Information Engineering Express
International Institute of Applied Informatics
2015, Vol.1, No.4, 53 – 63

Implementing a Collaborative Web Platform based on Transparent Layers

Akihiro Sugiyama* , Yusuke Niwa* ,
Shun Shiramatsu* , Tadachika Ozono* , Toramatsu Shintani*

Abstract

We have developed a novel collaborative web platform that shares a part of one's desktop. In the case of using a general desktop sharing software such as VNC and TeamViewer, all users interfere with each other on a shared desktop, resulting in disturbing user's work. Prior work has proposed a common collaborative workspace, but there are problems with collaborative workspaces using desktop applications. The platform uses a part of user's desktop as a collaborative workspace in consideration of interference to a user. To create the platform, we introduce a transparent layer with two aspects: visible- and event-transparency. The background of the transparent layer can be opaque. The transparent layer can forward received events to other layers as well as to other applications. The layer can interact with a user's desktop, thus taking a screenshot of a user's desktop and sharing the screen shot among other users. Furthermore, we have been developing a new web browser, Silhouette Web Browser, that consists of five transparent layers including existing desktop layers. The platform shares one's desktop by using transparent layers, thus using one's desktop as a collaborative workspace in consideration of interference to users. This paper describes the architecture and implementation of the proposed platform and its application.

Keywords: computer-supported collaborative work, transparent layer, desktop sharing

1 Introduction

We address issues of collaborative workspaces using multiple desktop applications. In the case of document editing, the author and corrector need to exchange PDF documents, TeX documents, Word documents, and so on. Despite the fact that users would prefer to share the contents more easily, they must currently use multiple communication tools. Solutions to this problem are now being widely discussed. One remedy is to implement a system that can be integrated into existing applications [1][3].

In this paper, we aim to implement a novel collaborative web platform that does not depend on a particular desktop application. To this end, we need to create an integrated framework of file, content, event, and screenshot sharing on one's desktop. We introduce

* Nagoya Institute of Technology, Aichi, Japan

transparent layers that have a fundamental function for collaborative workspaces. Furthermore, we have been developing a new web browser, Silhouette Web Browser, that enables seamless content, screenshot, and file sharing with transparent layers.

To implement the proposed system, we need to consider the interference to users. The system displays superimposed layers on existing desktop application's windows, thus hindering visibility and operability. Therefore, we design a transparent layer with two aspects: visible- and event-transparency. In other words, the background of the transparent layer can be opaque. The transparent layers can forward received events to other layers and other applications. Furthermore, we assume that we use the system with various devices such as PC and tablet. There is issue about layout of web contents. The layout of contents is dependent on the device. The system shares the contents based on image, thus holding the layout of the contents.

We describe a collaborative editing system as an application of our system. In this system, a teacher and a student share the student's desktop as well as documents and their annotations. One of the advantages of this system is that the student can restrict a synchronization area of his or her desktop using the transparent layers. The student selects a portion of the desktop to share with the teacher. The teacher can browse the student's desktop and add annotations to the document. The annotations are instantly displayed on the student's desktop.

In this study, we first describe the transparent layer with the two above-mentioned aspects and the layer's functions. Then we describe the Silhouette Web Browser. Finally, we describe an application using the proposed browser.

2 Transparent Layer

In this section, we describe the transparent layer and its functions. The transparent layer is similar to windows of existing browsers. Unlike the existing windows, the transparent layers can not only display contents but also interact between layers and cooperate with the desktop. Combining layer functions allows users to use a layer with a new combined function. One advantage of the layer is that it can have various shapes, e.g., rectangle, star, and cloud. Users can create layers with various shapes to display noticeable contents. Another advantage is its display level, e.g., it can be displayed between file icons and a desktop image. Layer dynamics (e.g., gravity) can be represented using animation. For example, some grouped layers move together.

