Universal Report Server
Installation Guide

VERSA REPORTS

Dunwoody, GA (USA) ♦ www.versareports.com

Copyright Notice

For information, please contact:

VersaReports, LLC.
Dunwoody, Georgia (USA)

Email: support@versareports.com
Web: *www.versareports.com*

Document Version Information

Document Name	Universal Report Server Installation Guide
Document Edition, Date	Edition 2, April 2010
For Release:	Universal Report Server Version 1.3.0

Table of Contents

Chapter 1
About This Guide

1.1 Purpose

This Installation Guide provides specific instructions to the person or persons who will be installing the Universal Report Server (URS) software for the first time, or who will be upgrading their existing Universal Report Server version to the current version.

1.2 Audience

This Guide is intended for the IT professional who will be installing URS and who is knowledgeable and qualified to make changes to certain installation files.

1.3 How to Use This Guide

This Installation Guide covers installation procedures for multiple site configurations that can be accommodated by the Universal Report Server. The Introduction to Universal Report Server chapter provides a description of the Universal Report Server product, lists the system requirements, and explains that there are pre- and post- installation steps you must undertake in addition to the software installation from the disk.

The pre-installation instructions are the same for all configurations. The post-installation instructions depend on the security infrastructure you are using and what report designers you have installed. It is important to follow the post-installation instructions that are specific to your configuration and, generally speaking, ignore those instructions that are not for your security infrastructure or report designer(s).

If you are upgrading your version of URS to the current release, refer to Appendix C on page 55 for important steps you must take that are specific to an upgrade installation.

Chapter 2
Introduction to Universal Report Server

2.1 Product Overview

The Universal Report Server is the first web report server to support nearly ALL report designers, including SAP Crystal Reports, DevExpress XtraReports, GrapeCity/Data Dynamics ActiveReports, Telerik Reporting, and others.

The Universal Report Server provides the following:

- An easy-to-use web interface
- Support for SAP Crystal Reports, DevExpress XtraReports, GrapeCity/Data Dynamics ActiveReports, and Telerik Reporting, including native report viewing support and support for parameters. Developer licenses for report designers are not included with Universal Report Server and you are encouraged to check with those vendors for licensing terms, pricing, and resellers.
- API support for integrating your favorite report designer if it's not already included.
- Support for ActiveDirectory (domains), Windows NT security (non-domains), Forms authentication, and no authentication.
- API support for integrating with your corporate security infrastructure if it's not already included.
- Report distribution to users within your network (via the web interface) or to mobile or remote users who rely on receiving their reports via e-mail.
- Report archival, including default and personalized retention rules.

Important Disclaimer: Universal Report Server operates with a number of third-party products. Those products are mentioned in this document or other marketing documents from VersaReports. However, those vendors own the copyrights and/or trademarks for their products. VersaReports documentation only mentions those products but does not endorse or imply any ownership in those products. Likewise, those third-party vendors do not endorse Universal Report Server, nor does VersaReports LLC -- the creator of Universal Report Server -- endorse any specific report designer or third-party product.

2.2 System Requirements

Universal Report Server is a Microsoft Windows Server-based application. Users and administrators interact with the software via a browser; no Universal Report Server software is loaded on the user's workstation to interact with this product.

2.2.1 Server Requirements

- Microsoft Windows Server 2003 SP2 or later (including Server 2008) with the latest updates installed.
- Microsoft SQL Server 2005 or later (can be on another server, if desired). Minimum of 100 MB is required for the SQL Server database for Universal Report Server.
- Microsoft .NET 2.0 framework (with the latest service packs and updates installed).
- At least 2 GB of RAM.

- For creating a Report Connector for the currently supported report designers, or to create a Report Connector for a report designer that is not included with Universal Report Server, you need the following:
  - Microsoft Visual Studio 2005 or later
  - Developer license for your report designer

  Note: The Report Connector is a compiled piece of software that provides an interface between the Universal Report Server software and the report designer's report engine software. Universal Report Server includes several example Report Connectors and instructions for creating and installing the Report Connector DLL.

Note: Most report designer packages support only 32-bit interfaces, so Universal Report Server is designed for 32-bit operation. However, as with all 32-bit products, you can run Universal Report Server on a server running a 64-bit operating system.

2.2.2 Workstation Requirements

Users and report administrators work with Universal Report Server via one of several popular browsers:

- Firefox 2.0+
- Internet Explorer 7.0+ (no support for Mac workstations)
- Safari 3.0+
- Google Chrome 1.0+

For all browser types, JavaScript and cookies must be enabled. For Internet Explorer, browser scripting must be enabled.

In addition to the browser requirements, you may also need one or more of the following programs installed locally on your PC to view the reports.

- Microsoft Excel (for spreadsheet reports)
- Adobe Acrobat (for PDF reports)
- Microsoft Word (for RTF and Word-based reports)

2.3 Installation Overview

Installing Universal Report Server requires some pre- and post- installation steps in addition to running the install program itself. The pre-installation steps are the same no matter what your operating system, user authentication method, or report designer you will be using.

The post-installation steps are dependent on your user authentication method, the report designer(s) you have installed, and to a somewhat lesser degree, the version of Windows Server you have. Once the installation program has been successfully run, you will need to edit the web.config file to adjust for your authentication type and to adjust certain settings, copy the correct global.asax file to the *InstallationDirectory*\web folder, and complete the configuration instructions for the specific report designer(s) you will be using.

The pre-installation steps, software installation steps, and post-installation steps are covered in separate chapters.

Chapter 3
Pre-Installation Steps for Universal Report Server

As with most server-based software, installation requires a series of steps to be completed.

3.1 Review System Requirements

Ensure your server meets or exceeds the Server Requirements mentioned in the previous chapter and that your server is properly configured within your corporate infrastructure.

The important system requirements are:

- Windows 2003 SP 2 or later (with latest updates)
- 2 GB of RAM
- A large enough disk to hold the reports your users want to run and save
- Microsoft .NET Framework version 2.0 installed (with latest service packs)

To compile a Report Connector on the server, you will also need:

- Microsoft Visual Studio 2005 or later
- Developer license for your report designer

3.2 Create a SQL Server Database

3.2.1 Identify or install the SQL Server you want to use for URS

The SQL Server database can be on the same server as Universal Report Server or it can be on a different server. Universal Report Server uses the database to store both information about the reports you want to run and the report contents once the report is run.

3.2.2 Determine the amount of disk space required

The amount of disk space required for the Universal Report Server database is based on how many reports you plan to use with the report server and how long you intend for users to retain reports that have been run. You may choose to store the actual archived reports within the SQL Server or on a local or networked disk. If you archive your reports within SQL Server, make sure to have adequate disk space on your SQL Server system as well as a way to monitor disk space usage on a regular basis. If you archive your reports to local/networked disk space, the SQL Server disk usage will be less than 100 MB.

3.2.3 Configure your SQL Server to support Mixed SQL

Using the SQL Server Management Studio, make sure that your SQL Server has been configured to support Mixed SQL security (as opposed to Integrated-only security, which requires a windows login to connect to the server).

3.2.4 Create a Database for Universal Report Server

Create a database for Universal Report Server. No special settings are needed for this database. The structure for this database will be created when you install Universal Report Server on your server system.

3.2.5 Create a SQL Login

Create a SQL login that has db_owner permission on the database and is mapped to the dbo schema.

3.2.6 Create an ODBC Connection

On the server where URS is to be installed, create (and test) an ODBC connection to the SQL Server database you just created for Universal Report Server. Use the credentials for the login you created as the db_owner of the database. If you can successfully complete this step, you are ready to do the installation. This ODBC connection will not be used by Universal Report Server and can be deleted after testing is successful.

Chapter 4
Installing Universal Report Server on your Host Server

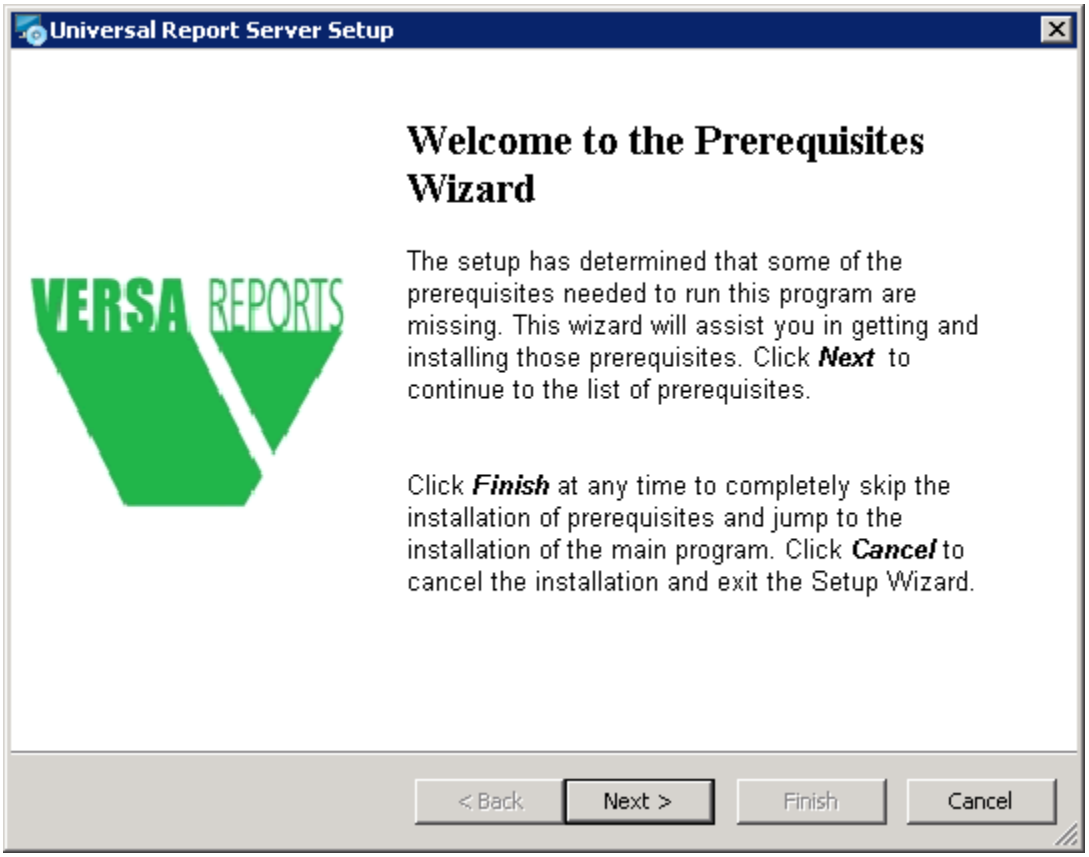
4.1 Overview

The Universal Report Server (URS) installation is performed via a wizard that requires just a few simple steps. The installation first checks for any missing programs that are needed for Universal Report Server. If there are missing programs, you are led through their installation by the **Prerequisites Wizard**. When that is finished, you will start the **Universal Report Server Setup Wizard**. You will be asked to read and accept the license agreement, choose the SQL server you are going to use, log in to the SQL server and choose the database you previously created, and select a folder location for the URS files.

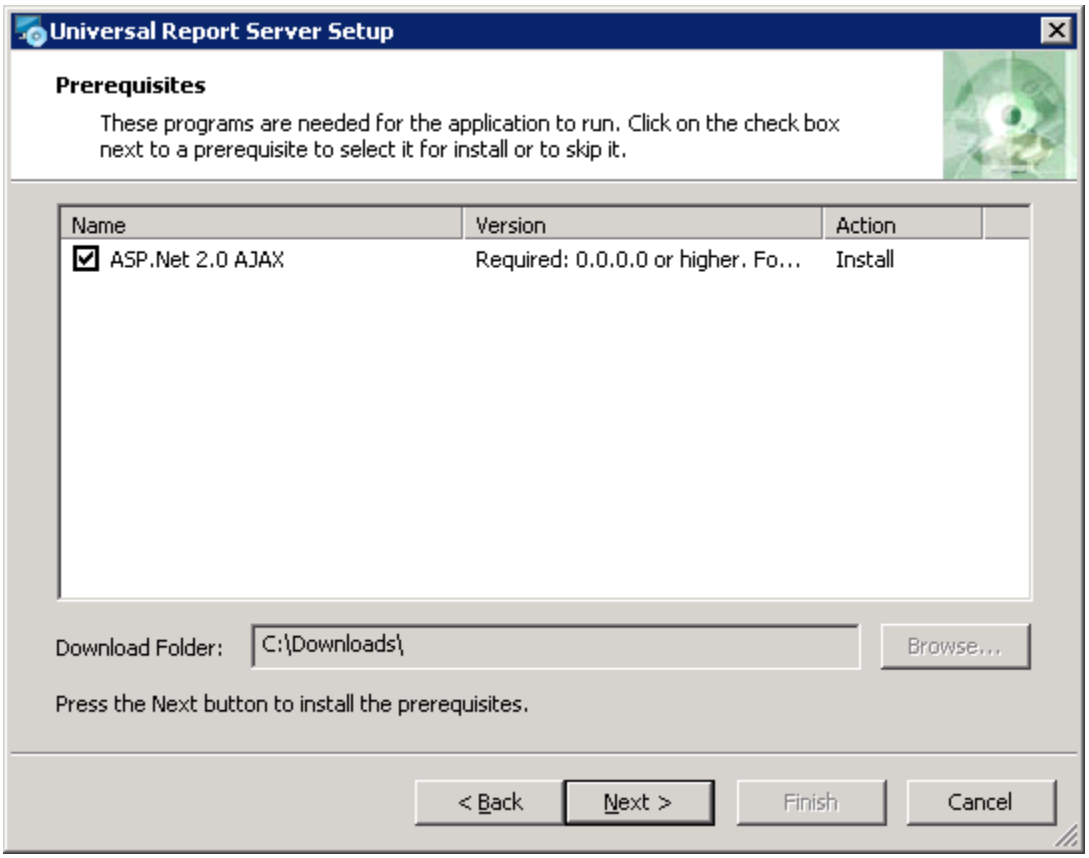
4.2 Installing Universal Report Server

1. On the server where you plan to load Universal Report Server, login as an administrator.
2. Identify the installation file on the Install CD and double-click it.

