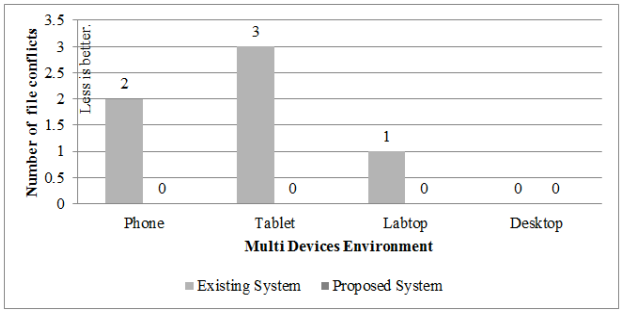

Figure 10. File conflicts in multi-devices

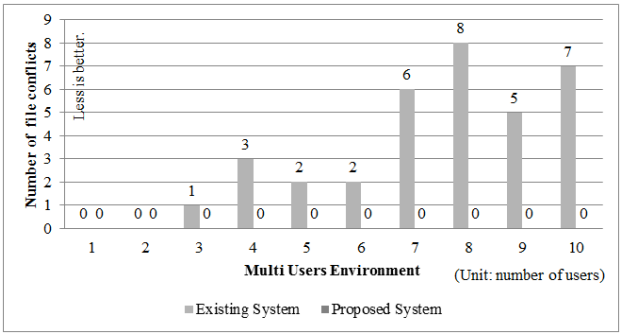

Figure 11. File conflicts in multi-users and single device

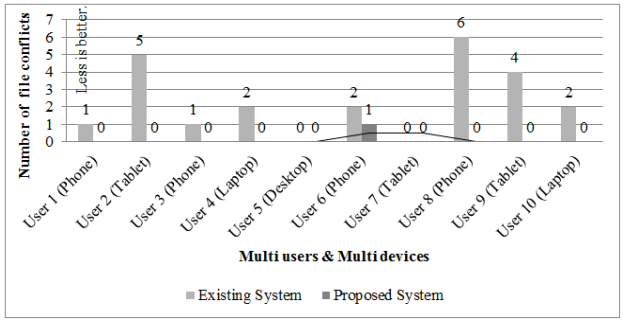

Figure 12. File conflicts in multi-users. Each user has one device.

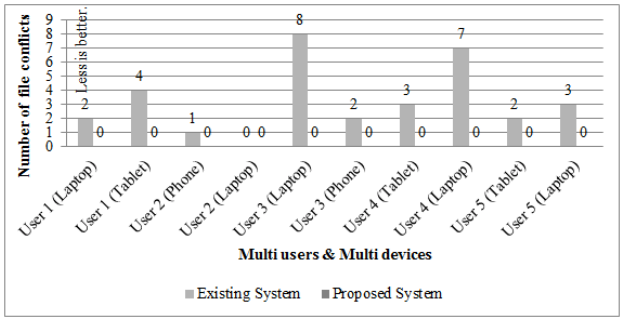

Figure 13. File conflicts in multi-users. Each user has two devices.

B. *Experimental result: Revision History of Cloud File*

Figure 9 shows how the modified content is saved into the distributed history management directory when many users execute the *check-in* command to merge modified content to the file. We can manage the modified content for an offline meeting by preserving the history of modified information [16] among the users from many different devices. We additionally made the *content sync agent* based on the distributed revision control software, called *Mercurial*. *Mercurial* is a distributed revision control tool that takes a peer-to-peer approach for software developers as open source project.

C. *Experimental result: Multi-devices*

Figure 10 shows the experimental result of a *file conflict* from a single-user and a multi-device environment. The proposed system guarantees zero *file conflict* even when a user tries to modify a shared file from multiple devices simultaneously, as compared to an existing system. In Figure 9, the x-axis refers to the name of the device of the user. The y-axis refers to the number of *file conflicts* when the user tries to modify a shared file using multiple devices at the same time. The existing system does not care the multiple access of the shared contents among the users. Therefore, the data synchronization mechanism of the existing system cannot solve *file conflicts* when the user tries to modify a shared file over two devices.

D. *Experimental result: Multi-users*

Figure 11 shows a comparison between the existing system and the proposed system in terms of a multi-user environment. Each user only has one device, is not multiple devices.

We evaluated the performance of our proposed system with respect to the data synchronization of the shared cloud file without any *file conflicts* for the case where many users modify the shared file at the same time. The x-axis refers to the number of users that access to the shared file simultaneously. The y-axis refers to the number of *file conflicts* of the shared file. From our experiment, we had zero *file conflict* resulting from the automatic merging operation such as *auto merging (experimental)* option indicated in Figure 6 when many users modify the content of the shared file from anywhere at the same time.

