NEC Sphere
300 Tristate International, 150 | Lincolnshire, IL 60069 | www.necsphere.com | 847.793.9600

August 2009

NEC UNIVERGE® Sphericall with Microsoft Office Sharepoint Server (MOSS) Webparts

Overall Architecture

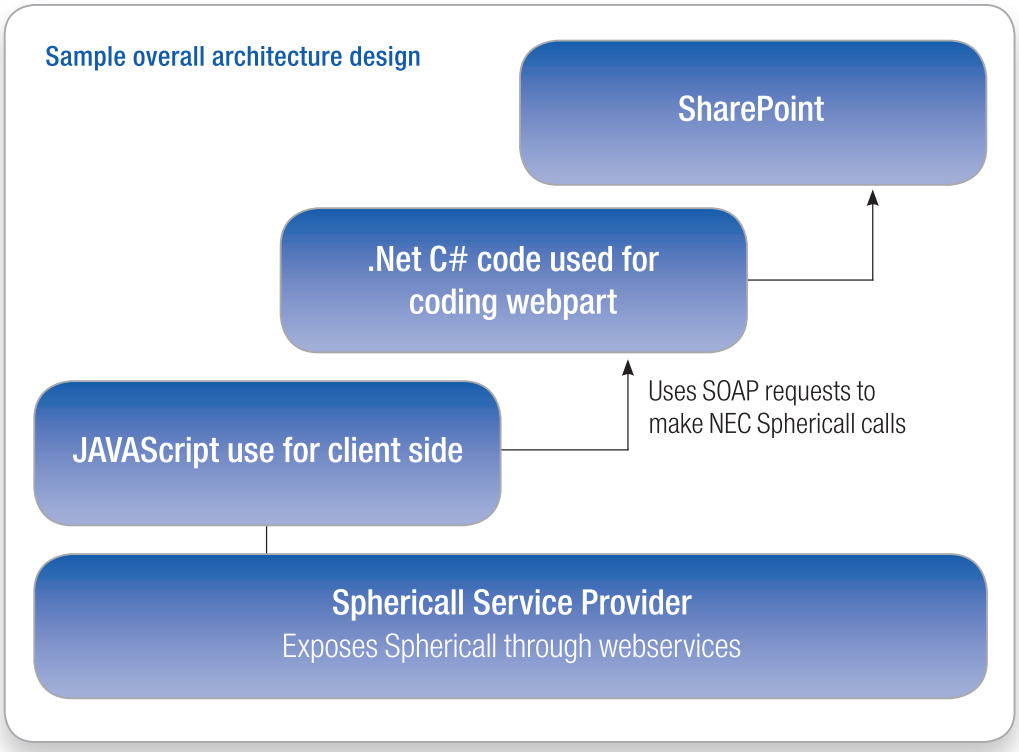
Collaboration is at the core of UNIVERGE Sphericall and NEC has created three portlets that are designed to provide voice communications within key business applications.

These applications are among the most popular software solutions in business today: Microsoft's SharePoint 2007 (MOSS) Webparts, Siebel ActiveX and SAP's NetWeaver are used to demonstrate interoperability and communication between the Sphericall platform and each Web portal technology.

Web services is the underlying technology used for each platform to communicate with Sphericall. As noted in the following diagram, each application uses HTTP to send SOAP commands for actions such as making calls or getting user directory from the Sphericall Web service server.

Each application uses the WSDL available through http://lava.soap.spherecom.com to get a list of the available operations or to use the information provided by the above Web server for automated code generation for the .NET components as used in the Siebel ActiveX control.

Please note the following diagram for the overall architecture and use of Web services in relation to each Web portlet.



SharePoint

Microsoft .NET (C#) is used to develop SharePoint user directory and quick conference Webparts. Additionally, JavaScript and jQuery are used for developing the client side code and lastly AJAX is used for sending SOAP/XML messages and XSL to transform server responses for HTML output to the browser.

Client side scripting was adopted for displaying SharePoint Webpart contents. Using JavaScript/jQuery allows tasks such as formatting output without the need to send information to the Web servers. Additionally, using client side coding, allows for easier code updates without the need for recompiling and releasing assemblies to the Web servers.

In this platform the following technologies are used:

Coding

C# Webpart

Custom controls, written in Microsoft C#, have been used in writing the Sphericall user directory and quick conference Webparts. These controls consist entirely of program code and no HTML template like the .ascx file is used for rendering HTML. For the most part each Webpart renders identical code other than changing a few ID's of container elements.