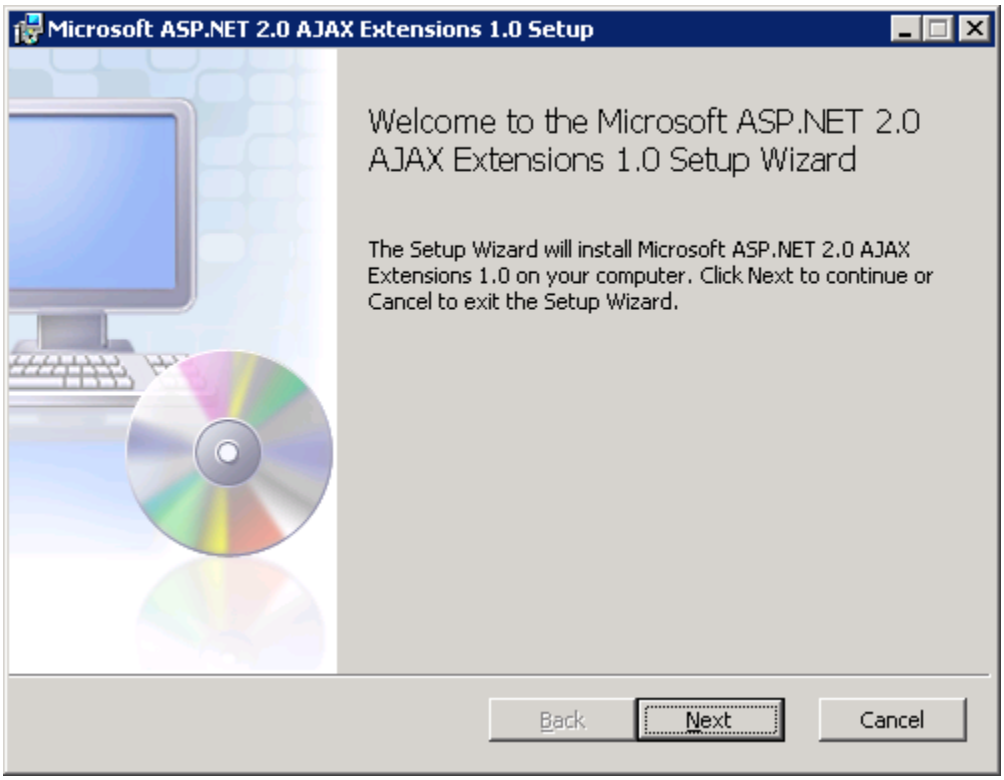
   The installation program checks to see if your system is missing any necessary programs. If there are any “prerequisites” found to be missing, the **Welcome to the Prerequisites Wizard** dialog box displays; if all prerequisites are found on your server, this dialog will not display and you can continue with Step 3.



2a. Click **Next** to see the list of prerequisites. The **Prerequisites** dialog displays.

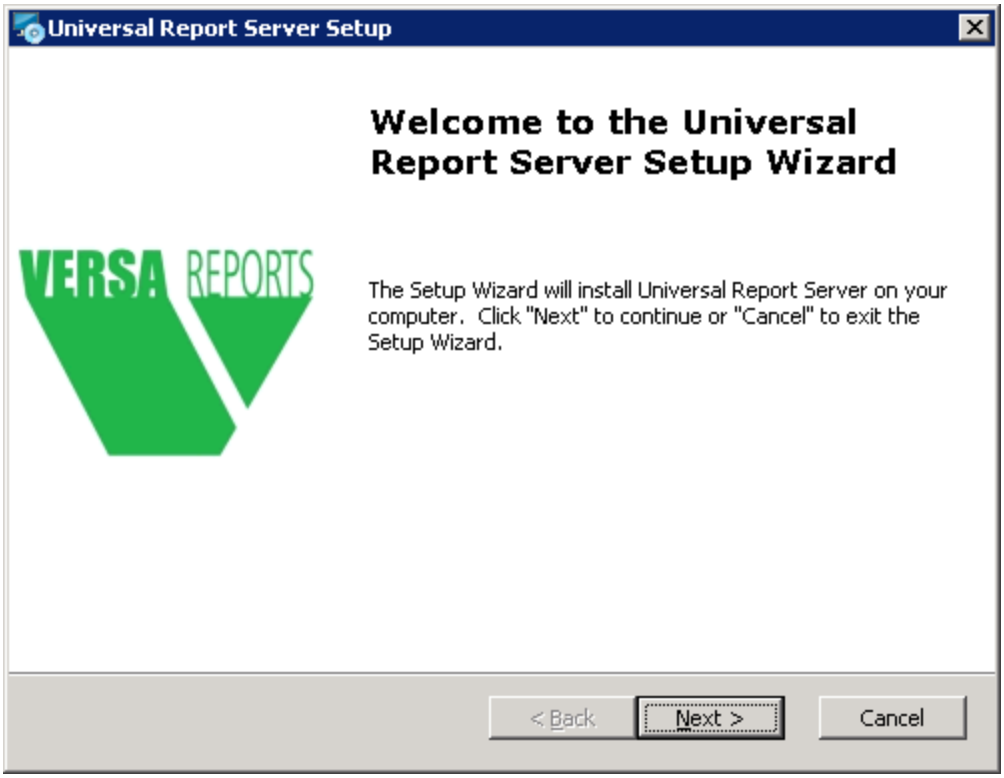


2b. Select the checkbox next to those programs that need to be installed and then click **Next**. The **Setup Wizard** for the program displays.

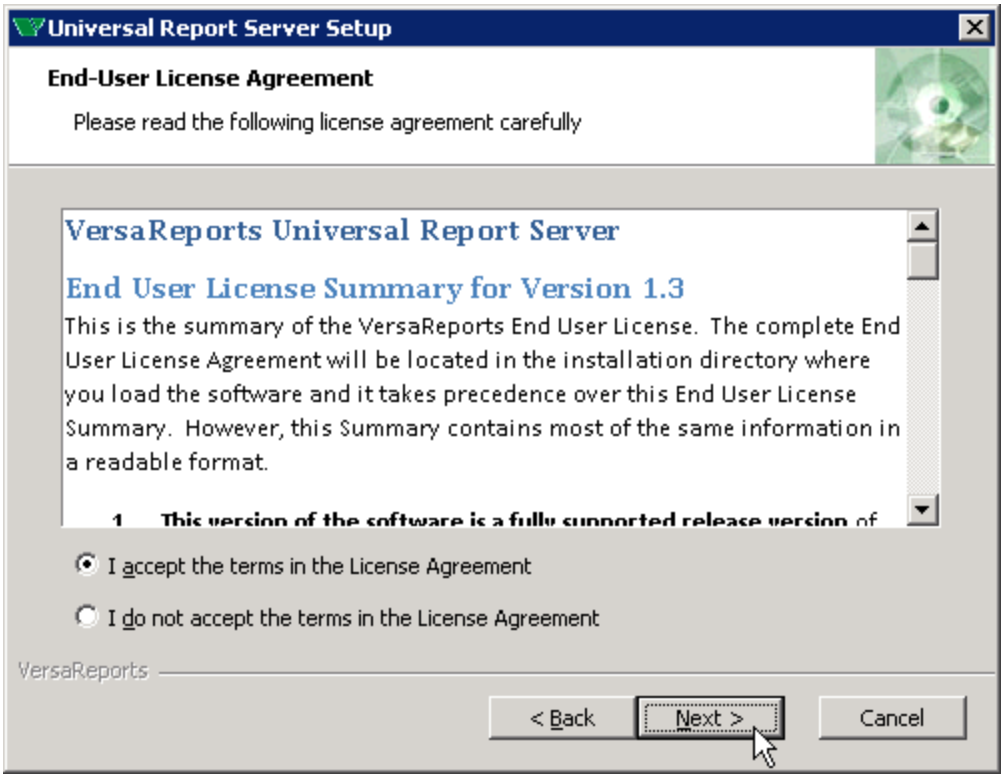


2c. Click **Next** to start the Setup Wizard that installs the Microsoft ASP.NET software. Accept the License Agreement and proceed through the wizard.

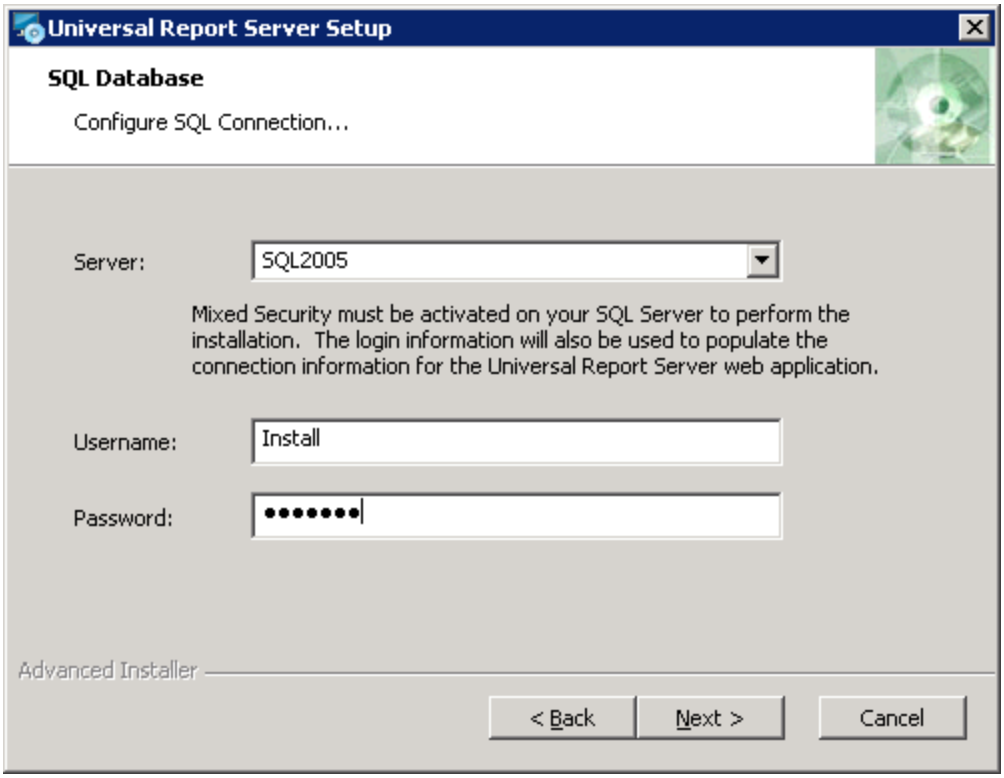
When finished installing all prerequisite programs, the **Welcome to the Universal Report Server Setup Wizard** dialog box displays.



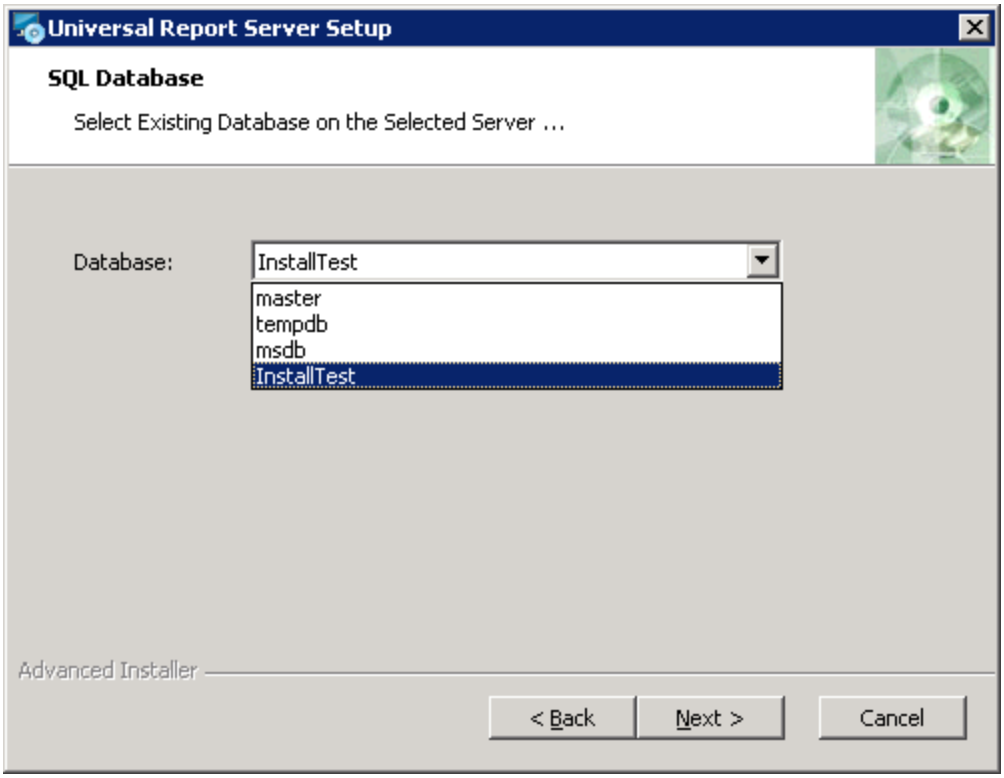
3. Click the **Next** button. The **End-User License Agreement** dialog box displays.