E. *Experimental result: Multi-users and multi-devices*

Figure 12 and Figure 13 show a comparison between the existing system and the proposed system in terms of a multi-user and a multi-device environment. The proposed system recognizes multiple users with multiple devices better than the existing system, without any *file conflicts*, in order to support a collaborative educational system.

Figure 12 shows the experimental results of the proposed system that generated one *file conflict* even though 10 users accessed the shared file at the same time.

Our analysis indicates that this problem sometimes happens since the user did not execute the *check-out* command after setting a manual lock without an automatic locking mechanism in order to merge the modified content manually.

F. *The difference between the existing system and the proposed system*

Table 1 summarizes the strong points of the proposed system against those of the existing system. The proposed system synchronizes modified content units in order, being aware of multiple users with multiple devices. Moreover, it supports an automatic history management technique for the modified cloud file in the case that a number of users try to modify the shared file simultaneously.

TABLE I.
FUNCTIONAL COMPARISON BETWEEN THE EXISTING SYSTEM AND OUR PROPOSED SYSTEM

Content	The Existing System	The Proposed System
Web-hard	Yes	Yes
Network File Sync	Yes	Yes
SCM Method	Modified-file based	Changed-content based
SCM Server Cost	Needed (Need additional server because there is not data locking mechanism)	Needless (Doesn't need additional server because cloud server is SCM server)
SCM Multi-user and SCM Multi-device	No (file conflict occurs)	FIFO-based spin-locking mechanism
SCM Linkage	No	Sync function of cloud client (Preference: Manual, Automatic)
SCM Merging	Coarse-grained method (file-unit based)	Fine-grained method (check-in time based)

The proposed techniques solve the problem of an increased server management cost through running a combined cloud storage and source code management system. We described the automatic merging technique for the case where many users modify the shared cloud file based on the distributed system structure for collaborative learning in mobile environments.

V. CONCLUSION

The proliferation of smart devices, such as smart phones, smart tablets, and smart laptop has quickly popularized mobile cloud services. Existing online education has moved from desktop environment to mobile due to the proliferation of smart devices. Therefore, it is important for many users to be able to edit content of cloud files in online environment from anywhere to overcome the collaboration limitation, which *Dropbox* [6] does not provide.

This paper proposes a cloud-based content cooperation system that can modify cloud files in multi-user multi-device environments. It consists of a modified content-based cloud file locking mechanism, distributed file history tracking, automatic *check-in* and *check-out* method, and automatic revision control components. Moreover, we addressed the automatic merging of changes in the content of a cloud file by using the modified content unit. This mechanism helps a number of users to access shared cloud files using a FIFO-based spin-lock locking mechanism to avoid file conflict problem. In our experiment, we verified that the proposed system could merge shared files without causing any file conflict against those of an existing system. The proposed system could browse the history of the shared file to track the task contents of each of the members.

ACKNOWLEDGMENT

We gratefully acknowledge TaeYun Kwon and Seungho Lee for their feedback and comments, which helped improve the content and presentation of this paper.

REFERENCES

[1] G. Zurita and M. Nussbaum, "A constructivist mobile learning environment supported by a wireless handheld network," *Journal of Computer Assisted Learning*, pp. 235-243, Aug. 2004. http://dx.doi.org/10.1111/j.1365-2729.2004.00089.x

[2] W. H. Quick, N. J. Wright, A. Rashid, and H. Herjanto, "Collaborative Networked Learning in Manufacturing," *International Journal of Advanced Corporate Learning*, Vol. 7, No. 4, 2014.

[3] X. Jian, "An Optimized Solution for Mobile Environment Using Mobile Cloud Computing," *in Proceedings of the Wireless Communications, Networking and Mobile Computing*, pp. 24-26, Sep. 2009.

[4] J. Wu, L. Ping, X. Ge, Y. Wang, and J. Fu, "Cloud storage as the infrastructure of cloud computing," *in Proceedings of the Intelligent Computing and Cognitive Informatics (ICICCI)*, 2010.

[5] D. Quick and K. R. Choo, "Dropbox analysis: Data remnants on user machines," *Digital Investigation*, 2013. http://dx.doi.org/10.1016/j.diin.2013.02.003

[6] H. Wang, R. Shea, F. Wang, and J. Liu, "On the impact of virtualization on dropbox-like cloud file storage/synchronization services," *in Proceedings of the IEEE international workshop on quality of service*, 2012.