There are two custom controls that are used in SharePoint's Webparts and each can be found in the NEC.Sphericall.Web. Webparts project.

The Webparts output the basic shell of the two controls. Each Webpart outputs references to the JavaScript files need, CSS style sheet needed, as well as the container div html elements that serve as the containers for the user directory and conference controls.

Login controls are first displayed to the user allowing them to enter their credentials that are used to validate them to the Sphericall system. Once the user has logged in, the actual scripts/container elements are rendered to the page.

Since client side JavaScript is heavily used, a few variables are written out during the rendering of the Webpart:

- Current SharePoint Site Theme
- Sphericall Web Service Url

JavaScript

The main part of the application is encapsulated in the NECSphericall.js file that can be found in the NEC.Sphericall project. This is the heart that drives the functionality of both the user directory and quick conference Webparts.

Writing the main component of the application in client site scripting was adopted to allow for easier code release and updates in addition to allowing for a more responsive and flashy UI.

Code example:

```
function sphereStartSessionRequest()
{
        var startSessionRequest = '<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/
envelope/"><soap:Body>' +
        '<startSessionRequest xmlns="http://ws.spherecom.com/">' +
        '<application />' +
        '<host />' +
        '<info />' +
        '</startSessionRequest>' +
        '</soap:Body></soap:Envelope>';
```

As shown in the above code sample, all of the communications and rendering are maintained in the NECSphericall.js file where functions such as starting a Sphericall session or making calls are accomplished through sending a SOAP request to the Web service server, using Ajax for communication. Depending on the type of client side functionality is required; the Ajax communications is done asynchronously or synchronously.

Additionally, the following supporting JavaScript code is used to support each Webpart:

1. /_layouts/NEC.Sphericall/scripts/jquery.js
   a. This is the main JavaScript library file which holds the functions used for DOM traversing, event handling, UI and Ajax interactions. If so desired, the jQuery.js file can be updated to the latest release of jQuery (http://www.jquery.com). Testing should be done in a sandbox environment prior to upgrading production to make sure that the current release of jQuery does not affect the Sphericall functionality.

   An additional consideration for using jQuery was to ensure for cross browser compatibility of the webparts.

2. /_layouts/NEC.Sphericall/scripts jqueryui.js
   a. This file includes modules and effects provided by jQuery.

3. /_layouts/NEC.Sphericall/scripts jquery_ext.js
   a. jQuery offers a lot of "plug-ins" that allow for additional functionality to be added to jQuery that were not part of the base functionality. Context menus and hover panels are some examples of jQuery plug-ins.
   b. All plug-ins used were combined into one JavaScript file to reduce the number of scripts that have to be output and thus downloaded by the clients browser.

Ajax

Ajax is used in posting and retrieving data from Sphericall's server asynchronously (sometimes synchronously) in the background without interfering with the display and behavior of each page.

In the following code sample jQuery's Ajax libraries are used by issuing the $.ajax command. Using jQuery's Ajax libraries allowed for a unified methodology for making Ajax calls throughout NECSphericall.js file.

Code example:

```
$.ajax(
        {
                type: "POST",
                url: sphericallServerUrl,

                dataType: "xml",
                data: startSessionRequest,
                processData: true,
                beforeSend: function(req)
```

XSL and Ajax

Ajax is used in posting and retrieving data from Sphericall's server asynchronously (sometimes synchronously) in the background without interfering with the display and behavior of each page.

In the following code sample jQuery's Ajax libraries are used by issuing the $.ajax command. Using jQuery's Ajax libraries allowed for a unified methodology for making Ajax calls throughout NECSphericall.js file.

Code example:

```
success: function(xml)
                {
                        //Call SpheriCall and instantiate a session and get back a sessionID
                        var strOut = applyXSL(xml, xslLibrary + "startSessionResponse.xsl");

                        sphereSessionID = strOut;

                        //Now that we have a sphericall sessionID, make a call to a get the
address entries
                        sphereAddressEntriesRequest();
```

Install/Release

Features are used in order to install the SharePoint Webparts. This allows for scheduling or retracting the Webpart through SharePoint administrative Web pages instead of the need for the system admins to copy files to the server or needing to modify server's config files on each code update.

Additionally and more importantly, releasing code through this solution allows for environments with multiple SharePoint Web front-end servers to synchronize automatically instead of the need for copying code individually to each server.

