HOWTO: Using Xcentrisity™ to Serve WOW Applications

The Xcentrisity Business Information Server (BIS) with WOW support—known as BISplus, allows WOW applications to run as BIS service programs, thus enabling the launching of the WOW application from a Microsoft Internet Explorer (IE) web browser running anywhere in the world. While the thin client configuration of WOW has always allowed the Internet to be used as a transport medium, it does so by using a proprietary protocol and special TCP/IP port numbers. BISplus allows this and more, while doing so using the standards and protocols of the World Wide Web.

To deploy and use a BISplus WOW application, no special network configuration is required on the client or server system. In addition, by using IE, BISplus supports—at the time of launch in the browser—the automatic downloading of the entire client software package and all other client resources needed to run the WOW application. Combined with these and other features, the advantages of BISplus include:

- Allowing WOW applications to be deployed in a web environment without code changes,
- Using standard web browsers to dynamically deploy and run all client software components,
- Permitting the use of the HTTPS protocol, web authentication, and X.509 client certificates for high levels of security without programming, and
- Allowing the mixing of BISplus WOW applications and normal BIS web applications in a single web server environment.

Simple "Hello World" Example

This section demonstrates how to move a WOW thin client application to BISplus. The "Hello World" sample (installed in the BIS Samples directory) is used as an example.

The easiest way to use BISplus is by means of the Microsoft Internet Component Download (MSICD) capability that is built into the IE browser. This service provides a method for downloading all of the necessary components of the application to the client machine. It also provides the means to update those components automatically when new versions become available. For more information on MSICD, visit this site:

http://msdn.microsoft.com/library/default.asp?url=/workshop/delivery/download/overview/overview.asp

The following steps will create "cabinets" that contain the necessary client parts. (A cabinet is a single file, usually with a **.cab** extension, which stores compressed files in a file library.) An information file (**.inf**) provides installation instructions that the MSICD service uses to install and register software components downloaded from the Internet, as well as descriptions of the individual files, where they are located, version numbers, and other installation-specific information. In this example, the server response file, **default.srf**, which is served by BIS, contains an HTML `<object>` tag that instructs the browser where to find the first-referenced cabinet, and initiates the download and execution of the BISplus WOW ActiveX client (**wowclient.ocx**) component. The first-referenced cabinet normally takes the name of the application and in this example is called **helloworld.cab**.

Note The following instructions assume a Windows BIS server. The BISplus WOW application can be served from UNIX versions of BIS, but the cabinets must be constructed on Windows and then transferred to the corresponding directory on the UNIX server system. See "Using UNIX as a Server" on page 6.

To move a WOW thin client application to BISplus:

1. Install BISplus on the server system (see the *BIS User's Guide* for information on how to perform this installation).
2. Create a virtual directory on the server machine by using the **bismkdir** utility installed with BIS. In these instructions, the virtual directory is assumed to have a path of **\inetpub\wwwroot\mywowapp**.
3. To achieve a more structured organization, Liant recommends that you create the following two sub-directories so that source files are not included in the same directory as binary files:

   ```
   src
   bin
   ```

4. Copy the following files, obtained from the **\inetpub\wwwroot\xbis10\samples\wowhelloworld** directory, into the **\inetpub\wwwroot\mywowapp** directory:

`default.srf`	required file
`helloworld.srf`	required file
`error.srf`	required file if referenced in **default.srf**
`layout.css`	required file if referenced in **default.srf**

   Note In your own program, you would substitute the name of your application for any files named "helloworld."