[7] I. Drago, M. Mellia, M. M. Munafo, A. Sperotto, R. Sadre, and A. Pras, "Inside dropbox: understanding personal cloud storage services," *in Proceedings of the ACM conference on Internet measurement conference*, 2012.

[8] X. Li, H. Zhang, and Y. Zhang, "Deploying Mobile Computation in Cloud Service," *in Proceedings of the Cloud Computing Lecture Notes in Computer Science*, vol. 5931, pp. 301-311, 2009.

[9] D. Quick and K. R. Choo, "Google Drive: Forensic analysis of data remnants," *Journal of Network and Computer Applications*, pp. 179-193, Apr. 2014. http://dx.doi.org/10.1016/j.jnca.2013.09.016

[10] D. Huang, T. Xing, and H. Wu, "Mobile cloud computing service models: a user-centric approach," *in Proceedings of the IEEE Network*, pp. 6-11, Sep. 2013.

[11] P. Lowenthal and D. Thomas, "Death to the digital dropbox: Rethinking student privacy and public performance," *Educause Quarterly*, 2010.

[12] Z. Li, C. Wilson, Z. Jiang, Y. Liu, B. Y. Zhao, C. Jin, Z. Zhang, and Y. Dai, "Efficient batched synchronization in dropbox-like cloud storage services," *Middleware 2013*, Springer Berlin Heidelberg, 2013.

[13] G. Lim, D. Lee, and S. Suh, "Cloud-Based Content Cooperation System to Assist Collaborative Learning Environment," *in Proceedings of the IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, Dec. 2014.

[14] Y. Zhang, C. Dragga, and A. Arpaci-Dusseau, "box: Towards reliability and consistency in dropbox-like file synchronization services," *in Proceedings of the Workshop on Hot Topics in Storage and File Systems*, 2013.

[15] T. Xing, D. Huang, S. Ata, and D. Medhi, "MobiCloud: A geo-distributed mobile cloud computing platform," *in Proceedings of the International Conference and Workshop on Systems Virtualization Management*, pp.164-168, Oct. 2012.

[16] B. d. Alwis, J. Sillito, "Why are software projects moving from centralized to decentralized version control systems?," *in Proceedings of the ICSE Workshop on Cooperative and Human Aspects on Software Engineering*, pp. 36-39, May 2009.

[17] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," *in Proceedings of the INFOCOM*, 2010.

[18] K. Kumar and Y. Lu, "Cloud computing for mobile users: Can offloading computation save energy?," *Computer* 43.4, pp. 51-56, 2010. http://dx.doi.org/10.1109/MC.2010.98

[19] D. J. Abadi, "Data management in the cloud: Limitations and opportunities," *IEEE Data Eng*. Bull. 32.1, pp. 3-12, 2009.

[20] D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: Deduplication in cloud storage," *Security & Privacy*, pp. 40-47, 2010.

[21] W. Zeng, Y. Zhao, K. Ou, and W. Song, "Research on cloud storage architecture and key technologies," *in Proceedings of the International Conference on Interaction Sciences: Information Technology*, Culture and Human, 2009.

[22] J. Ouyang and J. R. Lange, "Preemptable ticket spinlocks: improving consolidated performance in the cloud," *ACM SIGPLAN Notices*, pp. 191-200, 2013. http://dx.doi.org/10.1145/2517326.2451549

AUTHORS

Geunsik Lim received his B.S. degree in Computer Science and Engineering from Ajou University, in Korea in 2003. He received his M.S. degree in the College of Information and Communication Engineering from Sungkyunkwan University. His current research interests include system optimization, embedded operating systems, mobile platforms, and multicores. He is now Senior Software Engineer at Frontier CS Lab., Software R&D Center, Samsung Electronics, Suwon, Republic of Korea. (e-mail: geunsik.lim@samsung.com).

Donghwa Lee received his B.S. degree and M.S. degree in the School of Electrical Engineering and Computer Science from Kyungpook University. He is assistant software engineer at the Platform Solution Lab., Software R&D Center, Samsung Electronics Suwon, Republic of Korea. (e-mail: dh09.lee@samsung.com).

Sang-bum Suh is Vice President in Platform Solution Lab., Software R&D Center, Samsung Electronics and has developed Tizen Platform and led the Xen ARM virtualization project in Xen.org. He graduated with PhD in Computer Science from University of Cambridge, the United Kingdom. (e-mail: sbuk.suh@samsung.com).

This article is an extended and modified version of a paper presented at TALE 2014, the IEEE International Conference on Teaching, Assessment, and Learning for Engineering in the Workplace, held from December 8-10, 2014 in New Zealand. Submitted 13 February 2015. Published as resubmitted by the authors 10 March 2015.