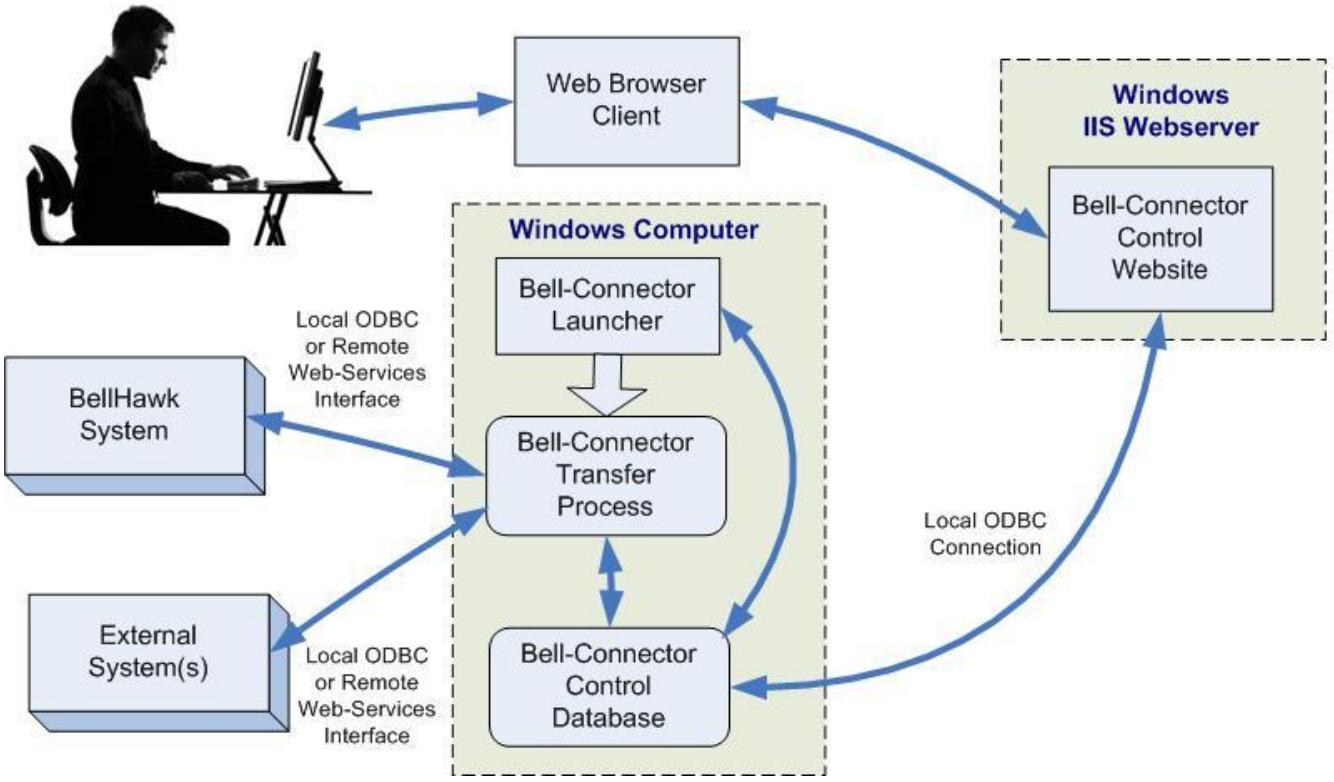
Installing and Configuring MilramX



Introduction

The MilramX is a framework and set of software tools used by the technical staff of BellHawk Systems, its Solution Partners, and it's Clients to implement custom data exchange interfaces between BellHawk and external systems such as ERP and accounting systems.

MilramX consists of a number of elements:

1. One or more MilramX transfer processes (.exe executable programs) whose job it is to perform the actual transfers between the external system(s) and BellHawk. Within the transfer process there may be one or more Data Transfer Objects (basically named subroutines) whose function is to transfer a specific type of data object. When the transfer function is run, the DTO name is specified as one of its arguments.
2. A MilramX Control Database – this is a SQL Server database that contains data about the latest time and date each DTO was run and data about when each DTO is scheduled to run next. It also contains tables into which to log failed transfers into BellHawk and from which they can be edited and retried. In the DEX2 interface, it contains Import and Export tables with which external applications can exchange data through an ODBC connection.

3. A MilramX Control Website that enables remote monitoring and control of the transfer processes. Through this interface, users can schedule and monitor the transfers, see and investigate errors, and edit and retry any transfers that failed.

4. A MilramX Launcher program. This Launcher.exe program runs continuously as a background user process. It monitors the setup and status data in the Control database to decide when to run each transfer process and with which DTO. It also tracks for transfer processes that have exceeded their maximum run-time and kills them if needed.

The launcher and transfer function programs are normally run on a Windows 7 Pro computer in the local plant along with the Control database, which typically uses SQL Server Express. They can also be run in a separate user process on a Windows 2008 Server.

The MilramX Website runs under the control of the IIS webserver. It can be installed on the same Windows 7 Pro Workstation computer as the transfer function or it can be run on a separate Windows 2008 Server computer, provided that the Windows Server is on the same LAN or VLAN network as the computer used to host the Control database.

Please note that the MilramX is not needed for the following interfaces to BellHawk:

1. A direct connection to the BellHawk database made using the BHDSK .Net DLL. This interface is typically used by external VB.Net programs that communicate directly with BellHawk at a business objects level.

2. A web-services interface where the external application communicates directly with BellHawk using SOAP/XML packets over the Internet or Intranet.

3. Excel or comma delimited file imports into BellHawk or exports from BellHawk.

Unlike with the use of the .Net SDK interface or the BellHawk web-services interface it is not necessary to setup a separate BellHawk device, with its own user name and password, in order to use the MilramX interface to BellHawk.

MilramX keeps the last-transferred times separate for each transfer function by tagging these times with the name of the transfer function in the Control database. This enables multiple transfer functions to be active at the same time.

Please note that the following instructions are based on installing MilramX on a Windows Workstation 7 Pro computer. They are essentially the same if you are installing this on a Windows 2008 Server computer but there are minor differences due to the different editions of the Windows operating system.

By convention, in this document, transfer functions are generally named BHxxI.exe where the xx is a mnemonic for the system with which BellHawk is exchanging data; hence BHQBI is the name for the BellHawk to QuickBooks Enterprise interface and BHQBI.exe is the transfer function process.

In this document, we will generically refer to BHxxI as the name of the transfer function. In practice the correct name for the transfer function should be substituted. Also we use the shorthand notation of BC Website for MilramX control website.

Adaptors

MilramX uses the concept of Adaptors as a way of connecting to different databases and systems. These Adaptors encapsulate the details of interfacing with different databases and so reduce the amount of programming work in developing interfaces. They also allow the same interface code to be used with different databases and different ways of interfacing with systems.