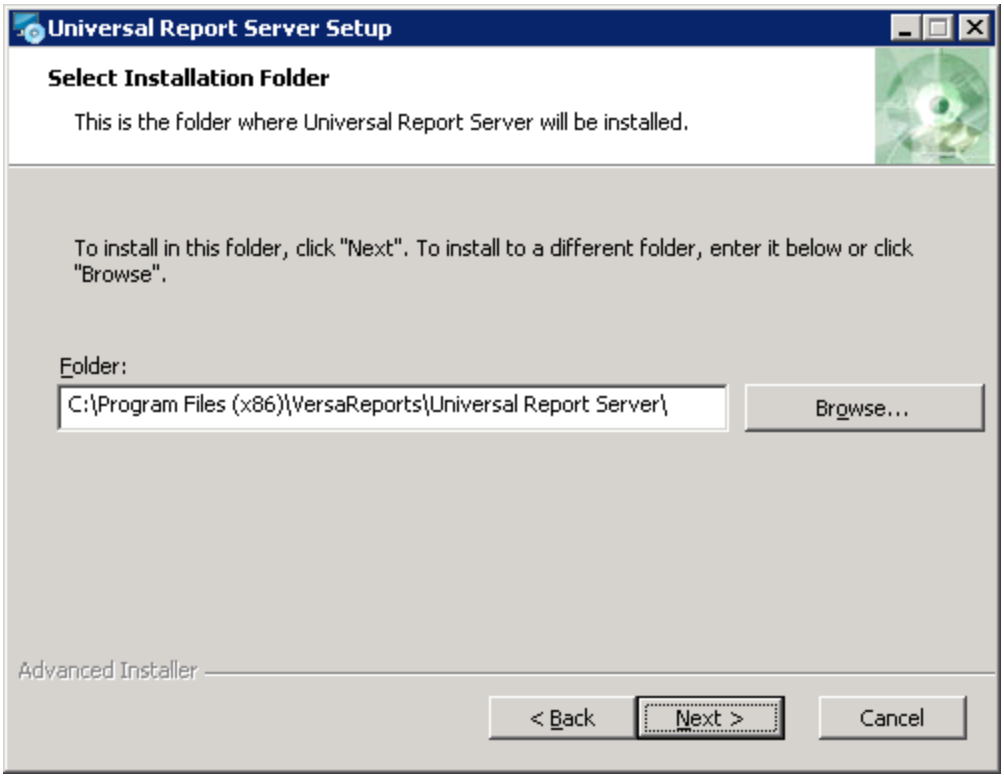
4. Read the License Agreement, click **I accept the terms in the License Agreement**, and then click **Next**. The **SQL Database (Configure SQL Connection…)** dialog box displays. Please note that although a summary of the license agreement is displayed during installation, you are bound by the full license agreement available for viewing in the file *InstallationDirectory*\Universal Report Server EULA.pdf that is loaded as part of the installation.



5. Select the SQL Server you are going to use with URS. This list is populated with the servers that were discovered on your network. If the SQL Server that you want to use does not display in this **Server** list, you can enter its name in the box. Enter the login name and password you previously created for the SQL Server database. Click **Next** when you are ready to continue. The **SQL Database (Select Existing Database on the Selected Server…)** dialog box displays.



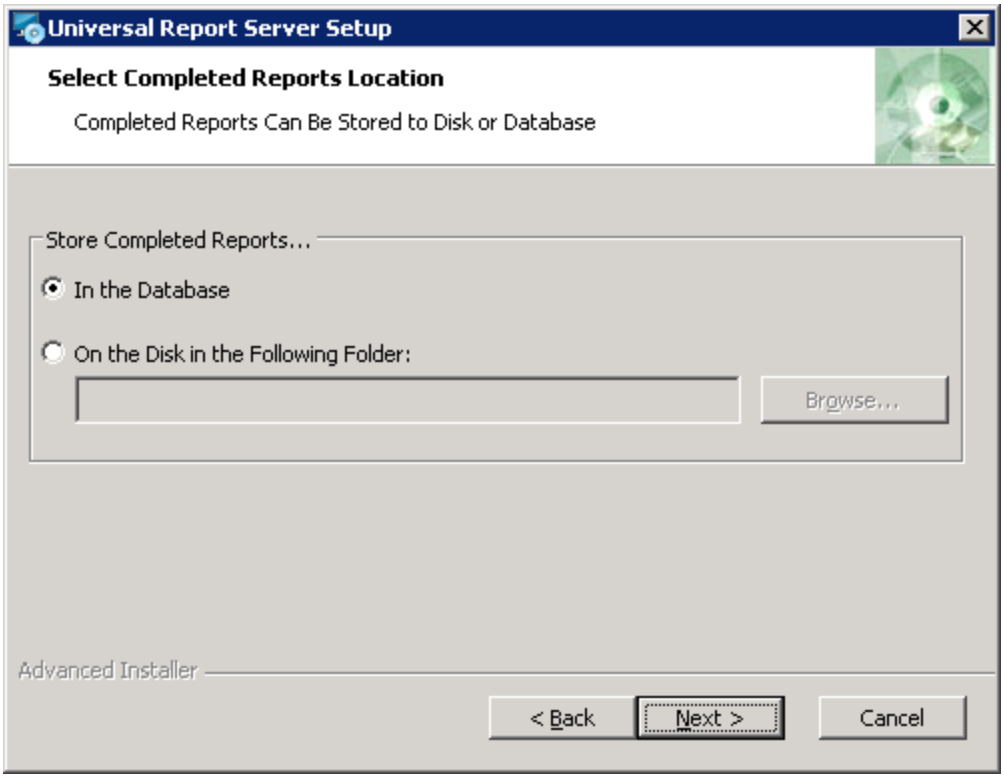
6. From the **Database** list, select the database name you created as part of the pre-installation steps. Click **Next**. The **Select Installation Folder** dialog box displays.



7. A folder name is suggested in the **Folder** box. If you wish to install in a different location or want to use a different folder name, make the change to the displayed path, or click the **Browse** button and select a new path.

   Note: The path you choose here will be referred to as the *InstallationDirectory* throughout this manual.

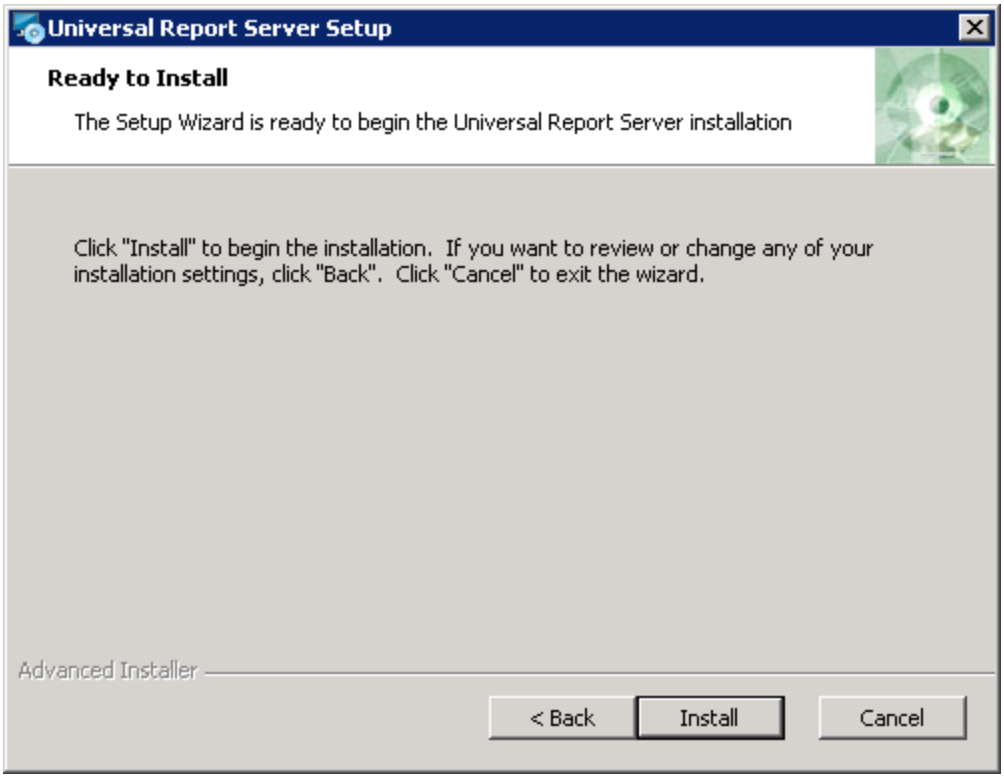
   Click **Next** when you are ready to continue. The **Select Completed Folders Location** dialog box displays.



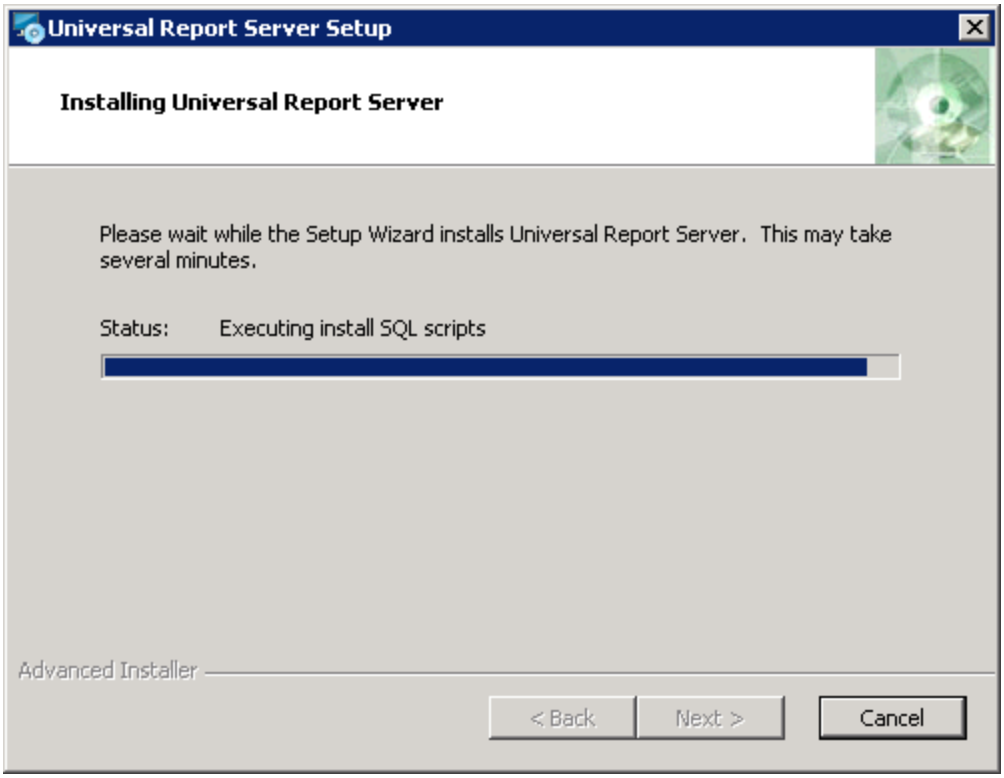
8. Universal Report Server gives you the option of storing completed reports in the database or on disk in a folder of your choosing.

   - You would choose to store completed reports in the database if you plan to access those files from another server or if you want stronger security than might be available to files on your server's disk.
   - You would choose to store completed reports on disk because files stored on disk are faster to retrieve and easier to backup and restore.

   Select the **Store Completed Reports** option of your choosing. If you select **On the Disk in the Following Folder**, click the **Browse** button and locate/create that folder. Click **Next** when you are ready to continue. The **Ready to Install** dialog box displays.



9. You can click the **Back** button to review and change any of your settings. When you are satisfied with your choices, click **Install**. The **Installing Universal Report Server** dialog box displays showing the progress of the installation.



Note that the installation program configures Internet Information Services (IIS) to create a new Application Pool for Universal Report Server (URS Pool). It also creates the required web application under the Default Web Site.

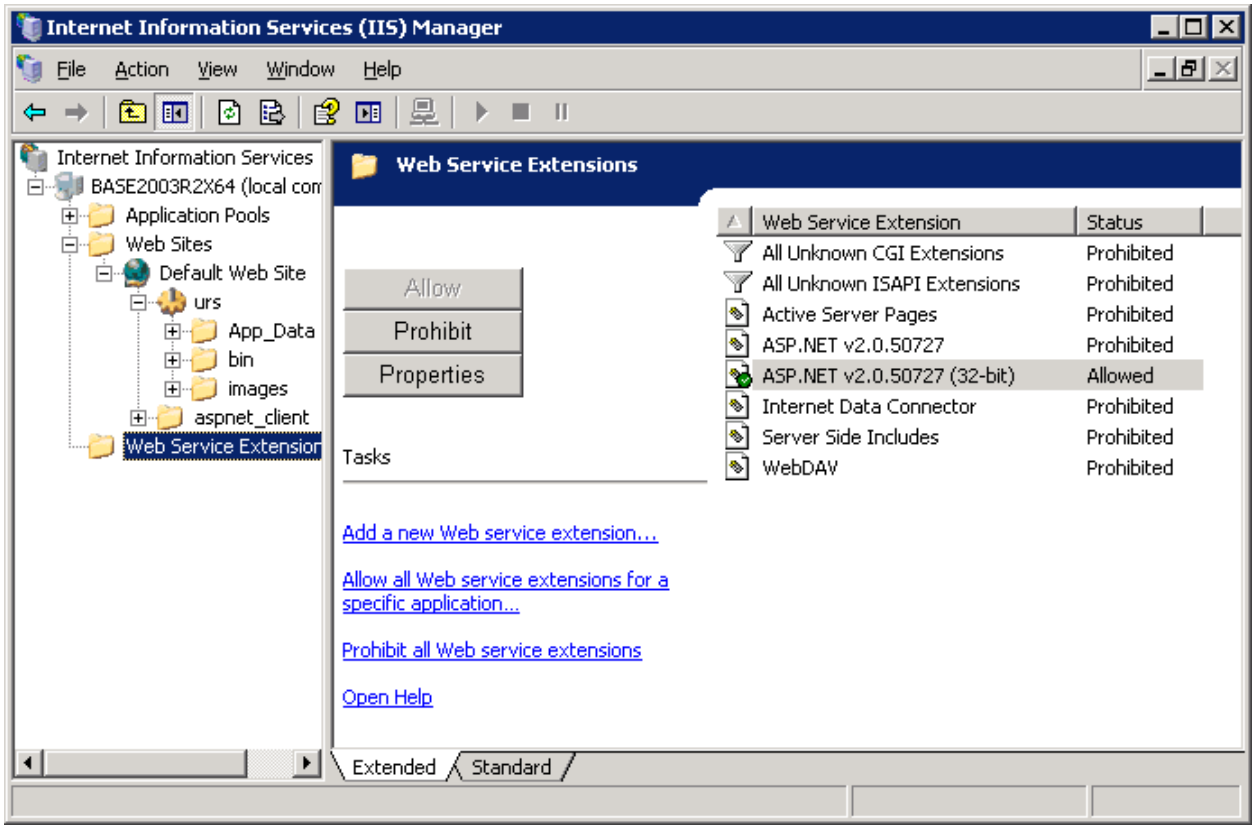
When the Setup Wizard has finished installing URS, the **Completing the Universal Report Server Setup Wizard** dialog box displays.