Interestingly, layers have two aspects: visible- and event-transparency. Layers can be configured for passing through transparency of background and mouse events, and window level to control interference to them. As an example of controlling whether they pass through events, layers are controllable in regard to whether a layer can be controlled with a mouse. For instance, the proposed browser enables desktop fusion: a mixture of a desktop and event-transparent layers. The transparent layer of a desktop fusion appears to fade into the desktop. The layer is displayed between the file icons and desktop images. By controlling the transparency of the background, the browser can display a layer superimposed on other layers and applications. For example, additional information can displayed on other layers and applications. Moreover, layers can dynamically change window level. In some situations, it is possible to display the layers at the top or back of the desktop.

Layers are sharable among users. Moreover, a supported cooperative work environment is feasible using layers. There are three types of layer: pair-sharing, public-sharing,

and private-sharing. In addition, there is a web desktop shot (WDS) function. The WDS function takes a screenshot of the area that the layer overlaps. Users can restrict the screenshot area by changing the layer area.

2.1 Sharing of Layers

We classify attributes of a layer in accordance with the range of users who share the layer: one for one, more than three, and an unspecified number, called the pair-sharing, private-sharing, and public-sharing layers, respectively. Users can configure the layer attributes and create the layer on all window levels. A pair-sharing layer is a private-sharing layer with only two users. To create a pair-sharing layer, a user sends an invitation to another user, with whom the pair-sharing layer is to be shared. Users can create a private-sharing layer on all window levels, assigning a token and name to one transparent layer to be used for authentication. For example, in the case of sharing a schedule in a group, group members can share the schedule among themselves by displaying the schedule on the desktop by using a private-sharing layer.

Users can create the public-sharing layer on their desktop. All users who run the Silhouette Web Browser can see the contents displayed on the layer by using the public-sharing layer. They can set whether the public-sharing layer is displayed or not. If they decide not to share the contents among all users, they can stop doing so.

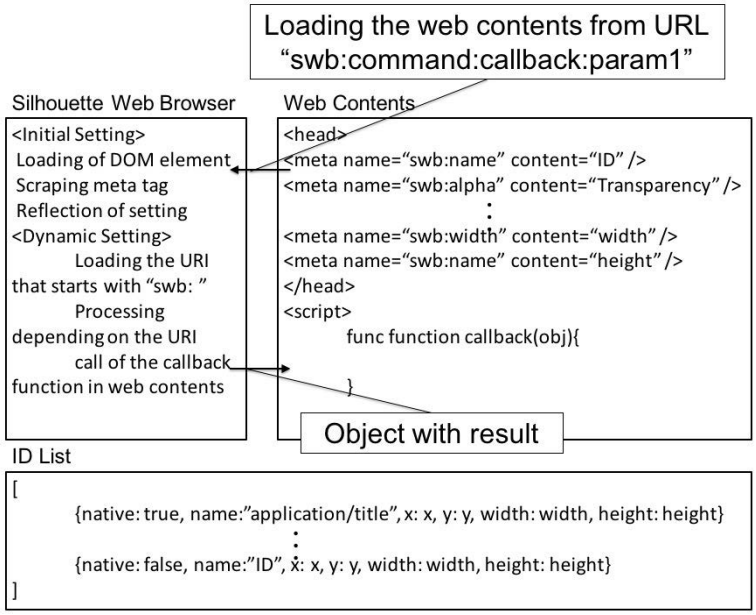

Figure 1: Overview of interaction with the Silhouette Web Browser.

2.2 Interaction between Layers

Fig. 1 shows an overview of interaction with the Silhouette Web Browser. In the browser, the layers can interact with web contents using URL Scheme. The layer reads web content specified by the user with a URL and scrapes meta tags in the HTML of the web content. Then, the layer reflects settings written in the meta tags. The elements of the meta tags

include name, transparency, desktop coordinates, width, and height. The ID list contains a set of desktop applications and transparent layers. All windows and transparent layers have a *native* property. If the *native* property of a window or layer is true, then it belongs to a desktop application.