The Adaptors mechanisms currently available with are:

1. Local direct ODBC access to a Microsoft SQL Server Database. This covers access to a wide range of systems including BellHawk, all of the Microsoft Dynamics ERP systems, and many other ERP systems.
2. Remote access to a BellHawk system using its SOAP/XML web-services interface
3. Local access to QuickBooks Enterprise through the QODBC driver
4. Local ODBC access to an Oracle Database through the Oracle ODBC driver

Other Adaptor mechanisms can be added as needed, including:

1. Local ODBC access to a MySQL Database through the MySQL ODBC driver.
2. Local ODBC read-only access to a Sage 100/MAS 90 database
3. Local ODBC access to Microsoft Access through the MDAC ODBC driver

The parameters to each adaptor mechanism, such as database server or web-services connection data, are given in the initialization files for the BC Website and transfer functions, or included in the ODBC DSN (Data Source Name) instance setup.

Generally we do not create adaptor mechanisms for web-services interfaces to external systems. Instead we use their WSDL to create a set of proxy server subroutines that are called within the appropriate DTOs.

We made an exception, in the case of BellHawk, so as to provide a programming interface that is the same whether BellHawk is installed locally in a client's plant or is accessed at a remote data center over the internet. This enables the installation location decision to be made at system configuration time rather than having to change the transfer function code.

Some ODBC drivers, such as the QODBC driver for QuickBooks, are capable of remotely accessing their corresponding systems/databases at remote sites, in the Cloud, over the Internet.

Setting up ODBC Data Sources

Please note that all ODBC drivers should be setup by a Windows system administrator as Systems Data Sources, so they are accessible to all users on the computer on which they are instantiated.

You will not need to setup and configure ODBC Data source instances for the Control database or for the BellHawk database but you will for other systems.

Connections to locally accessible Microsoft SQL Server databases use standard Microsoft SQL Server ODBC drivers, which are installed on the Workstation when you install SQL Server on the Windows Workstation. The ODBC connections to the control database and the BellHawk database use the special Microsoft SQL Server DSN-less connections. This means that you do not have to setup special DSN (data source name) ODBC connections for these connections. They also work irrespective of the version of SQL Server used.

You can setup a named DSN connection to the BellHawk database but not for the Control database, as this also uses the same setup data from the initialization file for direct connections to the database, which it uses wherever possible for efficiency. If you setup a named DSN instance for BellHawk, just make sure that you use the correct version of the ODBC driver (version 10 is for SQL 2008, version 11 is for SQL 2008 R2, version 12 is for SQL 2012).

The primary reason for setting up a DSN ODBC connection for the BellHawk database is for a systems administrator to hide the user name and password for this database.

To setup DSN based ODBC connections (assuming that you are running on a Windows 7 Pro workstation):

1. 64 Bit ODBC drivers can be setup and edited through
   Start=>Control Panel=>Administrative Tools=> Data Sources (ODBC)

2. 32 Bit ODBC drivers can be setup and edited by running
   C:\Windows\SysWOW64\odbcad32.exe

In general you will use the 64 bit drivers to access Microsoft SQL Server based databases on a 64 bit computer. But many other ODBC drivers, such as QODBC for QuickBooks are 32 bit drivers even when running on 64 Bit computers.

If you are running the Control Website on a Windows Server that is separate from the Workstation where the Control database is residing then you will need to make sure that you:

1. Install the client software for the SQL database you installed on the Workstation. This brings with it the appropriate edition of the ODBC driver.

2. Create TCP access for ports 1433 and 1434 through the firewall on the Windows Workstation computer to allow the ODBC driver on the Windows Server to communicate with SQL Server running on the Workstation.

If the MilramX transfer function is communicating with an external system by means of an ODBC connection then you need to make sure that you have installed the ODBC driver and setup an ODBC connection to the external systems as a system data source.

Connecting to a database other than Microsoft SQL Server may require purchasing or acquiring the ODBC driver from a third party and then installing it on the Workstation. For example, the read/write ODBC driver for QuickBooks Enterprise must be purchased from QODBC.com.

Please note that the transfer function uses a direct connection to the SQL server Control database and so does not need an ODBC connection for this database.

Please see appendices for installation and setup notes on specific ODBC drivers.

Software Distribution

The software is distributed as:

1. A zipped file containing a backup of the Control database.
2. A zipped file containing the Control Website.
3. A zipped file containing the Launcher code in installable .msi format
4. A zipped file containing the Transfer Function code in installable .msi format. There will be multiple of these is there are multiple transfer functions.
5. A zipped file containing an Excel import for the DTO setups for the MilramX for the specific transfer function(s) in use. This can be imported using the DEXEL capability of the MilramX website.
6. A zipped file containing the BC.Lex license files for the website and for the transfer function. These may be the same or they may be for different computers.

System Setup

The following procedures assume that the setup is performed on a Windows 7 Pro workstation. This may need to be modified if the MilramX is setup to run on a Windows Server computer. The setup procedure described below also assumes a non-domain security environment. It may need to be modified, as appropriate in a domain based security environment.

Set up a MilramX User Account

You need to create a non-admin user account on the Windows computer to be the user account in which the Launcher is run in the background. This user should be made a member of the User group and the IIS_IUSR group. We normally use the name BellConnector for this user. You do need to assign a password for this user.

Installing the .Net Libraries

If .Net 4.0 or 4.5 is not already installed on your computer, please download one of them from the Microsoft downloads site and install it. This is needed for JSON handling in the interfaces.

Setting up Firewall Ports

If you are running a firewall on your Workstation then you will need to enable access for web browsers using the web services to the MilramX software running under IIS.

