SAGE2: a Collaboration Portal for Scalable Resolution Displays

Luc Renambot, Thomas Marrinan, Jillian Aurisano, Arthur Nishimoto, Victor Mateevitsi, Krishna Bharadwaj, Lance Long, Andy Johnson, Maxine Brown, Jason Leigh (*)

University of Illinois at Chicago

renambot@uic.edu, thomas.j.marrinan@gmail.com, mvictoras@gmail.com, jillian.aurisano@gmail.com, arthur.nishimoto@gmail.com, krishnaknbharadwaj@gmail.com

long.lance@gmail.com, ajohnson@uic.edu, maxine@uic.edu

* **University of Hawai'i at Mānoa**

leighj@hawaii.edu

ABSTRACT

In this paper, we present SAGE2, a software framework that enables local and remote collaboration on Scalable Resolution Display Environments (SRDE). An SRDE can be any configuration of displays, ranging from a single monitor to a wall of tiled flat-panel displays. SAGE2 creates a seamless ultra-high resolution desktop across the SRDE. Users can wirelessly connect to the SRDE with their own devices in order to interact with the system. Many users can simultaneously utilize a drag-and-drop interface to transfer local documents and show them on the SRDE, use a mouse pointer and keyboard to interact with existing content that is on the SRDE and share their screen so that it is viewable to all. SAGE2 can be used in many configurations and is able to support many communities working with various types of media and high-resolution content, from research meetings to creative session to education.

SAGE2 is browser-based, utilizing a web server to host content, WebSockets for message passing and HTML with JavaScript for rendering and interaction. Recent web developments, with the emergence of HTML5, have allowed browsers to use advanced rendering techniques without requiring plug-ins (canvas drawing, WebGL 3D rendering, native video player, etc.). One major benefit of browser-based software is that there are no installation requirements for users and it is inherently cross-platform. A user simply needs a web browser on the device he/she wishes to use as an interaction tool for the SRDE. This lowers considerably the barrier of entry to engage in meaningful collaboration sessions.

Keywords

Scalable Resolution Display Environments, window manager, web-based, collaboration, multi-user interaction, large-scale displays, application development.

1. INTRODUCTION

Today, scientific data is collected, stored and analyzed digitally. Scientific phenomena are observed with new types of digital instruments, sensors and robotic autonomous vehicles capable of collecting data at ever-increasing resolutions. Natural phenomena from global weather systems to chemical reactions at the atomic level can now be simulated inside supercomputers, generating massive volumes of scientific data. These troves of data are invaluable to scientists as they explore the raw information and evidence needed for new insights and discoveries. However, making those insights is an increasingly complicated task, as the scale and complexity of data continue to grow at unprecedented rates. Since big data problems frequently require the combined efforts of many individuals from disparate fields, the next generation of visualization and interaction environments will need to enable collaboration and group work.

To deal with the scale and complexity of data, the *2007 DOE Visualization and Knowledge Discovery workshop report* [1] and the *2008 NSF Building Effective Virtual Organizations workshop report* [2] recognized that new modalities for accessing more visual information were necessary and described SRDEs as the type of environments that are crucial for next-generation collaborative cyber-enabled exploration. Furthermore, there is

Figure 1. Co-located collaborative session using SAGE2. Left – photo of ten users interacting with a SRDE tiled wall. Right – snapshot of a single user connected to SAGE UI, which provides interaction methods and a visual representation of the SRDE.

now conclusive evidence that large-scale display environments enable collaboration and significantly amplify the way users make sense of large-scale and complex data [3-11].

One software system that has addressed these issues is SAGE, the Scalable Adaptive Graphics Environment [12-15]. SAGE, in conjunction with SRDEs, represents a new type of "digital lens" – a high-resolution display that can effectively visualize large amounts of data in a collaborative environment. SAGE is an open-source middleware that provides users with a common operating environment, or framework, to access, display and share a variety of data-intensive information. The software allows each user to create a pointer on the SRDE by using their own personal device, or to directly approach the SRDE and interact through a multi-touch interface. In this manner, multiple users can simultaneously add and interact with content. SAGE utilizes pixel streaming over high-speed networks to display content ranging from high definition images and videos to PDF documents and laptop screens. While SAGE is being used to drive over 100 SRDEs around the world, its architecture was based on a monolithic software stack that made it increasingly difficult to integrate new capabilities as user requirements grew.

This paper presents the prototype for the next generation of SAGE, called SAGE2 for Scalable Amplified Group Environment. SAGE2 is a lightweight web-based middleware that runs in a browser. Along with the salient features of SAGE, SAGE2 (depicted in Figure 1) aims to expand upon SAGE's collaborative framework to support interaction with a wider variety of content and enable a richer set of input modes for multiple users. Web-based applications and JavaScript programming are becoming increasingly popular and when coupled with strong application development support, we believe that SAGE2 will attract a large community of developers for multi-user, high-resolution applications. HTML5 allows a browser to be used as a powerful rendering middleware. SAGE2 exploits this factor to transform a browser into an active environment capable of supporting diverse media content and advanced 2D and 3D visualizations. Additionally, SAGE2 becomes inherently cross-platform and requires no installation beyond having an up-to-date browser. Since web-based data portals and visualization tools are widely used for collaborative research in industry and academia, SAGE2 is capable of supporting a large array of collaborative research activities.

2. RELATED WORK

Scientific visualization on SRDEs takes advantage of parallel rendering frameworks. The "Distributed Graphics System" [16] was an early system for rendering content to tiled displays. A Silicon Graphics Onyx2 with eight R10000 processors drove the tiled display; multiple networked client applications could connect and share the display space at once. WireGL [17] is a sort-first parallel renderer with a parallel interface that distributes the rendering primitives to a cluster of nodes. WireGL does support scalable display sizes, however only a single application can drive the tiled display. Influenced by WireGL, Chromium [18] is a framework for interactive rendering on clusters, which uses OpenGL to move geometry across the network. Existing OpenGL desktop applications can be ported to work on a cluster by utilizing Chromium, with only a few modifications. Equalizer

[19] is an OpenGL parallel rendering system that supports scalable display environments. Using physical and logical abstraction, applications can run on a single or multiple displays.

OmegaLib [20, 21] is a middleware that uses Equalizer for parallel rendering and provides the tools to develop immersive 2D-3D applications for flexible systems ranging from tiled display walls to CAVE environments. OmegaLib also offers event handling that supports multiple heterogeneous devices. Cross Platform Cluster Graphics Library (CGLX) [22] is an OpenGL graphics framework for distributed, high performance visualization systems. CGLX allows users to easily develop new or adapt existing OpenGL desktop applications for visualization clusters such as tiled displays and multi-projector systems. CGLX supports collaboration through multiple multi-touch devices. Input events are synchronized with the display environment and scene information is streamed to the mobile device [23].

