Multimed Tools Appl (2015) 74:1557–1576
DOI 10.1007/s11042-013-1458-0

An interactive pervasive whiteboard based on MVC architecture for ubiquitous collaboration

Kangseok Kim · Wonjeong Ha · Okkyung Choi · Hongjin Yeh · Jai-Hoon Kim · Manpyo Hong · Taeshik Shon

Published online: 23 April 2013

Abstract Collaboration is about interactions among multiple users by sharing resources. With the advent of new generation of mobile access devices such as smartphone and tablet PC, we address this for ubiquitous collaboration—the capability of multiple users to link together with disparate access devices in anytime and in anywhere. The implementation and experiments of mobile applications for ubiquitous collaboration were challenge since cell phone (also referred to feature phone) prior to smartphone had high network latency and low computing performance. However, even though mobile device technologies and wireless networking are becoming more advanced with time, the research on the impact of ubiquitous collaboration is still immature. This paper extends our prior work with interactive multimedia services—whiteboard application on 3G and Wi-Fi Android platform based mobile devices, based on MVC (Model View Controller) architecture for ubiquitous collaboration.

K. Kim (✉) · W. Ha · O. Choi
Department of Knowledge Information Engineering, Graduate School of Ajou University, Suwon, Korea
e-mail: kangskim@ajou.ac.kr

W. Ha
e-mail: seacoolth@ajou.ac.kr

O. Choi
e-mail: okchoi@ajou.ac.kr

H. Yeh · J.-H. Kim · M. Hong
Graduate School of Information and Communications at Ajou University, Suwon, Korea

H. Yeh
e-mail: hjyeh@ajou.ac.kr

J.-H. Kim
e-mail: Jaikim@ajou.ac.kr

M. Hong
e-mail: mphong@ajou.ac.kr

T. Shon
Division of Information Computer Engineering, Ajou University, Suwon, Korea
e-mail: tsshon@ajou.ac.kr

Then, we present interaction and usability study with multiple users using (wire, wireless) small-sized devices (smartphones), mid-sized devices (tablet PCs), and large-sized device (desktops) to show how disparate access devices and networking have an effect on software design of shared applications in ubiquitous collaboration system.

Keywords CSCW · Ubiquitous Collaboration · Interactive Shared Whiteboard · Mobile Device · MVC

1 Introduction

With the advances in a variety of disparate access devices such as smartphone and tablet PC, and wireless networking such as Wi-Fi (wireless fidelity) and 3G / 4G (3rd and 4th Generation Mobile Telecommunications), there is coming a need for ubiquitous collaboration [14] which allows people to access multimedia information systems independent of their access device and their physical capabilities in anytime and in anywhere. The implementation and experiments for ubiquitous collaboration were challenge since cell phone prior to smartphone had high network latency and low computing performance. However, nowadays, users in ubiquitous collaboration can access shared multimedia contents in anytime and in anywhere via wired / wireless networks with disparate access devices. Moreover, users can more interactively communicate with others through those networking and disparate access devices. The following scenario motivates the design issues (with user study) of pervasive software like shared whiteboard for ubiquitous collaboration. Students in ubiquitous software lab at Ajou University are going to have a session for their colleague's research presentation. Some students join the presentation session in a shared conference room of lab and others join at remote locations by using conferencing collaboration tool built on disparate mobile access devices. During her presentation, she may use an application like shared whiteboard to discuss design issues of the research that she is doing on cloud computing. In shared workspace with the application, people in offline session see the same whiteboard canvas, while people in online session see their own canvases. Each student in the online session has their own canvas and a set of interfaces to the shared whiteboard application but they see the same results (or views) as others do. Therefore, pervasive applications need to support interactive user's actions by sharing (or presenting) contents such as text, graphics, audio, video, and so on.

Firstly, we extend our prior work [8, 14] by building ubiquitous collaboration framework on Android platform based smartphone and tablet PC with heterogeneous networking 3G and Wi-Fi. The ubiquitous collaboration framework, described in more detail in Section 3 and in [14], is a basic structure to hold consistent view or information of users' presences and sessions together. It also has a capability that allows a user to join a conference using networked heterogeneous computing devices in anytime and in anywhere. It is typical today and will be more typical in the future that all users can access information independent of their access devices and physical capabilities in anytime and in anywhere. In this paper we seek to answer to the following research questions: (*1*) *It seemed to be in offline real conference room even when using the shared whiteboard application on mobile devices at remote locations*?, and (*2*) *The response of a request with mobile devices was proper as compared to working with only desktops*?.

Secondly, in this paper we consider two classes of applications with disparate access devices and networking: wired personal computer based application and wireless mobile device based application. Development of wired personal computer based applications has already matured with a variety of computing fields. On the other hand, wireless mobile device based applications have shown tremendous growth with the advance of wireless mobile devices / networking in the last few years. This situation suggests that we further need to take another look at our prior work for ubiquitous collaboration with the advanced mobile devices and wireless networking. Therefore, in this paper we review our prior work with shared whiteboard that is a collaborative enabled drawing application which offers a set of drawing tools and a common canvas shared virtually among all participants joined in a session. The architecture of pervasively shared whiteboard application built on heterogeneous computing devices is based on MVC (Model View Controller) based collaboration model (also called interactive shared event / display collaboration model) [6, 7] as a uniform architecture for the development of wired desktop and wireless mobile applications in ubiquitous collaboration environment. The goal of MVC is, by decoupling models and views, to reduce the complexity in architectural design [20, 21]. This work also involves substantial study of approaches to ubiquitous collaboration. Therefore, in this paper we seek to answer to the following research questions: (*1*) *The shared whiteboard was useful for group discussion*? (*2*) *The shared whiteboard on mobile devices was stable during group communication*? (*3*) *To manage the network transactions and transformations for undo / redo of whiteboard contents* (*multimedia*) *was complex*? and (*4*) *The waiting time for turn-taking communication among participants with shared whiteboard was too long*?.

Finally, in this paper we seek to answer to the following research question: *We need the observations of users' behaviors with applications in ubiquitous collaboration environment considering response time vs*. *concurrency*, *response time vs*. *simplicity*, *and response time vs*. *principle of least privilege*?

To answer to the questions, we present user study with multiple users using wireless smartphones, tablet PCs, and wired desktops to show how disparate access devices have an effect on the software design for ubiquitous collaborative application.

