ISSN (Online) : 0974-5645
ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 8(S1), 145–153, January 2015* DOI : 10.17485/ijst/2015/v8iS1/57881

A Synchronization System for an Educational Content Viewer

Sun-Ock Kwon[1], Mihye Kim[2] and Kwan-Hee Yoo[1*]

[1]Department of Digital Informatics and Convergence and Software Engineering, Chungbuk National University, Cheongju, Chungbuk, 362-763, South Korea; khyoo@chungbuk.ac.kr
[2]Department of Computer Science Education, Catholic University of Daegu, Gyeongsansi, Gyeongbuk, 712-702, South Korea.

Abstract

With an increasing number of users having more than one smart device, an environment for delivering the same content through multiple devices has been established via wired and wireless network connections. Consequently, online education services that have provided content on one device screen by dividing it into multi pieces have changed to providing it through more than one smart device using a synchronization system embedded in a content viewer application or an e-book reader. However, most existing content viewers do not support content synchronization. Some do support it, but are platform-dependent. To address these shortcomings, we propose a synchronization system embedded in an educational content viewer that supports a wide range of content types and is platform-independent. To examine the performance of the proposed synchronization system, we conducted experiments on processing time for synchronization by applying it with different network connection types and between smart devices with different operating systems (OSs). The synchronization system performed within a reasonable time on wireless (Wi-Fi) and long-term evolution (LTE) networks, and most of the content, except three dimensional (3D) objects, synchronized in real-time regardless of OS type.

Keywords: Content Viewer, e-Book Viewer, Synchronization

1. Introduction

With the rapid rise in smart device users, various systems to support user convenience are being developed based on these smart devices[1, 2]. In addition, because an increasing number of users have more than one smart device, an environment for delivering the same content through multiple devices has been established via wired and wireless network connections[3]. Consequently, online education services that provide content on one device screen, dividing it into multiple aspects, have changed to providing it through more than one smart device using a synchronization process[4, 5]. In the field of education, there is ongoing research directed at improving learning experiences using N-screen services, which is a synchronization technology for a seamless display among multiple devices. Jang-Ho Lee[6] proposed a cloud-based real-time education system that provided lectures between the PC of an instructor and the smartphones of multiple users. Jae-Ho Kim et al[7]., introduced an educational system that enabled infants/toddlers to play games with their parents using multiple smart devices, in addition to allowing parents to monitor their children's learning. These systems, however, provide only limited features on some smart devices; they are also still in the research stage and have not been commercialized.

Content is delivered to smart devices using a content viewer application or an e-book viewer. To support synchronization services in a content viewer, it is necessary to include features to process content synchronization. The most common commercial content or e-book viewers include EPUB3.0-based viewers and Apple's "iBooks 2"[8, 9]. Such viewers support various e-book features and interfaces for the convenience of users. However, they have some limitations: EPUB3.0-based e-book viewers do not support interactive three-dimensional (3D) objects

*Author for correspondence

or content synchronization[8, 10]; iBooks 2 can only run on iOS-based platform devices and the synchronization system is difficult to use, although it can deliver interactive 3D objects and content synchronization services[11].

In this paper, we propose a synchronization system for an educational content viewer that provides content sharing (displaying the same content on multiple devices at the same time) and content continuity (continuity of displaying content) features to deliver educational content seamlessly to users anytime and anyplace. We implemented the proposed synchronization system in an educational content viewer that we previously developed[12, 13]. The educational content viewer is based on Extensible Markup Language (XML)[12] to enable the support of synchronization services between devices of different platforms. We designed the interface of the synchronization system to be easy to use. The content viewer supports not only basic controls such as text or images, but also interactive controls such as 3D objects, videos, and animations[12].

The remainder of this paper is organized as follows. Section 2 reviews the theoretical background on which this study was based, including synchronization systems and content viewers, and the smart educational content viewer introduced previously[12]. In Section 3, we describe the proposed synchronization system for an educational content viewer, including its configuration, database, server, and processes. In Section 4, we present the implementation of the synchronization system, and Section 5 describes our experimental results. Section 6 summarizes the paper with some directions for future research.