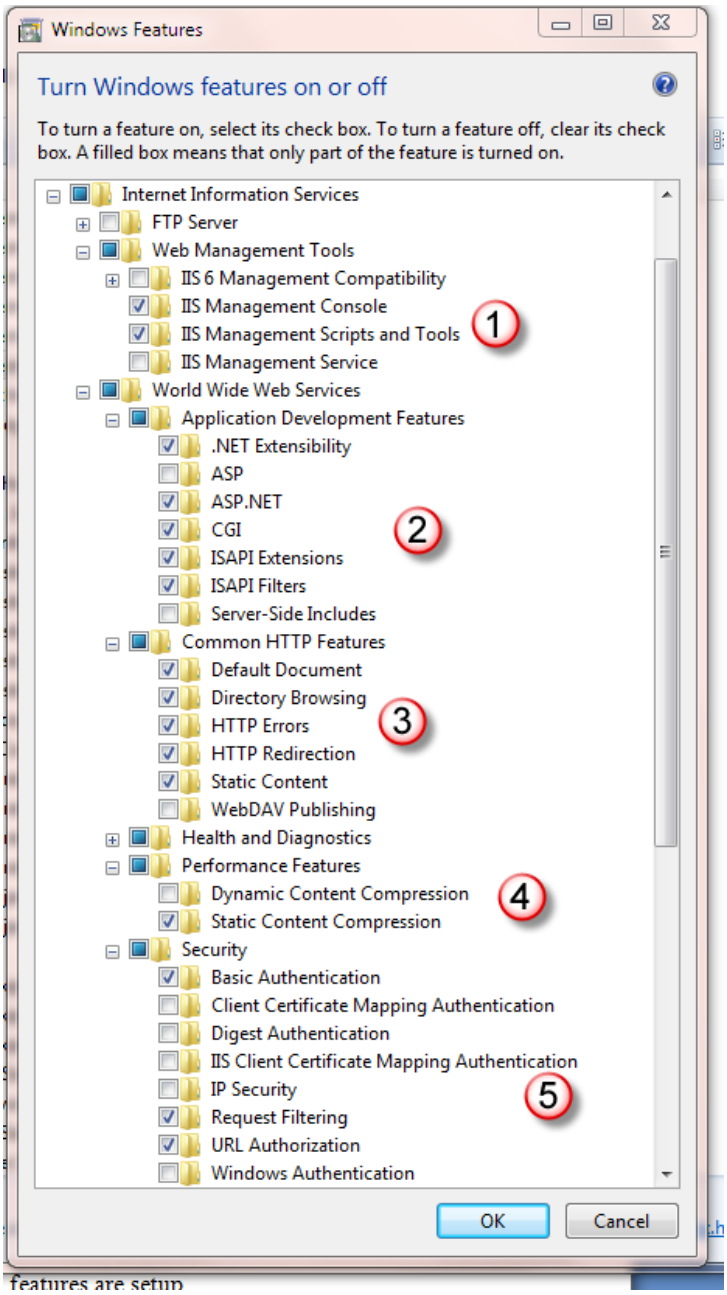
Normally MilramX is setup to use port 80 for receiving HTTP, which by default are routed to IIS by the operating system. But other port numbers may be used if setup as part of the website setup in IIS.

If your Windows Workstation is protected with a software or hardware firewall then requests on port 80 (or other port to be used for the website) from the Intranet need to be allowed through this firewall.

If the MilramX Control website is running on another server from the Control database then you need to enable TCP communications on Ports 1433 and 1434 for the ODBC driver in the Windows Server to access the SQL Server Control database on the WorkStation.

Setting up IIS

The MilramX website runs under the control of the IIS 7 webserver, which is provided as part of a Windows operating system on Windows Workstations (such as Windows 7 Pro) and Windows servers (such as Windows Server 2008) but not on Windows Home PCs.



The first step in setting up a MilramX system is to setup the IIS webserver, if this has not already been done on the Windows Server or Workstation you are using. By default IIS is not turned on, as a security measure by Microsoft.

In a Windows Workstation 7 Pro operating system, use Control Panel > Programs and Features > Turn Windows Features On or Off and then make sure that you have at least the settings shown at right setup

These include:

The Web Management Tools (1) which gives access to the IIS Management console.

The Application Development features (2) which are needed to run ASP.Net applications like MilramX.

Needed Common HTTP features (3) to allow the website to run.

Support for compressing static content (4) can improve performance but dynamic compression reduces performance.

The security settings shown here (5) are essential for correct operation.

Setting Up the MilramX Databases

Installing the Control Database

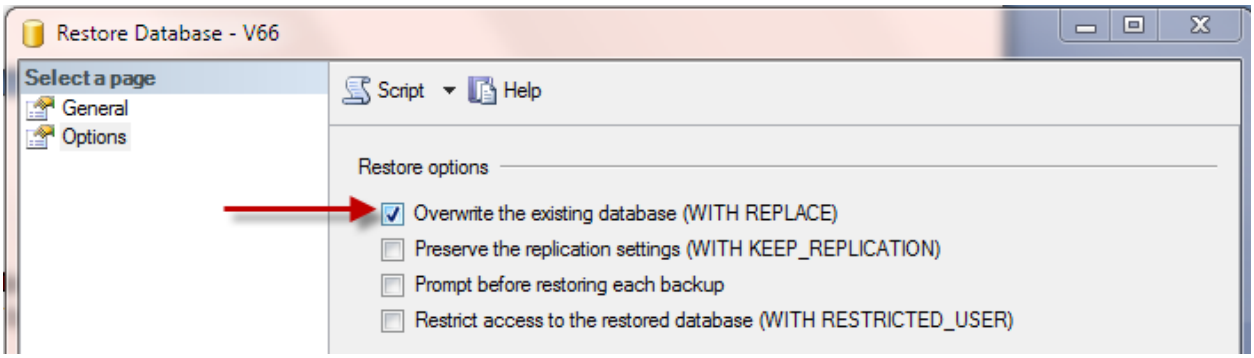
The control database comes as a Windows 2008 backup file in a zipped folder.

You should first install SQL Server Express on the Windows 7 workstation, making sure that you install the instance with mixed mode authentication and write down the system administrator (sa) password that you used.

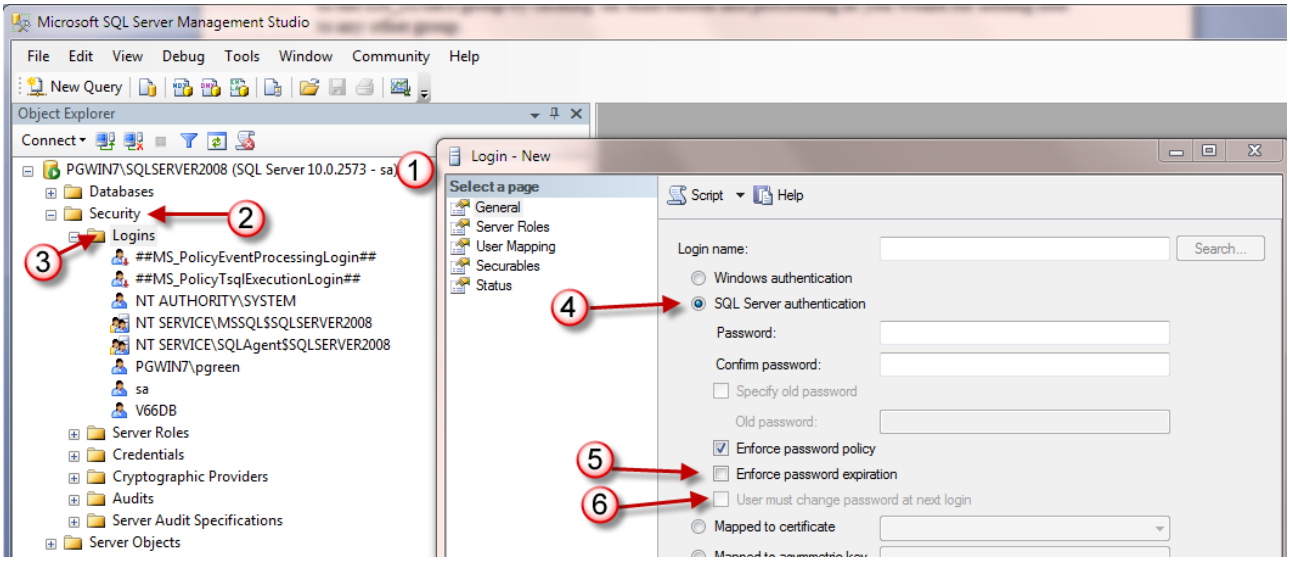
Then restore the control database from the provided backup. If desired for security you can create another user beside the system administrator for the database.