10. Click the **Finish** button to complete the setup and exit the installation wizard. Universal Report Server has been successfully installed, although to use it you must follow and complete the Post-Installation instructions in the next chapter.

4.3 Platform Considerations

The installation program automatically configures Internet Information Services (IIS) to create a new Application Pool for Universal Report Server (URS Pool) and the required web application under the Default Web Site.

If your platform is **Windows Server 2003 x64 Edition**, however, there is a step you need to perform manually. Universal Report Server is a 32-bit ASP.NET 2.0 application and it is that version of ASP.NET that needs to be allowed in the IIS Manager.



Open Internet Information Services (IIS) and make sure ASP.NET v2.0 for 32 bit is allowed.

For more information, please refer to the http://support.microsoft.com/kb/894435 Microsoft Knowledgebase article for switching between the 32-bit version and the 64-bit version of ASP.NET 2.0 on a 64-bit version of Windows.

Chapter 5
Post-Installation Steps for Universal Report Server

5.1 Overview

After the software is installed, there are some aspects of Universal Report Server that need to be configured to work with your security infrastructure and report designers.

5.2 Edit Settings in the web.config File

The installation of Universal Report Server includes a web.config file located in the *InstallationDirectory*\web folder. Most settings in this file are predefined with default values; however, you will need to verify, add, or change some settings to accommodate the structure and file location within your system.

5.2.1 Values in the appSettings section

Review these keys in the <appSettings> section of the web.config file and make changes as appropriate.

5.2.1.1 ReportTypesXML key

Verify the location of the report_types.xml file. This file is installed in the *InstallationDirectory*\App_Data folder shown in the example. This key should only be updated if the file has been moved.

For example:

```
<add key="ReportTypesXML" value="~\App_Data\report_types.xml"/>
```

5.2.1.2 RootDirectories key

This value is the *server's* directory location for any reports that the report administrators can load. These reports can be RPT files, DLLs, or whatever format of report that the report designer uses with the chosen Report Connector. (See Section 5.6 Report Designer Integration for more information about Report Connectors.)

Make sure the web application has permission to read and list the files in this directory and all subdirectories.

For example:

```
<add key="RootDirectories" value="C:\reports"/>
```

5.2.1.3 ReportStorage key

This key was set according to the choice you made during installation.

If you chose to Store Completed Reports **In The Database**, then this key will not be set, and all completed reports go into the database.

For example:

```
<add key="ReportStorage" value=""/>
```

If you chose to Store Completed Reports **On the Disk in the Following Folder**, then this key will be set showing that folder on the server for storing completed reports.

For example:

```
<add key="ReportStorage" value="c:\VersaReports\CompletedReports"/>
```

Note: Later, after using URS, you may decide you need to change the value for this key. Refer to the table below to see how to implement this change and the effect it may have on viewing the reports.

Reason for change	How to Implement / Effect of Change
You want to move your reports from one disk location to a new location (for example, you are getting a bigger disk or a SAN for them).	The Administrator needs to manually move all the reports from the old location to new location first, and then change the value of this key to the new location. Otherwise the users will not be able to view those reports.
You want to start storing completed reports in the database instead of in the folder where they are currently stored.	Change the value for this key from a disk location to an empty string. Existing reports on disk cannot be viewed (since the setting that conveys the disk location is destroyed), and new reports will be stored in the database.
You want to store completed reports on disk instead of in the database where they are currently stored.	Change the value for this key from an empty string to the new disk location. If a completed report is in the database, users will be able to continue to view that report until it is deleted; new reports are stored on disk and users can see those as well.

5.2.1.4 ReportFileViewer key

Set the value for this key to the web page that handles displaying completed reports that are not "native" reports. This page handles displaying pdf, word, and xls files for example. In most circumstances, you will not modify this setting's value.

Note: “Native” report viewers are described to URS in the report_types.xml file (see **Appendix B** on page 52 for more information).

For example:

```
<add key="ReportFileViewer" value="ReportFileViewer.ashx?id={0}"/>
```

5.2.1.5 Telerik.Skin key

This is the skin choice for the main pages of the application. Additional settings will be offered in future versions. Please do not adjust this setting.

5.2.1.6 ExternalURL key

Set this value to the URL that users outside of your intranet will use to reach the Universal Report Server web site on this server. This URL is used for all notifications and RSS feeds and must be properly configured before users outside your intranet can use the notifications and feeds features.

For example:

```
<add key="ExternalURL" value="http://www.versareports.com/urs"/>
```

5.2.2 Other Settings

Review the comment lines in the web.config file (other than the Authentication comments, which are covered in Section 5.3 Security Infrastructure) and take action on any lines appropriate for you system.

5.3 Security Infrastructure

Universal Report Server currently integrates with three common Windows security infrastructures, as well as accommodating “no authentication”. Refer to the table below to identify the security infrastructure you will be using, and then follow the instructions starting in the section noted for that infrastructure.

Note: Disregard the instructions for the other infrastructures. When you are finished with your security infrastructure instructions, continue with the Section 5.5 Configuring Your Site Settings in Global.asax on page 28.

Security Infrastructure	Instructions in Section	Starts on Page
Forms Authentication	5.3.1	25
Windows NT	5.3.2	26
ActiveDirectory	5.3.3	27
No Authentication	5.3.4	28

As additional infrastructures are identified and VersaReports adds support for them, we will publish Configuration Notes for those infrastructures in the Support area at *www.versareports.com*.

The basic steps for configuring Universal Report Server to work within your security infrastructure are:

- Copy the sample global.asax file for the security infrastructure you will be using to the *InstallationDirectory*\web folder and rename it to global.asax.
- Modify *InstallationDirectory*\web\web.config to reflect the type of authentication you want and the authorization rules for accessing the web site.

A general Configuration Notes document is also available if your security infrastructure is not one of the included choices.

5.3.1 Forms Authentication

Forms Authentication makes use of a SQL Server table that contains a list of users and their capabilities within the application. An example table named Users is already included in the SQL database and the forms and code from this setup use that table to store the user names, passwords, and security controls for each user that can access the report server.

Forms Authentication includes an administration page, UserAdministration.aspx, that provides a way for an administrator to add, edit, and delete new users.

To set up Forms Authentication, do the following (some of this may have already been done in the basic installation):

1. Copy Global.Forms_Auth.asax from the *InstallationDirectory*\Global.asax Samples folder to the *InstallationDirectory*\web folder and rename it to Global.asax (overwrite any existing Global.asax). This file contains the code needed to handle forms authentication and can be edited to adjust global report server settings in the Session_Start routine. This is discussed later in Section 5.5 Configuring Your Site Settings in Global.asax.
2. In the *InstallationDirectory*\web folder, adjust the web.config file to set the **authentication** to Forms if necessary. For example:

```
<authentication mode="Forms">
  <forms loginUrl="Login.aspx" slidingExpiration="true" timeout="180" protection="All"
     requireSSL="false" defaultUrl="ReportsArchive.aspx"/>
</authentication>
```

   The **authentication** node describes the type of authentication, and the **forms** subnode describes all the forms authentication information. For further information, please refer to the Microsoft documentation for Forms Authentication.

3. In the *InstallationDirectory*\web\web.config file, review the values for the **machineKey** node. This node is specified to force the Forms Authentication ticket to be encrypted and decrypted by the same key and will help to keep people logged in across IIS application pool restarts, which from our experience happens frequently. All this is documented by Microsoft at these URLs:

   http://support.microsoft.com/kb/910443
   http://msdn.microsoft.com/en-us/library/aa480476.aspx

Forms Authentication supports logout, so if you have a logout link defined in your site's master page, have that link's URL changed to use logout.aspx.

Note: The master page for the basic installation is named Default.master.

Once you have finished configuring authentication as described above, navigate your browser to the report server site, and enter the Administrator's login name and password when prompted. An administrator account with the user name of **Administrator** and the password **URSadmin** was automatically created during installation.

Important: Please change the Administrator password as soon as possible.

Once the **Reports Archive** page is displayed, manually navigate your browser to UserAdministrator.aspx to create and manipulate users.

If you accidentally delete the last administrator user, a quick way to gain access to the UserAdministrator.aspx page is to use the SQL Server Management Console to manually edit the Users table to add a new Administrator user.

5.3.2 Windows NT Authentication

For servers without a domain server, you can configure Universal Report Server to use Windows NT authentication with just a few minor changes to web.config. With this configuration, all users who access the server must have a local user account on the server in order to authenticate.

Once you have one or more users configured, set up Windows NT authentication by doing the following:

1. Copy Global.WinNT_Auth.asax from the *InstallationDirectory*\Global.asax Samples folder to the *InstallationDirectory*\web folder and rename it to Global.asax (overwrite any existing Global.asax). This file contains the code needed to handle Windows NT authentication and can be edited to adjust global report server settings in the Session_Start routine. This is discussed later in Section 5.5 Configuring Your Site Settings in Global.asax
2. In the *InstallationDirectory*\web folder, adjust web.config to set the **authentication** to Windows. For example:

   ```
   <authentication mode="Windows" />
   ```

   Be sure to comment out any other authentication blocks.
3. Adjust web.config to include <appSettings> key values to provide additional information. For example,

   ```
   <add key="AdministratorsGroup" value="ReportEditors"/>
   <add key="ProviderPath" value="WinNT://"/>
   <add key="ProviderLogin" value="AdminUser"/>
   <add key="ProviderPassword" value="AdminPassword"/>
   ```

   The **AdministratorsGroup** key defines a windows user group containing the users who can register reports with the Universal Report Server. Also, to edit a report, a user must be in this group. If you don't have a group of these users already defined, add that group via the Computer Management program (look for Users and Groups) and assign existing users to that group. Once the group is defined, enter its name as the value of the **AdministratorsGroup** key.

   ProviderPath shows the exact value to use for Windows NT-style authentication. Ensure that this setting in the web.config file is correct.

ProviderLogin and **ProviderPassword** are needed to provide appropriate credentials for the Universal Report Server application to access the users and groups on your server. Universal Report Server likely will run as the NETWORK SERVICE user on your system, and that user does not have access to the security subsystem to query information about users and groups. By providing an administrator's user name and password credentials for these two keys, Universal Report Server will have the necessary credentials to access that information. If your server is configured to not require special permissions to read user and group information from the local repository, leave the values for these two keys empty (that is, use value="").

Windows NT authentication does not support logout and does not need a login form, so you should modify the master page (Default.master) to remove the logout link that is included for Forms authentication scenarios.

5.3.3 Active Directory Authentication

For servers with an Active Directory domain server, you can configure Universal Report Server to use Active Directory authentication with just a few minor changes to web.config. With this configuration, all users who access the server must have an account on the server in order to authenticate.

Once you have one or more users configured, set up ActiveDirectory authentication by doing the following:

1. Copy Global.AD_Auth.asax from the *InstallationDirectory*\Global.asax Samples folder to the *InstallationDirectory*\web folder and rename it to Global.asax (overwrite any existing Global.asax). This file contains the code needed to handle Active Directory authentication and can be edited to adjust global report server settings in the Session_Start routine. This is discussed later in Section 5.5 Configuring Your Site Settings in Global.asax
2. In the *InstallationDirectory*\web folder, adjust web.config to set the **authentication** to Windows. For example:

   ```
   <authentication mode="Windows" />
   ```

   Be sure to comment out any other authentication blocks.
3. Adjust web.config to include <appSettings> key values to provide additional information. For example,

   ```
   <add key="AdministratorsGroup" value="ReportEditors"/>
   <add key="ProviderPath" value=" LDAP://dc=versareports,dc=com "/>
   <add key="ProviderLogin" value="AdminUser"/>
   <add key="ProviderPassword" value="AdminPassword"/>
   ```

   The **AdministratorsGroup** key defines a windows user group containing the users who can register reports with the Universal Report Server. Also, to edit a report, a user must be in this group. If you don't have a group of these users already defined, add that group via the Computer Management program (look for Users and Groups) and assign existing users to that group. Once the group is defined, enter its name as the value of the **AdministratorsGroup** key.

   ProviderPath is an LDAP URL for the domain server. An example would be "LDAP://dc=versareports,dc=com". You can find more information in this Microsoft document: http://support.microsoft.com/kb/196455.

   ProviderLogin and **ProviderPassword** are needed to provide appropriate credentials for the Universal Report Server application to access the users and groups in your domain. Universal Report Server likely will run as the NETWORK SERVICE user on your system, and that user does not have access to the ActiveDirectory security subsystem to query information about users and groups. By providing a user name and password with appropriate permissions to access your ActiveDirectory repository, Universal Report Server will have the necessary credentials to access that information. If your ActiveDirectory is configured to not require special permissions to read user and group information from the repository, leave the values for these two keys blank.

ActiveDirectory authentication does not support logout and does not need a login form, so you should modify the master page (Default.master) to remove the logout link that is included for Forms authentication scenarios.

5.3.4 No Authentication

Use the “no authentication” option if you plan to publish reports on the internet for anyone to view without requiring authentication.

Our “No Authentication” configuration includes a single page to select the type of user – guest or administrator -- and the code behind the scenes to let that selection govern what the user sees.

To setup the “No Authentication” setup:

1. Copy Global.No_Auth.asax from the *InstallationDirectory*\Global.asax Samples folder to the *InstallationDirectory*\web folder and rename it to Global.asax (overwrite any existing Global.asax).
2. Change the logout link on the master page (Default.master), to redirect users to the page Default_NoAuth.aspx.
3. Remove any **authentication** settings from web.config.
4. Configure IIS to set the default page for the Universal Report Server web application to Default_NoAuth.aspx (instead of the default of ReportsArchive.aspx).

The “No Authentication” setup defines two users: Guest and Administrator. Guest can view and possibly schedule reports, but not register or modify reports. Administrator can view, schedule, and define reports.

5.4 Testing the Basic Universal Report Server Configuration

Once you have completed setting up the interface to your server's security infrastructure (either for Forms, WinNT, or AD authentication as described in Section 5.3 Security Infrastructure), you should be able to go to the main URL on the web site and login. If everything has been set up correctly, the Universal Report Server application web pages will display. This will verify the basic configuration is ready. If you have trouble logging in or reaching the ReportsArchive.aspx page (the main site page), please review the correct authentication configuration section from above and verify that you have done all the steps before contacting VersaReports support for assistance.