2. Theoretical background

2.1 Synchronization System

Synchronization involves making the data on two devices identical[14]. Recently, it has been exemplified by displaying the same content on multiple devices at the same time, such as the N-screen system, or by seamless content display between different devices. There are various synchronization services, such as 'video on demand' (VOD) viewing and content sharing in educational services[15, 16]. Jang-Woo Yoon et al.[17–19] broadly classified synchronization service types into three categories: 'content sharing,' 'content continuity,' and 'collaboration among devices,' as shown in Figure 1.

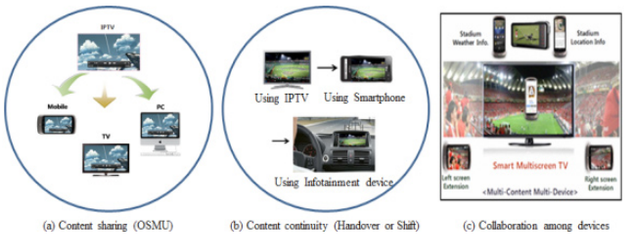

Figure 1. Three Classifications of Synchronization Services[17–19].

Content sharing provides the same content (or same service) identically on multiple smart devices with different resolutions, memories, and CPU speeds. Figure 1(a) shows examples of a content sharing service, called 'one-source, multi-use (OSMU)' in N screen[18–19]. Content continuity allows users to view content continuously on different smart devices without time or place limitations (seamless/continuous viewing of the content), as shown in Figure 1(b). For example, a video played for 10 min on device A then continues to play on smart device B after 10 min. The last service type is collaboration among multiple devices. For example, as shown in Figure 1(c), content can be delivered through multiple devices by dividing it into several aspects, or can be provided on a smart TV while supporting additional information, such as weather forecasts and advertisements through other smart devices. The synchronization system proposed in this paper provides content sharing and content continuity features to deliver educational content seamlessly anytime and anyplace.

2.2 Content Viewers

A content viewer or e-book reader is an application for displaying digital content or e books, including texts, audio/videos, images, and 3D objects, on various devices, such as PCs, tablets, and smartphones. The most popular content viewers include EPUB3.0-based viewers and Apple's iBooks 2[8, 9].

Representative EPUB3.0-based e-book viewers include Readium[8], Tabon eBook reader[20], and Ridibooks' viewer[21]. Readium was developed by Readium.org, a nonprofit organization, with the goal of accelerating the adoption of EPUB 3.0 as a universal digital standard publishing format[12]; it currently has more than 40,000 users since it was launched on the Chrome Web Store in 2012[10]. However, such e-book viewers have limitations in displaying Scalable Vector Graphics (SVG) and Cascading Style Sheets (CSS), and do not support interactive

3D objects and synchronization features[8, 10]. Apple's iBooks 2, which was developed for educational purposes, supports various types of content, including interactive 3D objects and photo galleries and pop up animations with supplemental learning features, such as highlights, notes, and content and glossary searches. It also supports synchronization services, including content sharing and content continuity[8, 11, 16]. However, it is only accessible on iOS-based smart devices, such as iPads, iPad minis, and iPhones.

To accommodate these shortcomings of existing content viewers, we designed synchronization features based on the smart educational content viewer we previously developed[12,13], supporting various types of content, including interactive 3D objects, SVG, videos, and big data, in addition to supporting compatibility with multiple device platforms.

2.3 Smart Educational Content Viewer

Recently, a cloud-based education system was developed for deploying smart media educational content services, enabling delivery and sharing of various enhanced content, including texts, images, videos, animations, and 3D objects[22]. The system supports six main features: a cloud platform, a compatible file format, an authoring tool, a smart educational content viewer, an inference engine, and a security system.

