# DESIGN AND IMPLEMENTATION OF SAGE DISPLAY CONTROLLER

BY

Javid M. Alimohideen Meerasa

M.S., University of Illinois at Chicago, 2003

PROJECT

Submitted as partial fulfillment of the requirements for the degree of the Master of Science in Computer Science in the Graduate College of the University of Illinois at Chicago, 2007

Chicago, Illinois

## Table of Contents

# Design & Implementation of SAGE Display Controller

## 1 Introduction

SAGE Display Controller, a web-based application designed and developed to access and control the SAGE environment. The application allows a novice user to interact with the environment from standard web browsers such as Firefox, Safari or Internet explorer. The user interface represents the tiled display along with the application windows currently running on the SAGE environment. Interactivity is achieved by allowing multiple users to move and resize the application windows simultaneously.

## 2 About SAGE

SAGE (Scalable Adaptive Graphics Environment) is a graphics streaming architecture for supporting collaborative scientific visualization environments with potentially hundreds of mega-pixels of contiguous display resolution. In collaborative scientific visualization, it is crucial to share high-resolution imagery as well as high-definition video among group of collaborators at local or remote sites.

The network-centered architecture of SAGE allows collaborators to simultaneously run various applications (such as 3D rendering, remote desktop, video streams and 2D maps) on local or remote clusters, and share them by streaming the pixels of each application over ultra-high-speed networks to large tiled displays.

SAGE's streaming architecture is designed so that the output of arbitrary M by N pixel rendering cluster nodes can be streamed to X by Y pixel display screens, allowing user-definable layouts on the display (Figure 1a). The dynamic pixel routing capability of SAGE lets users freely move and resize each application's imagery over tiled displays in run-time, tightly synchronizing the multiple visualization streams to form a single stream.

**Figure 1a: SAGE environment running on LamdaVision, a 100 mega-pixel tiled display. The figure also shows users sharing their desktop using the "Desktop Sharing" feature**

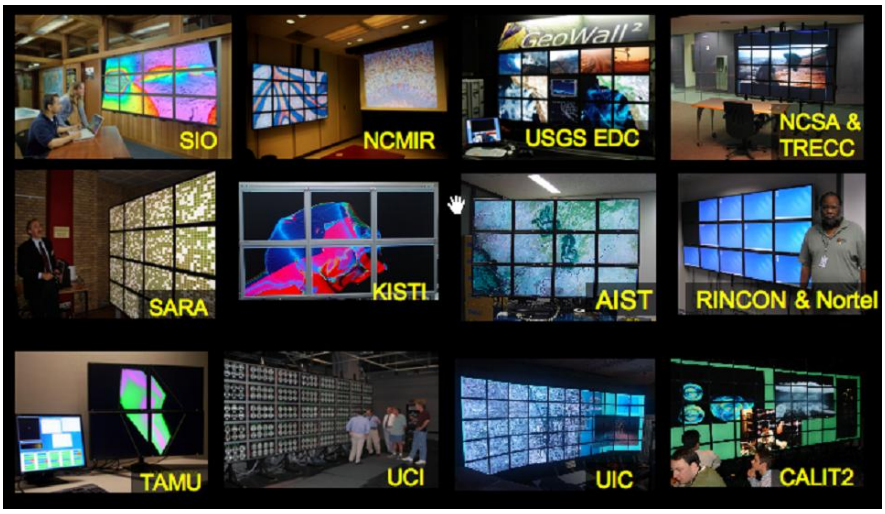


**Figure 1b: Expanding community of SAGE users**

## 3 Related Work

SAGE UI, a python based stand-alone application (Figure 2) is available to access and control the SAGE environment. The application provides all essential features like application window move, resize and close operations. The application also allows the user to share their desktop using a VNC server installed on their laptop, tablet or desktop PC. However, in a collaborative environment like SAGE (see Figure 1a), it is very possible that several users would like to use the display with ease and not being able to install any special application or software. And moreover,

the expanding community of users (Figure 1b) who have adopted SAGE for their everyday work includes scientists or researchers from several domains such as bioscience, geosciences and more. So, being able to make the environment easily accessible and not cumbersome to use is an important design factor. The SAGE Display Controller addresses the mentioned issues by being able to provide full access/control of the environment from a standard web browser such as FireFox, Safari or Internet Explorer, thus requiring minimal or no knowledge of underlying working architecture. Figure 2a & 2b, shows the screenshots of the stand-alone python user interface application and the browser based SAGE Display Controller application.