In many disciplines, collating data from various sources and juxtaposing content becomes an important task. In these scenarios, scalable window managers that support multi-user interaction are utilized to give SRDEs the appearance of one seamless desktop. The Distributed Multihead X [24, 25] is a parallel X11 server. A traditional window manager (like KDE or Gnome) needs to run on top of DMX to fully utilize its capabilities. DMX acts as an X11 proxy server that accepts server connections, therefore allowing the construction of scalable display environments. CubIT [26] is a multi-user presentation and collaboration framework designed for the large-scale projector and multi-touch LCD walls of The Cube exhibition and learning facility. Using either web-based or iOS interfaces, users are able to upload and manage multimedia content to the wall. The Python/Kivy multi-touch interface enables drag and drop interaction allowing moving, scaling and sharing of content across the shared canvas or between users. LACOME [27, 28] is a multi-user collaborative system that supports multi-user screen sharing through VNC. Participants can connect to the LACOME server using the standard VNC protocol, share their screen on a large display and interact with the other screens by moving them around. DisplayCluster [29] closely resembles the original SAGE, as it provides a windowing system for multimedia content across SRDEs. While claiming improved performance over SAGE, it relies purely on pixel streaming whereas SAGE2 can additionally take advantage of native distributed rendering.

	Scalable Resolution	Window Manager	Distributed Rendering	Pixel Streaming	Heterogeneous Multi-user Event Handling	Native Application API	Web-based	Cross-platform
WireGL	X		X			X		
CGLX	X		X		X	X		X
Equalizer	X		X			X		X
Chromium	X		X			X		
OmegaLib	X		X		X	X		X
CubIT	X	X	X		X		X	X
DisplayCluster	X	X		X	X			X
SAGE	X	X		X	X			
SAGE2	X	X	X	X	X	X	X	X

Table 1. Summary of features from select software that is related to SAGE2. Many software packages offer valuable components for collaboration on SRDEs, but SAGE2 is the first software to combine these features in one package.

Co-located and remote collaboration has been greatly affected by the advancement of web-based technologies. The web is being increasingly used to collaborate in a variety of scenarios such as teaching, corporate meetings and manufacturing. Binary Meetings [30], a web-based collaboration tool for lab tutorial classes, enables students to interact with each other and with lecturers through chat rooms, email, discussion forums and webcam teleconferences. DiCoDEv [31] is a web-based virtual collaborative platform that can be used during manufacturing product and process design evaluation. The DiCoDEv platform allows multiple users to work in collaborative and distributed way, decreasing considerably the time required for the designing phase to be completed. CollaBoard [32] uses web technologies to create a video, audio and data conferencing system that

imitates a life-sized face-to-face meeting. Recognizing the wealth of content on the web, Montage [33] aims to render multiple web pages on large tiled displays, using grouping and filtering techniques to make discoveries.

While systems exists to address the challenges of displaying rich information on SRDEs and enable multi-user simultaneous interaction from heterogeneous devices, SAGE2 presents a unified software to enable various forms of collaboration. Utilizing the power of the web, a wide variety of content can be displayed and interacted with, making SAGE2 an ideal collaborative framework across a wide range of disciplines in academia and industry. A summary of software features for various software packages is presented in Table 1, where SAGE2 is the only environment offering all aspects required (scalable resolution, window manager, distributed rendering, pixel streaming, heterogeneous multi-user event handling, native API, web-based, and cross-platform).

3. SAGE2

SAGE2 is a middleware interfacing between users and applications. It acts like an operating system for large-scale tiled display walls and supports direct and remote interaction. SAGE2 consists of one single web server that hosts a number of web pages and therefore any browser that visits one of these pages becomes a client to the SAGE2 server.

3.1 Display Clients

One or multiple machines can drive a SRDE. In the case of using multiple browser windows covering a large display, information about each window's relative location to the others is required in order to create the appearance of one seamless large desktop. Content can be opened and viewed in a windowed environment that is similar to a standard desktop operating system. All content is rendered in the browser using HTML elements. Images have built in tags to be viewed natively. Videos are decoded within the server and streamed in YUV form to the display clients (using a ffmpeg-based node module). YUV video blocks are streamed to display clients over high-speed local network (1Gbps and 10Gbps). This is necessary to offer tight video synchronization between display nodes (not afforded by the basic HTML5 video tag). For other common file types, such as PDF, we have built applications using the canvas tag. We have also included a framework for image streams, which can be used to display a remote application inside an image tag. The display clients continuously receive new images to replace the old ones, creating a live video. Custom application can also be loaded on the tiled display wall. Developers can write their own applications using the canvas tag to draw in 2D or utilize WebGL to draw in 3D and leverage hardware rendering. Any existing JavaScript package can be included in an application. These applications run natively in SAGE2, just as they would in a stand-alone browser. This makes application development for SAGE2 straightforward.

3.2 Audio Client

One browser, running on a machine connected to speakers, must visit the audio page hosted by SAGE2 in order to have sound play with videos. Any video that is shown on the display clients is dynamically added to the audio client web page. When a video is playing, the audio client plays the matching audio track. All display clients synchronize their video images at regular intervals to the position of the audio client's video. This ensures smooth seamless audio with consistently matched video. The video frames are decoded by the server, allowing us to have a frame-accurate synchronization. Precise timing is sent to the audio manager for audio/video synchronization. Short latency over local-area network should be negligible and offer a one video-frame of latency (~ 30ms). We are also investigating offering an audio API to applications to provide audio feedback.

3.3 Remote Interaction Clients

The SAGE UI page allows remote users to interact with the SRDE from their laptops. Users can conveniently interact with other applications on their local machine without SAGE2 taking up too much screen real estate. SAGE Pointer allows a user to drag and drop local files from their machine in order to upload them to the server and display them on the SRDE. Users can also choose to capture their computer screen and create a live video stream in order to make their desktop publicly visible on the SRDE (desktop pushing). Finally, SAGE Pointer allows users to create a virtual pointer on the SRDE with a custom color and label so that it can be distinguished from other users' pointers. This will lock the user's local pointer and send move and click events to the web server, which will control the virtual pointer shown on the SRDE. SAGE UI offers all the features of SAGE Pointer, but also depicts a graphical representation of the SRDE and provides a file browser allowing users to

access documents that have already been uploaded to the server. This affords users a second method of interaction as well as providing local visual feedback for their actions.

4. HARDWARE ARCHITECTURE

Collaboration spaces vary greatly, affording access to different display environments. In order to create a scalable resolution framework, SAGE2 needed to be compatible with environments ranging from a single computer with a single monitor, to a large cluster of compute nodes driving numerous tiled displays. Rendering content with the flexibility to display on any SRDE requires one of the two following modalities to be used.