Moreover, the settings of a layer can be changed dynamically by loading the URL that specifies *swb:command:callback:param 1:param 2:⋯:param n* changing the settings of the layer. There are *name, alpha, x, y, width, height,* and *windows* commands for identifier, transparency, x-coordinate of the upper left, y-coordinate of the upper left, layer width, layer height, and the set of window identifiers.

In addition, one layer can interact with another. By loading the URL that specifies *swb:send:callback:a identifier of destination:data*, the layer can send the data to the layer with the identifier. The layer that receives the data executes the callback function with the data.

2.3 Web Desktop Shot (WDS)

The WDS allows the user who runs the Silhouette Web Browser to share the desktop with another user using a layer. Users create layers with the same token and name. The WDS takes a screenshot of the desktop running the browser and sends the screenshot to a server. The another user browses the screenshot on the server to share the desktop. Users take a screenshot in three roles: sender, receiver, and free. The sender takes a screenshot and shares it with receivers. The receiver displays the sender's screenshot on a layer and takes a screenshot of the sender's desktop. The free user does not receive any screenshots.

The WDS takes a screenshot of a specified area in the desktop. The user can configure the size and coordinates of the layer to share the specified area. The WDS also takes a screenshot of the area that the layer overlaps. In other words, the user can share the part of the desktop that the user wants to share with other users.

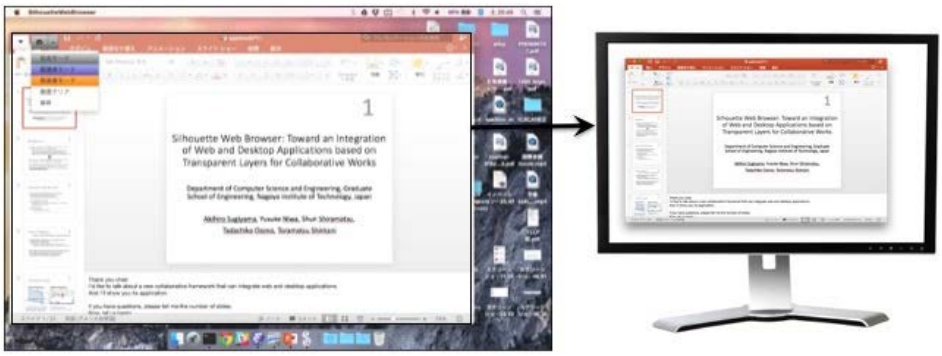

Figure 2: Range selection of desktop sharing with the Web Desktop Shot.

Fig. 2 shows an example execution of a range selection of desktop sharing using the WDS. First, users create a layer on each desktop and input the same token and name of the layer. Next, they decide their roles. Then, the sender defines a range of the desktop that he or she wants to share. To share the desktop, the user superimposes the WDS on it (Fig. 2, left). They share the sender's desktop by clicking the camera icon. The receiver browses the screenshot with the layer (Fig. 2, right). In this case, the sender takes a screenshot of creating document and the screenshot is displayed on the receiver's desktop.

Using the WDS, the sender shares the desired area in the desktop. In contrast to existing remote desktop software, the sender can share a part of the desktop. Users share the desktop and can change the shared area easily, because the WDS shares the area on which the sender superimposes the WDS.

3 Silhouette Web Browser

The Silhouette Web Browser can display superimposed transparent windows. We call a window in the browser a transparent layer. The browser manages the layers, enabling interaction between layers. Moreover, combining layer functions allows the user to use more complex functions. As noted, we describe a novel collaborative web platform using the proposed browser. In this section, we explain a novel desktop structure of the browser and new layer model.

Figure 3: Execution example of the Silhouette Web Browser.

Fig. 3 shows a browser execution. There is an existing PDF viewer (Fig. 3 A), and three layers (Fig. 3 B, C, D). The layer at the back of the icons allows users to share files (Fig. 3 B). The layer hovers on the PDF viewer and displays feedback from another user in the layer (Fig. 3 C) as well as web content (Fig. 3 D). Users can select whether the layers are displayed at the front or back of the desktop. Furthermore, users can work together on the user's desktop by using the layer that can share contents.