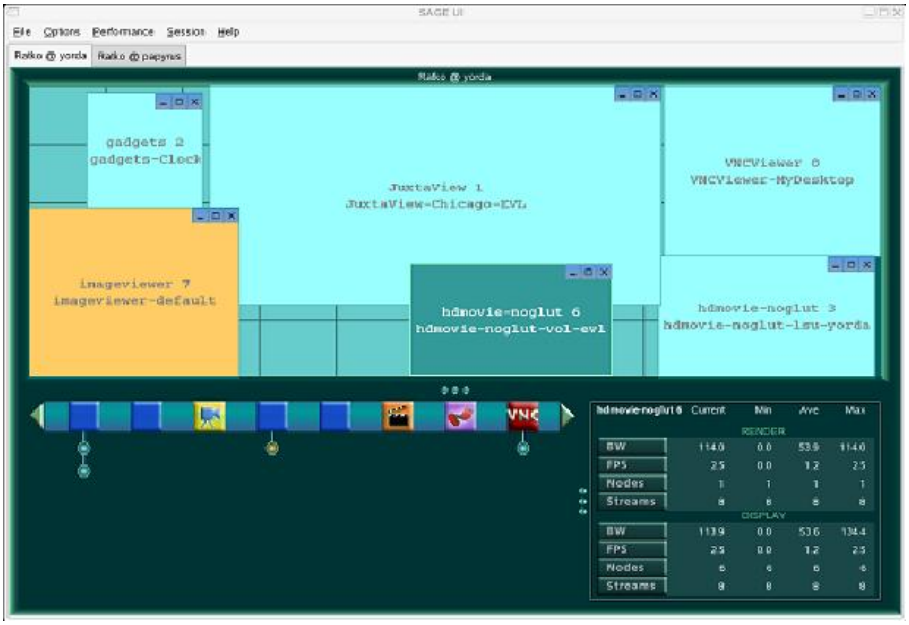


**Figure 2a: Stand-alone python based SAGE user interface application**

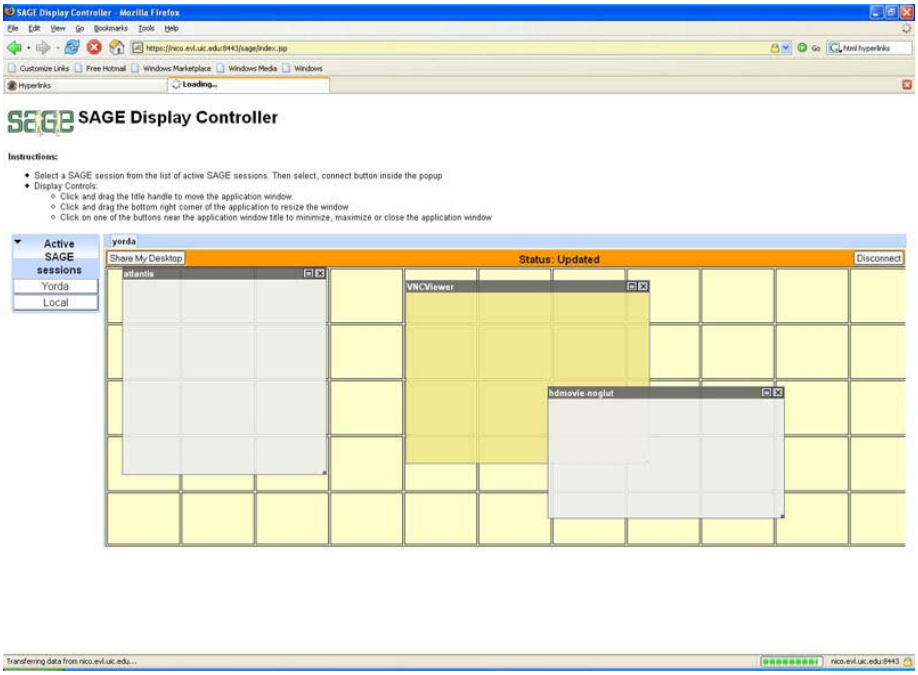


**Figure 2b: Web based SAGE Display Controller**

# 4 Features

The SAGE Display Controller is a browser-based application that uses the latest web technologies like AJAX (Asynchronous JavaScript) and DHTML (Dynamic HTML) to provide the most essential features needed to access/control the SAGE environment.

Some of the features are listed below:

- Application window operations – Move, resize and close the application windows on the SAGE display.
- Window Layout – Move multiple application windows on the display simultaneously.
- Multiple SAGE sessions – Connect to multiple active SAGE sessions simultaneously and control the sessions.
- Secured SAGE sessions – Password protected SAGE sessions permits limited access to available SAGE sessions.
- Desktop sharing – Share the desktop with one single mouse click.
  *Note: requires a VNC server installed on the client's machine.*

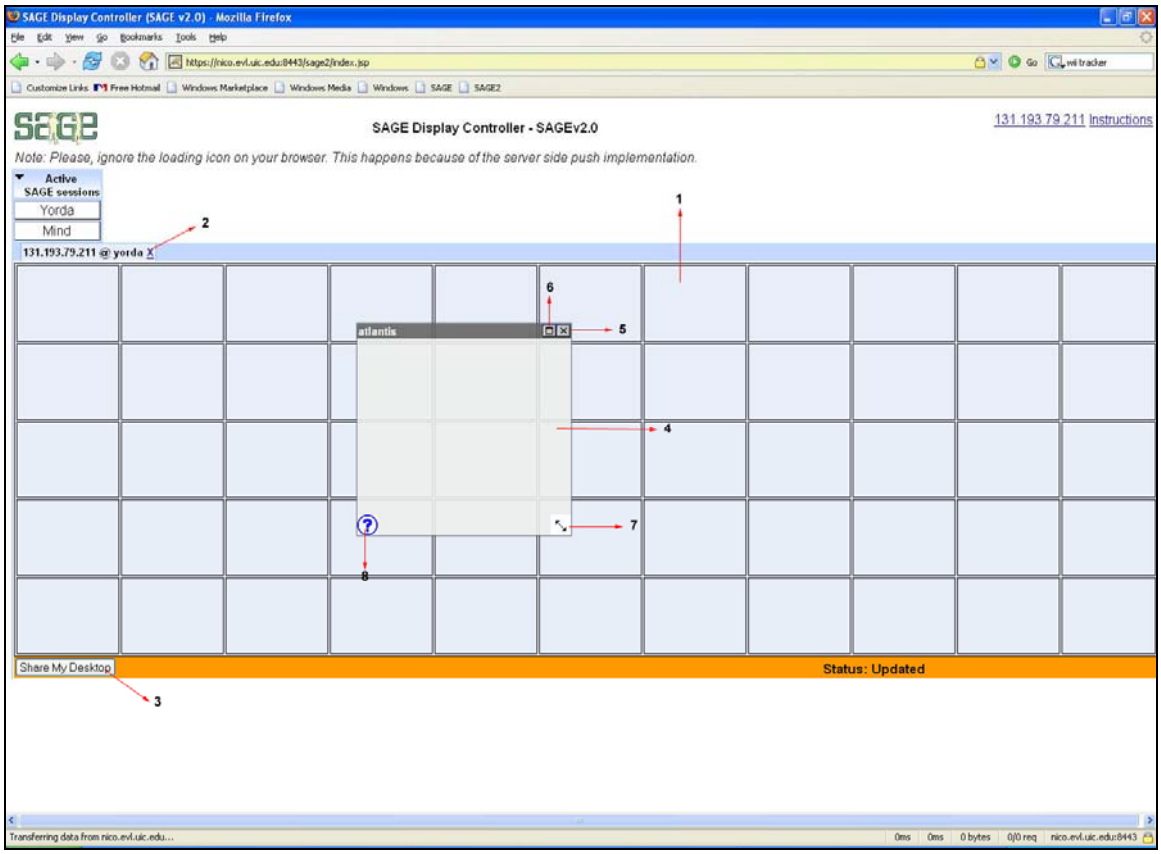


**Figure 3: SAGE Display Controller application to connected to SAGE session "Yorda"**

| **Legend** |
|---|
| 1. Tiled display. Figure3. Shows 11 x 5 tile configuration<br>2. Display tab with SAGE session name. Multiple tabs will be shown when connected to multiple sessions. Click on the "X" mark to disconnect from an active session.<br>3. Share desktop button. Uses VNC to share the client desktop.<br>4. Application window running on a SAGE environment.<br>5. Button to close an application window<br>6. Button to maximize an application window.<br>7. Button to resize an application window.<br>8. Information icon to display update messages. |

The SAGE Display Controller provides the user interface controls conforming very close to a standard desktop metaphor. The application window has user

interface controls very similar to a desktop application with buttons for minimize, maximize and close operations. The title bar handle allows the user to drag n drop window anywhere on the display and by dragging bottom right corner of the window, users can resize the application window. It also allows multiple window selection and move operation.

Access to multiple SAGE sessions is done by providing individual notebook like tabs for every SAGE sessions connected. Users can move through different sessions with simple mouse click on the display tabs. Active sessions can be disconnected by using the small "X" button on every individual display tabs.

The desktop sharing is an important feature of the application. With one simple mouse click users are able to share their desktop on the SAGE display. The application displays a popup dialog box with information like VNC server address, port number and an input field to enter VNC password. Note: To perform desktop sharing, a VNC server is required to be installed. Please, visit the documentation page (http://www.evl.uic.edu/cavern/sage/sagewebui/documentation.php) or section 11.2, to learn more about on instructions to install VNC.

The SAGE Display Controller's multi-browser feature (browser independent) makes the application very scalable and easy to use. The application takes care of all the browser compatibility issues and hence providing the same look and feel interfaces on all the web browsers.