1. *Content-Agnostic Display Clients*: Remote rendering hardware generates the frame buffer for the entire display environment and divides it into individual display viewports that each display client will render. The pixels are streamed to the appropriate display client and are directly rendered on the monitor without any further manipulation of the data.
2. *Content-Aware Display Clients*: Display client nodes calculate their viewport based on their relative position to other displays. Content data is maintained on each node and is used to render the pixels for the appropriate portion of the overall frame buffer.

SAGE2 follows the second modality; the web server distributes the rendering tasks, so each display client draws the portion it is responsible for. The main advantage is to provide a scalable resolution for dynamic media, not requiring a single machine to process an ultra high-resolution frame buffer.

To run SAGE2 on a SRDE, a web server, display environment and audio client are required. Optionally, an Omicron input server [34] can be used for advanced event handling and interaction, such as multi-touch. Any of these components can be hosted on the same or separate machines.

Power requirements are depending on the number and type of applications expected to be run at one time. Images and PDF documents use memory but require little CPU load once opened. Videos require a relatively high-CPU load on the master node (where decoded, using one core) and a bandwidth around 600Mbps for an HD-size video, while little CPU load is needed on the display side since we use WebGL to display YUV block onto textures (moderate GPU needed). The typical setup that we have been recommending is a high-end gaming PC for the master (multi-core CPU) and moderate gaming PC for the display nodes (moderate CPU, midrange GPU, and fait amount of memory to store content such as images and PDF documents). This can be combined into a single high-end gaming PC with multiple 4K monitors attached.

4.1 Components

A diagram illustrating the architecture of SAGE2 is depicted in Figure 2. The web server requires a machine with a network card and access to either the Internet or intranet, depending on the privacy needs of the SRDE. The web server will be used to host a secure website using the HTTPS protocol, which ensures data encryption to maintain fidelity of private collaboration sessions. Since SAGE2 deals with Scalable Resolution Display Environments, there are no pre-defined constraints for display clients. Machines and displays can be added to or removed from the environment, making SAGE2 uniquely capable of providing a scalable system. When the display clients are not driven by the same machine as the web server, the display machines also require a fast network connection to the web server. High-speed networking is becoming more affordable, even at 10Gbps speed. The audio client requires a machine with a sound card and speaker system. If the audio client is not driven by the same machine as the web server, the audio machine also requires access to the network of the web server. A dedicated input server provides multiple types of interaction devices through the Omicron input abstraction library [34]. Supported devices include PQLabs multi-touch overlays, game controllers, motion tracking, tablet devices and Microsoft Kinect.

4.2 Use Cases

For a simple use case, we have SAGE2 installed and running on a single machine driving a single display. A desktop running Windows 7 is used as the web server, display node and audio client. It is connected to a Planar 4K display (3840x2160 pixel resolution) and has a set of standard computer speakers.

For a more advanced use case, we have SAGE2 installed and running on a six-node cluster driving a 6x3 tiled display wall (each node controls three displays). The nodes are running Linux OpenSUSE 12.3 and are interconnected to an internal 10Gb network. An additional head node runs the web server, while each display node becomes a display client. A Mac Mini that is connected to an audio system acts as the audio client. Additionally, a PQLabs 32-touch overlay has been installed on top of the display wall. The overlay is connected to a Windows PC and streams touch events to SAGE2 using the Omicron input server.

5. SOFTWARE AND NETWORKING

SAGE2 operates as a series of web pages hosted by a single web server. Each page visited by a client communicates with the web sever via WebSockets, which allow real-time, two-way message passing and data exchange. This enables all components of SAGE2 to remain synchronized.

5.1 Web Server

The web server is built upon the Node.js platform, a JavaScript-based server framework. This allows both client and server side to be written in the same language, which makes passing and interpreting data objects simple. Along with Node.js, we leverage NPM (www.npmjs.com) modules, which are a collection of external scripts with classes and functions to simplify server development. The SAGE2 server utilizes packaged modules for decompressing zip files, parsing HTML form data, creating a WebSocket server, getting video information from YouTube URLs and getting the resolution of images, videos and PDFs. In addition to packaged modules, Node.js allows custom modules to be written and imported into the server. We have written a number of custom modules for SAGE2, including modules that serve public files to clients and that receive data from the Omicron input server.

In order to gather information about MP4 videos and PDFs, additional software packages were required. FFprobe, part of the FFmpeg multimedia framework, is used to get the resolution and length of an MP4 file. Poppler, a PDF rendering library, is used to get the width and height of each page in a PDF document. All dependencies of the web server can be built and installed on a machine running Windows, Mac OSX, or Linux.

Installation of SAGE2 on a new SRDE is straightforward and quick. All the major components are already multi-platform (Node.js and the Google Chrome browser). A configuration file describes the layout of the wall and its features (display resolution, Omicron input server, etc.). Instances of the Chrome browser open in full-screen mode on the displays and provide a large canvas to support collaborative work at ultra high resolution.

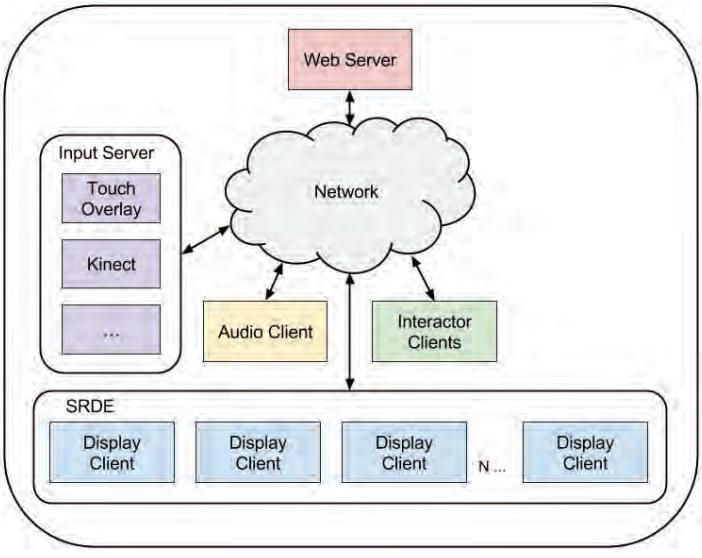

Figure 2. This figure depicts the SAGE2 architecture. The web server drives every component of SAGE2 and all communication goes through network connections.