We believe from our development experience that the computing performance of mobile devices as well as desktop computers will continue to improve and networks' bandwidth will continue to increase. Thus the infrastructure improvements of software, hardware, and networking will make ubiquitous collaboration and access more prevalent in the near future.

The remainder of this paper is organized as follows. We review related works in Section 2. Section 3 briefly describes the architecture of our collaboration framework built on disparate access devices as well as overall architecture of our collaboration system. Section 4 presents an interactive shared event / display collaboration model based on MVC for our pervasive whiteboard. Section 5 presents usability test results on user study for showing the viability of the interactive whiteboard application built on wire / wireless access devices and summarizes answers to the questions given in this section. Finally we give our conclusions with a brief discussion of future work in Section 6.

2 Related works

This section describes our related work in area: previous shared whiteboard technologies in CSCW (Computer Supported Cooperative Work). In the field of CSCW, a variety of collaboration systems for supporting shared display and drawing activities among collaborators were designed. Tivoli [19] is a shared whiteboard system with the human and technical considerations in disseminating a new shared object so that each host can have consistent shared object and each user can work on the disseminated object copy. VideoWhiteBoard [22] was developed to support a remote shared drawing activity with a shadow of gestures among collaborators. ClearBoard [12] is an eye contact image enabled shared whiteboard. Distributed Designers'

Outpost [5] is a synchronous collaboration system using projection technology for shared display among collaborators with two awareness mechanisms for gestures and presence. CollaBoard [17] is an interactive whiteboard with remote life-sized video image atop the shared canvas. Clear-Face [11] used translucent live face video images over a shared whiteboard to effectively use the limited screen size. VideoArms [23, 24] is a shared visual whiteboard using an embodiment technique that captures people's arms for presence. All the above systems were designed for supporting shared display and drawing activities among whiteboards built on homogeneous stationary devices. Unlike the above systems that were designed in homogeneous collaboration environment, our work focuses on supporting shared display and drawing activities by integrating heterogeneous networking (wired, wireless) and computing platforms (smartphone, tablet PC, desktop) environments into a single easy-to-use collaboration system.

3 Overall system architecture and Collaboration framework architecture

In this section we describe an overall system architecture and collaboration framework architecture built on heterogeneous (wire, wireless) computing environment. In our work we have used Google Android Platforms [1] for our software development in mobile (smartphone and tablet PC) computing environment and J2SE (Java 2 Standard Edition) [13] for our software development in non-mobile (stationary device like desktop) computing environment. Figure 1 shows an overall architecture view for our collaboration. For communication service, we have used NaradaBrokering [16, 18] for messaging and service middleware system as overlay built over heterogeneous networks to support group communications among heterogeneous communities and collaborative applications. The NaradaBrokering from Community Grids Lab (CGL) [3] is adapted as a general event brokering middleware, which supports publish/subscribe messaging model with a dynamic collection of brokers and provides services for TCP, UDP, Multicast, SSL, and raw RTP clients. The NaradaBrokering also provides the capability of the communication through

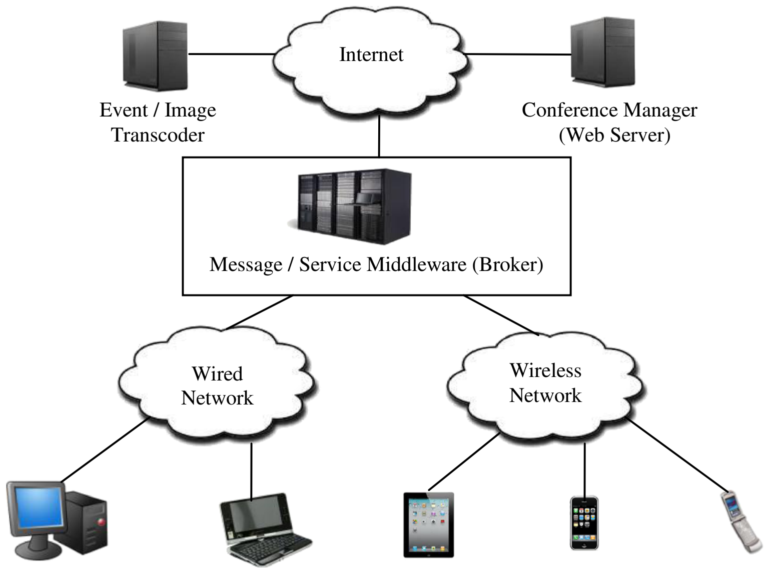
Internet
Event / Image Transcoder
Conference Manager (Web Server)
Message / Service Middleware (Broker)
Wired Network
Wireless Network

Fig. 1 Overall architecture view of our ubiquitous collaboration system

firewalls and proxies. It is an open source and can operate either in a client–server mode or in a completely distributed peer-to-peer mode. In this paper we use the terms “message and service middleware (or system)” and “broker” interchangeably. The conference manager, residing on web server running on tomcat, manages information related to all the conferences. The manager maintains registries of all scheduled conferences and registries of collaborative applications. The event/image transcoder, described in more detail in Section 4, is a kind of agent able to cooperate and to coordinate heterogeneous types of applications on heterogeneous platforms or devices like desktop, tablet PC, smartphone and so on. The purpose of using the transcoder is to convert one type of representation to other types of representations on heterogeneous platforms with different screen (or canvas) sizes and different representation formats.

The collaboration framework, described in more detail in our prior work [14], handles cooperation and communication among multiple users, and provides collaborative applications in the ubiquitous collaboration. A key function of the framework is to provide a generic solution for controlling sessions in a conference and accesses to resources, maintaining shared state consistency at application level and maximizing the use of various collaborative capabilities to users. Another function of the framework is to provide a structure for development and deployment of collaborative applications that can be used to support asynchronous collaboration by allowing different users of a session to access the same resource at different times, and synchronous collaboration by enabling the users to share the same resource in real time. As shown in Fig. 2, the collaboration framework is structured as three layers and six major components: control manager, session / membership control manager, access / floor control manager, policy manager, request / reply event message handlers, and communication channel. Note that our prior work [8, 14] focused on moderator mediated strict coordination mechanism for dealing with consistency in real time in