The process goes as follows:

First unzip the backup and use the SQL Server Management Console to restore the database. Right click on database then select Tasks => Restore =>Database and then selecting he backup file (.bak) on the General page and Options:



Then you need to add the login for the user you created to the logins for the SQL Server instance you are using:

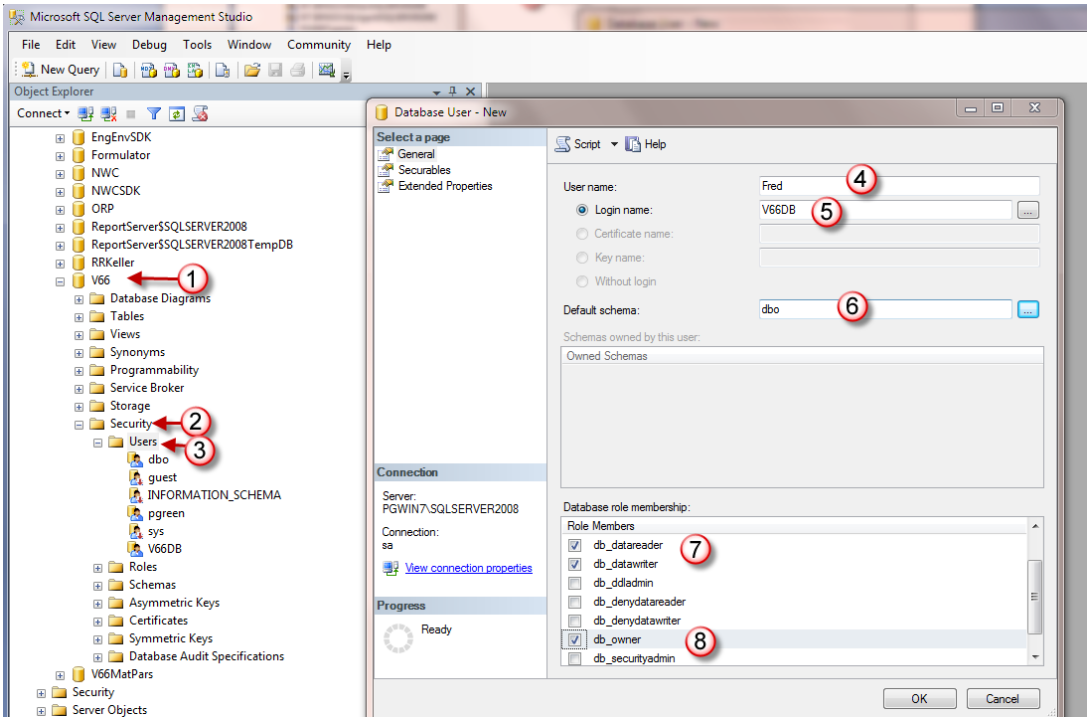


For the Server Instance (1) select Security (2) and Logins (3) to see a list of existing logins for the SQL Server instance. Then right click on Logins and select New Login and then add new

login as shown above. Select SQl Server Authentication (4) and Enter the User Name and Password that you want to Use.

Also turn off the Enforce password expiration (5) and User Must Change password (6).

Then you need to set the access privileges for the SQL Server user login that you created above to the specific database as shown below:



To do this:

1. Select your Control database (1)
2. Select Security (2)
3. Right Click Users (3) and select New User
4. Enter User Name (4) as to how you want to identify this user for this database. I typically use the same name as the Login name I setup for the SQL Server instance.
5. Enter the Login name (5) that was setup for the MilramX user for this SQL Server instance.
6. Select dbo as the Default schema (6)
7. Select db_datareader and db_datawriter as role membership (7).
8. I also usually select db_owner (8)

Session State Database

The MilramX website makes use of an ASP.NET SQL session state database as part of the IIS infrastructure. If such a database does not yet exist, it needs to be created. (A single session state database can serve multiple sites; if the database BellHawkASPState has already been created for BellHawk, you may advance to adding the SQL user credentials and editing of Web.Config.)

Although the program aspnet_regsql.exe has a wizard capable of creating a SQL Server database that supports a variety of ASP.NET applications, to install and configure the database to support session state requires that the aspnet_regsql.exe tool be run at the command line. The program can be found in the C:\Windows\Microsoft.NET\Framework\v2.0.50727 folder on your Web server. A copy is also located in the C:\Windows\Microsoft.NET\Framework\v4.0.30319 folder. It must be run as a system administrator to work correctly.

Supply the following information in the command line:

1. The name of the SQL Server instance, using the -S option.
2. The logon credentials for an account that has permission to create a database on SQL Server. Use the -E option to use the currently logged-on user, or use the -U option to specify a user ID along with the -P option to specify a password.
3. The -ssadd command-line option to add the session state database.
4. The –sstype c option to specify a custom session state database.
5. The name BellConnectorASPState, using the –d option

Example:

```
C:\Windows\Microsoft.NET\Framework\v2.0.50727\Aspnet_regsql.exe
-S PGWIN7\SQLSERVER2008 -E -ssadd -sstype c -d
BellConnectorASPState
```

The success of this step can be confirmed by using SQL Server Management studio. There should be a new database named BellConnectorASPState composed of the two tables dbo.ASPStateTempApplications and dbo.ASPStateTempSessions.

Then go to this table and add the credentials of the BellConnector database user login to this table, exactly the same as previously described for the BellConnector database.

Then edit Web.Config using Notepad or an XML editor such as Visual Studio (used in this example). Find the entry for <sessionState> and then modify it as shown below.

```
29  <system.web>
30    <!--
31          Set compilation debug="true" to insert debugging
32          symbols into the compiled page. Because this
33          affects performance, set this value to true only
34          during development.
35      -->
36    <!--<sessionState mode="InProc" cookieless="false" timeout="120" stateNetworkTimeout="60" />-->
37    <sessionState allowCustomSqlDatabase="true" cookieless="AutoDetect" mode="SQLServer" sqlConnectionString="Data Source=PGWIN7\SQLSERVER2008;
38                  Initial Catalog=BellHawkASPState; User ID=V66DB; Password=V66db45" stateNetworkTimeout="1440" timeout="1440"/>
39
40
41
42
```

Here, we enter our Session State database name (1) that matches the session state database we created previously, along with our MilramX database user-name (2) and password (3) that we setup for this database.