The educational content viewer developed enables displaying smart educational content on multiple device platforms based on XML, in cooperation with HTML5 and the SVG format[12, 13]. The viewer can display smart media content on multiple device platforms by converting the XML data to HTML format appropriate to each device using Extensible Style Sheet Language Transformations (XSLT). It is not easy to convert the XML data of the interactive 3D objects, including audio and animations to HTML format with XSLT. Thus, the viewer was developed with a mechanism for visualizing interactive controls by associating with another content viewer developed using the Open Graphics Library for Embedded Systems (OpenGL ES) rendering[12]. Consequently, this content viewer is platform-independent and deploys interactive 3D objects, including audio, videos, and animations. We implemented a synchronization system based on this smart educational content viewer. Figure 2 shows screenshots of the smart educational content viewer on an iOS-based smartphone (a) and an Android-based smartphone (b) for the same content.

3. Synchronization System for an Educational Content Viewer

3.1 System Configuration

The synchronization system for the educational content viewer proposed in this paper consists of four features: the 'content viewer,' 'synchronization server,' 'database,' and 'push message server,' as shown in Figure 3.

When a user executes the educational content viewer on a smart device with the content continuity feature, the content viewer requests from the synchronization server the content data last used by the user, after collecting the basic information of the user. Then, the server finds the last performed content data from the database based on the basic information of the user, and sends the last content status information to the content viewer. The content viewer analyzes the content information received, depending on the kind of content; that is, it applies the last content status information to the corresponding content by analyzing the content information differently according to the kind of content, and composes a content screen and displays it on the user device.

(a) Screenshots on an iOS-based smartphone (b) Screenshots on an Android-based smartphone

Figure 2. Examples of the content viewer on an iOS-based smartphone (a) and an Android-based smartphone (b)[12].

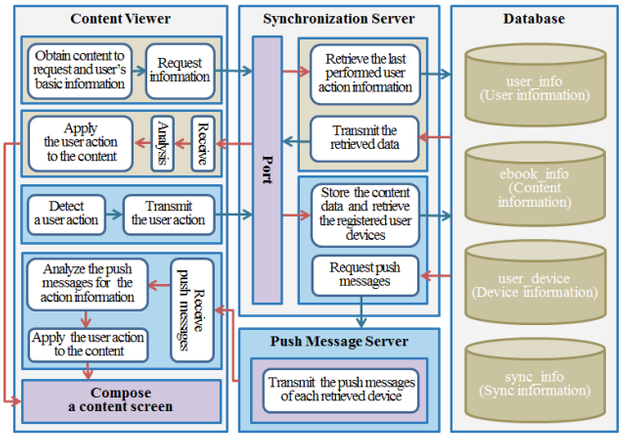

Figure 3. Configuration of the synchronization system.

If the content viewer detects a user action related to content synchronization, it transmits the detected action information to the synchronization server. The server then saves the information received into the tables of the database, searches smart devices registered in the database, and requests to the push message server to send the push messages of each retrieved device to the content viewer on each device. The push message server supports push messages based on the operating system of each smart device. For example, if a smart device from the databases is an Android-based device, then the push message server requests a message transmission to a Google server, which supports Google Cloud Messaging for Android (GCM). The content viewer on each device analyzes the transmitted push messages, based on the kind of content, and applies the action information to the content to compose a synchronized content screen.

3.2 Database for Synchronization

To manage the information required for synchronization, we designed a database, as shown in Table 1. The database consists of four tables: user_info, ebook_info, user_device, and sync_info tables. The 'user_info' table is for managing the user information of the content viewer, and the 'ebook_info' table is for managing the basic information on e-books (content) uploaded on the synchronization server. The 'user_device' table controls the information on users' smart devices by each user, and the 'sync_info' table manages the user actions performed for synchronization requests.

Table 2 shows the kinds of content supported by the synchronization system, and Table 3 describes the attributes of the user action information used in the content viewer for synchronization. They have different properties depending on the type of content.

3.3 Server for Synchronization

The synchronization system proposed in this paper processes device synchronization by retrieving smart devices from the database and requesting push messages to the push message server, through a web server. Consequently, the performance of the synchronization system depends on the network status of the smart devices and the processing speed of the web server. Thus, the synchronization processing algorithms on the server are important. For an efficient synchronization process on a Web server, we define three classes, as shown in Table 4.