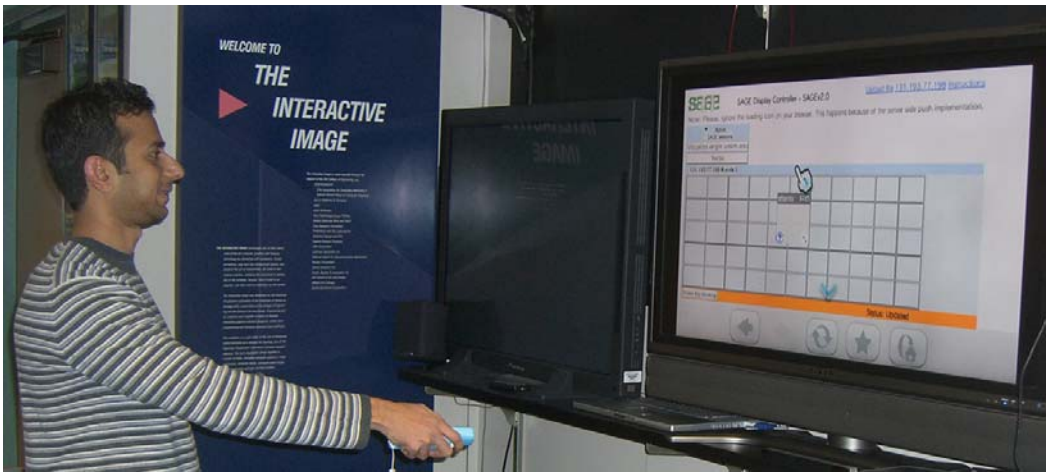


**Figure 4: SAGE Display Controller working on a Nintendo Wii (video game console)**

# 5 User Manual

This section illustrates the step-by-step instructions to connect and interact with an active SAGE session.

## 5.1 Welcome screen

The welcome page of the application lists all the active SAGE sessions along with links to instructions, user profile name or client IP address.

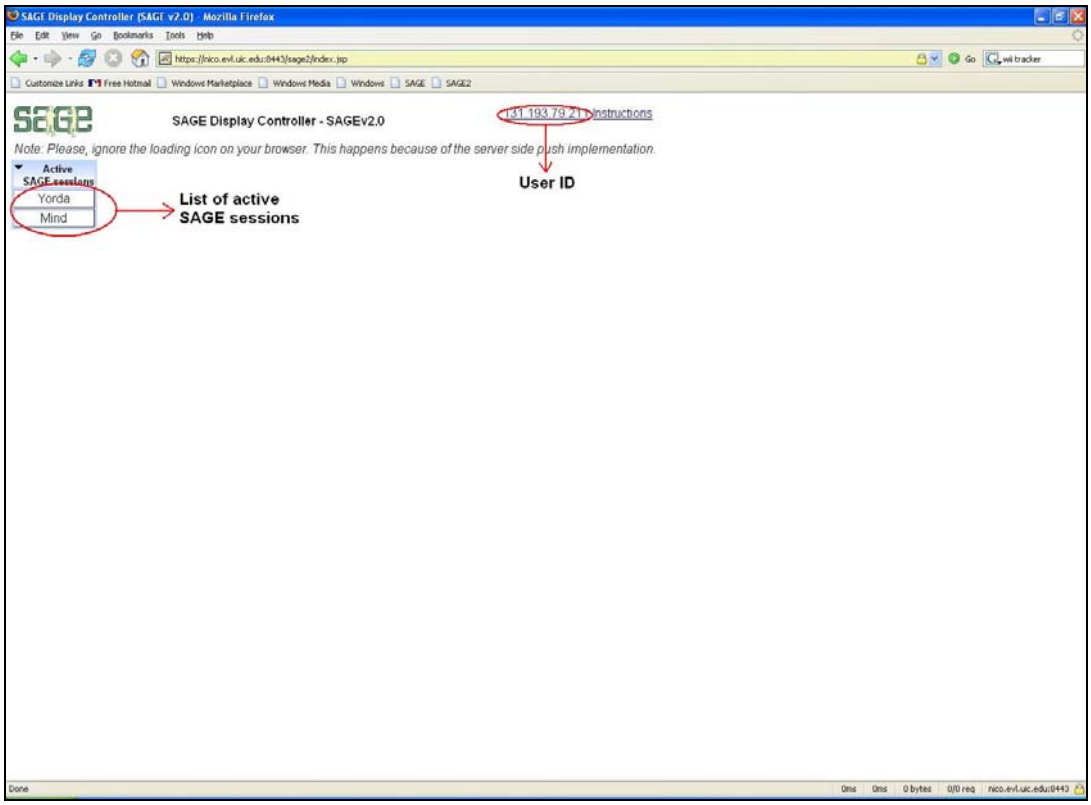


**Figure 5: Welcome screen**

## 5.2 Connect screen

Users can connect to an active SAGE session by performing the following steps:

- Click on the active SAGE session name
- Enter the secret session password.
- Click Connect button.

If password is authenticated the application displays the SAGE display along with the application windows else displays an invalid password error message.

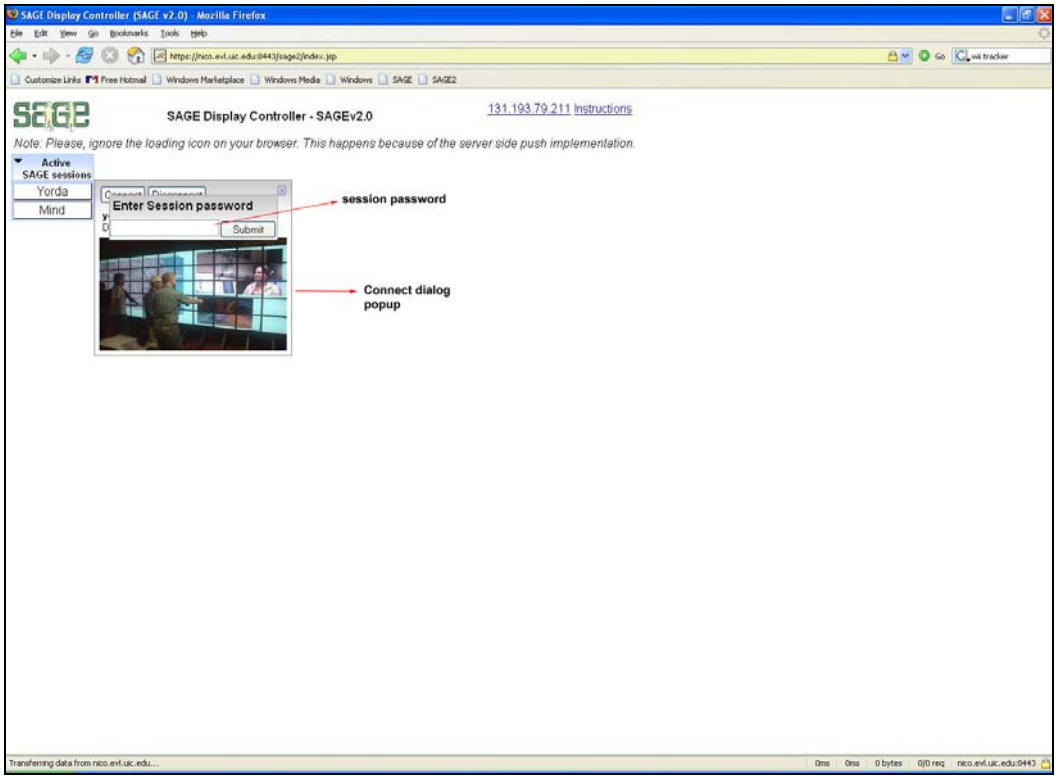


**Figure 6: Connect screen**

## 5.3 SAGE Display screen

The figure below shows the SAGE Display Controller screen after establishing an authenticated connection with the SAGE environment.