5.5 Configuring Your Site Settings in Global.asax

Before configuring the site settings in Global.asax, please review the subsections in Section 5.3 Security Infrastructure for setting up the interface to your server's security infrastructure. Those instructions had you copy the appropriate sample Global.asax in the *InstallationDirectory*\Global.asax Samples folder to the *InstallationDirectory*\web folder and rename it to Global.asax.

As part of Global.asax, there is a routine called **Session_Start**. This routine is run for every web session started by a user. Within **Session_Start** you can set a number of site settings by adjusting the properties of the **ReportState** class.

- ReportState.RequireAuthentication - set this value to **false** if your server is going to accept anonymous users without any type of authentication. Set it to **true** if you want authentication. By default this property is set to **true**; for most situations, do not adjust the default setting.
- ReportState.HideCopyrightNotice - for non-evaluation copies, you can set this property to **true** to hide the copyright notice that appears below the tab area on each main page. By default, this property is set to **false**.

In Global.asax, a public class that is derived from the **BaseConfiguration** class needs to be created if you are not using one of the sample Global.asax files. In the constructor for this class, you must set the following properties with user information:

1. ReportState.IsReportAdministrator - set this property to **true** for all report administrators and **false** for anyone else (including report schedulers who are not report administrators). This controls whether the user can view the Reports Manager page.
2. ReportState.UserId - the ID string that is assigned to the user. Assign this value once the user has signed in.

5.6 Report Designer Integration

Universal Report Server currently integrates with four common report designers: SAP Crystal Reports, DevExpress XtraReports, GrapeCity/Data Dynamics ActiveReports, and Telerik Reporting. Refer to the table below to identify the report designer you will be using, and then follow the instructions starting in the section noted for that report designer.

Note: Disregard the instructions for any report designer you will not be using. When you are finished with your integration instructions, continue with Section 5.5 Configuring Your Site Settings in Global.asax on page 28.

Report Designer	Instructions in Section	Starts on Page
GrapeCity/Data Dynamics ActiveReports	5.6.1	30
DevExpress XtraReports	5.6.2	31
Telerik Reporting	5.6.3	32
SAP Crystal Reports	5.6.4	33

As additional report designers are identified and integrated, VersaReports will publish Configuration Notes for those designers in the Support area at *www.versareports.com*.

The basic steps for configuring a report designer to work with Universal Report Server are:

- Load the report designer's deployment libraries and configure it for web access, if available.
- Create a "Report Connector" (software that Universal Report Server uses to interact with the report designer's libraries to run a report) and copy the Report Connector library to both the *InstallationDirectory*\ReportsRunner folder and the *InstallationDirectory*\web\Bin folder.
- Create a "native" report viewer web page for those designers that support a native report viewer (for example, Crystal Reports, XtraReports, and ActiveReports Professional). Sample report viewer web pages are included in the *InstallationDirectory*\Sample Report Viewers folder, and if you are integrating with one of the supported report designers, use the correct sample report viewer files from that folder. Place the report viewer web pages into the *InstallationDirectory*\web folder. Configure the <assemblies> section of the web.config file to include any libraries residing in the server's GAC that the web viewer requires in order to display a report for that report designer package. The report designer documentation will detail the lines that need to be included in web.config. There is a web.config in the *InstallationDirectory*\Sample Report Viewers folder that contains a sampling of what these assemblies' lines will look like.
- Modify the *InstallationDirectory*\web\App_Data\report_types.xml file to describe the Report Connectors available for report administrators to handle reports with Universal Report Server. The report_types.xml file is used by both the web site and the ReportsRunner service, and must be correct for reports to be properly loaded and run. The basic report_types.xml file that is included with the Universal Report Server installation defines the necessary information about the sample Report

Connectors included in the installation package. For more information about the structure of the report_types.xml file, refer to Appendix B on page 52.

Note: Report connectors can be compiled on the same system as the Universal Report Server or they can be compiled on a developer's workstation. To compile on a developer's workstation, you will need to copy VersaReports.ReportConnector.dll from the Universal Report Server installation to your development server, since that file is required to compile a report connector.

5.6.1 GrapeCity/Data Dynamics ActiveReports

Universal Report Server includes the source code for creating a Report Connector that will interact with ActiveReports for .NET version 2.0 and above, including support for report parameters and support for the Professional Edition's web viewer.

The Report Connector you will create supports accessing DLL-based reports instead of RPX-based reports, primarily because reports created with ActiveReports are so much more flexible if you have additional .NET-based code around them. However, because the source code for the Report Connector is included, you can modify it to support RPX-based reports instead.

To create a Report Connector for ActiveReports, you must do the following:

1. Deploy ActiveReports to the same server where Universal Report Server is running, as described in the ActiveReports for .NET documentation.
2. Using Visual Studio 2005 or later:

   Note: Please refer to Appendix A on page 40 for detailed instructions (including screen shots) on using Visual Studio to create and install the Report Connector DLL.

   a. Create a Visual C# Windows Class Library project giving it a name that identifies the Report Connector you are creating. For example, ActiveReportsDllConnector.

   b. Delete the default class (Class1.cs).

   c. Add ActiveReportsDllConnector.cs as an existing item to the project (from the *InstallationDirectory*\Sample Report Connectors\ActiveReportConnector folder).

   d. Add a reference to the VersaReports.ReportConnector.dll library (from the *InstallationDirectory*\ReportsRunner folder).

   e. Add references to the installed version of ActiveReports' libraries. Set Copy Local to **True** for all the added libraries.

   f. Make sure the Target Framework on the Applications tab (of Project Properties) is set to **.NET Framework 2.0** and the Platform target on the Build tab is set to **x86**.

   g. Build the project to create ActiveReportsDllConnector.dll.
3. Copy the ActiveReportsDllConnector.dll file from *ProjectFolder*\Bin\Debug into both the *InstallationDirectory*\web\Bin and *InstallationDirectory*\ReportsRunner folders on the Universal Report Server system.
4. If the ActiveReports for .NET libraries have been deployed to the server's GAC, modify the <assemblies> node in the *InstallationDirectory*\web\web.config file to include the list of assemblies that are needed to create and view an ActiveReports for .NET report. For example, one of these lines might be:

   ```
   <add assembly="ActiveReports.Web, Version=5.2.1331.2, Culture=neutral, PublicKeyToken=cc4967777c49a3ff"/>
   ```

   If the libraries have not been deployed to the system GAC, copy them into both the *InstallationDirectory*\web\Bin and the *InstallationDirectory*\ReportsRunner folders. The list of deployable libraries is described in the ActiveReports for .NET documentation and you must include ALL of them for ActiveReports reports to execute and be viewed properly.

5. Make sure the ActiveReports for .NET report type lines are included in the *InstallationDirectory*\web\App_Data\report_types.xml file. This report type is included as part of the default installation.

   If the ActiveReports for .NET libraries have not been deployed to the system GAC or into the directory where the VersaReports.ReportConnector.dll file is located, then edit the report_types.xml file to create **add** nodes in the **supporting_assemblies** node to enumerate this Report Connector's supporting DLLs.

   Review the documentation on the report_types.xml file in Appendix B on page 52 for more information about the XML structure for this file.

   Note: If you have *ActiveReports for .NET Professional Edition*, your edition includes a way to display native ActiveReports for .NET reports using a web viewer. To configure this viewer:

   1. Modify report_types.xml to include the following line in the <output_formats> section for the ActiveReportsDll report type.

      ```
      <format extension="rdf" name="Preview-Only Format" native="true"
      display_page="ActiveReportsViewer.aspx?id={0}" icon="images/report.png" />
      ```

   2. Next, copy the ActiveReportsViewer.* files from *InstallationDirectory*\Sample Report Viewers folder to the *InstallationDirectory*\web folder.

   3. Finally, copy the file arview2.cab from the ActiveReports for .NET installation to the *InstallationDirectory*\web folder.

You're ready to register and run an ActiveReports for .NET report. When you register the DLL containing the report, the Report Definition Wizard will request the "Class Name of Report", which should be the fully-defined class name (including namespace) for the report class within the DLL.

5.6.2 DevExpress XtraReports

Universal Report Server includes the source code for creating a Report Connector that will interact with DevExpress XtraReports version 2008.3 and above, including support for report parameters and support for its web viewer.

The Report Connector you will create supports accessing DLL-based reports created with DevExpress XtraReports. If you want to implement REPX (XtraReports' native report designer file format) report handling, the source code just needs some minor modification to work with REPX files.

To create a Report Connector for XtraReports, you must do the following:

1. Deploy DevExpress XtraReports to the same server where Universal Report Server is running, as described in the XtraReports documentation.
2. Using Visual Studio 2005 or later:

   Note: Please refer to Appendix A on page 40 for detailed instructions (including screen shots) on using Visual Studio to create and install the Report Connector DLL.

   a. Create a Visual C# Windows Class Library project giving it a name that identifies the Report Connector you are creating. For example, XtraReportsDllConnector.

   b. Delete the default class (Class1.cs).

   c. Add XtraReportsDllConnector.cs as an existing item to the project (from the *InstallationDirectory*\Sample Report Connectors\XtraReportConnector folder).

   d. Add a reference to the VersaReports.ReportConnector.dll library (from the *InstallationDirectory*\ReportsRunner folder).

   e. Add references to the installed version of XtraReports. Set Copy Local to **True** for all the added libraries.

f. Make sure the Target Framework on the Applications tab (of Project Properties) is set to **.NET Framework 2.0** and the Platform target on the Build tab is set to **x86**.

g. Build the project to create XtraReportsDllConnector.dll.

3. Copy the XtraReportsDllConnector.dll file from *ProjectFolder*\Bin\Debug into both the *InstallationDirectory*\web\Bin and *InstallationDirectory*\ReportsRunner folders.

4. If the DevExpress XtraReports libraries have been deployed to the GAC, modify the <assemblies> node in the *InstallationDirectory*\web\web.config file to include the list of assemblies that are needed to create and view an XtraReports report. For example, one of these lines might be:

```
<add assembly=" DevExpress.Data.v8.3, Version=8.3.2.0, Culture=neutral, PublicKeyToken=B88D1754D700E49A "/>
```

   If the libraries have not been deployed to the system GAC, copy them into both the both the *InstallationDirectory*\web\Bin and the *InstallationDirectory*\ReportsRunner folders. The list of deployable libraries is described in the XtraReports documentation and you must include ALL of them for an XtraReports report to work properly.

5. Make sure the DevExpress XtraReports report type lines are included in the *InstallationDirectory*\web\App_Data\report_types.xml file. This report type is included as part of the default installation.

   If the DevExpress XtraReports libraries have not been deployed to the system GAC or into the directory where the VersaReports.ReportConnector.dll file is located, then edit the report_types.xml file to create **add** nodes in the **supporting_assemblies** node to enumerate this Report Connector's supporting DLLs.

   Review the documentation on the report_types.xml file in Appendix B on page 52 for more information about the XML structure for this file.

   Note: DevExpress XtraReports includes a way to display native-format reports using a web viewer. To configure this viewer,

   1. Modify report_types.xml to include the following line in the <output_formats> block for the XtraReportsDll report type.

```
<format extension="prnx" name="Preview-Only Format" native="true"
display_page="XtraReportsViewer.aspx?id={0}" icon="images/report.png" />
```

   2. Finally, copy the XtraReportsViewer.* files from *InstallationDirectory*\Sample Report Viewers folder to the *InstallationDirectory*\web folder.

You're ready to register and run a DevExpress XtraReports report. When you register the DLL containing the report, the Report Definition Wizard will request the "Class Name of Report", which should be the fully-defined class name (including namespace) for the report class within the DLL.

5.6.3 Telerik Reporting

Universal Report Server includes the source code for creating a Report Connector that will interact with Telerik Reporting version 2009.1 and above, including support for report parameters.

The Report Connector you will create supports accessing DLL-based reports created with Telerik Reporting.

To create a Report Connector for Telerik Reporting, you must do the following:

1. Deploy Telerik Reporting to the same server where Universal Report Server is running, as described in the Telerik Reporting documentation.

2. Using Visual Studio 2005 or later:

Note: Please refer to Appendix A on page 40 for detailed instructions (including screen shots) on using Visual Studio to create and install the Report Connector DLL.

a. Create a Visual C# Windows Class Library project giving it a name that identifies the Report Connector you are creating. For example, TelerikReportingDllConnector.

b. Delete the default class (Class1.cs)

c. Add TelerikReportingDllConnector.cs as an existing item to the project (from the *InstallationDirectory*\Sample Report Connectors\TelerikReportingConnector folder).

d. Add a reference to the VersaReports.ReportConnector.dll library (from the *InstallationDirectory*\ReportsRunner folder).

e. Add references to the installed version of Telerik Reporting. Set Copy Local to **True** for all the added libraries.

f. Make sure the Target Framework on the Applications tab (of Project Properties) is set to **.NET Framework 2.0** and the Platform target on the Build tab is set to **x86**.

g. Build the project to create TelerikReportingDllConnector.dll.

3. Copy the TelerikReportingDllConnector.dll file from *ProjectFolder*\Bin\Debug into both the *InstallationDirectory*\web\Bin and *InstallationDirectory*\ReportsRunner folders.

4. If the Telerik Reporting libraries have been deployed to the GAC, modify the <assemblies> node in the *InstallationDirectory*\web\web.config file to include the list of assemblies that are needed to create and view a Telerik Reporting report.

   If the libraries have not been deployed to the system GAC, copy them into both the *InstallationDirectory*\web\Bin and the *InstallationDirectory*\ReportsRunner folders. The list of deployable libraries is described in the Telerik Reporting documentation and you must include ALL of them for a Telerik Reporting report to work properly.

5. Make sure the Telerik Reporting report type lines are included in the *InstallationDirectory*\web\App_Data\report_types.xml file. This report type is included as part of the default installation.

   If the Telerik Reporting libraries have not been deployed to the system GAC or into the directory where the VersaReports.ReportConnector.dll file is located, then edit the report_types.xml file to create **add** nodes in the **supporting_assemblies** node to enumerate this Report Connector's supporting DLLs.

   Review the documentation on the report_types.xml file in Appendix B on page 52 for more information about the XML structure for this file.

   Note: The Telerik Reporting Report Connector does not support the native Telerik Reporting web viewer. Only exportable formats are allowed at this time.