5.2 Web Clients

SAGE2 has three categories of clients – display clients, audio clients and remote interaction clients. All clients are standard HTML pages that use JavaScript to perform actions and communicate with the web server. The machines that are connected to the SRDE each open a full-screen browser and visit the display client web page. The machine that is connected to the audio system opens a browser and visits the audio client web page. Anybody wishing to interact with the SRDE can open a browser on their laptop and visit one of the remote interaction client web pages. The remote interaction client pages use JavaScript event listeners to capture user events (mouse, keyboard, etc.). The only software required by any client is a modern web browser – we recommend Google Chrome (version 40 or higher) for full feature functionality. Other major browsers (FireFox, Internet Explorer, Safari) are known to work for display clients, and with some limited functionalities (desktop sharing) for user clients.

5.3 WebSockets

The WebSocket protocol was standardized in 2011 and provides a way for the web server to send data to a browser without being solicited by the client. It also maintains open connections, which allows for continual message passing between client and server. In SAGE2, the server acts like an operating system and keeps track of all the applications and their window's size, position and content. When a user connects to a remote interaction client and wishes to modify an item on the SRDE, the user events are passed to the server and the server calculates the new size, position and content of the selected window and broadcasts it to all clients (including the user's remote interaction client). Figure 3 provides a flowchart for when a user wants to open a file that exists on the server. From the user's perspective, when they click the load file button, a file browser pops up showing the files on the server. Once a file is selected, it appears on the SRDE and a corresponding icon is displayed on their SAGE UI. Internally, messages and data are passed through WebSockets and actions are synchronized through the web server.

6. USER INTERACTION

6.1 Window Management

Windows on the SRDE can be added, organized and removed using a remote interaction client. Through a remote interaction web page, users can open a file that is already stored in the server library, drag-and-drop local multimedia files or web URLs and share their computer screen. All of these actions will create a new window on the SRDE and display the desired content. Additionally, the SAGE Pointer can be activated in order to create a personalized pointer that interacts with content on the SRDE. Using a SAGE Pointer, a user can organize content that is displayed on the SRDE by dragging the window to a new position or scaling the window to a new size. Finally, windows can be closed, thereby removing them from the SRDE.

The original SAGE interface features an onscreen media browser for opening stored multimedia directly from the main display.

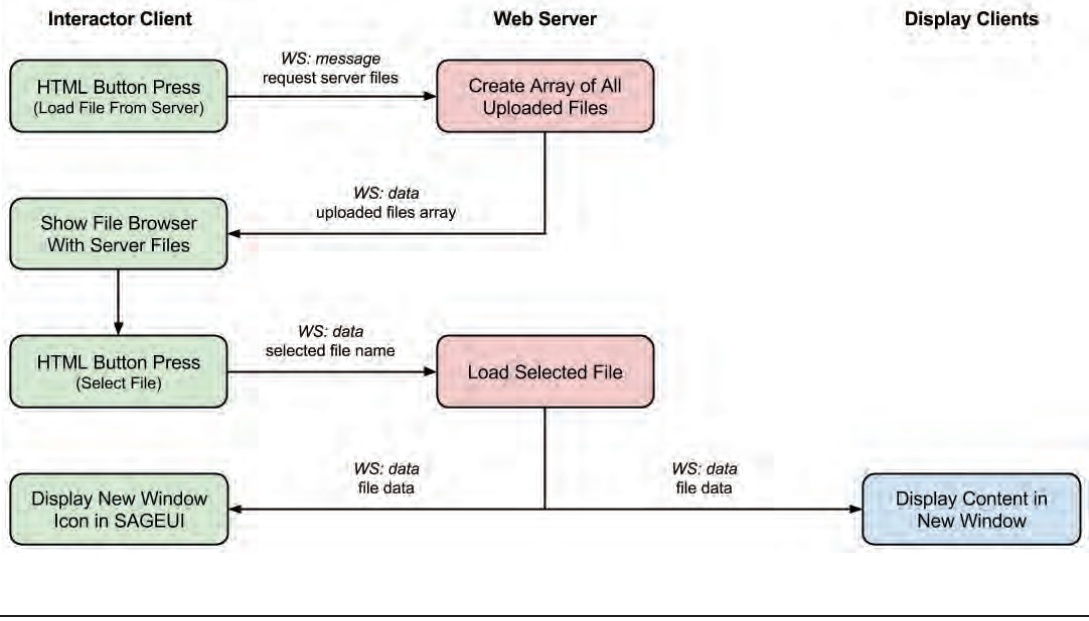

Figure 3. This chart depicts the flow of operation for a sample user interaction – selecting and displaying a file stored on the web server.

The media browser consists of a series of icons representing different media types in a fixed section of the screen. Selecting an icon would open a window containing thumbnails for that media type and display metadata information. Clicking on the thumbnail would then open the media content on the wall [35]. SAGE2 introduces a scalable and multi-user approach to the media browser. Using the SAGE Pointer or touch interface, a personal radial menu can be opened at the pointer or touch position allowing multiple users to browse and open content. This scalable approach allows access to the media browser and other options such as local organization of content anywhere along the SRDE regardless of size or configuration.

6.2 Application Interaction

New to SAGE2, SAGE Pointer events can also be used to directly interact with HTML5 canvas and WebGL applications. By switching interaction modes, SAGE Pointer events are forwarded from the web server to the application. Application developers can handle incoming events to effect changes in their application, giving the appearance of direct content interaction to the users. Each window can toggle modes between window management and application interaction.

6.3 Multi-modal Interaction

In addition to the remote interaction clients, SAGE2 supports multiple input devices through the Omicron input abstraction utility library [34]. Omicron is capable of receiving data from different types of input devices including touch overlays, motion tracking systems, game controllers and speech recognition tools. Figure 4 depicts a user interacting with a touch overlay system. The Omicron input server processes incoming events to stream outgoing events in a standardized format. All outgoing events have fields such as position, orientation and state flags. Pointer events, typically consisting of mouse or touch events, have a position and state (click/down, drag, release/up) while 6DOF devices additionally use orientation and use the flags to store multiple button states. More complex devices like Microsoft Kinect may use additional data fields to store skeletal joint positions. Using the Omicron library, SAGE2 can support application interaction and window management in four major interaction zones for SRDEs: directly at the display using touch gestures, standing near the display using motion tracking or 6DOF devices, seated near the screen using a gyro-mouse or laptop pointer, or indirect control further away from the screen using laptop pointers or the SAGE UI [36].

7. APPLICATION DEVELOPMENT

7.1 Applications in SAGE

Figure 4. Touch interaction through the Omicron library. This photo depicts a user executing a pinch-zoom touch technique to modify the size of the window on the SRDE.