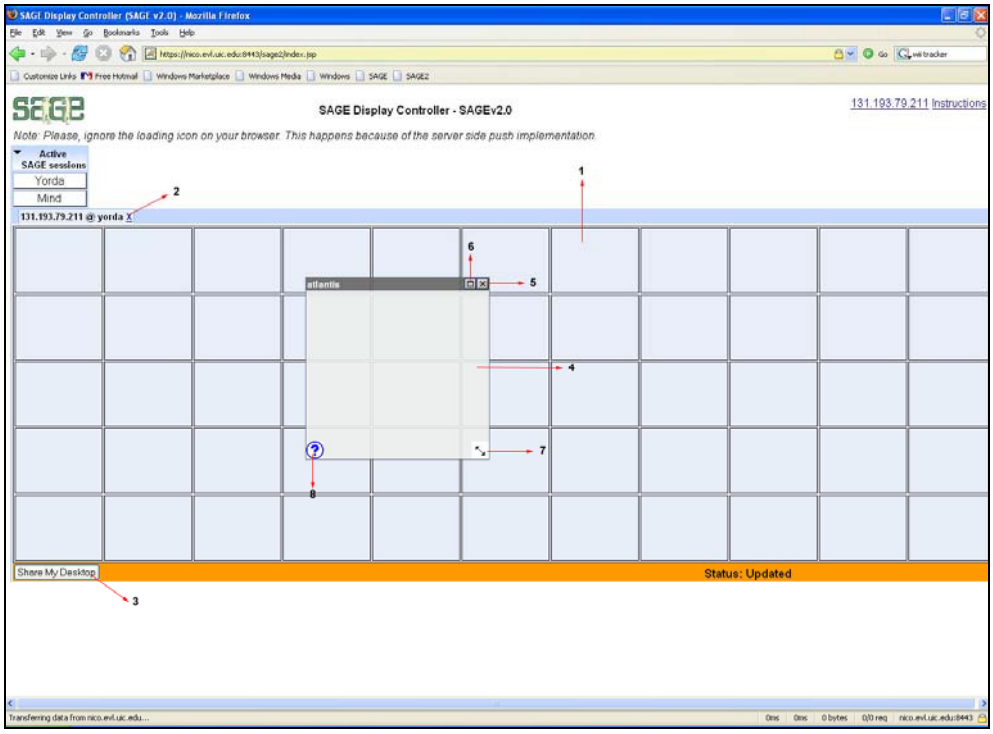


**Figure 7: Display screen**

| **Legend** |
|---|
| 1. Tiled display. Fig. Shows 11 x 5 tile configuration<br>2. Display tab with SAGE session name. Multiple tabs will be shown when connected to multiple sessions. Click on the "X" mark to disconnect from an active session.<br>3. Share desktop button. Uses VNC to share the client desktop.<br>4. Application window running on a SAGE environment. |

5. Button to close an application window
6. Button to maximize an application window.
7. Button to resize an application window.
8. Information icon to display update messages.

## 5.4 Share the desktop

Users can share their desktop on the SAGE environment by following the steps below:

1. Click on the “Share My Desktop” button located on the lower left corner of the tiled display widget.
2. On the popup dialog box, enter the VNC session password and verify the display information populated automatically in the dialog box.
3. Click “Ok” to launch the session.

*Note: If the launch was successful, you will see a new application window with title, as “VNC” appear on the lower left corner of the display else an error message is displayed.*

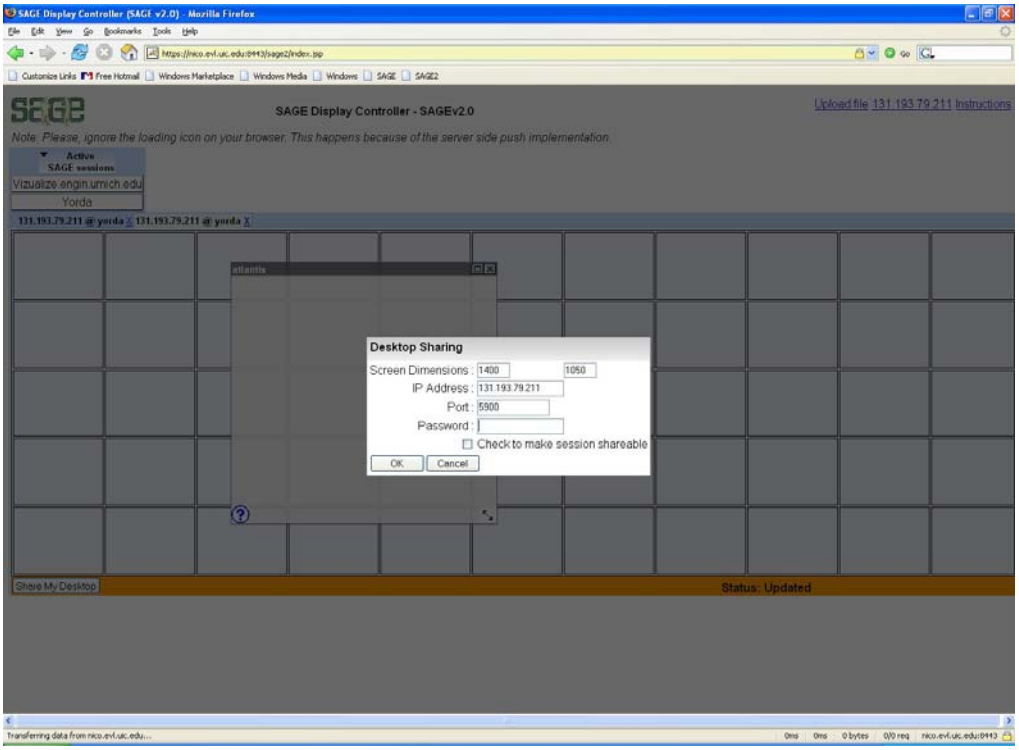


**Figure 8: Desktop sharing popup dialog**

# 6 Software Tools

This section lists all the third party toolkit and dependency software packages that were used for the application development. JAVA and Python were the primary language used to develop the application. Table 1, shows the tools used along with their use.

- Google Web Toolkit (GWT): A JAVA toolkit for developing dynamic, rich and interactive web applications. The unique feature of this toolkit is to translate all the JAVA code into JavaScript code, which allows the application to be launched using standard web browser. (http://code.google.com/webtoolkit/)
- Apache XML-RPC: Is a Java implementation of XML-RPC, a popular protocol that uses XML over HTTP to implement remote procedure calls.
- JSON Library: Is a java library for transforming beans, maps, collections, java arrays and XML to JSON and back again to beans and DynaBeans. JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language.
- Apache Tomcat: A web container developed by the Apache Software Foundation (ASF). Tomcat implements the servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, providing an environment for Java code to run in cooperation with a web server.

| Tools | Version | Purpose |
|---|---|---|
| Google Web Toolkit (GWT) | 1.2 | User-interface controls & Asynchronous Javascript calls. |
| Apache XML-RPC | 1.2-b1 | Communicate with the SAGE proxy server |
| JSON Library | 0.9 | Transform Java type data structures to JSON format. Primary data exchange format |
| Tomcat Server | 5.1 | Server to host the web application module and JAVA servlets. |