There are multiple batch files for building and deploying the solution/Webpart to SharePoint.

The following steps provide and create the needed files for code release:

1. Rebuild Latest Code

The first step needed for a releasing a new solution is to create a .WSP file. A .WSP file is simply a cabinet file with the .wsp extension that holds all the needed files, folder structure for releasing a solution package. The \NEC.Sphericall\PACKAGE\create_solution.cmd file can be used to create this file.

This file references a .DDF file that is a simple text file with building instructions for the makecab utility. This cab.ddf can be found in \NEC.Sphericall\SOLUTION folder. This .ddf file sets parameters such as the output folder for complied package, its name (the .WSP file name) and the needed files and folders that needs to be included within the .WSP file.

Next, deploy the solution package to the intended servers. As before, use the \NEC.Sphericall\PACKAGE\deploy_solution_dev.cmd file for this purpose. Simply modify the URL in the batch file so that it points to the appropriate SharePoint site.

2. Add Webparts

Lastly, if the Webpart has never been added to a site previously, it is necessary to add the Webparts from the Webpart list. If the Webparts aren't showing up, go to the Webpart gallery and make sure that both UserDirectory and Quick Conference Webparts are loaded.

Installed Files

After the solution deployment, the following files and directories will be created and released to each Web front-end server.

Please note that the actual server path will be the server's \Program Files\Common Files\Microsoft Shared\web server extensions\12 folder.

1. JS files

LAYOUTS\NEC.Sphericall\scripts\jquery.js
LAYOUTS\NEC.Sphericall\scripts\jqueryui.js
LAYOUTS\NEC.Sphericall\scripts\jquery_ext.js
LAYOUTS\NEC.Sphericall\scripts\necsphericall.js

2. XSLT

LAYOUTS\NEC.Sphericall\styles\addressEntriesResponse.xsl
LAYOUTS\NEC.Sphericall\styles\answerCallResponse.xsl
LAYOUTS\NEC.Sphericall\styles\conferenceEntriesResponse.xsl
LAYOUTS\NEC.Sphericall\styles\fetchEventsResponse.xsl
LAYOUTS\NEC.Sphericall\styles\hangUpCallResponse.xsl
LAYOUTS\NEC.Sphericall\styles\holdCallResponse.xsl
LAYOUTS\NEC.Sphericall\styles\makeCallResponse.xsl
LAYOUTS\NEC.Sphericall\styles\sendTextChatResponse.xsl
LAYOUTS\NEC.Sphericall\styles\startSessionResponse.xsl
LAYOUTS\NEC.Sphericall\styles\terminalStatusResponse.xsl
LAYOUTS\NEC.Sphericall\styles\terminalStatusResponse ConfCall.xsl

LAYOUTS\NEC.Sphericall\styles\transfercall.xsl
LAYOUTS\NEC.Sphericall\styles\unHoldCallResponse.xsl
LAYOUTS\NEC.Sphericall\styles\conferenceBridgeName Response.xsl
LAYOUTS\NEC.Sphericall\styles\conferenceDropDown EntriesResponseMakeCall.xsl
LAYOUTS\NEC.Sphericall\styles\conferenceDropDown EntriesResponseTxtMsg.xsl
LAYOUTS\NEC.Sphericall\styles\presenceAddressEntries Response.xsl
LAYOUTS\NEC.Sphericall\styles\presenceTerminalStatus Response.xsl

3. CSS

LAYOUTS\NEC.Sphericall\styles\necsphericall_styles.css

4. IMAGES

IMAGES\NEC.Sphericall\expand_large.gif
IMAGES\NEC.Sphericall\open_panel_small.gif
IMAGES\NEC.Sphericall\presence_away_small.gif
IMAGES\NEC.Sphericall\presence_online_small.gif
IMAGES\NEC.Sphericall\sphere_logo.gif

5. Deploy DLL to GAC and add safecontrols entry

```
<Assemblies>
    <Assembly DeploymentTarget="GlobalAssembly Cache" Location="NEC.Sphericall.Web.WebParts.dll">
        <SafeControls>
            <SafeControl Assembly="NEC.Sphericall.Web.WebParts, Version=1.0.0.0, Culture=neutral, PublicKeyToken=831791eb53198545" Namespace="NEC.Sphericall.Web.WebParts" TypeName="*" Safe="True
        </SafeControls>
    </Assembly>
</Assemblies>
```

Remove Webpart/Retract Solution

To retract an already existing package, use the following NEC.Sphericall\NEC.Sphericall\PACKAGE\retract_solution_dev.cmd batch file. The batch file uses SharePoint STSADM commands to remove the solution from all content urls and then removes the solution from the farm. By using the STSADM commands and a SharePoint solution, SharePoint will properly remove all of the files so that nothing is left on the file system.