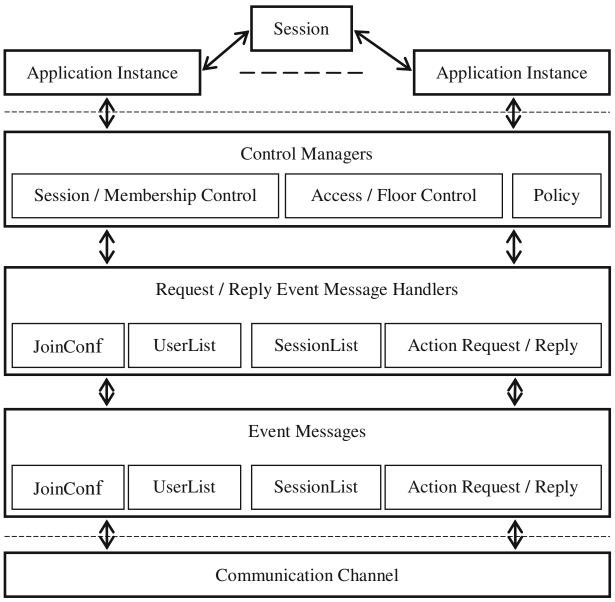

Fig. 2 The framework architecture consists of three layers (collaborative applications, managers, and communication service) and six major components (control manager, session/membership control manager, access/floor control manager, policy manager, request/reply event message handlers, and communication channel)

ubiquitous collaboration environment. For user study described in Section 5.3, the strict coordination mechanism as well as relaxed coordination mechanism without locking mechanism was implemented in our collaboration framework. Therefore, in future work we will implement access / floor control manager component with policy manager component in the collaboration framework considering the results of user study described in Section 5.3. We briefly describe the components in turn.

- Control Manager
  The control manager is an interface component for providing conference management services such as presence, session, and access / floor control managements for participants in collaboration.
- Session and Membership Control Manager
  The session / membership control manager component manages information about who is currently in the conference and has access to what applications, and which sessions are available in the conference.
- Access and Floor Control Manager
  The access / floor control manager component is responsible for handling accesses to collaborative applications through the request / reply event message handlers which are one of components in the framework.
- Policy Manager
  Access / floor control policies are written in XML and put into the conference manager which resides on web server running on tomcat for globally consistent use. In future work we will implement this component defining a policy according to the floor control mechanism.
- Request and Reply Event Message Handlers
  An event message handler is a subroutine that handles request / reply event messages. The control manager manages the associations between incoming and outgoing event messages with each of event message handlers. According to the associations, generated outgoing event messages are first processed by the associated request event message handlers. Incoming event messages are also serviced by the associated reply / response event message handlers. The messages are sent to a broker (our messaging and service middleware—NaradaBrokering) via the communication channel shown in Fig. 2. The broker disseminates the messages to other participants connected to the collaborating session.
- Communication Channel
  The channel uses topic-based publish-subscribe mechanism that defines a general API for group communication. In the mechanism, the topic information contained within messages is used to route the messages from publisher to subscriber. The messages containing topic information are sent to a broker through the channel. And the messages are disseminated through router nodes, referred to as brokers to subscribers which registered a subscription to the topic.

4 Interactive shared event / display collaboration model based on Model-View-Controller (MVC)

In this section we present MVC (Model View Controller) based collaboration model (also called interactive shared event / display collaboration model) [6, 7] depicted in Fig. 3. In the model, each client in collaboration shares one copy of the same application with others (or master). Then sharing (View) is achieved by intercepting drawing object or image events among clients, transcoding (Model) the events into other events in the event / image transcoder shown in Fig. 3, and sending copies of the transcoded (or transformed) events to all participants—shared event / display. The example is interactive shared whiteboard built on disparate access devices which use event / image transcoder to trap user input (Controller) on a whiteboard. Each of clients in the collaboration has their own canvas and drawing activities on the canvas through an interactive shared major event—drawing object event which results in same view among them.

The event / image transcoder in Fig. 3 is a kind of agent able to cooperate and to coordinate heterogeneous types of applications on heterogeneous platforms or devices, but is not considered as intelligent agent. The purpose of the transcoder use in our ubiquitous collaboration system is to convert one type of representation to other types of representations on heterogeneous platforms with different screen (or canvas) sizes and different representation formats. Arriving objects into the transcoder are immediately transcoded or filtered and converted to other types of objects, and are broadcasted to all participating clients (devices) through our event messaging and service middleware. The communication channel (publish/subscribe) of the transcoder enables one type of collaborative application to exchange event objects with other types of collaborative applications. Note that the homogeneous collaborative applications with the same type of representations can communicate directly through the broker without the use of the transcoder. Therefore arriving messages are converted to (N-1) types of event messages where N is the number of heterogeneous types of clients (devices) supported in collaboration. In this architecture, each application does not know how to convert its own representation into other representations, that is, our

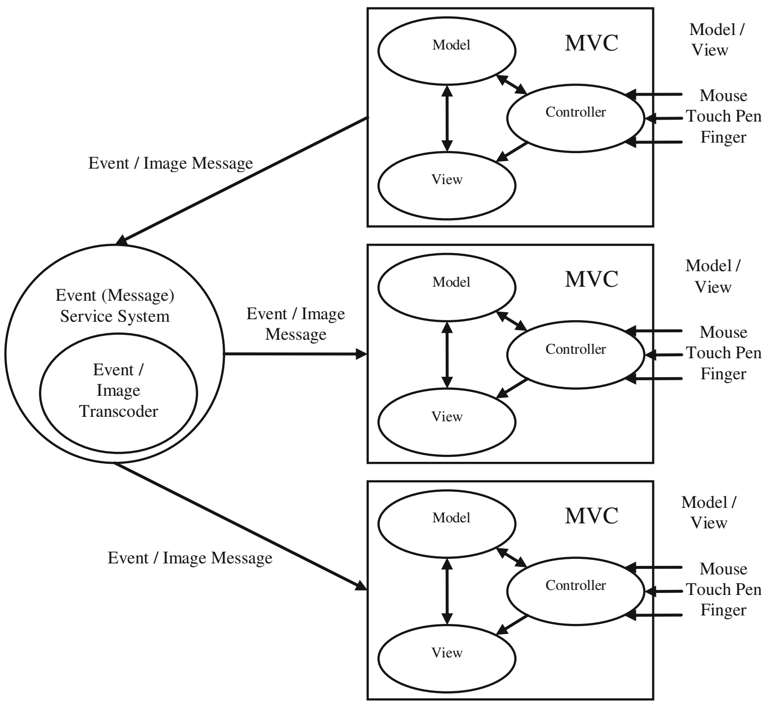

Fig. 3 Interactive shared (event / display) collaboration model based on MVC with event (message) service system and event / image transcoder to reduce the complexity in architectural design