**Table 1: Software tools**

# 7 Architecture

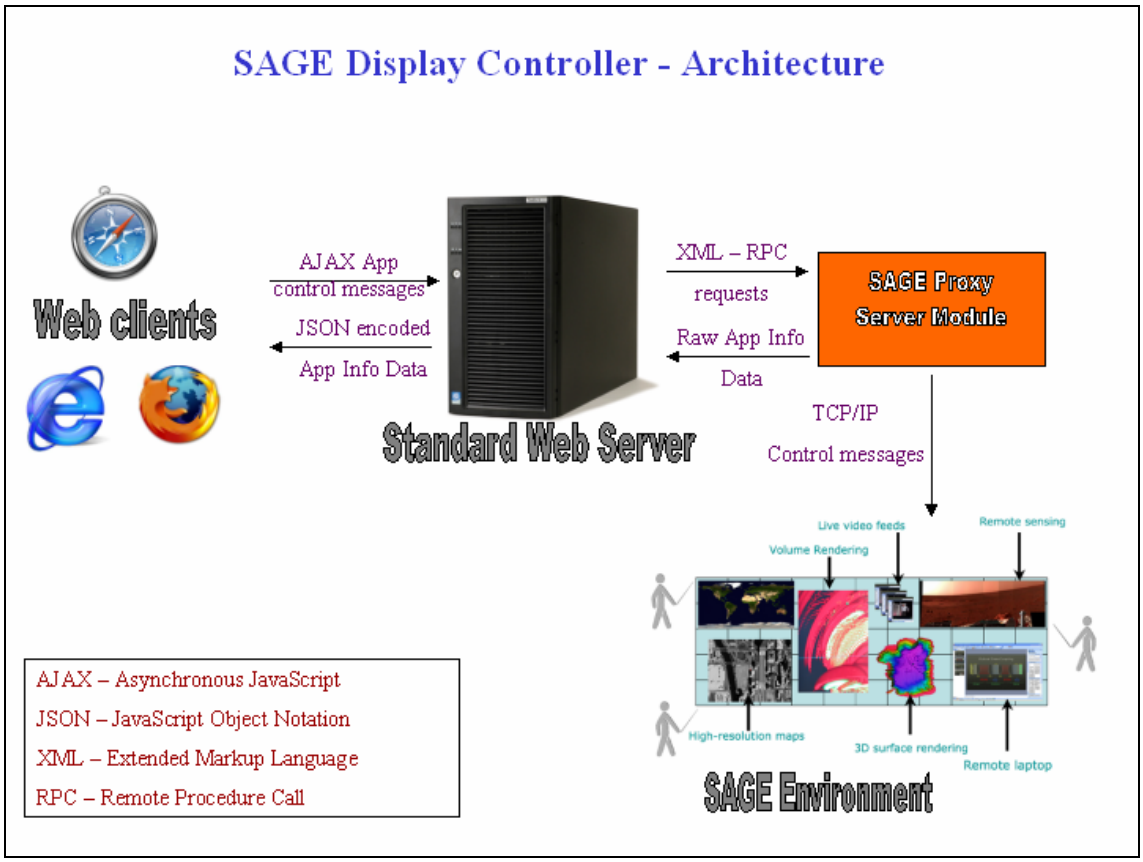


**Figure 9: SAGE Display Controller application architecture**

The application architecture mainly comprises of three important modules:

- JavaScript client module
- JAVA servlet module
- XML-RPC proxy server module

## 7.1 JavaScript client module

The JavaScript client module is responsible for all the user interface components of the application and the client-side validation rules. The GWT library translates all the widgets to their corresponding HTML, DHTML elements and also generates the necessary JavaScript code to manipulate the DOM (Document Object Model). Using the AJAX functionality, the client makes asynchronous calls to the server to

obtain updated information from the server based on user's interaction with the environment. The AJAX feature lets the widgets update themselves with new information without requiring reloading of the pages, thus giving a feel of desktop applications.

## 7.2 Java servlet module

The servlet module handles all the requests sent by the client and acts as a proxy between the native SAGE application and the clients. The communication between the server and the native SAGE application is done using the python based SAGE proxy server module described in section 6.3. Using a servlet helps avoid the same server JavaScript security violation that does not allow the browser to send requests to any other server/domain from where they were hosted originally.

## 7.3 Proxy server module

The proxy server module exposes the interface needed by the servlet module to communicate with the native C++ SAGE application as XML-RPC service. The presence of a proxy server removes the restriction of the servlet module to be installed on the same machine where SAGE application is running. In real world, the SAGE application could be launched in a remote cluster and the server hosting the client module and servlets can be installed on any web server and with a proxy server in between communication between the web client and the SAGE application is handled seamlessly.

## 8 Application flow pipeline

The SAGE Display Controller application is hosted on a standard web server, which could be accessed using the server URL from the web browser. All the DHTML based user interface components are downloaded automatically on the client machine with the necessary JavaScript to manipulate and control the user interface components. When the client performs an action (e.g. move a window), an Asynchronous JavaScript (AJAX) control message is sent to the server along with necessary data needed to control the SAGE environment. The Servlet module accepts the client's request and makes an XML-RPC request to the proxy server module to communicate with the native SAGE application. Upon successful of the XML-RPC request, the Servlet returns the browser's call with the updated data structures. Since, all the communications are done asynchronously, the client or the server does not block on any request, which allows the user interface components to be updated automatically and thereby giving a feel of multi-threaded application. This kind of asynchronous communication lets the application to behave like a standard desktop application and avoiding page refreshes typically encountered by standard HTML based web applications. Also, using the latest JSON (JavaScript Object Notation) format for data exchange between the client and the servlet improves the performance of the application, since JSON uses minimal encoding of control messages and data structures compared to commonly used data exchange format like XML.

## 9 Software Download

The latest version of the SAGE Display Controller application can be downloaded using the URL http://www.evl.uic.edu/cavern/sage/sagewebui/download.php. A version compatible with the new SAGE2 is also made available.

## 10 Software/Hardware Requirements

- Apache Tomcat Servlet Container (Version 4 & above)
- VNC client/server application. (Required for desktop sharing feature only)

You can download a copy of the VNC client/server from the following links specific to your platform.

- Windows: client and server: http://www.tightvnc.com/
- Mac: server: OSXvnc http://www.macupdate.com/info.php/id/11283
- Mac client: Chicken of the VNC http://www.macupdate.com/info.php/id/9517
- Linux: vncviewer/vncserver package or tightvnc package

Here is the link to common FAQ on Apache Tomcat. Links include how to install Tomcat as service, enable SSL (Security encryption)

## 11 Network Requirements

The SAGE Display Controller uses network intensive technologies such as AJAX and XML-RPC for its remote communications with server. Any client, with a minimum of a broadband or DSL Internet connection would be able to achieve a high performance rate.

*Note: This application has never been tested with a dial-up connection.*

## 12 Installation instructions

*Note: You need a working Apache Tomcat installed before you proceed with the installation.*

- Download the SAGEUI.WAR file from the downloads page.
- Copy the downloaded file into the webapps directory found under the Apache Tomcat installation root directory

Based on your tomcat installation configuration, sometimes you might have to re-start your tomcat server for the application to work.

**Example:**

APACHE_ROOT_DIRECTORY = /home/user/apache-tomcat

Copy the SAGEUI.WAR file into /home/user/apache-tomcat/webapps directory.

Restart the apache tomcat server with the command 'catalina.sh restart' found in the /home/user/apache-tomcat/bin directory. If the tomcat has been installed on a machine named xyz.uic.edu and running on port 8443, you should be able to access the application using the URL http://xyz.uic.edu:8443/sageui/index.jsp

## 12.1 Additional instructions

Once the application is installed, you have to run the SageProxyServer script every time you start the SAGE application.

- Download the SageProxyServer.zip file from http://www.evl.uic.edu/cavern/sage/sagewebui/download.php page. Please, make sure that you download the compatible proxy server.
- Extract the zip file contents to your preferred working directory.
- Run the proxy server script with the command
    - *'python SageProxySecure.py host port password'*
      where host - host name of the machine running SAGE application, port - SAGE application port number, password - SAGE session password that you would like to use.

*Note: The SAGE application has to be running before you start the SageProxyServer script.*

## 12.2 Desktop sharing

Clients Only:

To share your desktop you have to install a VNC server on your personal device (e.g., laptop, desktop). Currently, the application does not support NAT addresses, so if you were behind a router you would have to enable port forwarding in your router to share your desktop. Please, refer to the user manual section on how to start a VNC session.