We describe the structure of multiple transparent layers in the Silhouette Web Browser. In other words, we describe the novel layer model that adds new layers using the browser to an existing desktop layer structure. The most significant aspect of displaying contents with the proposed browser is its focus on control of interference to users. For example, the contents can be displayed without obstructing the user's work through minimal interference. However, content can also be displayed by interfering with it aggressively. We create this capability by introducing a five-layered structure. The browser changes the levels of the layers by adjusting the degree of interference in the user's activity. The optimal level of a layer depends on its context. For example, content is displayed in the forefront of the desktop, when it needs to be noticed as soon as possible. However, content can be integrated with a desktop when the content is to be shared discretely.

Figure 4: Left: Overview of layer model. Right: Layers on a desktop.

The Silhouette Web Browser contains presentation (L_p), application (L_A), work (L_W), file (L_F), and desktop layers (L_D) ordered from the front to the back of the desktop. Fig. 4 shows an overview of the layer model. The three layers (L_P, L_W, and L_D) are extra layers. Users can select the window level corresponding to the content displayed on the layers by adding new layers to the existing layer structure.

A layer closer to the front is much more visible on the desktop, whereas that closer to the back is less visible on the desktop (Fig. 4, left). In the case in Fig. 4, L_P is displayed in the foreground, and the desktop picture is displayed behind it. Users see the layers on top of one another on their desktop (Fig. 4, right).

L_P is the layer displayed in the foreground among the windows on the desktop (Fig. 4, L_P). Users can avoid overlooking contents displayed on L_P since L_P can pass through mouse events. Users can control applications on L_A while browsing contents displayed on L_P by passing through the mouse events on L_P. On the other hand, an increase in the use of L_P will lead to a decline in the region of the layers behind L_P. Users frequently work on L_A. An increase in the content on L_P will lead to a decline in such a user's workspace, interfering with the user's work. However, users can decrease interference by controlling transparency of the background of layers. Users can display contents in the foreground on the desktop in an optimal way by controlling visible- and event-transparency.

L_W is the layer displayed behind in L_A, i.e., behind the windows that the user opens (Fig. 4, L_W). Hence, L_W can display contents without disturbing the user's work. L_W can pass through mouse events. The layers can cooperate with existing applications. Therefore, by creating L_W and using the function of sharing one's desktop, as described above, users can share the desktop for a specific application to collaborate with other users.

L_D is the layer displayed behind all the other layers, i.e., L_D displays contents as if there is an integration of L_D and the desktop picture (Fig. 4, L_D). L_D is displayed at the back of the desktop. Therefore, L_D can display contents without disturbing a user's work. L_D is configured at the window level close to the desktop picture. Users can see the contents easily displayed on L_D by using the shortcut key to show the desktop. L_D passes through mouse events. Users work on L_A frequently. Therefore, L_D can display contents without disturbing a user's work. However, an increase in windows on L_A will lead to a decline in the space of L_D. It is preferable that L_D displays less urgent contents, similar to a calendar.

We introduce a five-layered structure to change a layer's window level corresponding to the content displayed on the layers. Using multiple layers enables the browser to display

contents in consideration of interference with users.

4 Evaluation

We conducted a programming contest as an evaluation of the Silhouette Web Browser in June 2015. On the programming contest, 19 participants had to develop each collaborative work support system with the Silhouette Web Browser for four weeks. The participants were consisting of 8 undergraduate students and 11 graduate students of our laboratory. 10 of 19 students were able to develop collaborative work support systems as Web applications without the Silhouette Web Browser. We conducted a questionnaire to the participants shown in Table 1, and we received 16 answers.

Table 1: Description of questionnaire.

	Description
Q1	Was “ visible-transparency ” function effective for your development?
Q2	Was “ event-transparency ” function effective for your development?
Q3	Was “ Web Desktop Shot” function effective for your development?
Q4	Was “ sharing of layers ” function effective for your development?
Q5	Do you think that you cannot develop the same system for the same period without the Silhouette Web Browser?