architecture supports post-transcoding mechanism. The transcoder is responsible for translating data from one type of representation into other types of representations. In the figure, interactive shared display collaboration model shows that clients share the graphical image display and the state is maintained (shared) among clients by transmitting the changes in the display through the event message service. The supporting heterogeneous clients require that sophisticated shared display environments automatically change size and display representation formats to suit each client. The interactive shared display model has key advantages—it can immediately be applied to all shared objects. Also it can reduce significant network transit overhead using transcoder—it reduces the size of image by transcoding since complete graphical image display between desktop and smartphone applications as well as among desktop applications needs substantial network bandwidth. To demonstrate the effectiveness of the transcoder, we show the experimental results on the effectiveness of the transcoder with interactive shared whiteboards on heterogeneous computing platforms in Section 5.2. Also, the architecture of our collaboration framework follows this approach for request / reply presence and session control mechanism built on it with our messaging and service system as a derivative of interactive shared event model.

We define fine-grained actions in our whiteboard application as interactive smallest major events (semantic events). For example, drawing a line includes clicking, dragging, and releasing a mouse on the whiteboard canvas. For a user working alone with whiteboard, user input events (low level events such as mouse click, drag, and release) can be interactive major events between the user and whiteboard application. For users working with others sharing the application, the smallest major event means “drawing a line” and the user input events will then be an event data (mouse click—the first point of the line, and mouse release—the second (or last) point of the line). After the completion of each drawing (as the mouse is released), the semantic event (drawing an object) is dispatched to other multiple users as the interactive smallest major event. Then the user input events will be an event data for drawing an object affecting the whiteboard (view-sharing) of others. Therefore, the fine-grained action in our collaboration means an interactive smallest major event affecting the shared view (or result) among collaborators.

5 User study and analysis with interactive shared whiteboard built on disparate access devices

The purpose of this user study is to demonstrate the effectiveness of the interactive pervasive whiteboard based on MVC architecture with various disparate access devices. In this section we summarize answers to the questions given in Section 1 through this user study: questions described in the questionnaire of Table 2.

5.1 Baseline performance result

In this section we show the baseline performance result (round trip time) of heterogeneous (wired, and wireless 3G and Wi-Fi) network used for communication between our messaging/service middleware (broker) and collaboration framework built on smartphone, tablet PC, and desktop devices. Note that the results are not to show better performance enhancement but to quantify the network performance of disparate access devices for a variety of datasets. The quantified results will be used as a reference of the experimental result of the performance measurement used in the following section. Figure 4 show the round trip time to transfer bytes data between collaboration framework and a broker through

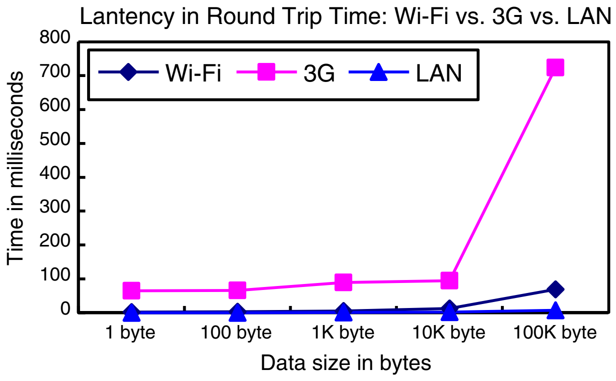

Fig. 4 Latency in round trip time: Wi-Fi vs. 3G vs. LAN

wired and wireless network including the corresponding execution time of the broker. As the size of data increases, the time for transferring the data increases as well, as shown in the figure. Note that 3G has pervasive availability as relatively compared to Wi-Fi but the performance is a little slower than Wi-Fi until a certain point (around 10 KB in our experiment), but since the point, the latency is sharply increased, introducing unstable state. Figure 4 shows that mobile devices are still sensitive to the network delay as compared to desktop.

5.2 A case study of shared whiteboard with image annotation

In this experiment, two users interact with a smartphone and a tablet PC respectively while a user interacts with a desktop. In this collaboration with heterogeneous computing devices, the disparate access devices cannot directly display objects (drawings and images) represented from each other because of different canvas sizes (smartphone: 480 by 800 pixels vs. tablet PC: 600 by 1024 pixels vs. desktop: 1024 by 768 pixels). Instead, a converted object is retrieved from the application transcoder through which the object can be represented into other disparate access devices in accommodating the capabilities of the devices. The transcoder is connected to a broker as an application server and is located in the same place where the broker is residing to reduce the latency of the object transfer time for transcoding. To transcode (convert) a large graphical image data sent from shared whiteboard on desktop into a shrunk image data on shared whiteboard of mobile devices, an image is created from the large binary image data and then a Buffered Image class is created, which is a subclass of Java Image API [13] because the class provides a more structured internal design with much greater access to the image data. The created Buffered Image is scaled (shrunk) to the canvas size of mobile devices and converted to an image file type for the mobile devices. The binary data of the image is sent to mobile devices through a broker and represented into the image format adapted on the mobile devices. The image conversion data is shown in Table 1 as an example of the case study in our experiment, and vice versa.

Figure 5 shows JPEG images after transcoding an image from desktop into images adapted on mobile devices. We do not show the performance gain of the network latency because we can intuitively know it as shown in Fig. 4. The latency is divided in two components: network delay time and processing time in transcoder. One interesting observation from these results is: improved performance in time and increased image detail loss. The transfer time is reduced as the size of image data increases, relatively compared to time

Table 1 Original size (PC) vs. Shrunk size (Tablet PC, Smartphone) (*left table*) and Original size (Smartphone) vs. Enlarged size (PC, Tablet PC) (*right table*)

	Original Size (PC) 1024×768	Shrunk Size (Tablet PC) 600×1024	Shrunk Size (Smartphone) 480×800	Original Size (Smartphone) 480×800	Transformed Size (PC) 1024×7684	Transformed Size (Tablet PC) 600×1024
Image 1	780,831 byte	108,344 byte 0.59×1.33 scaled	75,240 byte 0.8×0.78 scaled	281,438 byte	85,092 byte 2.13×0.96 scaled	71,823 byte 1.25×1.28 scaled
Image 2	561,492 byte	54,103 byte 0.59×1.0 scaled	37,582 byte 0.47×0.78 scaled	137,516 byte	47,100 byte 2.13×0.96 scaled	39,056 byte 1.25×1.28 scaled

performed without transcoding. The performance enhancement in time is due to transformation from large graphic image size to small size for adapting to the image format and screen size of other access devices. In the experiment, 780 KB image size is transformed into 75 KB image size for smartphone in transcoder. Then the 75 KB image is sent to smartphone through the transcoder. After the JPEG image is transcoded and scaled by the transcoder as shown in Fig. 5, much detail of the original JPEG image was lost—one of drawbacks in transcoding among disparate access devices. In our future work we will explore different transcoding and scaling algorithms based on our case study to overcome technical limitation occurring as porting from wired stationary computer based applications into wireless mobile device based applications.