You can download a copy of the VNC client/server from the following links specific to your platform.

- Windows: client and server: http://www.tightvnc.com/
- Mac: server: OSXvnc http://www.macupdate.com/info.php/id/11283
- Mac client: Chicken of the VNC http://www.macupdate.com/info.php/id/9517
- Linux: vncviewer/vncserver package or tightvnc package.

*Note: Configure your firewall to allow port 5900 (default VNC port). If the VNC server is running on some other different port, please allow exceptions for that port number on your firewall.*

# 13 Code structure

The code base is grouped into different packages based on the Java and GWT recommendations. The Figures 10 & 11 below shows the different packages and the dependencies between them.

## 13.1 packages

- evl.sage.client.ui : The Google Web Toolkit was extended to built custom user interface widgets specific to the SAGE Display Controller application. This package contains all the widgets and controls. It is always a good practice to separate the user interface components in a package different from the packages that use the components.
- evl.sage.client: Package contains the classes and data structures used to build the application. The remote interfaces needed for asynchronous communication with the server are defined in this package. Classes in this package accesses or calls classes defined in the evl.sage.client.ui and evl.sage.common packages.

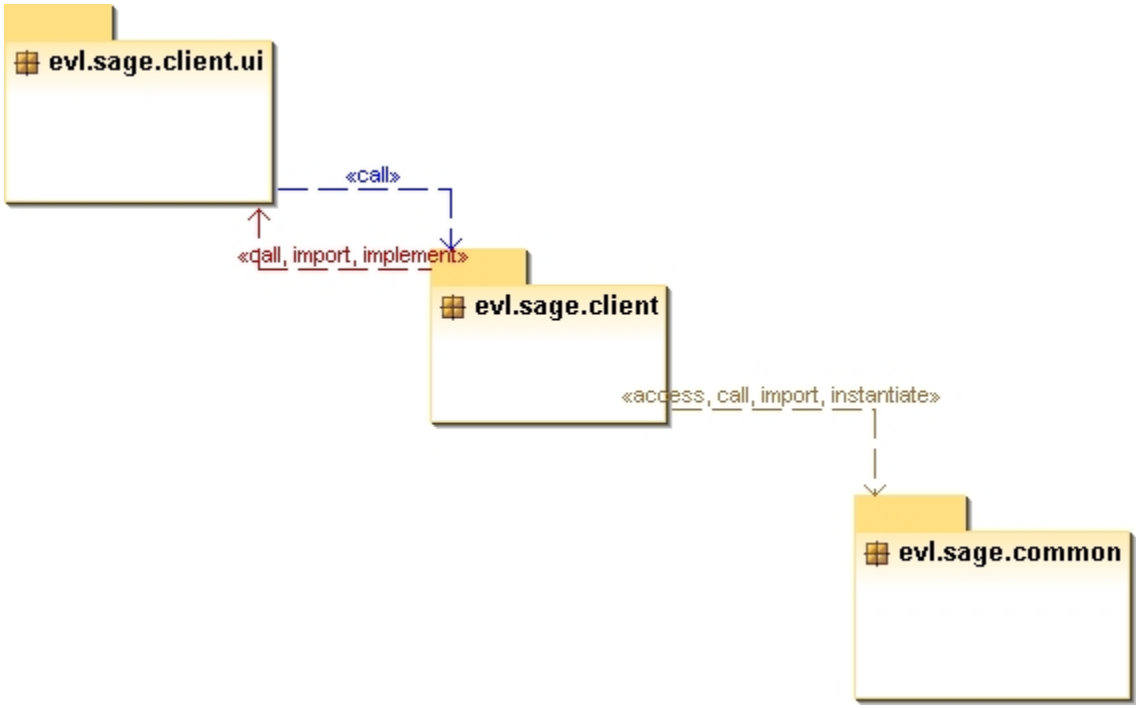


**Figure 10: package dependency diagram of client package.**

- evl.sage.common : package contains classes that define the custom data structures used in the application. Classes contained in this package are accessed by both the evl.sage.client and evl.sage.server packages.
- evl.sage.server: package holds the entire server related classes accessed by the clients. Classes in the package access the evl.sage.common package for creating application specific data structures to be sent back to the clients.

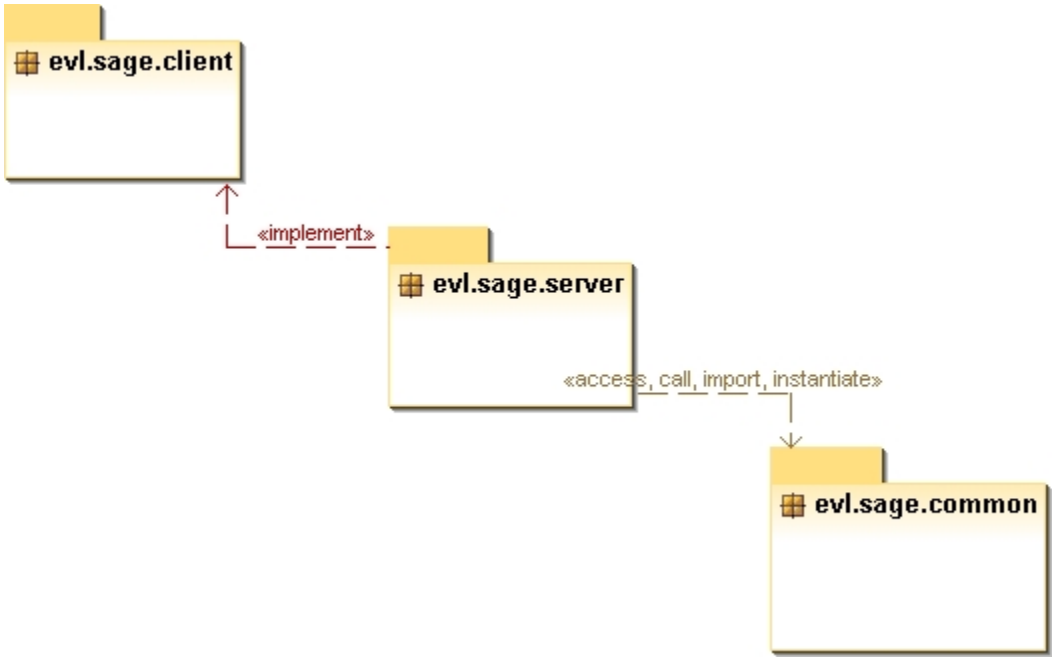


**Figure 11: package dependency diagram of package evl.sage.server**

## 13.2 class diagrams

This section lists all the relationship between the classes in individual package and the functionality of the classes.