Setting up the MilramX Website

Unpacking the Website

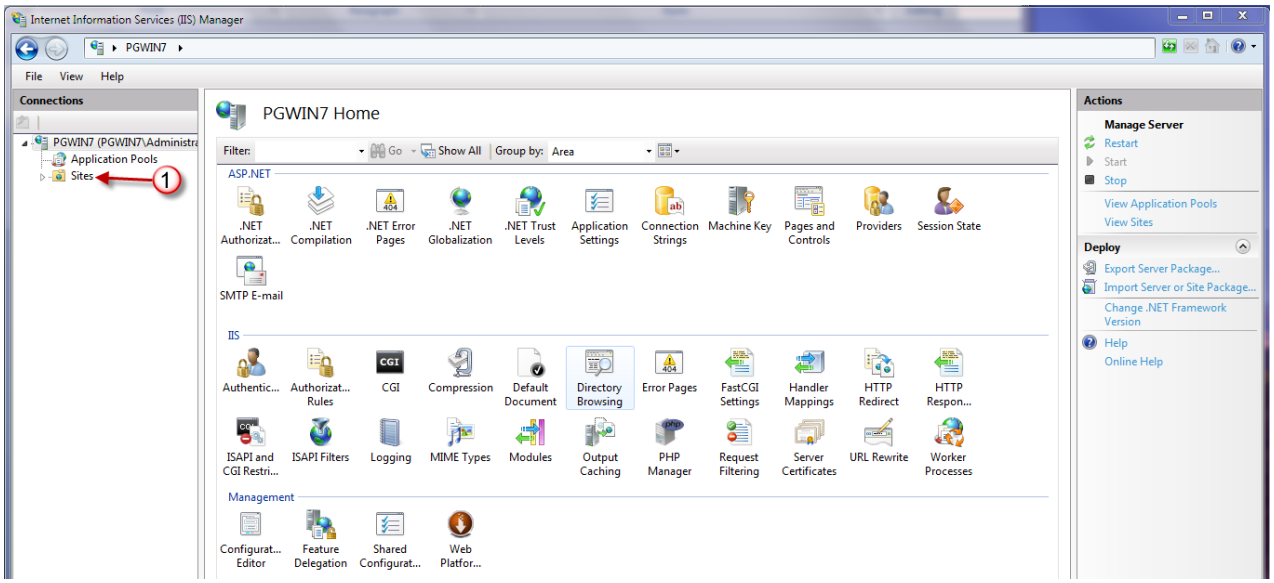
Create a folder C:\inetpub\wwwroot\BellConnector\ or an alternate folder to hold the website. If you use a different folder make sure that members of the IIS_IUSR group have read privileges.

Unzip the website file into this folder. Then make sure that members of IIS_IUSR group have write and update privileges to the Logfiles folder.

Accessing the IIS Management Console

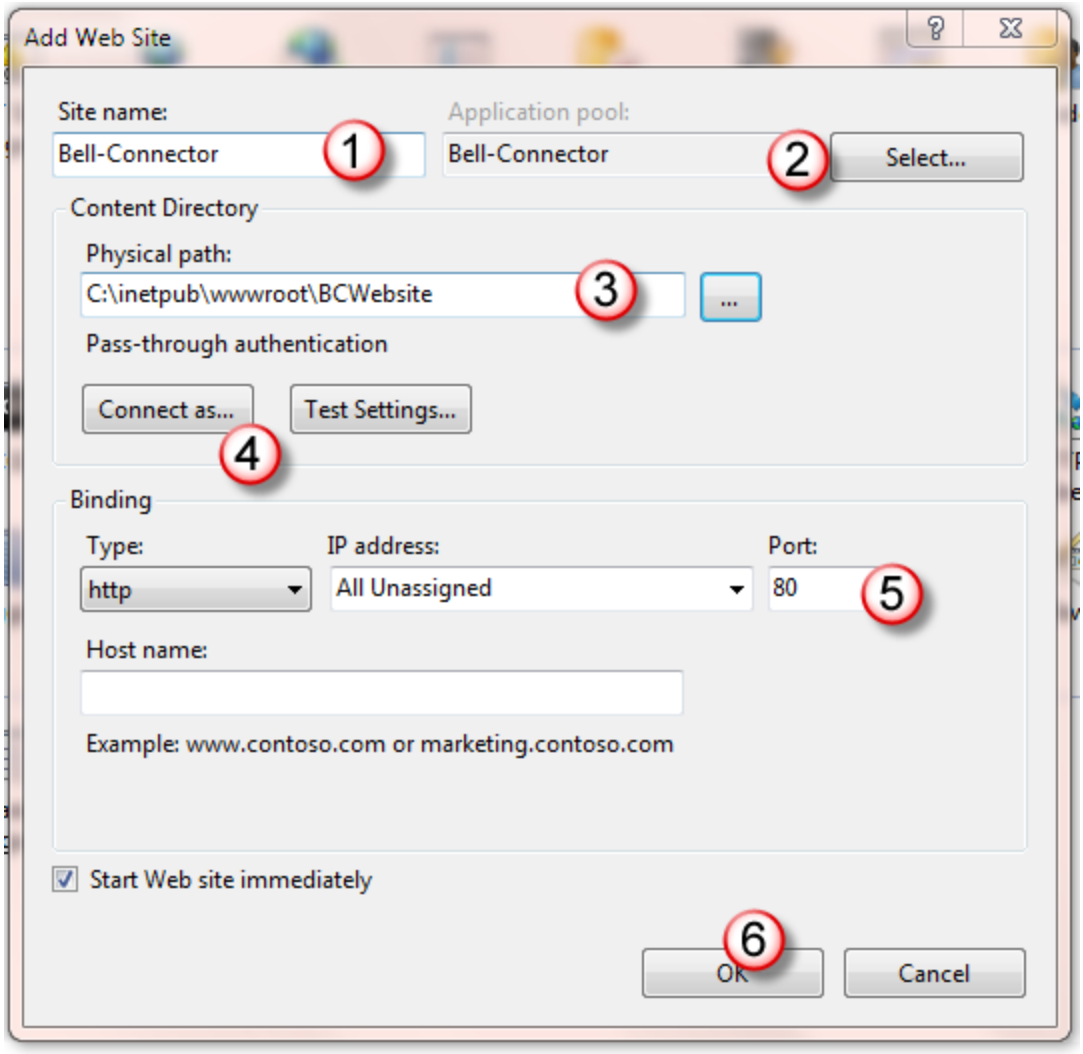
On a Windows Workstation, under Start > Control Panel > Administrative Features you will find the IIS Management Console. In a Windows Server environment this can be reached through the server manager. Alternately you can search for InetMgr.exe in the start menu search box.