5. Copy the following files, obtained from the **\inetpub\wwwroot\xbis10\samples\wowhelloworld\src** directory, into the **\inetpub\wwwroot\mywowapp\src** directory:

`makeinf.exe`	utility program to make the information (**.inf**) file
`helloworld.cob`	main COBOL program
`helloworld.org`	required file
`helloworld.lst`	list of files in **helloworld.cab**
`liant.lst`	required file (list of files in **liant.cab**)
`allegris.dll`	required file
`codebrdg.dll`	required file
`pockethttp.dll`	required file
`rmprop.dll`	required file
`rmhttp.dll`	required file
`rmguife.dll`	required file
`rmremprt.dll`	required file
`wowclient.ocx`	required file
`wowmfcrt.dll`	required file
`wowrt.dll`	required file

6. Obtain a copy of the cabinet file manipulation utility, **cabarc.exe,** from the Microsoft site, http://support.microsoft.com/?id=310618, and place it in the **\inetpub\wwwroot\mywowapp\src** directory.

 Liant Software Corporation © Copyright 2006. All rights reserved.

7. Build a file named **build.bat** that contains the following command lines and place it in the **\inetpub\wwwroot\mywowapp\src** directory:

```
REM Copy the Cobol programs
if exist ..\bin\helloworld.cob attrib –r ..\bin\*.cob
copy *.cob ..\bin

REM Build the .inf file

set bis_program_dir=%ProgramFiles%\Liant\BIS10\
set    path="%bis_program_dir%";%PATH%

makeinf helloworld.org helloworld.inf

REM Build the cabinets
cabarc n ..\helloworld.cab @helloworld.lst
cabarc n ..\liant.cab @liant.lst
```

8. Execute the **build.bat** batch file. This will copy the RM/COBOL® program(s) to the **\inetpub\wwwroot\mywowapp\bin** directory and build the cabinets in the **\inetpub\wwwroot\mywowapp** directory.

   Note The **makeinf** program is a utility that simplifies the insertion of "`FileVersion=`" lines into the **.inf** file. It replaces "`#GetVersion=<filename>`" lines in the **.org** file with "`FileVersion=…`" lines and writes the new lines to the .inf file along with all other lines in the .org file.

9. You now have everything that is needed to access this test application from a client browser. From your client machine, bring up IE and enter the following address (substituting your server's name for "*MYSERVER*"):

   http://*MYSERVER*/mywowapp

   This last step will download the appropriate files to your client machine. The files will be placed in the IE cache directory, which is normally **\windows\downloaded program files**. If you use IE to navigate to that directory, you will see an entry called "BIS WOW Extensions Thin Client". If you need to remove those downloaded files (for example, during testing when you want to start over from scratch), right-click on that entry and select "Remove".

To more fully understand the Internet Component Download process, use the editor of your choice and examine the following files:

```
default.srf
helloworld.srf
helloworld.inf
```

The **default.srf** file is the file that IE will access when the following URL is referenced:

http://*MYSERVER*/mywowapp

Take a look at the "`<object … >`" tag in that file. Notice that it references the **helloworld.cab** and **helloworld.srf** files. The reference to **helloworld.cab** initiates the download process, and the reference to **helloworld.srf** serves the first HTTP page to start the BISplus WOW process.

You will probably want to modify **default.srf** and **layout.css**, as desired, to create a starting Web page (the "look-and-feel") that is appropriate for your company and application.

A More “Real World” Example

Frequently, an application may require separate data sets for different yet similar components within a company, for example, yet use a common set of programs. The following directory structure is an example of one way to accomplish this:

```
\inetpub\wwwroot\mywowapp
          common
               cabinets
                    clientparts.cab
                    myapplication.cab
                    liant.cab
                    common.srf
                    error.srf
                    layout.css
               programs
                    mainpgm.cob  (and other RM/COBOL programs)
          division1
               data
                    Data files for Division1
               default.srf
               myapplication.srf
          division2
               data
                    Data files for Division2
               default.srf
               myapplication.srf
          division<n>
               data
                    Data files for Division<N>
               default.srf
               myapplication.srf
          src

               …
```

When “Division1” wants to run the application, it references the URL:

http://*MYSERVER*/myapplication/division1

When “Division2” wants to run the application it would reference

http://*MYSERVER*/myapplication/division2

and so on.