5.3 User study results and analysis

In this section we show user study with two groups (Group A: 5 graduate students and Group B: 10 graduate students) on interactive pervasive application design through activities occurred in cooperating shared whiteboards on disparate access devices. In our experiment, we utilized two desktop devices, multiple smartphones and tablet PCs, and one broker. The broker ran on a desktop with 2.13 GHz Intel Core i3 CPU desktop and 4 GB RAM located in Ajou University. The experiment results were measured from executing shared whiteboard application built on our collaboration framework running on Android based tablet PC with

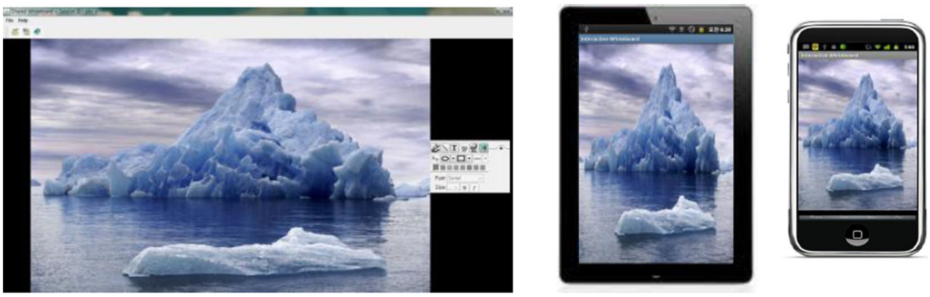

Fig. 5 1024×768, 600×1024, and 480×800 JPEG images on desktop, tablet PC, and smartphone respectively

Cortex A8 S5PC110 1GHz and 512 MB RAM, Android based smartphone with QSD8250 1GHz and 512 MB RAM connected to 3G Internet network and Wi-Fi, and running on Windows 7 platform with 2.13 GHz Intel Core i3 CPU and 4 GB RAM connected to Ethernet network respectively. Table 2 shows the questionnaire for user study. Likert-type scale from 1 (strongly disagree) to 5 (strongly agree) is used. Figures 6, 7, 8, 9, 10, 11 and 12 show the results of the questionnaire. The boxplots for the questionnaire in the figures show the median values. The lower and the upper end of the box are the 25th and the 75th percentile.

The survey result for question 1 shown in Fig. 6 presents that even though the shared whiteboard on desktop for group discussion was more effective than that on mobile devices, the shared whiteboard on mobile devices was rated to be useful for group discussion except for unstable states occurred as depicted in Fig. 7. Compared to non-mobile computing environment, whiteboards on mobile platforms have not been stable as shown in the box plot of Fig. 7 as survey result on question 2. In particular, during our experiments and through the user study as well, we have experienced unstable states on smartphone—fail to steadily function due to the low computing performance and energy efficiency of smartphone device. Technologies of applications and underlying operating system on mobile device are under tremendous development with the major companies like Apple, Google, Microsoft, Samsung, and so on. The development is expected to continue. Therefore we expect applications on mobile platforms to become more stable.

It is important to users joining a conference that it seems to be in the same conference room even when using heterogeneous computing devices at remote locations. However, the

Table 2 Questionnaires for user study with two groups (Group (subjects) A: 5 graduate students and Group (subjects) B: 10 graduate students). Likert-type scale from 1 (strongly disagree) to 5 (strongly agree) is used

Questions	
Usefulness of shared whiteboard on mobile devices	
1. The shared whiteboard was useful for group discussion.	Smartphone
	Tablet PC
	Desktop
2. The shared whiteboard on mobile devices was stable during group communication.	
Presence awareness & Integration of mobile devices using wireless network into desktops	
3. The response of a request with mobile devices was proper as compared to working with only desktops.	Group A
	Group B
Relaxed (Asynchronous) interaction vs. Strict (Synchronous) interaction	
4. To manage the network transactions and transformations for undo / redo of contents (multimedia) on whiteboard was complex.	Group A
	Group B
5. The waiting time for turn-taking communication among participants with shared whiteboard was too long.	Group A
	Group B
Response time vs. Concurrency	
6. The response time was improved by decreasing the amount of concurrency to the shared whiteboard.	
Response time vs. Simplicity	
7. The improvement of the response time by decreasing the amount of concurrency to the shared whiteboard introduced complexity.	
Response time vs. Principle of Least Privilege	
8. The unnatural granularity had an effect on limiting access to the minimal level.	

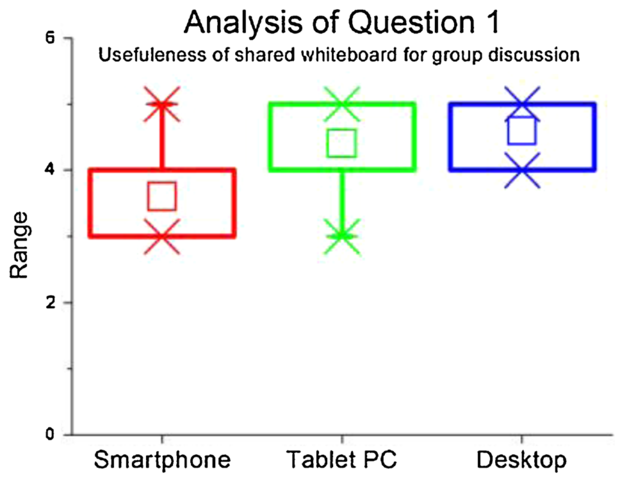

Fig. 6 Analysis of question 1: Usefulness of shared whiteboard on mobile devices

survey results for questions 3 depicted in Fig. 8 show that the result for question 3 was rated significantly worse than that for question 4 as expected. It is because the current our collaboration framework on mobile devices does not support audio / video conferencing tool with the application being shared with other participants for WYSIWIS (What You See Is What I See) which is an inclusion of collaboration. In future work we will implement the tool to support audio / video streams on mobile devices with various interactive consistency control mechanisms through the results obtained from this user study. Thus, it will seem for roaming participants to be more in real conference room even when using disparate mobile devices at remote locations.