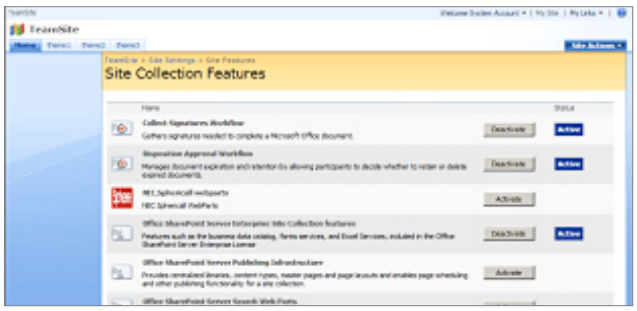
Installation Process

1. Deploy NEC_Sphericall_WebParts.wsp to SharePoint farm

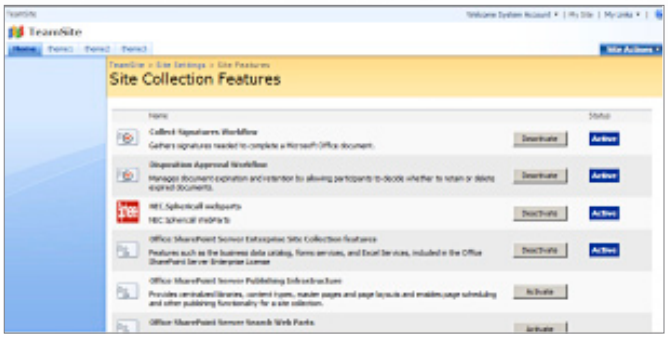
a. Run deploy_solution_dev.cmd (or use other cmd depending on environment)

2. Activate NEC Sphericall Site Collection Feature

a. Navigate to the site collection where you deployed the code
   i. Ex. http://testsite/
b. Navigate to Site Settings
   i. Ex. http://testsite/_layouts/settings.aspx
c. Navigate to Site Collection Features
d. Look for NEC.Sphericall Webparts Feature

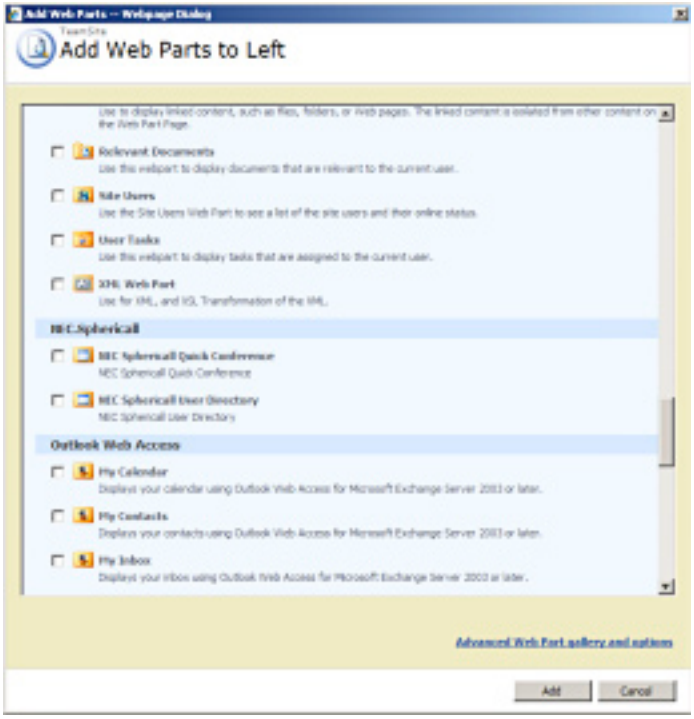
i. 

e. Active the NEC.Sphericall Webparts Feature

i. 

f. Navigate to page where you want to add the NEC.Sphericall Webparts to

g. Edit Page to add Webparts to correct zone and scroll down to section named “NEC.Sphericall”



h. Add selected NEC.Sphericall Webparts to page.

NEC Sphere Communications Inc.
300 Tri-State International, Suite 150
Lincolnshire, Illinois 60069

www.necsphere.com
info@necsphere.com
888.774.3732
847.793.9600