The result of the questionnaire is shown in Fig. 5. More than 80% answers on Q1 and Q2 are agree, strongly agree, or very strongly agree. It means that visible-transparency and event-transparency function are effective for developments of collaborative work support systems. More than 80% answers on Q5 are agree, strongly agree, or very strongly agree. It means that the Silhouette Web Browser effectively supports the development of collaborative work support systems. However, more than 40% and 20% answers of Q3 and Q4 respectively are agree, strongly agree, or very strongly agree. Therefore, we need further improvement of the sharing function.

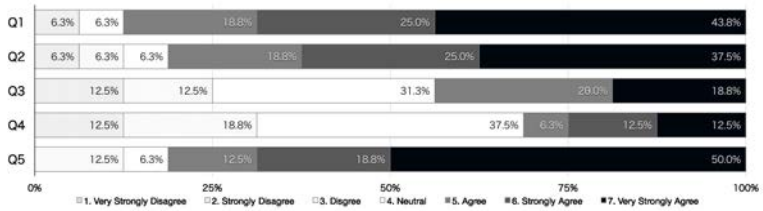

Figure 5: Result of questionnaire.

5 Application

For application of the Silhouette Web Browser, we describe an editing system as an example of the proposed browser. The production of research documents such as papers or presentation documents is fundamental to academic practice. Multiple roles occur in

document editing, such as a teacher and a student. The student writes a document and has his or her document corrected by the teacher. In face-to-face document editing, they always share the same document and can verbally explain the details, gesture over documents, and write manually in real-time. In editing documents by using computers, they need to share the document when the student or teacher modifies it. For example, the teacher corrects a document and sends it to the student. Then the student modifies the document on the basic of the received document and sends the modified document to the teacher. This system shares a part of the student's desktop, thus viewing the same document seamlessly on the desktop in real-time. Furthermore, the teacher can write manually by using the tablet.

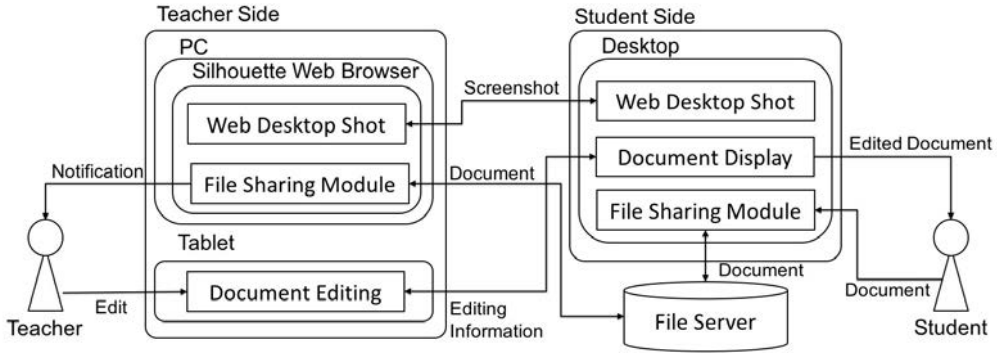

Figure 6: System architecture of editing system.

Fig. 6 shows an overview of the system architecture of the editing system. A student creates a document and shares it using the proposed browser on a PC. A teacher edits the document using a PC and tablet. The system on the PC uses the Web Desktop Shot functions in the proposed browser, whereas that on the tablet computer uses the editing module. The student and teacher can share the document with the File Sharing Module. When they share the document, the system announces the file has been uploaded by the student. The teacher browses the document and creates annotations in the shared document using the tablet computer. On the student's desktop, the layer displays the annotations created by the teacher. The student takes a screenshot of the edited document with the WDS and shares the screenshot with the teacher.