In the ubiquitous collaboration linked with smartphone devices which still have high network latency and low computing performance as shown in Fig. 4, the interaction mechanism may have to be considered differently according to overheads—the network transactions and transformations for undo / redo operations in relaxed interaction mechanism and the waiting time for turn-taking among participants in strict interaction mechanism,

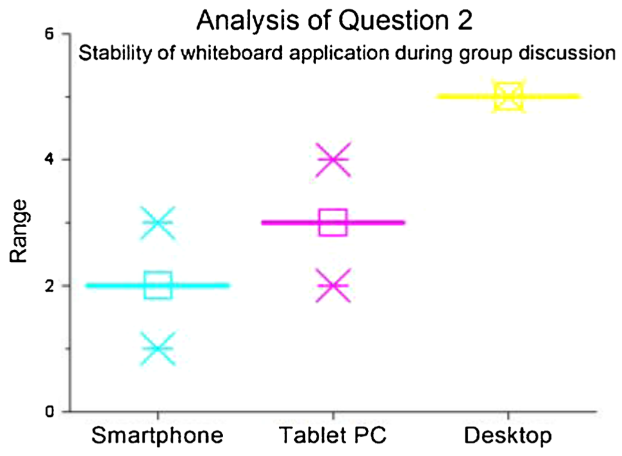

Fig. 7 Analysis of question 2: Stability of shared whiteboard application during group communication

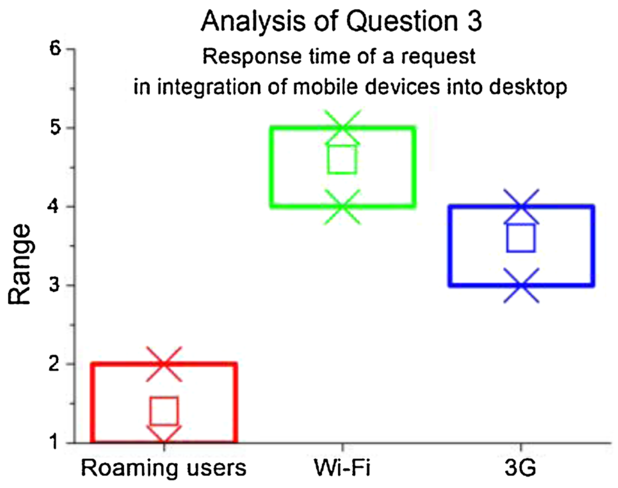

Fig. 8 Analysis of question 3: Response time of a request in integration of mobile devices using wireless network into desktops

where the strict interaction mechanism means to avoid conflicts and the relaxed interaction mechanism means to allow updates by any host (or participant) on any object (or data) and to resolve the uncoordinated updates [4]. When participants want to send drawing event messages in our distributed collaboration system, to do so may was satisfactory in small group size (Group A) since the possibility of direct conflicts of events occurred was rare as compared to moderate group size (Group B). As concurrent activities among participants increase, the number of network transactions (for undo / redo of interactive multimedia contents) increased as shown in the box plot of Fig. 9 for question 4, leading to the increase of complexity for managing the transactions and transformations of objects on shared whiteboard and the increase of overhead time for processing them with specifically smartphone devices. The result for question 5 depicted in Fig. 10 shows that the turn-

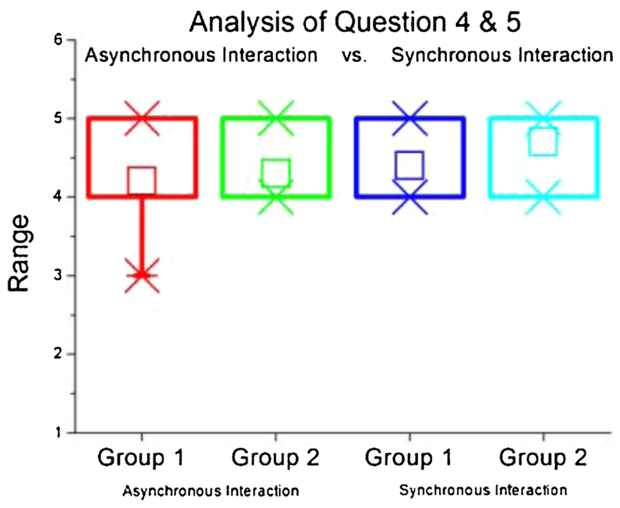

Fig. 9 Analysis of question 4 & 5: Asynchronous interaction vs. Synchronous interaction

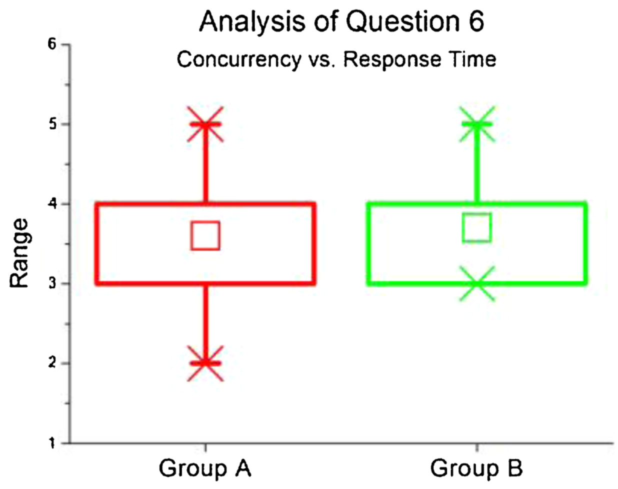

Fig. 10 Analysis of question 6: Concurrency vs. Response time

waiting time to provide a turn for only one participant at a time increased according to the increase of group size. If concurrent activities are small, the number of network transactions for undo / redo operations will be small and the waiting time for turn-taking will decrease.