Table 1. Database for synchronization services

Table	Column Name	Data Type	Function
User_Info	user_index	int	User index
	user_ID	varchar(50)	User identification
	user_pass	varchar(50)	User password
	user_name	char(10)	User name
	user_tel	varchar(15)	User phone number
	user_date	datetime	Subscription date
eBook_Info	ebook_index	int	e-book index
	ebook_id	varchar(50)	e-book identification
	ebook_writer	varchar(50)	Author identification
	ebook_kinds	varchar(50)	Kind of e-book
	ebook_update	varchar(MAX)	Modified content
	ebook_ver	varchar(50)	Version of e-book
	ebook_date	datetime	Written date of e-book
User_Device	device_index	int	Device index
	device_token	varchar(50)	Device token
	device_kind	varchar(50)	Device OS
	user_ID	varchar(50)	Device user ID
Sync_Info	sync_index	int	Sync index
	sync_page	int	Page number
	sync_kind	varchar(50)	Kind of content
	sync_action	varchar(MAX)	Content status info.
	sync_time	datetime	Synchronized time
	user_ID	varchar(50)	User ID
	books_id	varchar(50)	E-book ID

Table 2. Kinds of content supported by the synchronization system

Category	Kind of Content	Function
Basic Content	Text	Page content or texts
	Image	Images in the *.png, *.jpg, and *.gif formats
	SVG	Simple 2D objects such as lines, circles, and polygons
	Content continuity (common)	Display content by content continuity
Sync Content	Page	Synchronization for page movement actions
	Video/Audio	Synchronization for pause and replay actions
	3D object	Synchronization for camera movement actions

Table 3. Content attributes for synchronization actions

Content	Attribute	Function
Page	pageNum	Page number
Video/Audio	frame	Play and stop time or frame information
	state	Status of pause, stop, or play
3D Object	eye_*x*, eye_*y*, eye_*z*	*x*, *y*, *z* coordinates of a camera position
	at_*x*, at_*y*, at_*z*	*x*, *y*, *z* direction vectors of a camera view
	m_dist	Distance between a camera and an object
	m_rad*x*	*x* rotation information of a camera
	m_rad*y*	*y* rotation information of a camera

Table 4. Classes for synchronization in a web server

Class	Method	Function	Return value
Continuity	cur_content	Retrieve the content information performed most recently	Action info
Sharing	sync_action	Called by the content viewer when an action is occurred	Boolean
	save_action	Save the action information to the database	
	get_device	Retrieve the device information to send messages	
	send_Android	Request the push messages of the Android-based device	
	send_iOS	Request the push messages of the iOS-based device	
Common	save_device	Save user device information	
	getBookXML	Retrieve the eBook content requested by users	XML
	XMLtoHTML	Convert XML data to a HTML document	HTML

Table 5 presents the SQL statement and algorithm of the 'cur_content' method defined in the 'continuity' class. When a content viewer is executed on a smart device, the method 'cur_content' is called from the viewer with parameters 'user_ID' and 'ebook_id'. It finds the content information performed most recently from the 'sync_info' table of the database, and returns them to the viewer. Then, the content viewer composes a content screen for the smart device with a continuous view using the returned information and displays them on the smart device. We retrieve the most recently generated top 500 data from the database. For the 500 retrieved data, if a data belongs to the page where the last synchronization occurred, it is added to an action information list (contentsArray). If data are not on the page, the list is returned to the user device. The list includes the information of content, action, and page number. The other methods presented in Table 4 are also implemented in a similar way to this algorithm.

3.4 Synchronization Process in the Educational Content Viewer

Figure 4 describes the synchronization process in the content viewer, divided into three phases. A synchronization service (content continuity and sharing) is initiated from Phase 1 (request the XML data of the content and analyze them). First, the XML data for an e-book's content are requested from the content viewer to the web server, and the transmitted XML data are parsed with each page unit using the 'XMLtoHTML' method of the 'Common' class. When a user selects the content continuity feature of synchronization, the server requests the user's last-performed content information to the viewer (Phase 2).