In the original SAGE software, applications could be displayed within SAGE either by streaming frame buffer pixels to an OpenGL texture or by running the application on a user laptop displaying the content through VNC. Using this approach, high-demand applications were developed, such as a video player and PDF viewer. Application windows could be moved and resized within SAGE by multiple, simultaneous users through touch or SAGE Pointers. In order to interact with the content inside the window, simple button widgets could be appended to the bottom of a window. This allowed, for instance, any user to change pages in a PDF or play and pause a video. The successful adoption of SAGE in collaborative environments suggests that this model of content sharing and interaction did facilitate collaborative work by allowing users to rapidly share ideas and juxtapose previously disparate information on a large, shared interface [13].

However, the streaming pixels approach for interactive applications in SAGE presented limitations to collaborative work sessions. In SAGE, content within applications viewed through shared VNC sessions can only be interacted with by its owner. Pixel streaming applications do not handle input beyond custom button widgets. While this was technically possible, the SAGE interaction paradigm was designed around window manipulation (moving or scaling a window) not passing events into a window's application. Further, without an established application development API, it would have been difficult to generate a community of application developers to drive the development of multi-user applications for group collaboration.

7.2 Applications in SAGE2

In contrast to the original SAGE, SAGE2 is web-based and content is directly rendered on the displays. Many research and industrial collaborative projects take advantage of web-based content. Online data portals and visualization tools have proliferated, generating demand for direct interaction with this content in a group setting. Web development has a large community supporting many APIs, while browsers are becoming as powerful as stand-alone applications at rendering high-quality and 3D content. Porting applications to SAGE is difficult and in many cases impossible, whereas SAGE2 is designed to allow web-based applications to run natively, only requiring minimal modification for scalability and synchronization.

We provide an application development API for scalable resolution 2D and 3D rendering with multi-user interaction that is compatible with SAGE2. Additionally, SAGE2 also supports pixel block streaming, providing legacy compatibility with original SAGE.

The SAGE2 framework has been designed to enable rapid application development for a community interested in supporting multi-user collaborative environments. There are three primary ways to develop or port an application to SAGE2.

1. *Direct development for SAGE2*: Using the HTML5 canvas element, applications can be rendered using JavaScript 2D and 3D contexts. Libraries such as D3.js, Snap.svg or Three.js can be utilized for higher-level graphics abstraction.
2. *Pixel streaming applications*: Like SAGE, SAGE2 supports pixel streaming from an external application. SAGE2 additionally sends interaction events to applications, allowing for interactivity.
3. *Porting existing web content*: SAGE2 also provides support for users porting web-based applications, providing handles to connect SAGE2 events to application events.

The SAGE2 API enables application developers to create applications that can run on cluster-based display environments and accept inputs from a variety of devices, without worrying about low-level details. SAGE2 handles distributing and synchronizing an application as it runs across a display cluster. When each node finishes rendering a frame, a signal is sent to the server. Once all active display nodes have finished rendering, the next draw event is fired.

The SAGE2 API breaks application development into four parts: application initialization, rendering, window resizing and event handling. Application initialization creates a SAGE Application object, which internally takes care of viewport size and position as well as maintaining synchronized time across parallel display nodes. The rendering section is the core of the application where developers create the graphics using JavaScript APIs and libraries. Resizing handles how an application should respond when a window is scaled larger or smaller. Finally, the event handling section receives SAGE2 events from the web server. Events are received from pointers, omicron clients or widget elements defined by the programmer (button, text box, etc.). Application developers can utilize this section of the SAGE2 API to enable multi-user collaborative interaction from users with varied input devices.

The SAGE2 Application API provides hooks into applications to initialize, render, resize and interact. We also created an online forum to support a community of developers interested in creating applications for collaborative environments. SAGE2 serves as an excellent platform for HCI research on collaboration, ultra-resolution visualizations and multi-user application development.

7.3 Interactive Applications

We have developed various demonstration applications showcasing the SAGE2 API. One of them is a notepad application using the HTML5 canvas element where SAGE2 allows multiple users to simultaneously place a cursor at a location in the document and type to add text. Previously, using other software for SRDEs, users would be forced to take turns using a mouse and keyboard to interact with content. This application demonstrates the usefulness of SAGE2, not only for organizing content in collaborative sessions but also for collaborative application interaction.

We have also developed a web browser application that allows users to launch a browser within the SAGE2 display environment and interact with HTML rich content. This is an external C++ application built using CEF, the Chromium Embedded Framework, a simple framework for embedding Chromium-based browsers in other applications. It renders a browser window into an off-screen buffer. This buffer is sent to SAGE2 using the pixel-streaming paradigm, while the application receives user interaction from SAGE2. Both interactive applications are depicted in Figure 5.

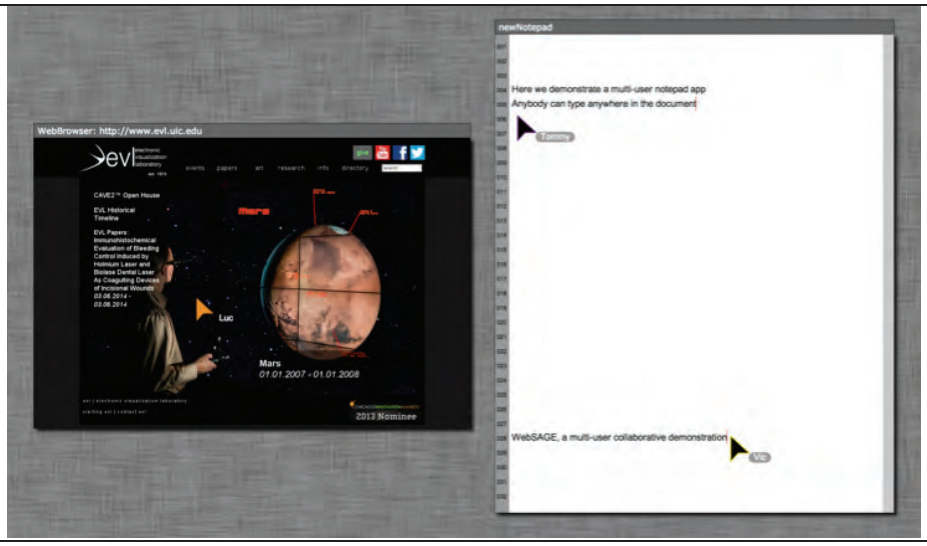

Figure 5. Interactive applications in SAGE2. On the left is a web browser application built with C++ that generates a dynamic image buffer and utilizes pixel streaming to display content in SAGE2. On the right is a multi-user notepad application where numerous users can type at various locations in the document simultaneously.