When mobile devices using wireless network were involved in our collaboration, the response time of a request was increased as shown in the result of Fig. 8 for question 3 since the wireless network has high latency. Hence, in the case of the use of smartphone, we made the granularity of fine-grained actions larger to reduce the wireless network overhead. The shared whiteboard application uses fine-grained actions with the smallest major events. When a user requests an image loading action, it may be natural to simultaneously request it with other drawing actions. This natural request with larger-grained action improved the response (delay) time of a request but decreased the amount of concurrency as shown in the result on question 6 depicted in Fig. 10 and introduced complexity as shown in the result on question 7 depicted in Fig. 11. However, the unnatural granularity to improve response time

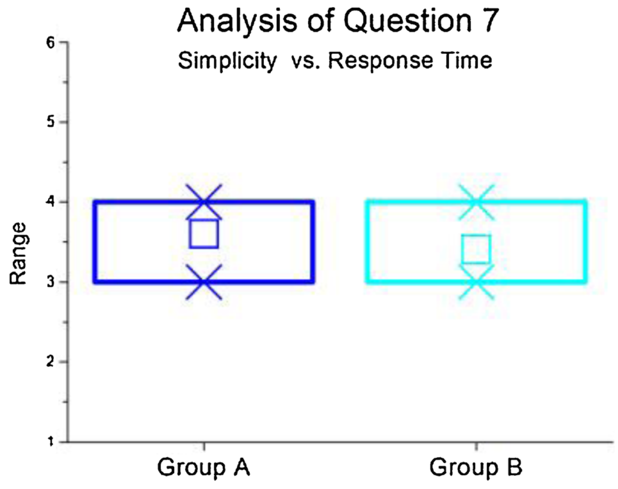

Fig. 11 Analysis of question 7: Simplicity vs. Response time

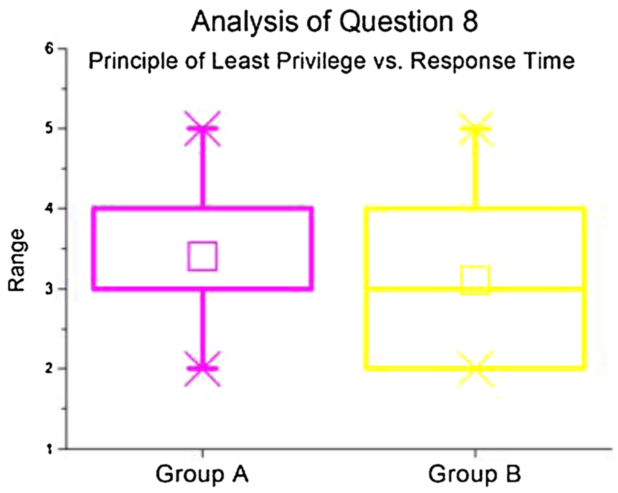

Fig. 12 Analysis of question 8: Principle of Least Privilege vs. Response time

had an effect on limiting access to the minimal level as shown in the result on question 8 depicted in Fig. 11. Therefore, without user's point of view [9] for the granularity of actions, unnatural granularity may violate the principle of least privilege [2] because it may give a user more privilege than needed. Thus, we need to consider access control mechanisms as well as consistency control mechanisms in future work. In future work we will implement various consistency control mechanisms based on this user study.

6 Conclusion

In this paper we have attempted to provide a virtually shared workspace for not only remote or co-located users but also roaming users with disparate access devices. This attempt has been driven by extending our prior ubiquitous collaboration system with the evolution from 2G cell phone to 3G / Wi-Fi mobile devices. As ubiquitous collaboration and access becomes more prevalent in the future, it will become more important to provide collaborative applications running on virtual workspace in which dispersed roaming users at remote locations can work together with disparate access devices.

The user study shows that the pervasive shared whiteboard can improve response time by decreasing the amount of concurrency to the shared whiteboard. However, the improvement of the responsiveness by decreasing the amount of concurrency to the shared whiteboard can introduce complexity with specifically smartphone devices. Also, the study shows the unnatural granularity can have an effect on limiting access to the minimal level. All of these results motivate us to re-think about research of software design issues with further empirical evaluations for ubiquitous collaboration to clarify the usability and limitation of current technologies.

Further research is planned to solve an interactive mechanism in synchronous or asynchronous fashion. The interactive mechanism for coordinating activities occurred in cooperating applications shared among multiple users is one of the significant challenges in ubiquitous collaboration. As ubiquitous collaboration and access with the extensive development of technologies becomes more prevalent, it will become more important to provide coordination mechanisms for pervasive collaborative applications running on virtual workspace.

Fundamentally collaboration is about sharing resources through interactions among people. The cooperation on the resources shared among users may hence produce new results on the shared resources. On the contrary, security is about restricting unauthorized access to resources and thus it is essential that the security of collaboration environments as well as of collaborative applications running on them is ensured while providing the openness only to users that are authorized to access them. Therefore, difficulties to deal with the conflicting goals of allowing and restricting accesses for resources among a group of users may happen in collaboration environment. Hence in future work we will provide a solution for controlling accesses as well as authentication for protecting secured computing environments and resources from unauthorized users as well as unsecured disparate access devices since the environments and resources can be compromised by inadequately secured entities—human, devices, software, data, and so on.

Also, we will focus on extending our work with various applications to new generation of smartphone such as LTE (Long Term Evolution) [15] 4G iPhone [10] in future work.

Acknowledgments This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the "Employment Contract based Master's Degree Program for Information Security" supervised by the KISA (Korea Internet Security Agency).