Table 5. SQL and Algorithm for the 'cur_content' method

SQL statement for retrieving the content information performed most recently

```
SELECT TOP 500 log_page, log_contents, log_action
   FROM [CloundServerDB].[dbo].[Log_CurInfo]
   WHERE user_ID = uID and book_id = bID ORDER BY
      log_time DESC;
```

Algorithm for retrieving the content information performed most recently

```
public contentsArrary cur_content(String uID, String bID) {
 // the page where the last synchronization occurred
 int page = DBtable[0].log_page;
 for (int i=0; i<DBtable.count; i++) {
  if ((page == DBtable[i].log_page) &&
   (!contentsArray.search(contents.DBtable[i].log_content))
    contentsArray.add(DBtable[i]);
  else return contentsArray;
 }
}
```

The server transmits the last-performed content to the content viewer, retrieving them from the database. Then, the viewer analyzes the transmitted content on the basis of two categories: basic content and sync content. For basic content, the viewer requests the server to convert the XML data of the content to HTML format. For sync content, the viewer initializes the content data to the status when the synchronization occurred. Then, the viewer composes a content screen using the converted HTML data and the initialized sync content. When a user selects the content sharing feature, the viewer detects that a user action occurred and transmits it to the web server (Phase 3). To process content sharing services, the server stores the received action information to the database and retrieves the user's device information, registered in the database. Then, the server sends the push messages for each device retrieved to the content viewer on each corresponding device. The content viewers on each device analyze the received push messages and provide the synchronized content on the screen of each corresponding device, based on the push messages.

Table 6 describes the algorithms for requesting the necessary data from the content viewer to the web server, and for analyzing the data received in the viewer. In the algorithm for data analysis, if the data received are in the XML format, the viewer parses the data to the HTML format. In the case of HTML data, the viewer displays the data on the user device screen using WebView, which enables web pages to be displayed on a smart device. For action data to be synchronized, the viewer composes a sync screen differently according to the kinds of sync content. In the case of a video, audio, or 3D object, the viewer first extracts the attributes of the action information using the 'dataSplit' function, and applies them to the content to synchronize by calling each function corresponding to the kind of the content.

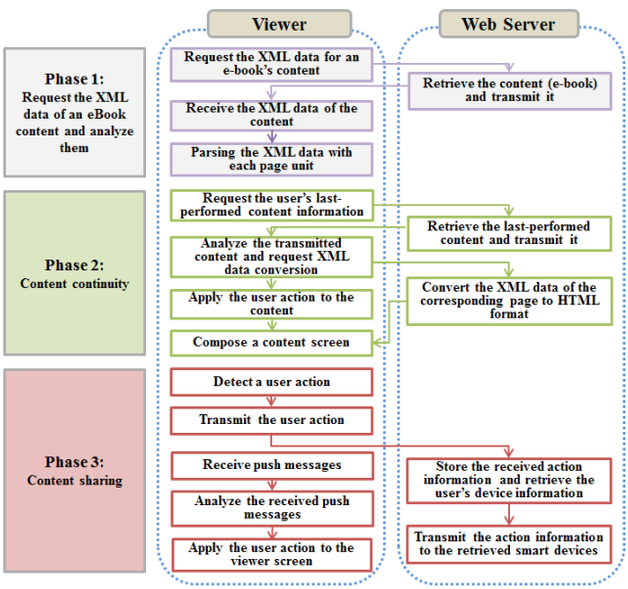

Figure 4. Synchronization process in the educational content viewer.