7.4 High-resolution Image Viewer

Another application ported to SAGE2 is a multi-resolution image viewer for high-resolution imagery based on the *Openseadragon* JavaScript package ("*An open-source, web-based viewer for zoomable images, implemented in pure JavaScript*" http://openseadragon.github.io). Using the VIPS image-processing library (http://www.vips.ecs.soton.ac.uk/), we processed large images as test samples. The image shown in Figure 6 is from the National Center for Microscopy and Imaging Research from UCSD (http://ncmir.ucsd.edu): it is a 324Mpixel microscopy image (17,360 by 18,701 pixels) from a cerebellum of a rat, originally a 724MB TIFF file. Using VIPS, it is processed into a multi-resolution tiled pyramidal format that can be loaded by *Openseadragon*. We integrated the *Openseadragon* toolkit into a SAGE2 application and added event handling to let a user zoom in and navigate through the dataset at interactive speed. The user can resize the SAGE2 window to see more pixels and navigate within the application to get a more detailed view. This application is extremely valuable to many users handling high-resolution imagery in science and creative domains.

Figure 6. High-resolution Image Viewer application on SAGE2 (image from NCMIR/UCSD)

7.5 Network-Intensive Streaming

We support the SAGE pixel-streaming model through the use of an external C++ websocket library (Websocketpp http://www.zaphoyd.com/websocketpp). This helps leveraging high-speed networks as we did in the SAGE project for several years. For instance, we are porting our high-definition video-capture C++ application (used for video conferencing) with initial success. The question that arose is what is the network capability of such a JavaScript-based environment. Websocket uses TCP for streaming data, while the initial handshake is handled through HTTP. We envision high-data streams to be mostly done in local area networks where TCP performs well. Various efforts exist to bring UDP-based protocol to the browser (for instance, RTP based) for wide-area streaming. To test this question, we wrote a Node.js based server written in JavaScript streaming uncompressed RGB images to Chrome browser instances over a 10Gbps network: a server preloads a series of RGB files and streams the pixels to a WebGL application running inside a Chrome browser. The following table shows early networking results, which tells us that a single server can deliver close to a 10Gbps bandwidth given sufficient clients, and that a display node can render around 2 Gbps of pixel data, which is enough to display an uncompressed high-definition video stream. These results are encouraging, and could be optimized in the future for instance to support 4K video streams.

	1 display node	*2 display nodes*	*4 display nodes*	*6 display nodes*
Bandwidth from a NodeJS server	1.8 Gbps	4.3 Gbps	7.0 Gbps	9.3 Gbps

7.6 Mirrored Portals

By design, SAGE2 allows to multiple display clients to connect the same SAGE server. This allows for the construction of mirrored walls or portals, where two or multiple sites show exactly the same content in synchronous fashion. This has been a long-standing request from the SAGE community to support various scenarios such as remote teaching (teacher at one location presenting content to a remote location), creative sessions (in CineGrid scenario over high-speed network, where a creative director directs and instructs a remote team). To demonstrate this capability, we setup our wall display with two SAGE2 sessions connected to the same SAGE2 server, as shown in Figure 7: our 6x3 display wall is configured into two 3x3 sessions. One can notice in the figure that all the content (movies, desktop sharing, images, pointers) are kept in perfect synchronization.

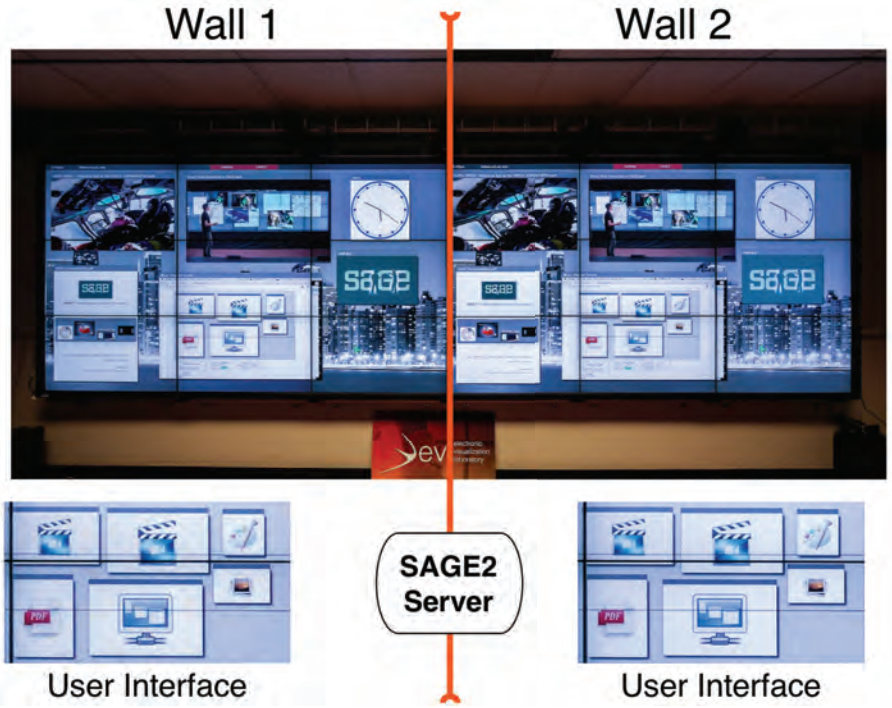

Figure 7. Mirrored Displays, connected on a single server

8. COLLABORATION USE CASES

International groups in academia, research and industry, and in particular CineGrid, have adopted the original SAGE software to support collaboration. We have documented over 95 institutions using SAGE on SRDEs, indicating that the approach embodied by SAGE is effective at enabling collaboration. Like SAGE, SAGE2 provides features that enable group work, such as providing a shared interface for collaborators and making it easy for users to share media content. SAGE2 takes SAGE into the next generation by allowing application interaction and accessing the wealth of web-based content. We present use-case scenarios below to illustrate the power of this tool in collaborative settings.

8.1 Large Lab-wide Meeting

We used SAGE2 during a large group meeting. All group members (n=15) were able to connect to SAGE2 using SAGE UI. The only instruction provided was to install an up-to-date version of the Google Chrome browser. Group members presented progress on projects by streaming their laptop screen to SAGE2 and sharing supporting documents, such as videos or high-resolution images. As members of the group presented, everyone was able to simultaneously move and resize elements in SAGE2 using their SAGE Pointers. This led to streamlined transitions between presenters, since content from upcoming presenters could be prepared while the current presenter interacted with their content. It also allowed group members to respond to the presenter and contribute to the discussion, by bringing up relevant documents or media.

Based on previous use of SAGE during large group meetings, we noted that the barrier to entry in this collaborative setting was significantly reduced. Connection to the network and the Google Chrome browser was all that was required and there was no need to install special applications or modify system settings. This helped the meeting run smoothly, allowing everyone to participate.