References

1. Android Platform. http://www.android.com/
2. Bishop, M., Introduction to Computer Security. Addison Wesley. 2004.
3. Community Grids Lab (CGL) at Indiana University, http://pti.iu.edu/cgl
4. Dommel HP, Garcia-Luna-Aceves JJ (1997) Floor Control for Multimedia Conferencing and Collaboration. ACM Multimedia Systems 5(1)
5. Everitt KM, Klemmer SR, Lee R, Landay JA (2003) Two worlds apart: Bridging the gap between physical and virtual media for distributed design collaboration. In: Proceedings of the SIGCHI conference on human factors in computing systems, CHI'03, pp.553–560, doi:10.1145/642611.642707
6. Fox G (2006) "Collaboration and community grids", Special Session VI: Collaboration and Community Grids in Proceedings of IEEE 2006 International Symposium on Collaborative Technologies and Systems (CTS 2006) Las Vegas; IEEE
Computer Society, Ed: Smari, Waleed & McQuay, William, pp 419–428. ISBN 0-9785699-0-3 DOI.
7. Fox G, Bulut H, Kim K, et al. (2003) Collaborative Web Services and Peer-to-Peer Grids. In: Collaborative Technologies Symposium (CTS'03)
8. Fox G, Wu W, Uyar A, Bulut H, Pallickara S (2003) "Global Multimedia Collaboration System" in Proceedings of the 1st International Workshop on Middleware for Grid Computing co-located with Middleware, Rio de Janeiro, Brazil.
9. Greenberg S, Marwood D (1994) Real time groupware as a distributed system: Concurrency control and its effect on the interface," Proceedings of the ACM CSCW Conference on Computer Supported Cooperative Work. ACM Press, North Carolina
10. iPhone, http://www.apple.com/iphone/
11. Ishii H, Arita K (1991) ClearFace, Translucent Multiuser Interface for TeamWorkStation". In Proceedings of the Second European Conference on Computer-Supported Cooperative Work. Sep. 25–27, Amsterdam, The Netherlands, 163–174
12. Ishii H, Kobayashi M (1992) "ClearBoard: A seamless medium for shared drawing and conversation with eye contact". Proceedings of CHI '92 Proceedings of the SIGCHI conference on Human factors in computing systems., pp 525–532. doi:10.1145/142750.142977
13. Java, http://www.oracle.com/technetwork/java/index.html
14. Kim K, Fox G (2011) Modeling, simulation, and practice of floor control for synchronous and ubiquitous collaboration. Multimedia Tools and Applications 53(1):213–236. doi:10.1007/s11042-010-0508-0
15. LTE (Long Term Evolution). http://en.wikipedia.org/wiki/3GPP_Long_Term_Evolution
16. NaradaBrokering, http://www.naradabrokering.org

17. Nescher T, Kunz A (2011) An interactive whiteboard for immersive telecollaboration. Visual Computer 27(4):311–320. doi:10.1007/s00371-011-0546-2
18. Pallickara S, Gadgil H, Fox G (2005) On the discovery of topics in distributed publish/subscribe systems. Proceedings of the IEEE/ACM GRID 2005 Workshop, Seattle, pp 25–32
19. Pederson E et al. (1993) "Tivoli: an electronic whiteboard for informal workgroup meetings," in Proceedings INTERCHI'93: Human factors in computing systems. Amsterdam, Netherlands: pp.391–398
20. Qiu X, Carpenter B, Fox G (2003) "Collaborative SVG as a Web Service", SVG Open 2003 Conference and Exhibition. Vancouver, Canada
21. SVGArena, 2003. http://www.svgarena.org/
22. Tang JC, Minneman S (1991) "VIDEOWHITEBOARD: video shadows to support remote collaboration". In Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology. CHI '91, 315–322, doi:10.1145/108844.108932
23. Tang A, Neustaedter C, Greenberg S (2004) "VideoArms: Supporting remote embodiment in groupware". In: Video Proceedings of CSCW, Vol. 4
24. Tang A, Neustaedter C, Greenberg S (2007) "Videoarms: embodiments for mixed presence groupware". In: People and Computers XX-Engage, 85–102

Kangseok Kim received Ph.D. in Computer Science from Indiana University at Bloomington, IN, USA. He is currently research professor of Knowledge Information Engineering department at Graduate School of Ajou University, Suwon, Korea. His main research interests include Mobile Computing, Ubiquitous Computing, Scalable Distributed Database System, Pervasive Collaborative Applications with Smart Phones, Mobile Network Security, Wireless Sensor Networks, Data Mining, and Bioinformatics.

Wonjeong Ha received M.S. degree in Knowledge Information Engineering at Ajou University, Suwon, Korea. She is currently working as a research staff in M2Soft Co., Ltd. Her main research interests include mobile computing, ubiquitous computing, and mobile security.

Okkyung Choi received her master's and Ph.D. degrees in Computer Science and Engineering from Chung-Ang University, Seoul, Korea. Now, she is working at Graduate School of Ajou University as a research professor. She previously worked at Sejong University as a visiting professor and Yonsei Graduate School of Information as a Post Doctor. She also has many experiences as a lecturer in several Universities. Her main research interests include Semantic Web Services, Ubiquitous Computing, Auction, and Mobile Security.

Hongjin Yeh received B.A in Mathematics Education from Seoul National University, M.S in Computer Engineering from Ajou University, Korea, M.S. in Applied Mathematics from Universite Joseph Fourier - Institut Nationale Polytechnique de Grenoble and Ph.D. in Computer Science from Universite Claude Bernard - Ecole Normale Superieure de Lyon, France. He is currently associate professor of Information and Computer Engineering department at Ajou University, Suwon, Korea. His main research interests include Mobile Computing, Mobile Security, and M2M Security.

Jai-Hoon Kim received the B.S. degree in Control and Instrumentation Engineering, Seoul National University, South Korea, in 1984, M.S. degree in Computer Science, Indiana University, IN, U.S.A., in 1993, and his Ph.D. degree in Computer Science, Texas A&M University, TX., U.S.A., in 1997. He is currently a professor of Department of Information and Computer Engineering at Ajou University, South Korea. His research interests include Distributed Systems, Real-time Systems, Mobile Computing and Ubiquitous Computing.

Manpyo Hong received the B.S., M.S, and Ph.D. degrees in Computer Science, Seoul National University, South Korea. He is currently a professor of Department of Information and Computer Engineering at Ajou University, South Korea. His research interests include Mobile Computing, Ubiquitous Computing, Cloud Computing, Mobile and Network Security, and Artificial Immune System with Secure Anonymous Communication.

Taeshik Shon received Ph.D. in Information Security from Korea University, Seoul, Korea and M.S. and B.S. in Computer Engineering from Ajou University, Suwon, Korea. From Aug. 2005 to Feb. 2011, Dr. Shon was a senior engineer in the Convergence S/W Lab, DMC R&D Center of Samsung Electronics Co., Ltd. He is currently assistant professor at the Division of Information and Computer Engineering, College of Information Technology, Ajou University. His research interests include Convergence Platform Security, Mobile Cloud Computing Security, Mobile / Wireless Network Security, WPAN / WSN Security, Anomaly Detection.