Table 6. Algorithms for data request and analysis

Algorithm for requesting the necessary data from the content viewer to the web server

```
SoapObject request = new SoapObject(NameSpace, Method);
for (int i = 0; i < numberOfParameter; i--) request.
  addProperty(parameterName, data);
SoapTrandsport.call(Soap_Action,SoapSerializationEnvelope.
  setOutputSoapObject(request));
```

Algorithm for analyzing the data received from the web server in the content viewer

```
for (int i = 0; i < table.count; i++) {
 if (table[i].data == HTML) WebView.setview = table[i].data;
 else if (table[i].data == XML) DataParsing(table[i].data);
 else if (table[i].data == actionInformation) {
  if (table[i].contnet == Page) PageMove.set(table[i].page);
  else {
   String temp [] = dataSplit(table[i].action);
   if (content == Video) VideoState.set(temp.Frame, temp.
     stopORplay);
   else if (content == Audio) AudioState.set(temp.Frame,
     temp.stopORpaly);
   else if (content == 3Dobject) 3Dobject.set(temp.camera,
     temp.at, temp.dist, temp.rad);
  }
 }
}
```

4. Implementation

The synchronization system, implemented in the educational content viewer in this paper, supports content continuity and content sharing features. The synchronization system is platform-independent. Content continuity is a synchronization service that enables users to continuously view content, starting from the content that they were viewing on the last content viewer. Content continuity can be processed between the same devices or among multiple different devices. Figure 5 shows examples of content continuity using three different smart devices. Content sharing is a synchronization

service that provides the same content identically on multiple smart devices, despite different system performance characteristics. The proposed synchronization system was implemented to work across multiple smart devices running on not only the same platform, but also on different platforms. Figure 6 shows examples of content sharing between Android-based smart devices. Figure 6(b) and (d) are examples of synchronization for a video play and pause. When a video is played or paused on one smart device, other synchronized devices are also synchronized with the video play or pause mode. At this time, there could be a difference in video play time due to differences in time of synchronization processes on each different smart device, as seen in Figure 6(b). Figure 7 shows examples of content sharing between iOS-based and Android-based smart devices while performing page movements.

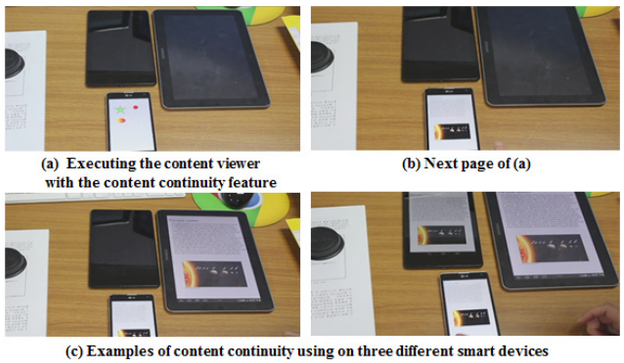

Figure 5. Examples of content continuity on different smart devices.

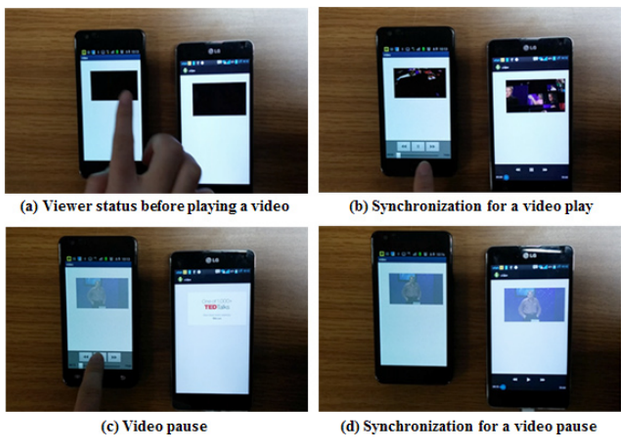

Figure 6. Examples of content sharing between Android-based smart devices.