8.2 Small Group Project Meeting

We also used SAGE2 for a small group meeting, shown in Figure 8. In this meeting, a team of four researchers worked together on a project in front of an SRDE configured as a daily use workspace running SAGE2. We used this interface to share information relevant to the project such as related publications, documentation and timelines. The multi-user notepad application allowed team members to take notes during conversations. Users

reported that SAGE2 facilitated rapid information sharing and exchange of ideas, indicating that this platform facilitates collaboration in an active work environment.

Figure 8. Small group research session using SAGE2. This photo shows a team of four looking and interacting with various multimedia elements, juxtaposing content in order to extract valuable information.

9. CONCLUSION

Collaboration has always been an essential dimension of work in research, academia and industry. In the next decade, collaboration will be even more essential, as teams of multidisciplinary researchers tackle complex, big-data problems. While SRDEs have been shown to be powerful tools for collaborative work, specialized software is required to effectively leverage these environments. The original SAGE middleware provided such a platform by allowing collaborators to easily share and interact with content on the wall. As a result, SAGE is widely used on SRDEs to enable collaboration.

However, to better address the needs of contemporary collaborative projects, we built the next generation of SAGE: SAGE2, a browser-based SRDE middleware. SAGE2 now stands for Scalable Amplified Group Environment. SAGE2 taps into the SAGE user community and has the potential to expand this community, because of its enhanced support for development and integration of multi-user applications and lower barrier to entry. With the ability to provide support for various input devices, its seamless extension towards multi-user scenarios and ability to leveraging the web infrastructure, SAGE2 is a powerful tool for group work.

10. DISCUSSION / FUTURE WORK

Research on SAGE2 is ongoing and we released an alpha version of the software by November 2014. One of the major features of SAGE that we plan to integrate and expand upon in SAGE2 is support for remote, multi-SRDE collaboration, where users in distant SAGE2 sessions can share content. We will also explore ways to enable simultaneous multi-site interaction with SAGE2 content and applications, which will better enable remote collaborative work. This work is essential, given the increasingly global spread of collaborative research projects. To enhance multi-site collaboration, we will also integrate cloud services, which will free data from being tied to a physical location.

Since data intensive problems are complex, technologies that aspire to support collaborative infrastructures must be flexible enough to aid in solving individual problems of varying domains. We plan to study how SAGE2 can help with three use cases based on real-world scenarios: 1) imitate co-located collaboration with a physically distributed team, 2) collaboration between multiple teams working on different aspects of a related problem, and 3) allowing a single remote user to join a collaborative session.

A major priority of SAGE2 is to continue investigating approaches to support multi-user application development and the integration of existing web-based visualizations and data portals. This research will lead to deeper exploration of novel visualization approaches that take advantage of SRDEs, as well as the support the architecture needed to support multi-user research. We anticipate that the SAGE2 API will evolve to accommodate the integration of a wider variety of applications, growing the user community to include many research fields. SAGE2 will continue as an open source project, designed to be accessible to developers interested in exploring multi-user and collaborative research on SRDEs.

11. ACKNOWLEDGEMENTS

This publication is based on work supported in part by the National Science Foundation (NSF), awards ACI-1339772, OCI-0943559 and CNS-0959053. Any opinions, findings and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the funding agency. SAGE and SAGE2 are trademarks of the University of Illinois Board of Trustees.

12. REFERENCES

[1] C. Johnson, R. Ross, S. Ahern, J. Ahrens, W. Bethel, K. L. Ma, M. Papka, J.V. Rosendale, H. W. Shen, and J. Thomas. 2007. Visualization and knowledge discovery: Report from the DOE/ASCR workshop on visual analysis and data exploration at extreme scale. *Salt Lake City October*.

[2] J. Cummings, T. Finholt, I. Foster, C. Kesselman, and K.A. Lawrence. 2008. Beyond being there: A blueprint for advancing the design, development, and evaluation of virtual organizations.

[3] C. Andrews, A. Endert, and C. North. 2010. Space to think: large high-resolution displays for sensemaking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '10). ACM, New York, NY, USA, 55-64. doi=10.1145/1753326.1753336

[4] R. Ball, and C. North. 2005. Analysis of user behavior on high-resolution tiled displays. In *Proceedings of the 2005 IFIP TC13 international conference on Human-Computer Interaction*(INTERACT'05), Maria Francesca Costabile and Fabio Paternò (Eds.). Springer-Verlag, Berlin, Heidelberg, 350-363. doi=10.1007/11555261_30

[5] R. Ball, and C. North. 2005. Effects of tiled high-resolution display on basic visualization and navigation tasks. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '05). ACM, New York, NY, USA, 1196-1199. doi=10.1145/1056808.1056875

[6] R. Ball, C. North, and D. A. Bowman. 2007. Move to improve: promoting physical navigation to increase user performance with large displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '07). ACM, New York, NY, USA, 191-200. DOI=10.1145/1240624.1240656

[7] X. Bi, and R. Balakrishnan. 2009. Comparing usage of a large high-resolution display to single or dual desktop displays for daily work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '09). ACM, New York, NY, USA, 1005-1014.

[8] M. Czerwinski, G. Smith, T. Regan, B. Meyers, G. Robertson, and G. Starkweather. 2003. Toward characterizing the productivity benefits of very large displays. In *Proceedings of Interact*, Vol. 3, 9-16.

[9] D. S. Tan, J. K. Stefanucci, D. R. Proffitt, and R. Pausch. 2001. The Infocockpit: providing location and place to aid human memory. In *Proceedings of the 2001 workshop on Perceptive user interfaces* (PUI '01). ACM, New York, NY, USA, 1-4. doi=10.1145/971478.971526

[10] D. S. Tan, D. Gergle, P. Scupelli, and R. Pausch. 2003. With similar visual angles, larger displays improve spatial performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '03). ACM, New York, NY, USA, 217-224. doi=10.1145/642611.642650

[11] B. Yost, Y. Haciahmetoglu, and C. North. 2007. Beyond visual acuity: the perceptual scalability of information visualizations for large displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '07). ACM, New York, NY, USA, 101-110. doi=10.1145/1240624.1240639

[12] B. Jeong, L. Renambot, R. Jagodic, R. Singh, J. Aguilera, A. Johnson, and J. Leigh. 2006. High-performance dynamic graphics streaming for scalable adaptive graphics environment. In *Proceedings of the 2006 ACM/IEEE conference on Supercomputing* (SC '06). ACM, New York, NY, USA, Article 108. doi=10.1145/1188455.1188568

[13] B. Jeong, R. Jagodic, L. Renambot, R. Singh, A. Johnson, and J. Leigh. 2005.Scalable Graphics Architecture for High-Resolution Displays. In *Proceedings of IEEE Information Visualization Workshop 2005*. Minneapolis, MN, October 2005.