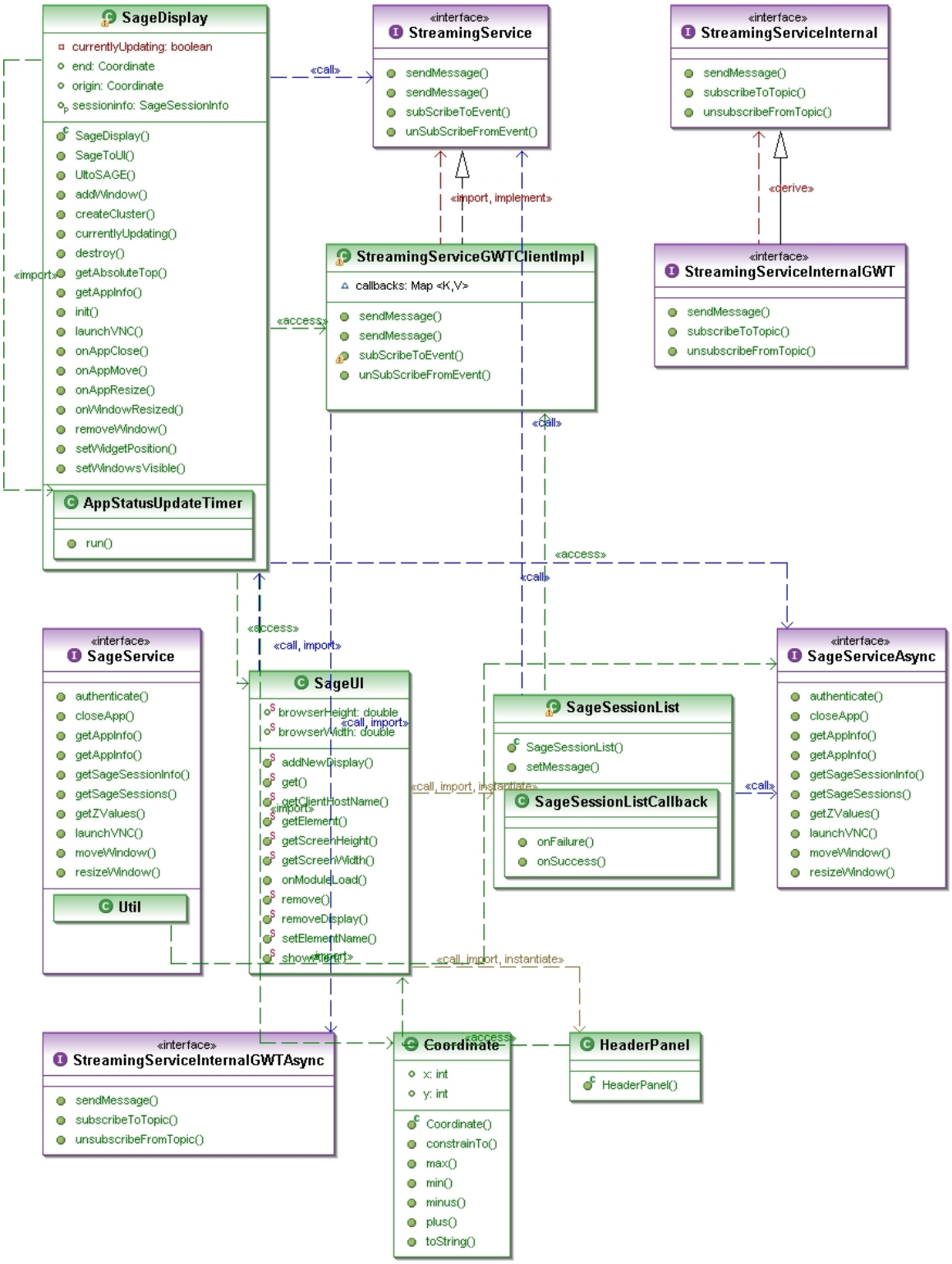


**Figure 12: class diagram of evl.sage.client package**

| Type | Name | Description |
|---|---|---|
| Class | SageUI | Main entry point class of the SAGE Display Controller application. This class contains the RootPanel to which all the user interface widgets and components are added. |
| Class | SageSessionList | Holds the data structures and widgets for all the available active SAGE sessions that a user could connect to. |
| Class | SageDisplay | Holds the data structures of all the application windows of type SageApp and implements SageServiceAsync interface to invoke the asynchronous SAGE services. |
| Interface | SageService | An interface implemented by the server side servlets allowing the clients to communicate with the SAGE services asynchronously. |
| Interface | Streaming Service | An interface implemented by the server side servlets to broadcast application update messages to all the |

| | | |
|---|---|---|
| | | clients accessing the SAGE session asynchronously. |

**Table 2: Client package description**

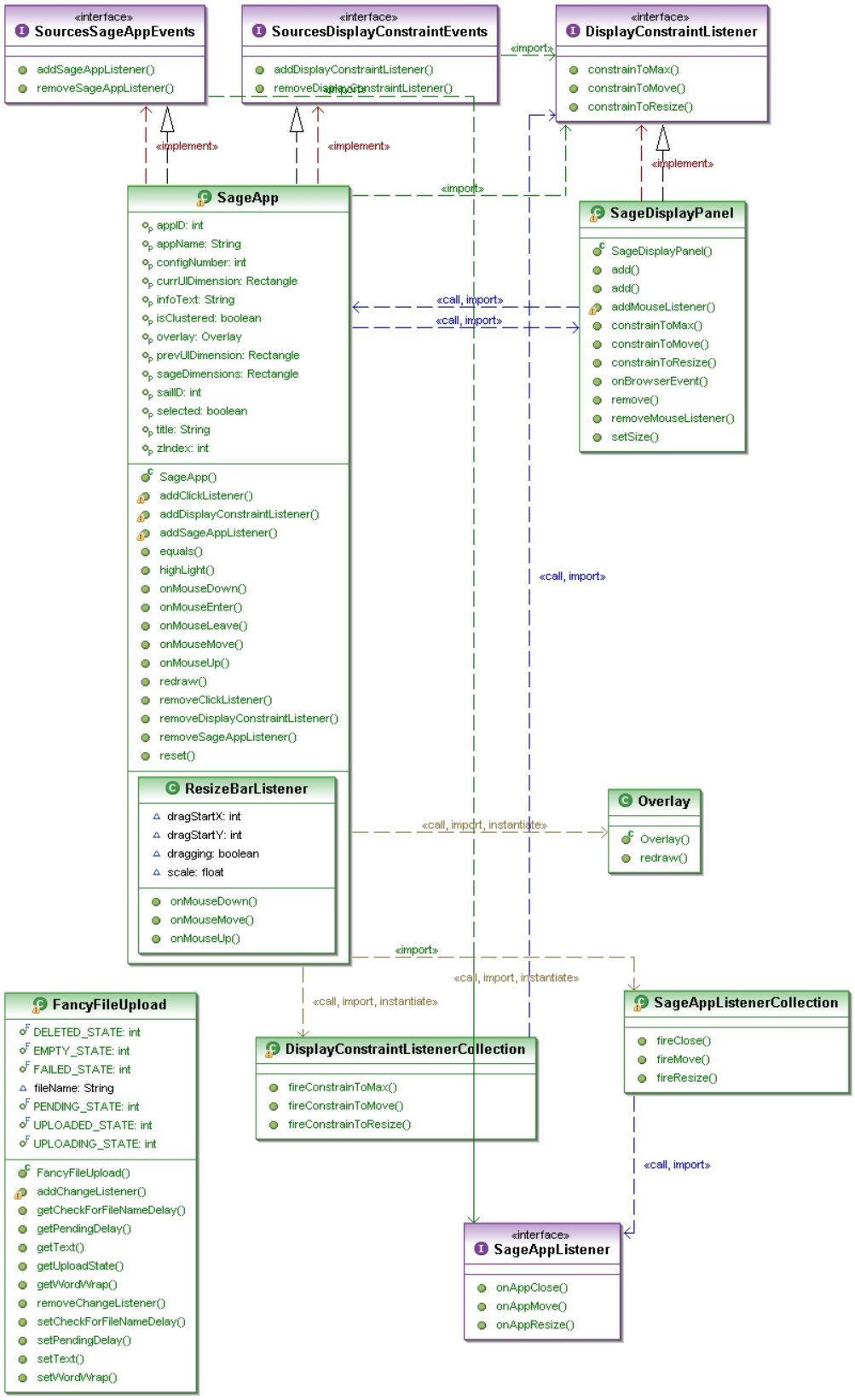


**Figure 13: class diagram of evl.sage.client.ui package**

| Type | Name | Description |
|---|---|---|
| Class | SageApp | User interface widget for a SAGE application window. The widget represents a window with buttons to minimize, maximize and close. It also implements EventListener interfaces to provide drag'n'drop and resize functionality. |
| Class | SageDisplayPanel | Widget to represent the tile display. It can contain only widgets of type SageApp as child widgets and also holds the user interface components to launch the VNC session (desktop sharing). It also invokes the asynchronous SAGE services whenever an application window event occurs. |
| Interface | SageAppListener | Event listener interface for SAGE application window events. |
| Interface | DisplayConstraintListener | Event listener interface for display constraint events. This interface is implemented by the SageDisplayPanel to |

| | | |
|---|---|---|
| | | constrain the movements of window within the display panel. |