When you select this, you will see:



Creating the MilramX Website

In the IIS Manager, right-click on Sites (1) above and select Add Web Site. This will bring up the screen shown below:



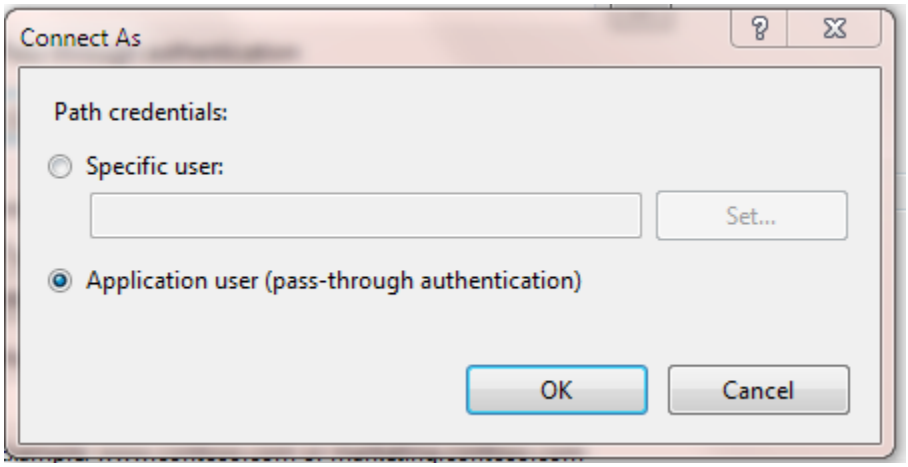
Enter the site same (1) and IIS should automatically create an application pool of the same name. Please note that the automatic creation of the Application Pool is a feature of IIS 7.5. In IIS 7.0, you will need to manually create the application pool first by right clicking on Applications Pools in the left frame and selecting Add Application Pool before creating the Website. Then you can select (2) the application pool to use.

Enter or select the physical path (3) where the website files are installed.

Assign a port number through which this website will be accessed. If MilramX is the only website on this computer then use Port 80. If other websites are already installed on this Windows computer then choose a port number like 8099 that will allow this website to be accessed as http://localhost:8099/ on the Windows computer on which it is installed or by a URL such as http://BCWorkstation:8099/ from another computer on the network (provided that you allow port 8099 as an inbound TCP input through the Windows Firewall).

If Port 80 is the used then you can simply use a URL such as http://localhost/ as Port 80 is assumed by default.

The Connect As (5) should simply default to the Application User, which by default is IUSR, which is a member of IIS_IUSR special group of website users.

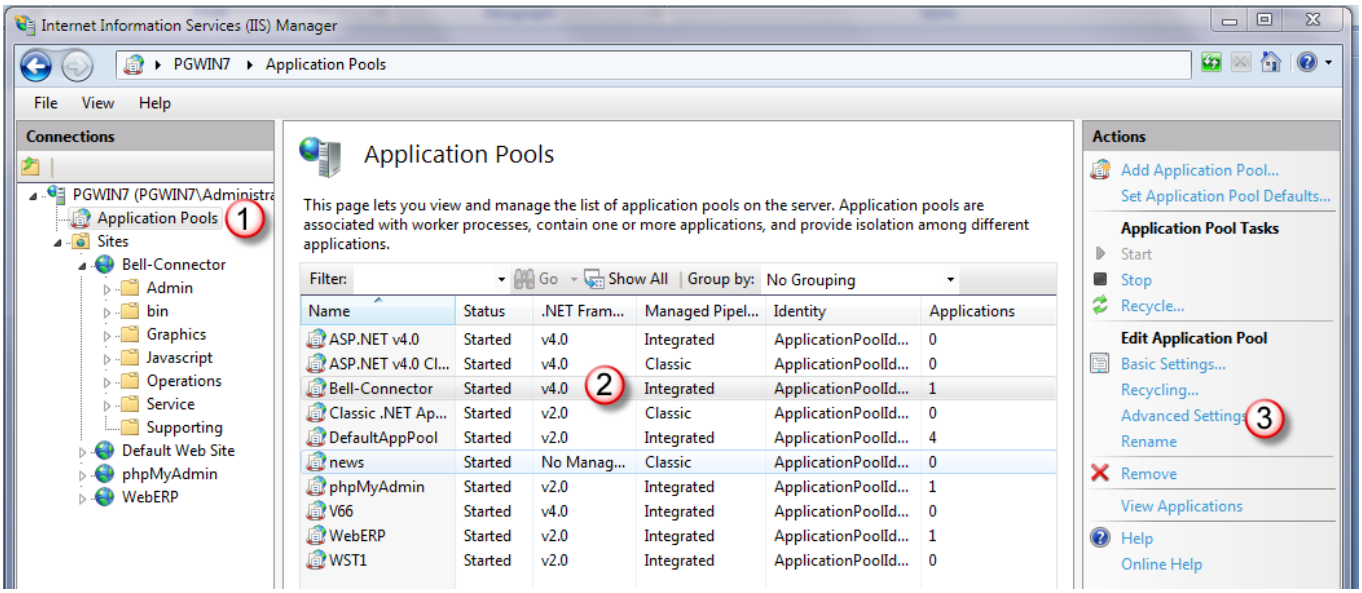


This will work fine for MilramX when installed as described here. If you need to give special access privileges to the location where the website is stored then you can create a special user, give that user special privileges, and select that user here.

Then select [OK] (6) to create the website and its Application Pool (a dynamic set of web pages and the pool of processes and their supporting data which are used to respond to incoming requests from web-browsers).

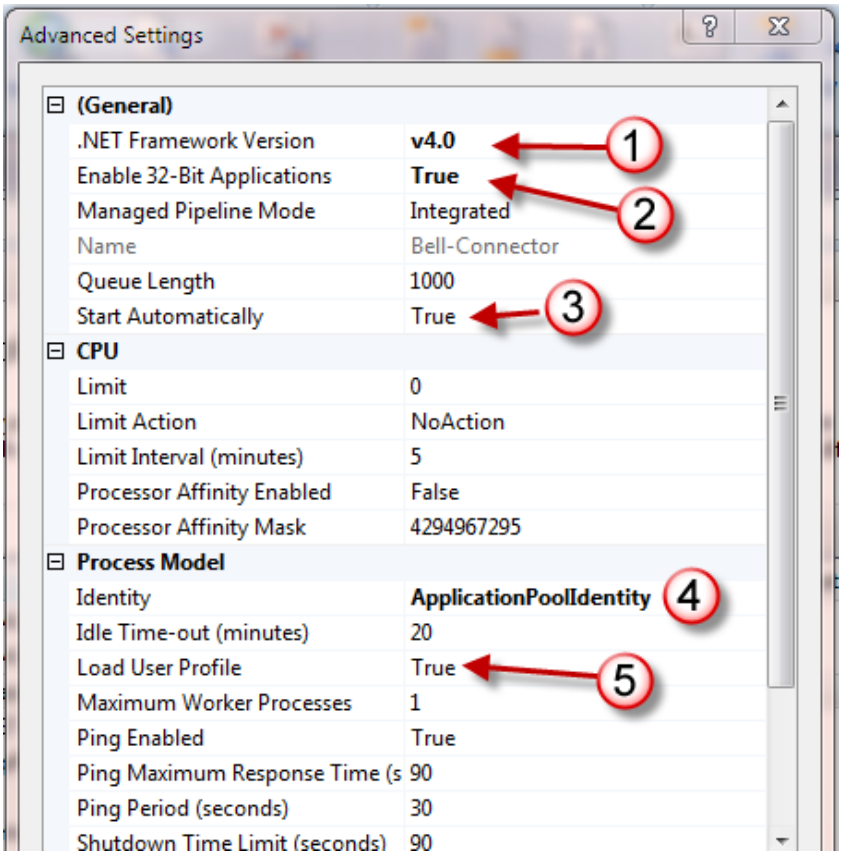
Assign a Host name if you want to assign more than one host name to one computer that uses a single IP address. If you specify a host name, clients must use the host name instead of the IP address and optionally a port to access the Web site. This requires setup of DNS routing on the client network, which is outside of the scope of this manual.