You're ready to register and run a Telerik Reporting report. When you register the DLL containing the report, the Report Definition Wizard will request the "Class Name of Report", which should be the fully-defined class name (including namespace) for the report class within the DLL.

5.6.4 SAP Crystal Reports

Universal Report Server includes the source code for creating a Report Connector that will interact with SAP Crystal Reports, including support for report parameters. Similar Report Connectors have been tested with several versions of Crystal Reports.

The Report Connector you will create supports accessing RPT file-based reports created with Crystal Reports Designer.

To create a Report Connector for Crystal Reports, you must do the following:

1. Install the SAP Crystal Reports deployment package to the same server where Universal Report Server is running, as described in the Crystal Reports documentation.
2. Using Visual Studio 2005 or later:

   Note: Please refer to Appendix A on page 40 for detailed instructions (including screen shots) on using Visual Studio to create and install the Report Connector DLL.

   a. Create a Visual C# Windows Class Library project giving it a name that identifies the Report Connector you are creating. For example, CrystalReportsConnector.

   b. Delete the default class (Class1.cs).

   c. Add CrystalReportsConnector.cs as an existing item to the project (from the *InstallationDirectory*\Sample Report Connectors\CrystalReportsConnector folder).

   d. Add a reference to the VersaReports.ReportConnector.dll library (from the *InstallationDirectory*\ReportsRunner folder).

   e. Add references to the installed version of Crystal Reports. Set Copy Local to **True** for all the added libraries.

   f. Make sure the Target Framework on the Applications tab (of Project Properties) is set to **.NET Framework 2.0** and the Platform target on the Build tab is set to **x86**.

   g. Build the project to create XtraReportsDllConnector.dll.
3. Copy the CrystalReportsConnector.dll file from *ProjectFolder*\Bin\Debug into both the *InstallationDirectory*\web\Bin and *InstallationDirectory*\ReportsRunner folders.
4. The Crystal Reports deployment package installs its libraries into the system GAC, so modify the <assemblies> node in the *InstallationDirectory*\web\web.config file to include the list of assemblies that are needed to create and view a Crystal Reports report. The list of libraries in the GAC is described in the Crystal Reports documentation and you must include ALL of them for a Crystal Reports report to work properly.
5. Make sure the Crystal Reports report type lines are included in the *InstallationDirectory*\web\App_Data\report_types.xml file. This report type is included as part of the default installation.

   Review the documentation on the report_types.xml file in Appendix B on page 52 for more information about the XML structure for this file.

   Note: Crystal Reports includes a way to display native-format reports using a web viewer. To configure this viewer,

   1. Modify report_types.xml to include the following line in the <output_formats> block for the CrystalReportsRpt report type.

      ```
      <format extension="rpt" name="Preview-Only Format" native="true"
      display_page="CrystalViewerMain.aspx?id={0}" icon="images/report.png" />
      ```

   2. Next, copy the Crystal`*.*` files from *InstallationDirectory*\Sample Report Viewers folder to the *InstallationDirectory*\web folder.

   3. Finally, check with the Crystal Reports documentation for information about the virtual directory you will need to create in the Web folder (the name of the directory and its physical location are specific to the version of CR that you installed). This step is VERY important for successful use of the Crystal Reports web viewer.

You're ready to register and run a Crystal Reports report.

5.6.5 Other Report Connectors

Periodically, VersaReports will release report connectors for other report generators. You will find the source code and documentation for these connectors on the VersaReports support web site at http://www.versareports.com/support.

5.7 Start ReportsRunner

To run all reports, Universal Report Server includes a "ReportsRunner" program in the *InstallationDirectory*\ReportsRunner folder. This program can either be run from a Command Prompt window or it can be installed as a service that restarts when the computer restarts and never needs to have a user logged in.

To install ReportsRunner as a service using the Local System Account, open a Command Prompt window on server, and navigate (using the CD command) to the *InstallationDirectory*\ReportsRunner folder. Enter the command:

```
ReportsRunner -install
```

When ReportsRunner is installed as a service, all error and status messages from it are logged to the Application Event Log.

To run ReportsRunner in interactive mode from the Command Prompt window, start a command prompt window, CD to *InstallationDirectory*\ReportsRunner and enter the command

```
ReportsRunner
```

without the "-install" command line argument. Running ReportsRunner in interactive mode is an excellent way to debug your reports and the Report Connectors, because the program displays error and status messages indicating the reports it is running and any errors about them.

5.8 Licensing

Universal Report Server version 1.1.0 is licensed on a per-server basis. You may run the web site and the ReportsRunner on exactly one physical/logical server. If you want to split these pieces up or create a web farm for the Report Server, you are obligated to purchase a separate license for each physical/logical server.

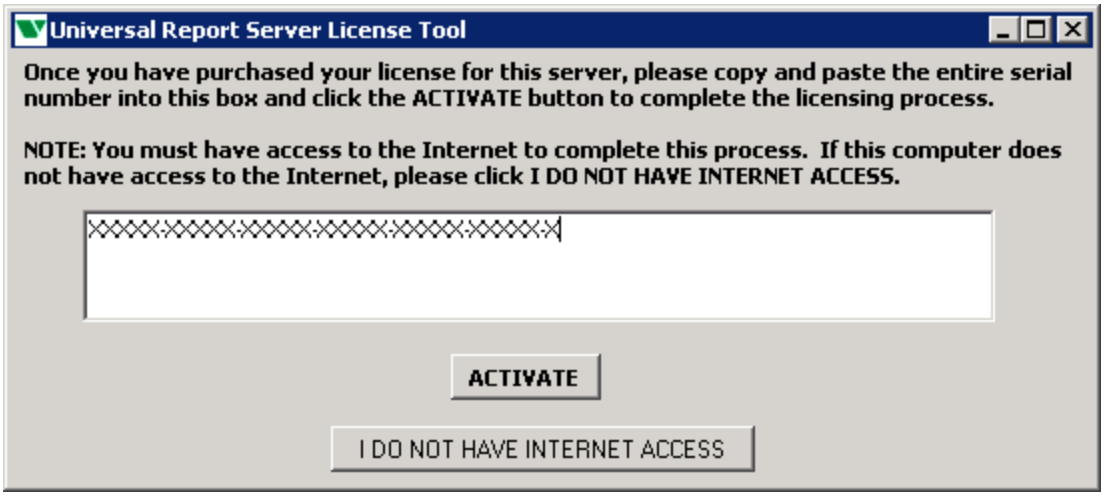
Universal Report Server comes in two editions: Standard Edition and Enterprise Edition. The products are essentially the same, except that Enterprise Edition supports an unlimited number of CPUs on a single machine and Standard Edition supports no more than 2 CPUs on a single machine. Each CPU can have any number of "cores" or "hyperthreads" for either edition.

The installation program installs an evaluation copy of the application. The evaluation copy will have an expiration date that is displayed along with the copyright message at the bottom of the main web pages. If you decide to purchase the license any time during or after the evaluation period, run the **LicenseTool** program to activate your Universal Report Server license.

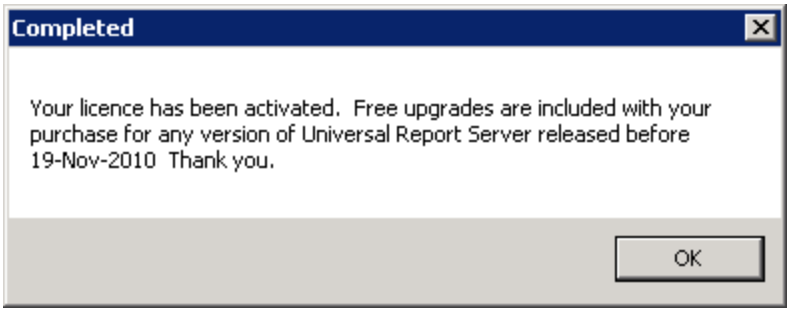
5.8.1 Activating Your License via the Internet

Use the following instructions to convert your evaluation copy to a purchased program if you have internet access.

1. Run the **LicenseTool** program from the *InstallationDirectory*\Licensing folder. The **Universal Report Server License Tool** dialog box displays.

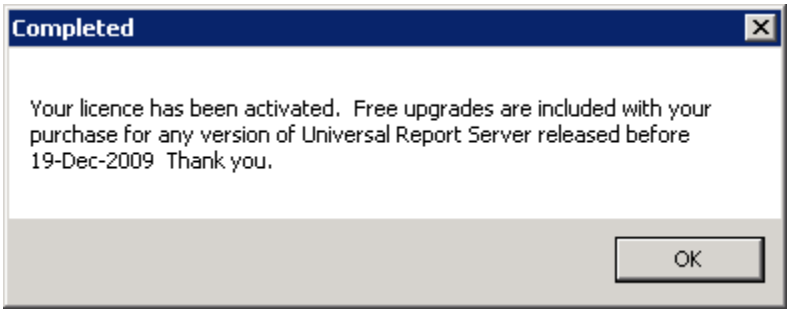


2. Enter (Copy and Paste) the serial number provided by VersaReports or its authorized reseller when you completed your purchase and click **ACTIVATE**.
3. If your license activation is successful, a dialog box similar to the following displays:



This example, activated on November 19, 2009, illustrates the purchase of a Support contract providing for free upgrades for one year.

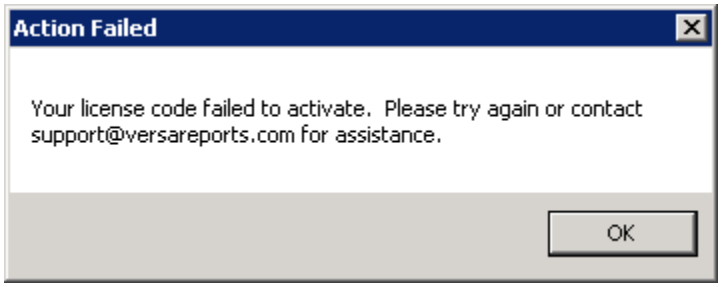
You also may see a dialog box similar to the following:



This example illustrates the activation of Universal Report Server without having purchased a Support contract (only 30 days of upgrades).

4. Click OK on this **Completed** dialog box. Both this dialog box and the **Universal Report Server License Tool** dialog box disappear.

Note: If you enter the wrong serial number, if you do not have an internet connection, or for some other reason the activation is not successful, a dialog box similar to the following displays:

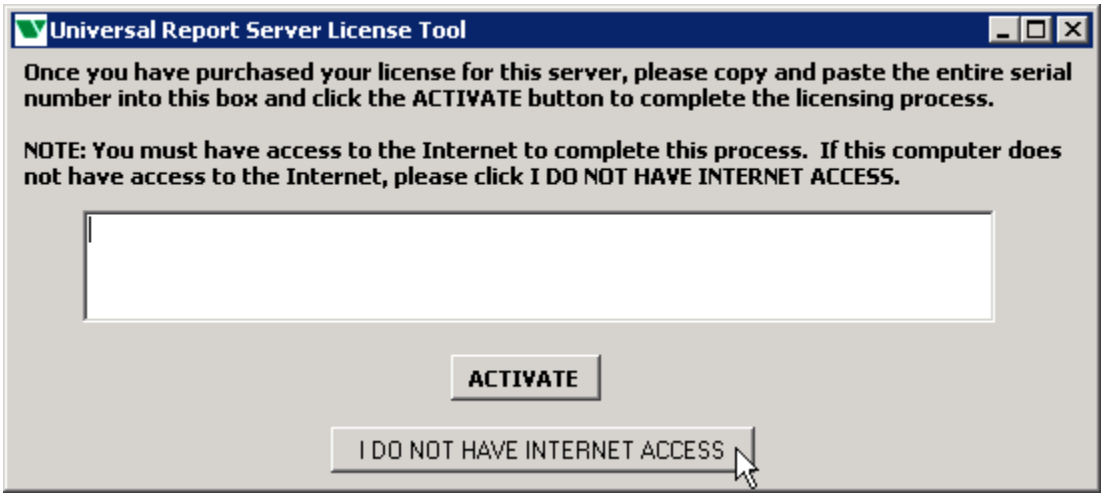


Click OK and try again or contact VersaReports for assistance.

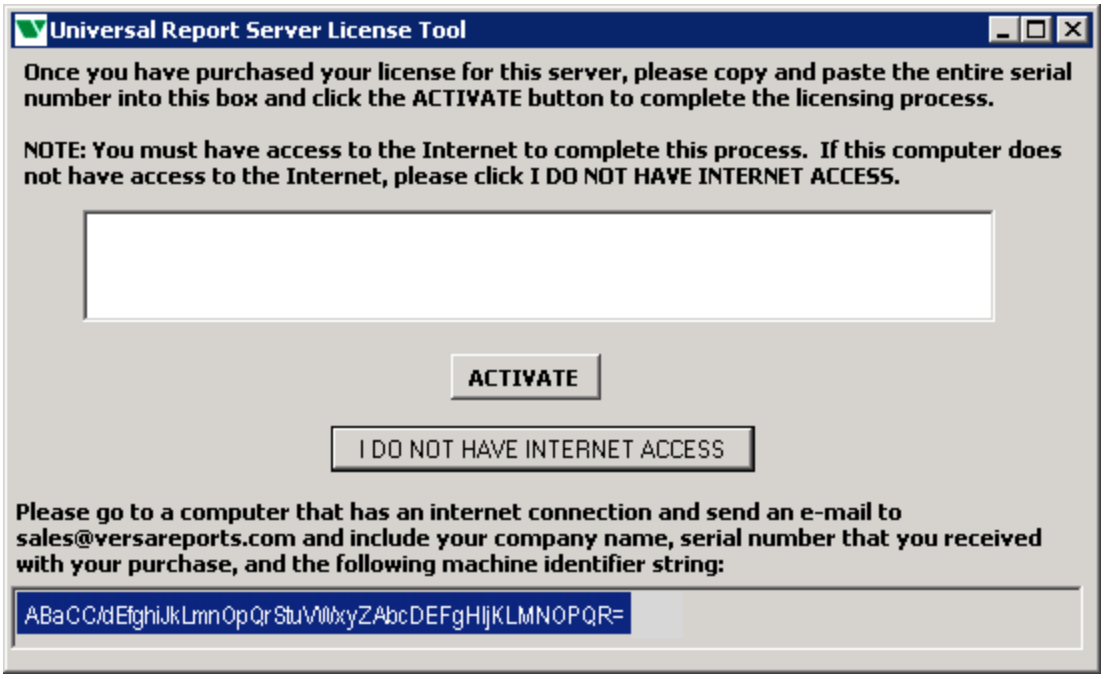
5.8.2 Activating Your License Without Internet Access

Use the following instructions to obtain your license when you do not have internet access from the machine where Universal Report Server is installed.