**Table 3: Client UI package description**

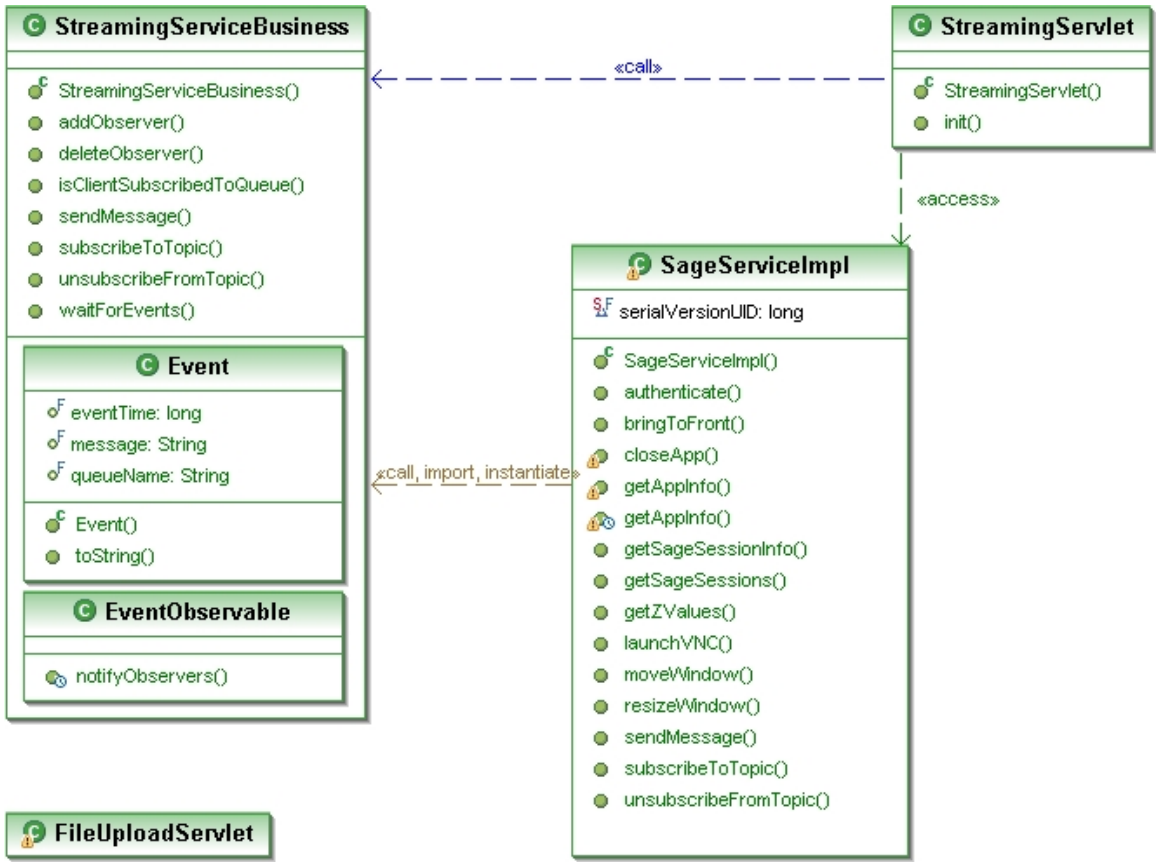


**Figure 14: class diagram of evl.sage.server package**

| Type | Name | Description |
|---|---|---|
| Class | SageServiceImpl | Servlet to handle the client requests and to communicate with the native SAGE application using XML-RPC. It implements the SageService interface to handle the asynchronous SAGE service calls requested by the client. |

| | | |
|---|---|---|
| Class | StreamingServlet | Servlet to broadcast SAGE or application specific update messages to all the clients accessing the SAGE session. |

**Table 4: Server package description**

## 13.3 Sequence diagram

To illustrate the working of the application better, a sequence diagram for a normal application window move operation is shown below.

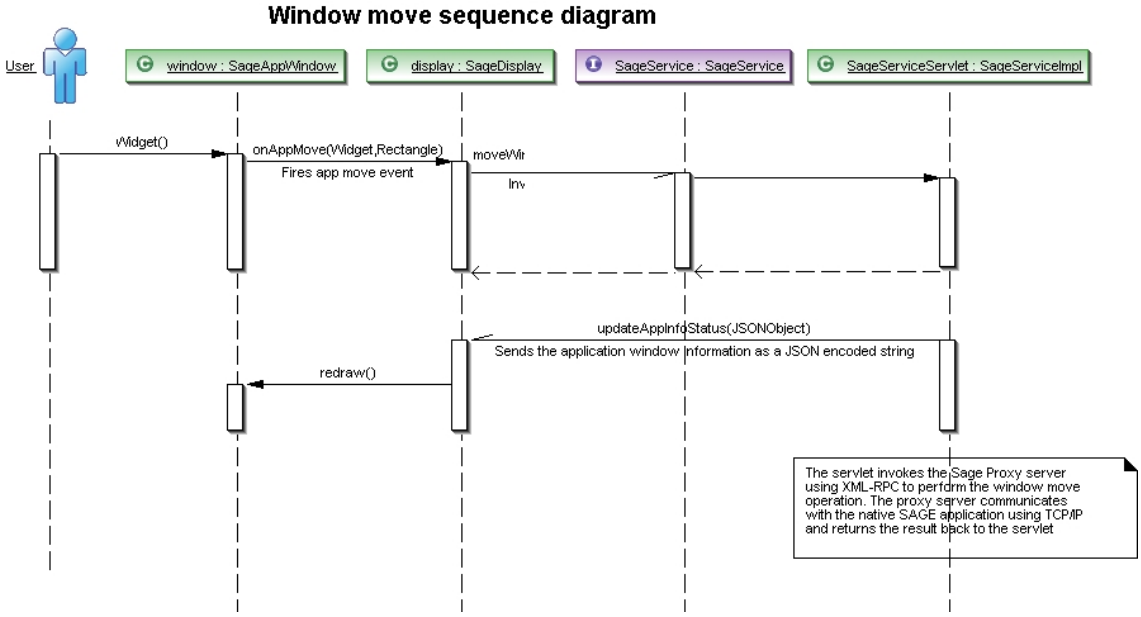


**Figure 15: Sequence diagram of an application window move operation**

# 14 Conclusion

SAGE Display Controller has been successfully deployed at several research laboratories around the globe such as Electronic Visualization Laboratory-UIC, Scripps Institute of Oceanography-UCSD, SARA-Netherlands and more.

.