Setting up the Application Pool for the MilramX Website



First click on Application Pools (1), as shown above, then select the MilramX application pool (2) and then click on Advanced Settings (3). This will bring up the following screen:

On this screen:



Select v4.0 (or higher) of the .Net Framework Version (1).

Select Enable 32 Bit Applications (2) to be True.

Set Start Automatically (3) to be True

Set the Identity to Application Pool Identity (which translates to our old friend IUSR) (4). You can also use the special user that you created for website access but, if you follow the installation instructions given here this should not be necessary.

Set Load User Profile (5) to be True.

Setting Up URL Routing for MilramX

In its most simplistic form the URL can be something like 192.168.0.105, where this is an example of the local network IP address of the server on which the MilramX website runs. By default this routes to the default website on the server, which can be setup as the MilramX website.

You can also setup the URL to get to the website as something like 192.168.0.105:8099 and then map port 8099 to the MilramX website, when you setup the IIS website parameters for MilramX.

If you are running the web-browser on the same computer on which the MilramX website is setup then you can refer to the website by http://localhost:8099/ or even http://localhost/ if the MilramX website is mapped to port 80. Alternately, provided inbound access is granted for the website port through the firewall, then an address such as http://BellConnectorWorkStation:8099/ can be used, where BellConnectorWorkstation is the name of the computer on which the MilramX website is installed.

If you have a DNS name server on your network, you can set it up to route requests for BellConnector.ServerName.com to your BellConnector host server (such as 192.168.0.105 in our example). This will make it easier for users as they will not have to remember the numerical IP address. If you are also running the BellHawk software then you will need to setup a separate website URL for this website, such as BellHawk.ServerName.com, so using DNS routing makes it easier.

You may also have to put the routing information into the /etc/localhosts/ file on each computer if you are running a Workgroup network that is not under Domain control.

If MilramX is to be accessed over the Internet then you can register an external URL name for your webserver (such as ServerName.com) with an ICANN registrar and then have a URL such as MilramX.ServerName.com routed to the MilramX website from anywhere in the world. This can be used to give access to the MilramX website for IT staff from anywhere they have an Internet connection.

If you allow external access then you need to setup your Internet firewall to route the external HTTP (or HTTPS) and SOAP/XML packets directly to the MilramX server in a simple network setup or to your DNS server in a more complex network setup.

License Files

As part of the installation you will be given a license file called BC.Lex for the website. This should be copied into the main folder of the website. This file specifies the name of the computer on which the MilramX website is licensed to run.

Setting up the Initialization File for the Website

```
;Initialization File for Bell-Connector Website
[Control]
XMLfile = Control.XML
Type = MSSQL
; Replace the values on the next 4 lines with your BellHawk BHCTL SQL server database/login settings
Server = PGWIN7\SQLSERVER2008
Database = BellConnector
UserID = *******
Password = ******
LogFolderName = C:\Program Files (x86)\BellHawk Systems Corp\BHQBI\LogFiles\
DebugLevel = 0

[Adaptors]
; Adaptors should be a comma-separated list of adaptors beside Control that DEXEL will access.
Adaptors = BellHawk,QuickBooks

[BellHawk]
XMLFile = BHMeta_v6.40.XML
Type = MSSQL
Server = PGWIN7\SQLSERVER2008
Database = V66
UserID = *******
Password = *********

[QuickBooks]
XMLFile = QBMeta.XML
DSN-Name = QuickBooks Data
Type = QODBC
```

The file BCWesbite.ini is provided in the /Supporting folder of the website files. It should be copied to the root directory of the website and then modified. This file is used to setup all the database connections for the MilramX website.

This initialization file contains a number of sections:

1. The [Control] section (1). In this:
    a. The XMLFile provides the meta-data descriptions for the Control database. It enables the use of DEXEL to import setup data, such as the list of DTOs, into the Control database, via Excel spreadsheets. The metadata file needs to be
    b. Type=MSSQL specifies the use of the native SQL Server driver to access the Control database. Please do not change this.
    c. The Server, Database, User ID and Password entries provide the information that the BC Website uses to directly access the SQL Server Control database.
    d. The LogFolderName is the default path name that the Transfer function(s) will use as the folder into which they will write their daily log files. This can be over-ridden through the website. Note that this is different from the folder that the BC Website uses to write its daily log files, which is the LogFiles folder in the BC Website files directory.
    e. The DebugLevel sets the level of data logging that the website makes to its own logfile. A Debug level of 0 only means that errors are logged to the daily logfile. A Debug level of 1 means that warning and alerts are logged and a Debug level of 2 means that details of most website activities are logged. Normally this should be set to 0 unless you are trying to track down a specific problem. Errors and messages are

also logged to the console unless Debug is set to -1 (log errors) or -2 (log all messages) when console output is suppressed.

2. The [Adaptors] section (2) lists the database adaptors that are accessible through the BC Website DEXEL functionality, in addition to the Control adaptor, for use in developing metadata definitions for interfacing to different databases. Each Adaptor name must have a corresponding section in the initialization files for the BC website and for each transfer function that is activated using the website. When setting up each DTO, using the BC Website, the Adaptor names are captured and stored. Subsequently, the Launcher will pass these Adaptor names to the transfer function to be used by the selected DTO for accessing the source and destination databases. There can be as many Adaptor names/sections as needed.

3. The [BellHawk] section (3) specifies:

    a. The XMLFile which sets the metadata to be used to access BellHawk. The named metadata file needs to be copied into the root directory of the Website so that it can be accessed by the DEXEL function. This should be the XML metadata file that is used by the version of BellHawk that you are using. This is referenced in the initialization file for your BellHawk website and is found in the root directory of your website.

    b. If the Type is MSSQL then this is a DSN-less ODBC connection to a Microsoft SQL Server database (not made through a pre-defined DSN connection). The following data specifies the connection data for that database.

    c. If the Type is BellHawkWS then this is a remote connection to BellHawk using its SOAP/XML web-services interface – see section below on Adaptors.