1. Run the **LicenseTool** program from the *InstallationDirectory*\Licensing folder. The **Universal Report Server License Tool** dialog box displays.



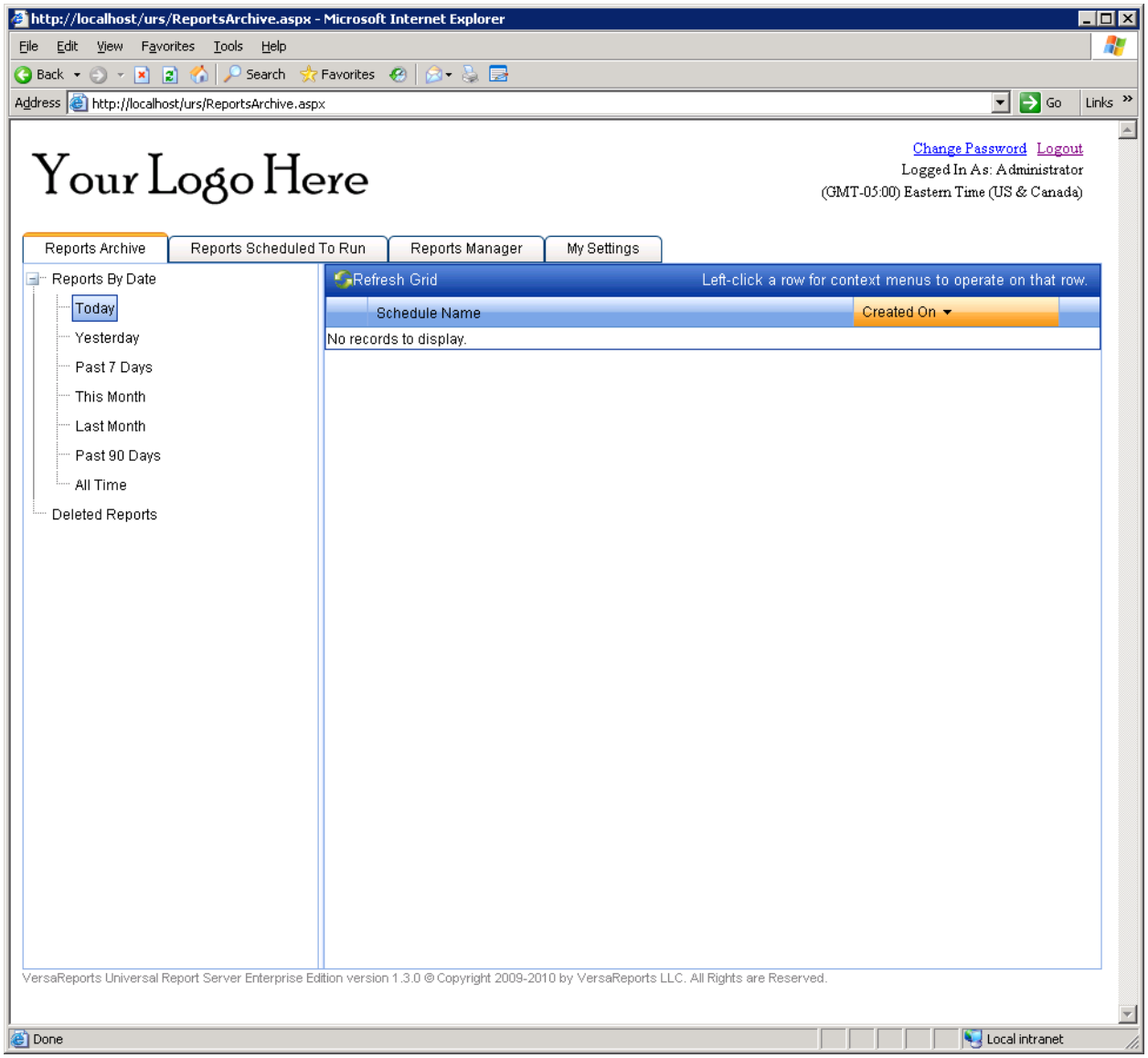
2. Click the **I DO NOT HAVE INTERNET ACCESS** button. The following dialog box displays.



3. Follow the instructions on the screen that tell you to send an email to sales@versareports.com and send the displayed machine identifier string (it can be copied) along with your company name and the serial number you receive with your purchase.
4. Upon receipt and verification of this information, VersaReports will activate your license and email back to you a new license code.
5. Take this new license code to the server where URS is installed and enter it into the first box in the License Tool. Click **ACTIVATE**. A **Completed** dialog box displays informing you of your support term. Click **OK** in this dialog box.

5.8.3 Run Universal Report Server

When you run Universal Report Server, the URS edition and version is printed at the bottom of the screen:



Appendix A
Detailed Instructions for Creating a Report Connector

The Report Connector is a compiled piece of software that provides an interface between the Universal Report Server software and the report designer's report engine software. Universal Report Server includes several example Report Connectors. This appendix provides detailed instructions for creating and installing the Report Connector DLL.

The currently supported report designers are:

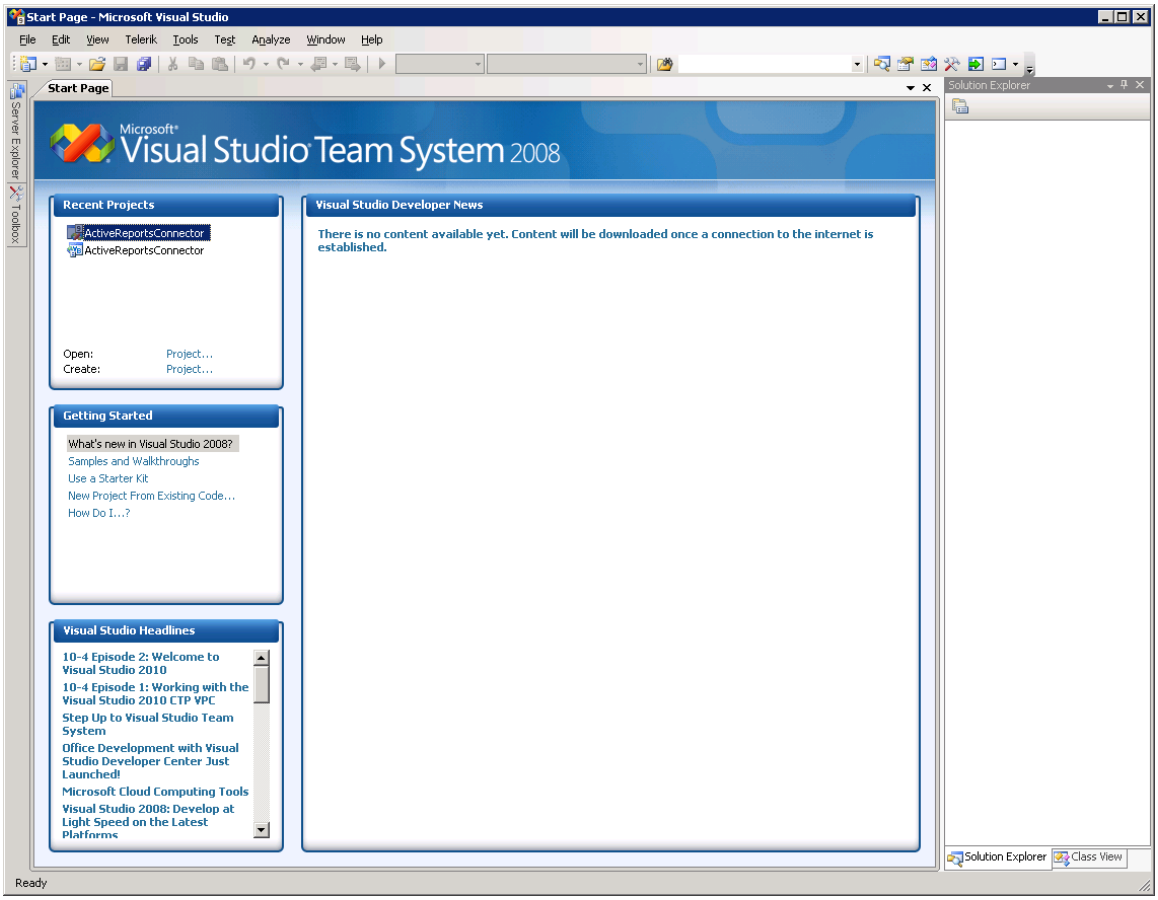
- SAP Crystal Reports
- DevExpress XtraReports
- GrapeCity/Data Dynamics ActiveReports for .NET
- Telerik Reporting

For creating a Report Connector for these report designers, or to create a Report Connector for a report designer that is not included with Universal Report Server, you need the following:

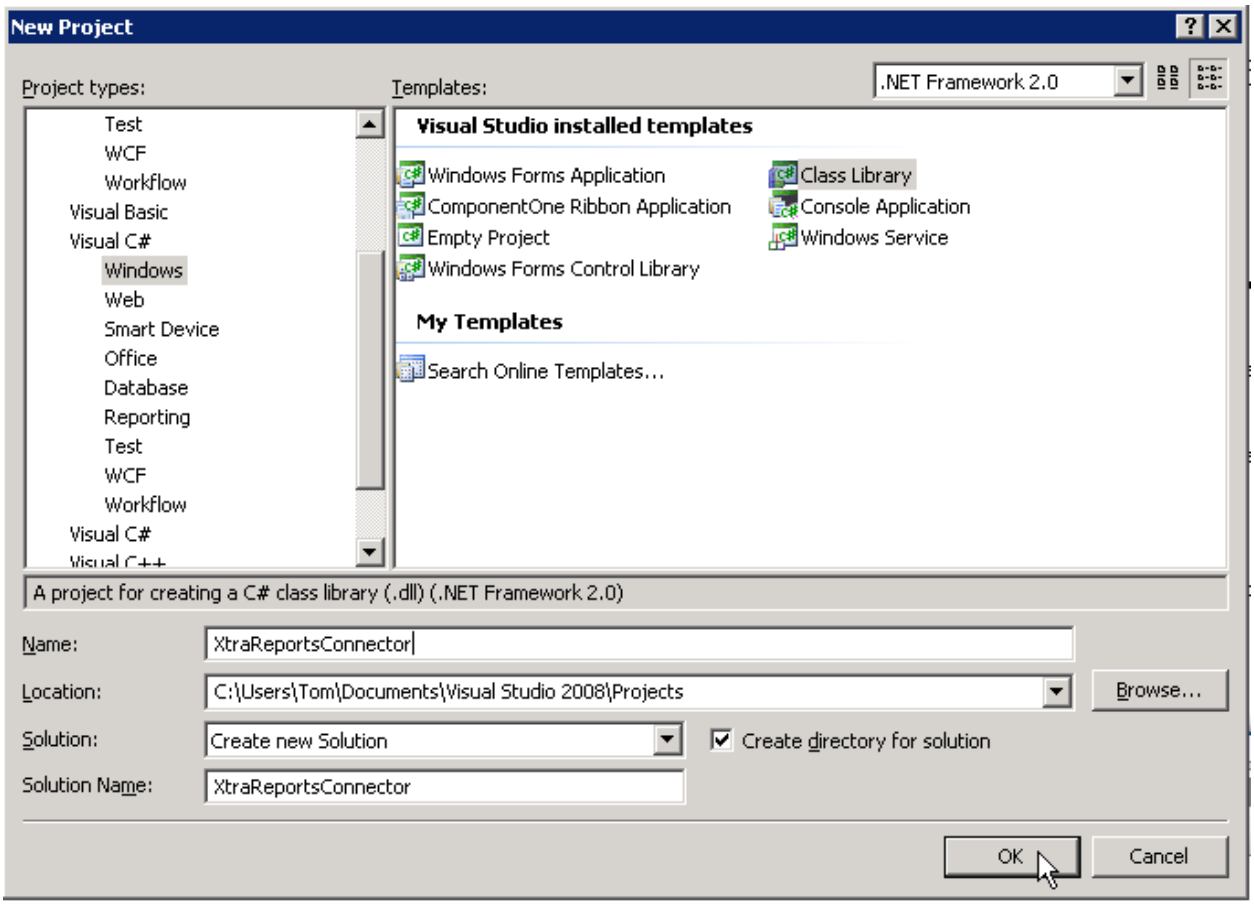
- Microsoft Visual Studio 2005 or later
- Developer license for your report designer

Note: Report connectors can be compiled on the same system as the Universal Report Server or they can be compiled on a developer's workstation. To compile on a developer's workstation, you will need to copy VersaReports.ReportConnector.dll from the Universal Report Server installation to your development server, since that file is required to compile a report connector.