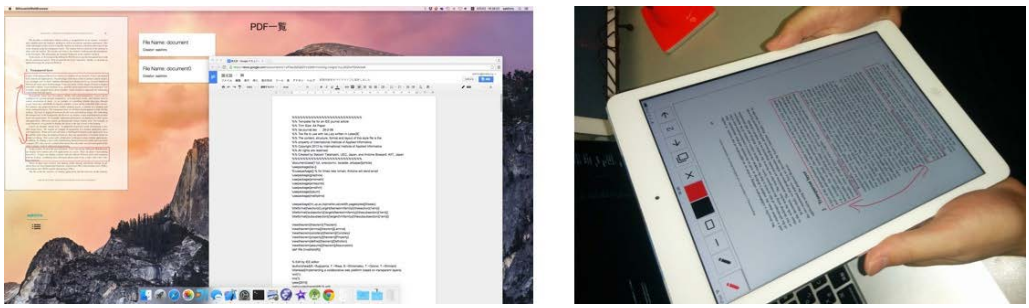

Figure 7: Example of collaborative editor based on the Silhouette Web Browser. Left: desktop of student's PC. Right: the teacher edits a document using a tablet.

Fig. 7 shows an example of editing execution on the research document. The student writes a TeX document and shares it with the teacher using Google Docs (Fig. 7, left). The

system displays the shared files on the back of the desktop. The teacher edits the document using the tablet (Fig. 7, right). First, the student shares the document file. Next, the teacher opens the document and creates annotations using the system. The teacher can select four tools for editing the document: freehand memo, text annotation, color, and figure, including circle, rectangle, and straight. When the teacher opens the document, the system displays the editing document on the student's desktop.

The teacher can freely create complex annotations such as a curve using freehand memo, but it is difficult for the teacher to draw a straight line. However, the teacher can create exact circle, rectangle, and straight annotations using figure annotations and create smaller characters than freehand memo annotations using text annotations. The annotations created by the teacher are displayed on the layer. In other words, the Silhouette Web Browser overlays the annotations on the student's desktop. This system helps edit a document directly on the desktop when the teacher edits the document, and it enables the student to use familiar applications.

6 Related Works

Collaborative workspaces are shared to facilitate awareness for the remote user [8][9]. Tracs is a system that uses dual-sided see-through displays through which a user can control the transparency of individual areas. Tracs allows users to collaborate to resolve issues regarding personal privacy, screen content privacy, and visual interference. A 3D-Board adds to the virtual 3D embodiment of a remote user on remote collaborative workspaces. By blending the front-facing 3D embodiment of a remote collaborator with the shared workspace, an illusion is created as if the observer were looking through the transparent whiteboard into the remote user's room. In the case of using transparent contents, it is important to take rivalry into account [10].

Another study on facilitating more immersive interaction with the remote environment in general developed a fully functional system for mobile remote collaboration. A shared visual information helps collaborators understand the current state of their task and enables them to communicate and ground their conversations [5]. The remote user can communicate via spatial annotations that are immediately visible to the local user [12]. RichReview [2] is a system for creating annotations on the front of digital documents using freeform inking, voice for narration, and deictic gestures in support of voice.

Prior work has created tools by integrating into an existing application. InterTwine [1] is a system for linking information regarding search history in web browsers with information regarding application usage, helping a user to re-find information in both applications. Brandt et al. [3] has proposed a Web search interface integrated into the Adobe Flex Builder development environment. Pongnumkul et al.'s Pause and Play system [4] demonstrates how a 3D modeling application can work together with a video player. This system automatically pauses and plays the video by detecting important events in it and linking them with corresponding events in the target application. In contrast to the related work, the Silhouette Web Browser creates a collaborative workspace on a user's desktop. Users can work in their favorite environments. In other words, users can use familiar desktop applications for collaboration work. Users can control the transparency and window levels of the layers to avoid interference and rivalry on the desktop.

The proposed browser is also inspired in part by pair programming [11], in which two programmers work in the same workspace, one writing code, while the other watches and

reviews the typed code. The two programmers typically work on the same project. Pair Research [7] is a kind of interaction within a research group in which group members are assigned as pairs each week to work together on each other's projects. Pair Research motivates participants, helps them learn about their colleagues, and makes opportunities for further collaboration. The browser differs in that it is applicable to not only programming but also other research tasks such as creating presentation materials and writing papers without any need for partners to work in the same workspace.