4. The [QuickBooks] section (4) is an example of an ODBC connection to a database, through a pre-defined ODBC DSN, which specifies:

    a. The XMLFile which sets the metadata to be used to access the database. The named metadata file needs to be copied into the root directory of the Website so that it can be accessed by the DEXEL function. This can be found amongst the Setup files for your transfer function.

    b. The DSN-Name which must correspond to an ODBC name in a pre-defined DSN or setup in the ODBC driver at installation time. Here all the connection information is contained in the Data Source Name (DSN) setup.

    c. Type=QODBC tells the system that this is a special DSN for QuickBooks, which requires special ODBC syntax that is not part of the ODBC 1992 “standard”.

Please note that Type=Oracle changes the generated ODBC/SQL syntax to be compatible with the Oracle ODBC driver. You still have to setup a DSN ODBC connection to the Oracle driver and select the appropriate version of the driver.

Please note that the first line of the initialization file needs to be blank or a comment (starting with a semicolon ;).

Setting up Adaptors

Please follow the above examples, where applicable, for BellHawk and QuickBooks.

If BellHawk is not accessible by a direct connection (the most efficient) then you can use the web-services Adaptor description:

```
[BellHawk]
XMLFile= BHMeta_v6.40.XML
Type = BellHawkWS
SVC_URL = http://BellHawkSystemURL/
UserID = *****
Password = *****
```

You can also use Aliases when you have more than one BellHawk databases (such as for different plants):

```
[MySecondPlant]
Alias = BellHawk
Type = MSSQL
XMLFile = BHMeta_v6.40.XML
Server = LOCALHOST\SQLExpress2008
Database = BellHawk_2
UserID = *****
Password = *****
```

In this case, the DTO code in the transfer function will be told that the Adaptor is called BellHawk, when it is run, but there can be multiple adaptor sections, one for each plant database. In this way a single DTO can be used to transfer data from multiple BellHawk databases into a single ERP or accounting system.

Note that the Adaptor name is in the XML Metadata file and the name the alias refers to (such as BellHawk in the above example) must be the Adaptor name used in the metadata file, such as BellHawk in the above example.

Adding Adaptors to Website

In order for the DEXEL functionality to work, you need to include the DLL files for each of the Interfaces (such as MSSQL) in the /Bin directory and the XML Metadata files for each of the Adaptors in use in the main website folder.

Testing the MilramX Website

If you are running on a PC that is different from that on which installed the MilramX website, first test that you can Ping the website. If you cannot, then you may well have firewall issues.

Use a PC and enter the URL that you setup for the MilramX website. You should see the pre-login MilramX splash screen, shown here.

If not then you will probably get an error message from IIS that it cannot find your MilramX website. Alternately you may get a rather imposing but badly structured long message from ASP.net which indicates that the specified databases or some other component of MilramX cannot be found or are inaccessible.

For details of how to use the BC website, please see the MilramX User's Manual.

Installing the Launcher

You will first need to create a user account on the Workstation that will be used for the launcher, as previously described.

Normally the Launcher will be run as a user program in a user account that you will login into, start the launcher running, and then switch accounts, leaving the launcher running in the background.

We do this, rather than making the Launcher an operating system service, so that the BHxxI.exe process has all the access privileges of the user account in which the Launcher runs. If we make the Launcher run as a service then the BHxxI.exe process has very limited access privileges (such as not being able to communicate with other computers).

While this arrangement does require that the Launcher be manually restarted whenever the Workstation is rebooted, it avoids some incredibly convoluted system configuration activities that are highly domain dependent and so requires the intervention of a very knowledgeable IT security specialist.

The launcher comes as a zip file that contains BHLauncher.msi and Setup.exe. Unzip this file into a temporary folder and run Setup.exe. This will install the Launcher files in:

```
C:\Program Files (x86)\BellHawk Systems Corp\BHLauncher\
```

This will also install an icon  on the desktop of the user's account in which it is installed.

This is used to run the launcher.

Configuring the Launcher

Inside the BHLuncher folder you will find a file BHLauncher.ini:

```
[Control]
; Replace the values on the next 4 lines with your BellHawk BHCTL SQL server database/login settings
Type = MSSQL
Server = PGWIN7\SQLSERVER2008
Database = BellConnector
UserID = V66DB
Password = *******
SingleProcessMode = True
DebugLevel = 0

[BHQBI]
PATH = "C:\Program Files (x86)\BellHawk Systems Corp\BHQBI\BHQBI.exe"
```

(1) (2) (3) (4)

This [Control] section is similar to that for the MilramX website. It defines the Type of connection and the components of the database connection string (1).

Normally the launcher runs one transfer process at a time and waits until it has completed before running the next. This is the default Single Process Mode (2). If SingleProcessMode is set to False then the Launcher uses a launch-and-forget policy and does not monitor the processes it launches.

The launcher creates a log file in its local directory folder in which it writes any errors it encounters. If the DebugLevel (3) is set to 0 (the default) then only errors are logged. If it is set to 1 then additional information is logged to aid in diagnosing systems problems, such as not being able to contact the control website. If DebugLevel is set to 2 then diagnostic messages are logged. These error, warning and diagnostic messages also displayed in a scrolling console screen of the workstation if the DebugLevel is a positive number. If it is negative then all console output is suppressed. If it is -1 then just errors and warnings are logged in the log file. If it is -2 then diagnostic messages are logged to the log file.

You also need to setup the path to the BHxxI.exe transfer function as shown in (4). If there are multiple transfer functions then each will have its own [BHxxI] section.

Please note that the first line of the launcher's .ini file needs to be blank. Comment lines can begin with a semi-colon.

Installing the Transfer Function

The provided installation file, when executed, will install the transfer function files in:

```
C:\Program Files (x86)\BellHawk Systems Corp\BCTransfer\
```

Please note that, if there are multiple transfer functions, each will have its own install files and its own installation directory.

You will need to install the BC.Lex licensing file that you received in each BCTransfer folder. This licenses the transfer function to run on a specific named computer.

Configuring the Transfer Function

Each transfer function needs an initialization file in its working folder. This can contain all the same setup elements as the website but usually it contains a single [Redirect] section

```
[Redirect]
Path = "C:\inetpub\wwwroot\HampdenBC.bellhawk3.com\BCWebsite.ini"
```

This causes the Transfer Function to use the initialization data, as well as the XML metadata files, from the BC Websites main folder. This saves copying these over if a change is made in the metadata and saved to the BC website.

If the BC website is installed on a different computer then you might need to change a separate initialization file with settings appropriate to that computer. Also you many need to setup local ODBC drivers on the computer on which the transfer function is running and to change firewall settings allow access to databases in a similar manner described for the BC website.

Please note that the first line in an initialization file must be empty, as it is ignored.