5. Experiments

Experiments were conducted to examine the performance of the proposed synchronization system, in terms of processing time of synchronization. Table 7 shows the results of the average processing time for synchronization according to network connection types: third generation (3G), long-term evolution (LTE), and wireless (Wi-Fi) networks. The results show significant differences, depending on network connection types regardless of OS type. On average, the synchronization was processed in 2-7 s on Wi-Fi and LTE networks, but required more than 10 s on the 3G network. This indicates that the synchronization services proposed in this paper can be processed within a reasonable time under Wi-Fi and LTE networks. Currently in South Korea, most areas support Wi-Fi and LTE networks. Thus, it seems that the limitation with 3G will not be a major issue.

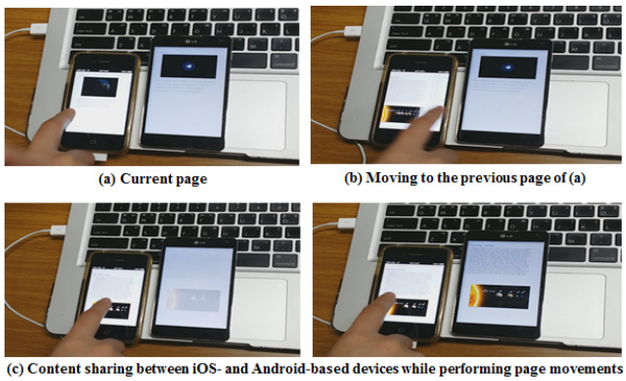

Figure 7. Examples of content sharing between iOS-and Android-based devices.

Table 7. Performance for synchronization according to connection type

OS	Synchronization Service Type (Content Type)		Speed (unit: second)		
			3G	Wi-Fi	LTE
iOS	Content continuity		2~3	2	1~2
	Content sharing	Page	Over 10	2~5	2~4
		Video/Audio	Over 10	2~5	2~4
		3D object	Over 10	4~7	3~7
Android	Content continuity		2~3	1~2	1~2
	Content sharing	Page	Over 10	2~4	2~4
		Video/Audio	Over 10	2~5	2~4
		3D object	Over 10	3~7	3~7

Table 8. Results for synchronization between different platform devices

Synchronization Type		Result
Content continuity		Synchronized in real-time
Content sharing	Page	Synchronized in real time, except an unstable network state
	Video/Audio	Synchronized in time, although the time of play/pause could differ depending on the arrival time of push messages
	3D object	Synchronized in slightly different time, due to differences in system performance of each device's OS, and in object representation methods implemented on each device

Table 8 shows the results of synchronization system application between smart devices with different OSs (between Android-based and iOS-based devices). The synchronization of content continuity and content sharing for page movements were performed in real-time, unless the networks were in an unstable state. Video/audio play and pause motions were also well synchronized in time, although the time of play/pause could differ depending on the arrival time of push messages. In the case of 3D objects, there were slight differences in user actions due to differences in system performance of each device's OS, and in object representation methods implemented on each device. Thus, it will be necessary to develop a standardized method for representing 3D objects between different OSs.

6. Conclusion

We described the development of a synchronization system for a content viewer, which supports content sharing and content continuity features. The synchronization system was implemented in the education content viewer that was previously developed for a cloud-based educational content service[12,13]. The most popular content viewers, such as EPUB3.0-based e-book readers and iBooks 2, have some limitations: EPUB3.0-based e-book viewers do not support 3D object controls or content synchronization, and iBooks 2 can only run on iOS-based platform devices. Our content viewer accommodates these shortcomings by making it possible to process various types of content and to do so independent of device platform.

To examine the performance of the synchronization system, we applied our system over different network connection types and between smart devices with different OSs. The results showed a significant difference depending on network connection types regardless of OS types; the system performed in a reasonable time on Wi-Fi and LTE networks, but it showed speed limitations on 3G networks. In addition, most content types, except for 3D objects, were synchronized in real-time. In the case of 3D objects, there were slight differences on user actions due to differences in the system performance of each device's OS, and in the object representation methods implemented on each device.

The educational content viewer with the proposed synchronization system has a number of advantages over existing content viewers; however, there are still a number of areas to be extended. First, a study on the improvement of content sharing for 3D objects should be performed to enhance performance. Second, further implementations are required on a collaboration feature of synchronization to deliver content through multiple devices by dividing it into several aspects, or to provide content services of different types on a user's multiple devices simultaneously.