7 Conclusion

We have developed a web platform for collaborative work that shares a part of a user's desktop in the consideration of interference to a user. The core innovation is transparent layers to share a part of one's desktop and control interference to users. The transparent layers have two features: First, the background of the transparent layer can be opaque. Second, the transparent layer can forward received events to other layers as well as to other applications. Users can share the layers in three modalities: one for one, more than three, and an unspecified number. A layer can interact with web contents or another layer using a URL Scheme. Transparent layers allows users to use their desktop as a collaborative workspace. We believe that these features can help users to collaborate with other users.

Furthermore, we have been developing the Silhouette Web Browser, a WebKit-based web browser that consists of five transparent layers: presentation, application, work, file, and desktop layers. The user can change the depth level of the layers in accordance with the contents to be displayed. The browser enables seamless content, screenshot, and file sharing among users by using transparent layers.

For an application of the Silhouette Web Browser, we described an editing system. This system shares a part of a student's desktop, sharing the document and annotations on the student's desktop in real-time. The Silhouette Web Browser overlays the layers displaying the annotations and the document on the student's desktop. This system helps edit a document directly on the student's desktop when the teacher edits the document.

The student selects a region that he or she wants to edit using the WDS. The teacher browses and edits the region using a tablet computer. The annotations that the teacher writes are displayed on the student's desktop. Therefore, the student can edit documents on his or her desktop by using familiar desktop applications.

Acknowledgments

This work was supported in part by JSPS KAKENHI Grant Number 15K00422, 25870321.

References

[1] Fourney, A. et al., "InterTwine: creating interapplication information scent to support coordinated use of software," Proc. of the 27th annual ACM symposium on User interface software and technology, 2014, pp. 429-438.

[2] Yoon, D. et al.,"RichReview: blending ink, speech, and gesture to support collaborative document review," Proc. of the 27th annual ACM symposium on User interface software and technology, 2014, pp. 481-490.

[3] Brandt, J. et al., "Example-centric programming: integrating web search into the development environment," Proc. of the SIGCHI Conference on Human Factors in Computing Systems, 2010, pp. 513-522.

[4] Pongnumkul, Suporn, et al., "Pause-and-play: automatically linking screencast video tutorials with applications," Proc. of the 24th annual ACM symposium on User interface software and technology, 2011, pp. 135-144.

[5] Gergle, Darren, et al., "Action as language in a shared visual space." Proc. of the 2004 ACM conference on Computer supported cooperative work, 2004, pp. 487-496.

[6] Amores, J. et al., "ShowMe: A Remote Collaboration System that Supports Immersive Gestural Communication," Proc. of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems, 2015, pp. 1343-1348.

[7] Miller, R. C., et al., "Pair research: matching people for collaboration, learning, and productivity", In Proc. of the 17th ACM conference on Computer supported cooperative work & social computing, 2014, pp. 1043-1048.

[8] Lindlbauer, D., et al., "Tracs: transparency-control for see-through displays", In Proc. of the 27th annual ACM symposium on User interface software and technology, 2014, pp. 657-661.

[9] Zillner, J., et al., "3D-board: a whole-body remote collaborative whiteboard", In Proc. of the 27th annual ACM symposium on User interface software and technology, 2014, pp. 471-479.

[10] Laramee, R. S., et al., "Rivalry and interference with a head-mounted display", ACM Transactions on Computer-Human Interaction, 9(3), 2002, pp. 238-251.

[11] Beck, Kent. "Extreme programming explained: embrace change", Addison-Wesley Professional, 2000.

[12] Gauglitz, S., et al., "World-stabilized annotations and virtual scene navigation for remote collaboration", In Proc. of the 27th annual ACM symposium on User interface software and technology, 2014, pp. 449-459.