[14] B. Jeong, J. Leigh, A. Johnson, L. Renambot, M. Brown, R, Jagodic, S. Nam, and H. Hur. 2010. Ultrascale Collaborative Visualization Using a Display-Rich Global Cyberinfrastructure. In *IEEE Computer Graphics and Applications*, 30(3), 71,83. doi=10.1109/MCG.2010.45

[15] J. Leigh, L. Renambot, A. Johnson, B. Jeong, R. Jagodic, N. Schwarz, D. Svistula, R. Singh, J. Aguilera, X. Wang, V. Vishwanath, B. Lopez, D.J. Sandin, T. Peterka, J. Girado, R. Kooima, J. Ge, L. Long, A. Verlo, T.A. DeFanti, M. Brown, D. Cox, R. Patterson, P. Dorn, P. Wefel, S. Levy, J. Talandis, J. Reitzer, T. Prudhomme, T. Coffin, B. Davis, P. Wielinga, B. Stolk, G.B. Koo, J.Y. Kim, S.W. Han, J.W. Kim, B. Corrie, T. Zimmerman, P. Boulanger, and M. Garcia. 2006. The global lambda visualization facility: an international ultra-high-definition wide-area visualization collaboratory. In *Future Generation Computer Systems*, *22*(8), 964-971.

[16] G. Humphreys and P. Hanrahan. 1999. A distributed graphics system for large tiled displays. In *Proceedings of the conference on Visualization '99: celebrating ten years* (VIS '99). IEEE Computer Society Press, Los Alamitos, CA, USA, 215-223.

[17] G. Humphreys, M. Eldridge, I. Buck, G. Stoll, M. Everett, and P. Hanrahan. 2001. WireGL: a scalable graphics system for clusters. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '01). ACM, New York, NY, USA, 129-140. doi=10.1145/383259.383272

[18] G. Humphreys, M. Houston, R. Ng, R. Frank, S. Ahern, P. D. Kirchner, and J. T. Klosowski. 2002. Chromium: a stream-processing framework for interactive rendering on clusters. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '02). ACM, New York, NY, USA, 693-702. doi=10.1145/566570.566639

[19] S. Eilemann, M. Makhinya, R. Pajarola. 2009. Equalizer: A Scalable Parallel Rendering Framework. In *IEEE Transactions on Visualization and Computer Graphics,* 15(3), 436,452. doi=10.1109/TVCG.2008.104

[20] A. Febretti, V. A. Mateevitsi, D. Chau, A. Nishimoto, B. McGinnis, J. Misterka, A. Johnson, and J. Leigh. 2011. The OmegaDesk: towards a hybrid 2D and 3D work desk. In *Proceedings of the 7th international conference on Advances in visual computing - Volume Part II* (ISVC'11), George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, and Song Wang (Eds.), Vol. Part II. Springer-Verlag, Berlin, Heidelberg, 13-23.

[21] A. Febrett, A. Nishimoto, T. Thigpen, J. Talandis, L. Long, J. D. Pirtle, T. Peterka, A. Verlo, M. Brown, D. Plepys, D. Sandin, L. Renambot, A. Johnson, and J. Leigh. 2013. CAVE2: a hybrid reality environment for immersive simulation and information analysis. In *Proceedings of* The Engineering Reality of Virtual Reality 2013 (SPIE 8649). doi=10.1117/12.2005484

[22] K. U. Doerr, and F. Kuester. 2011. CGLX: a scalable, high-performance visualization framework for networked display environments. In *IEEE Transactions on Visualization and Computer Graphics*. 17(3), 320-332.

[23] K. Ponto, K. Doerr, T. Wypych, J. Kooker, and F. Kuester. 2011. CGLXTouch: A multi-user multi-touch approach for ultra-high-resolution collaborative workspaces. In *Future Generation Computer Systems*. 27(6), 649-656.

[24] Xdmx, Wikipedia. 2006. Retrieved February 13, 2014 from http://en.wikipedia.org/wiki/Xdmx

[25] Distributed Multihead, X. Project .2004. Retrieved February 5, 2014 from http://dmx.sourceforge.net

[26] M. Rittenbruch. 2013. CubIT: large-scale multi-user presentation and collaboration. In*Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces* (ITS '13). ACM, New York, NY, USA, 441-444. doi=10.1145/2512349.2514923

[27] R. MacKenzie, K. Hawkey, K. S. Booth, Z. Liu, P. Perswain, and S. S. Dhillon. 2012. LACOME: a multi-user collaboration system for shared large displays. In*Proceedings of the ACM 2012 conference on Computer*

Supported Cooperative Work Companion(CSCW '12). ACM, New York, NY, USA, 267-268. doi=10.1145/2141512.2141596

[28] Z. Liu. 2004. *Lacome: a cross-platform multi-user collaboration system for a shared large display.* Master's thesis. The University of Electronic Science and Technology of China.

[29] G. P. Johnson, G. D. Abram, B. Westing, P. Navr'til, and K. Gaither. 2012. Displaycluster: An interactive visualization environment for tiled displays. In *Cluster Computing (CLUSTER), 2012 IEEE International Conference on* (pp. 239-247). IEEE.

[30] K. Curran. 2002. A web-based collaboration teaching environment. In *IEEE Multimedia*. 9(3), 72-76.

[31] M. Pappas, V. Karabatsou, D. Mavrikios, and G. Chryssolouris. 2006. Development of a web-based collaboration platform for manufacturing product and process design evaluation using virtual reality techniques. In *International Journal of Computer Integrated Manufacturing*. 19(8), 805-814.

[32] M. Kuechler, and A. M. Kunz. 2010. Collaboard: a remote collaboration groupware device featuring an embodiment-enriched shared workspace. In *Proceedings of the 16th ACM international conference on Supporting group work* (GROUP '10). ACM, New York, NY, USA, 211-214. doi=10.1145/1880071.1880107

[33] D. Lee, S. A. Munson, B. Congleton, M. W. Newman, M. S. Ackerman, E. C. Hofer, and T. A. Finholt. 2009. Montage: a platform for physically navigating multiple pages of web content. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '09). ACM, New York, NY, USA, 4477-4482. doi=10.1145/1520340.1520686

[34] Omicron. Retrieved February 13, 2014 from http://github.com/uic-evl/omicron

[35] R. Jagodic, L. Renambot, A. Johnson, J. Leigh, and S. Deshpande. 2011. Enabling multi-user interaction in large high-resolution distributed environments. In *Future Generation Computer Systems*. *27*(7), 914-923.

[36] J. Leigh, A. Johnson, L. Renambot, T. Peterka, B. Jeong, D. Sandin, J. Talandis, R. Jagodic, S. Nam, H. Hur, and Y. Sun. 2013. Scalable resolution display walls. In *Proceedings of the IEEE*. 101(1), 115-129.