8. Acknowledgement

This research was financially supported by Ministry of Education (MOE) and National Research Foundation of Korea (NRF) through the Human Resource Training Project for Regional Innovation.

9. References

1. Kwon, SO, Jeong JS, Kim JH, Kim HR, Yoo KH, Chungbuk N. An user location based news retrieval system. J of the Korea Contents Assoc. 2012; 12(12):524–32.
2. Park C, Jeong JS, Yoo KH. WeWhere: A multiple user location tracking system of smart phones. J of the Korea Cont Assoc. 2011; 11(4):18–26.
3. Yoon YI, Kim S, In MG, Lee GC, Lee SY. Mobile cloud-based N-screen service strategy. J of the Korea Instit of Commun and Inform Sci. 2011; 28(10):38–43.
4. YoonYI, Oh JH. A study on N screen synchronization techniques based on time button configuration for smart learning space. J of the Korean Soc of Comp and Inform. 2013; 18(9):91–9.
5. Kim GH. Web technology and multi-screen service. J of Smart Media. 2012; 1(3):16–23.

6. Lee JH. Cloud-based mobile distance learning system. Ubiquit Inform Technol and Appl. 214. 2013; p. 399–405.
7. Kim JH and Kim GH. A development of web-based educational game for the multiple devices. Proceedings of the Korea Computer Graphics Society 2013 Conference; 2013 Jun 19-21; Pusan, Korea. p. 235–36.
8. Readium.org. Readium. 2014; Available from: http://readium.org.
9. Apple Corp. iBooks. 2014; Available from: http://www.apple.com/ibooks, http://www.macworld.com/
   a. article/2013168/review-ibooks-3-is-the-best-ibooks-yet.html,.
10. Chrome Web Store. 2014; Available from: http://chrome.google.com/webstore/detail/empty-title/ fepbnnnkkadjhjahcafoaglimekefifl.
11. Park CS. Analysis of the Apple's Digital Textbook Ecosystem. KERIS Issue Report. 2012; RM 2012–3.
12. Ju SY, Jeong JS, Kwon SO, Kim M, Yoo KH. Design and implementation of a smart educational content viewer. J of Next Gener Inform Technol. 2013; 4(8):412–22.
13. Kwon SO. Design and implementation of synchronization system for a smart educational content viewer [Master Thesis]. Chungbuk National University; 2014.
14. Wiki, Synchronization. 2014; Available from: http://en.wikipedia.org/wiki/Synchronization.
15. Lee JH, Kim TH, Lee MK, Kim JH, Jung HM, Sung et al. Usefulness evaluation on elements for visualization of technology intelligence. J of the Korea Cont Assoc. 2011; 11(12):543–51.
16. Kim YW. Analysis on N-Screen strategy and trend. KISDI Broadcast and Commut Poli; 2010; 22(20):1–3.
17. Yoon CW, Jeon SH, Lee HY, Heo GW, Lee HY, Ryu W. N-Screen Service Classification and standardized technology. Commun of the Korea Inform Sci Soci. 2011; 29(7):23–3.
18. Yoon CW, Um TW, Lee HW. Classification of N-screen and its standardization. Proceedings of the 14th International Conference on Advanced Communication Technology (ICACT); 2012 Feb 19-22; PyeongChang, Korea: p. 597–602.
19. Yoon CW. Technology standard issue: N-screen service scenarios for FMC. ICT Stand Weekly. Telecommun Technol Assoc; 2013; 12.
20. Tabon Books Inc. Tabon eBook Reader. 2014; Available from: http://www.tabonbooks.com/.
21. Rido Corp. RidiBooks's Viewer. 2014; Available from: http://ridibooks.com/support/ introduce_appdown.
22. Jeong JS, Kim M, Yoo KH. A content oriented smart education system based on cloud computing. Internat J of Multimed and Ubiquit Engineering. 2013; 8(6):313–28.