The **default.srf** file in all divisions contains the following:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
{{ handler * }}
{{ include ../common/cabinets/common.srf }}
```

 Liant Software Corporation © Copyright 2006. All rights reserved.

The **common.srf** file, referenced in **default.srf**, contains the following:

```
{{ handler * }}
<html>
<head>
      <title>Customer Application</title>
      <style type="text/css" media="all">
            @import "../common/cabinets/layout.css";
      </style>
</head>
<body onBeforeUnload="CustApp.Stop()">
<h1 id="Header">Customer Application on the WEB</h1>
<div class="Content">
<p>
Please wait while Customer Application for the WEB downloads...
</p>
<p>
If you see a warning from Internet Explorer that the download sample does not have
a valid certificate, you should click on "Install" to allow this sample to run.
</p>
<object
      id="CustApp"
      codebase="../common/cabinets/myapplication.cab#version=-1,-1,-1,-1"
      CLASSID="clsid:6DED256A-08A2-4CA8-839E-8422BF920B5E"
      Height=0 Width=0 Border=0>
      <PARAM NAME="URL"
                  VALUE="{{Value('myapplication.srf?trace=file',MAKEABS)}}”>
      <PARAM NAME="Cookies"
                           VALUE="{{Value(HTTP_COOKIE,URLENCODE)}}">
      <PARAM NAME="CommandLine"
            VALUE="">
      <PARAM NAME="Trace"
            VALUE=0>
      Error loading Customer Application!
</object>
</div>
</body>
</html>
```

The **myapplication.srf** file contains the following:

```
{{//There must be no whitespace rendered before the exchange tag, hence the
newline-eating comment tags }}{{//}}
{{ Handler * }}{{//}}
{{//}}
{{ Trace(start,queryparam=trace,ip=127.0.0.1) }}{{//}}
{{ SessionParms(ServiceTimeout=30,InactivityTimeout=600) }}{{//}}
{{//}}
{{ If Value(_logMask,QP,MATCH="^[1]$") }}{{//}}
{{   SetEnv(RM_WOWWEB_LOG_MASK=1) }}{{//}}
{{ EndIf }}{{//}}
{{ If Value(_logMask,QP,MATCH="^[23]$") }}{{//}}
{{   SetEnv(RM_WOWWEB_LOG_MASK=3) }}{{//}}
{{ EndIf }}{{//}}
{{ If Value(_logMask,QP,MATCH="^[45]$") }}{{//}}
{{   SetEnv(RM_WOWWEB_LOG_MASK=5) }}{{//}}
{{ EndIf }}{{//}}
{{ If Value(_logMask,QP,MATCH="^[67]$") }}{{//}}
{{   SetEnv(RM_WOWWEB_LOG_MASK=7) }}{{//}}
{{ EndIf }}{{//}}
{{//}}
{{ RunPath(../common/programs,./data) }}{{//}}
{{//}}
{{ StartService(mainpgm -V,bispluswow) }}{{//}}
{{//}}
{{ XMLExchange(OnExit=../common/cabinets/error.srf) }}{{//}}
```

Notice the use of relative pathnames in the **.srf** file. Those references are relative to the Division<n> directory.

To avoid excessive network traffic every time the application is started, make certain that the first-referenced cabinet (in this case, **myapplication.cab**) contains only the **custapp.inf** file. Doing so allows this one short file to be downloaded quickly. IE will check version numbers of the files listed in the information file (**.inf**) and determine whether any other cabinets need to be downloaded.

This example should serve as a starting point for your application. Simply substitute your own programs and files as necessary.

Using UNIX as a Server

It is recommended that you first get your application running with a BIS Windows (IIS) server. Subsequently, you can zip up your server directory and transfer it to a UNIX server. You can then configure Apache, as documented in the *BIS User's Guide*, to enable your BISplus WOW application directory to be "served" from this machine.

 Liant Software Corporation © Copyright 2006. All rights reserved.