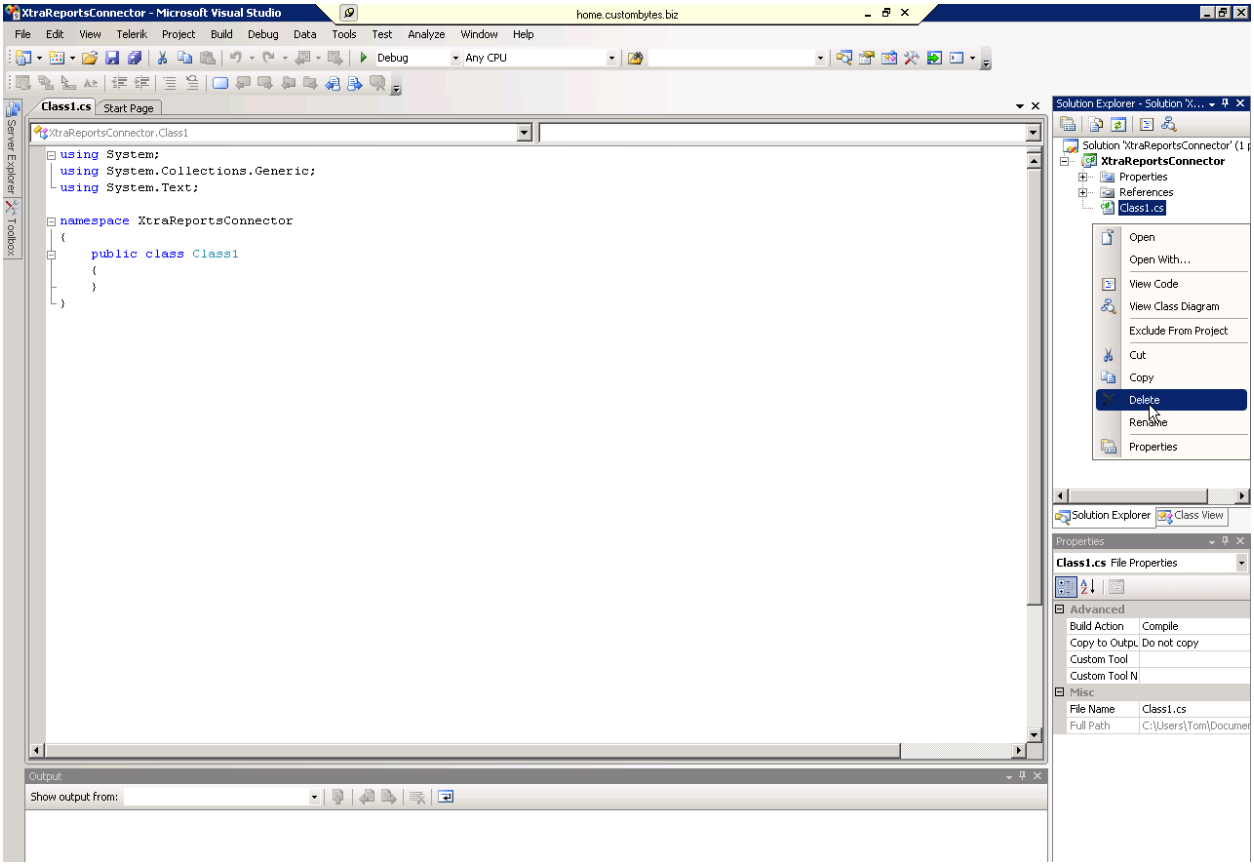
Note: The screens in the following steps show XtraReports as the report designer for which we are creating the Report Connector. The steps and screens would be virtually the same for the other report designers. Also, the screens show Microsoft Visual Studio 2008. Visual Studio 2005 screens, although not identical, are very similar.



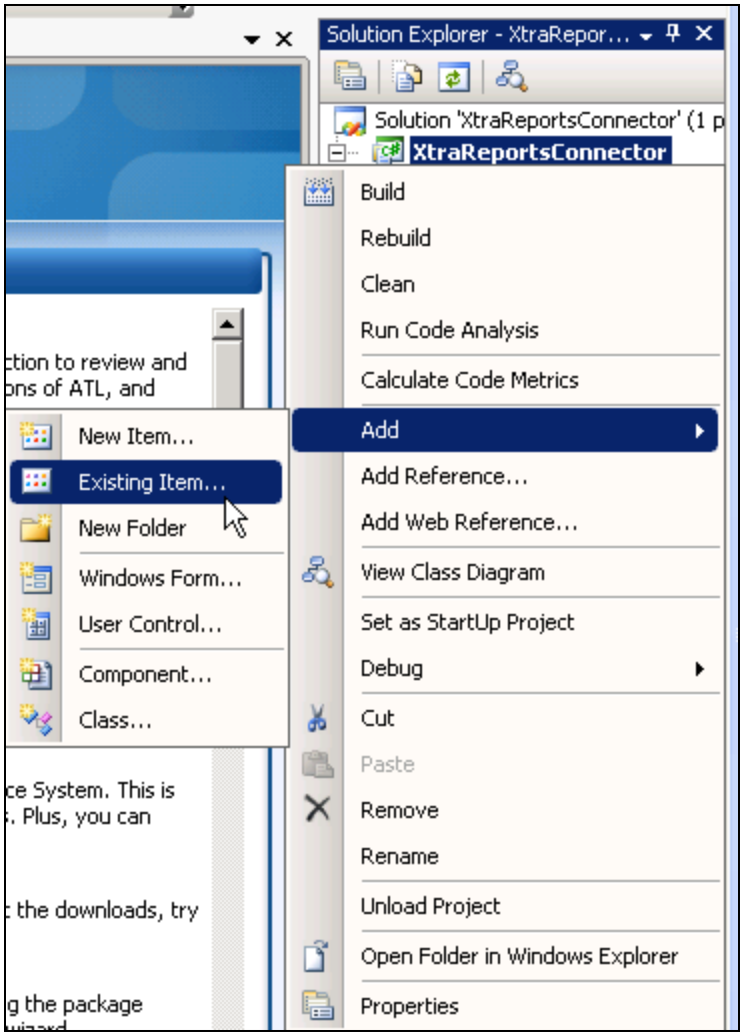
1. Open Visual Studio and create a new Visual C# Windows Class Library project. One way to do this is on the **File** menu, point to **New**, and then click **Project**.



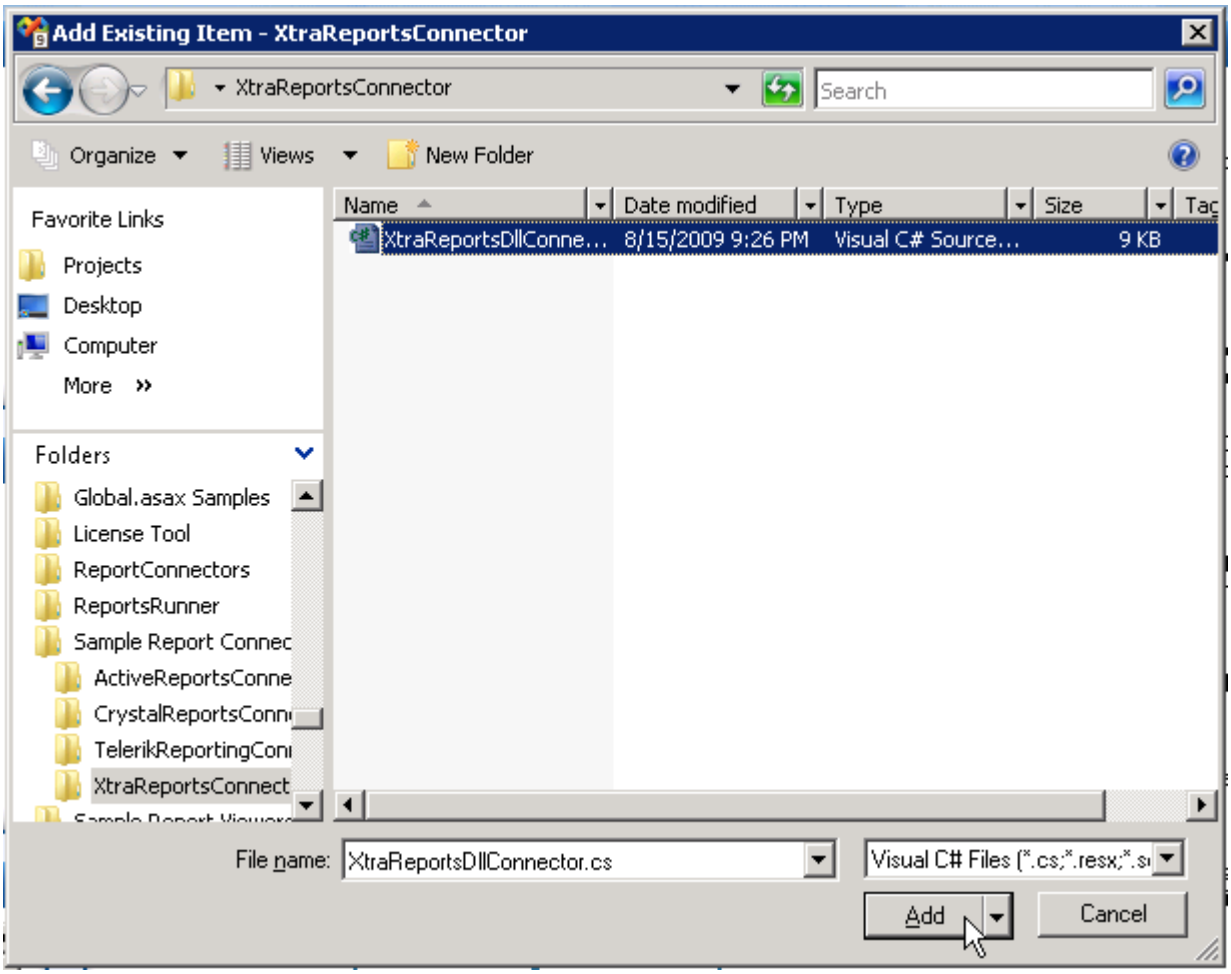
2. Be sure to select **Windows** from the **Visual C#** Project types list, and **Class Library** from the VS installed templates list. Select **.NET Framework 2.0** from the upper-right list. Give the project a name that identifies the Report Connector you are creating. In our example, XtraReportsConnector.



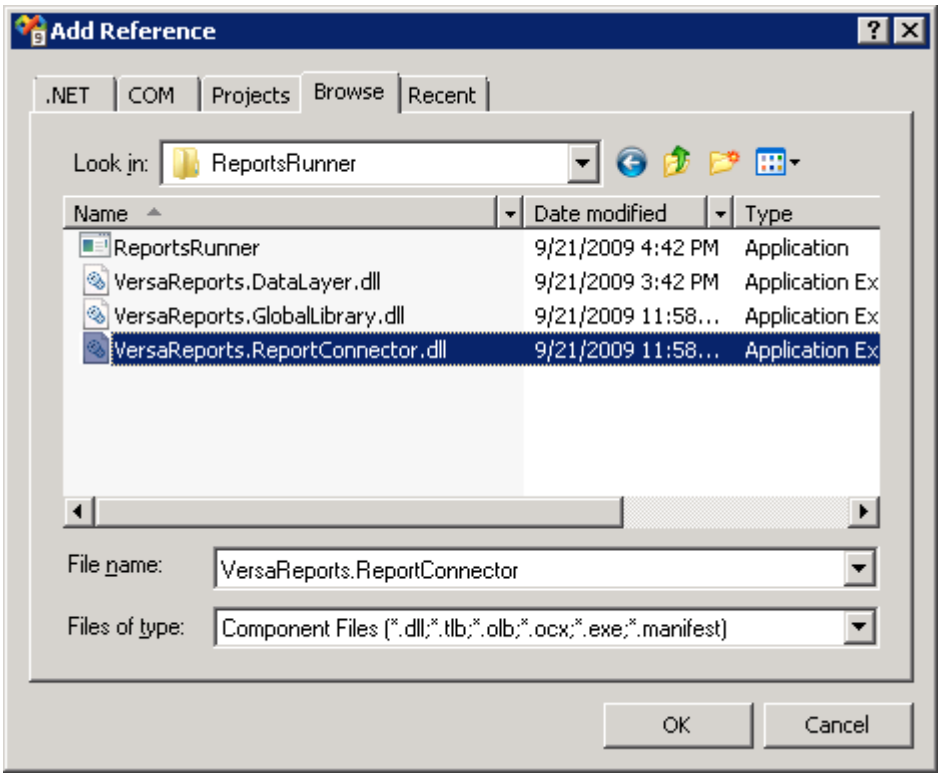
3. Delete the default class file (Class1.cs) by right-clicking on **Class1.cs** and selecting the **Delete** option. Click **OK** on the confirmation dialog.



4. Add XtraReportsDllConnector.cs as an existing item to the project. To do this, first right click the project, point to **Add**, and then click **Existing Item…**

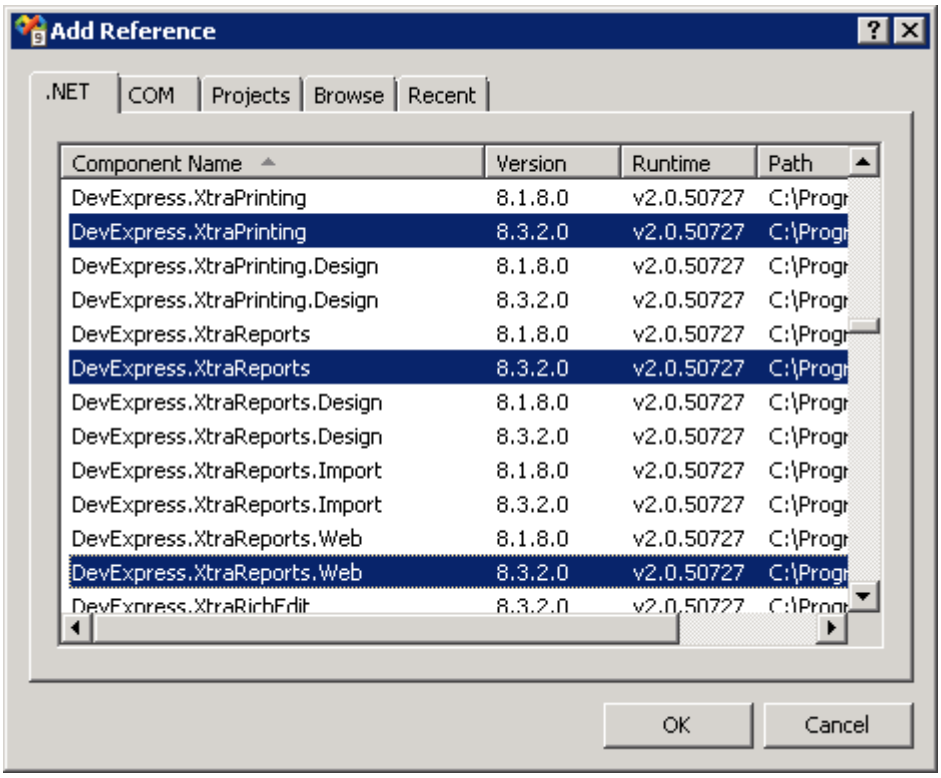


5. Then navigate to the *InstallationDirectory*\Sample Report Connectors\XtraReportsConnector folder and select the **XtraReportsDllConnector.cs** file.



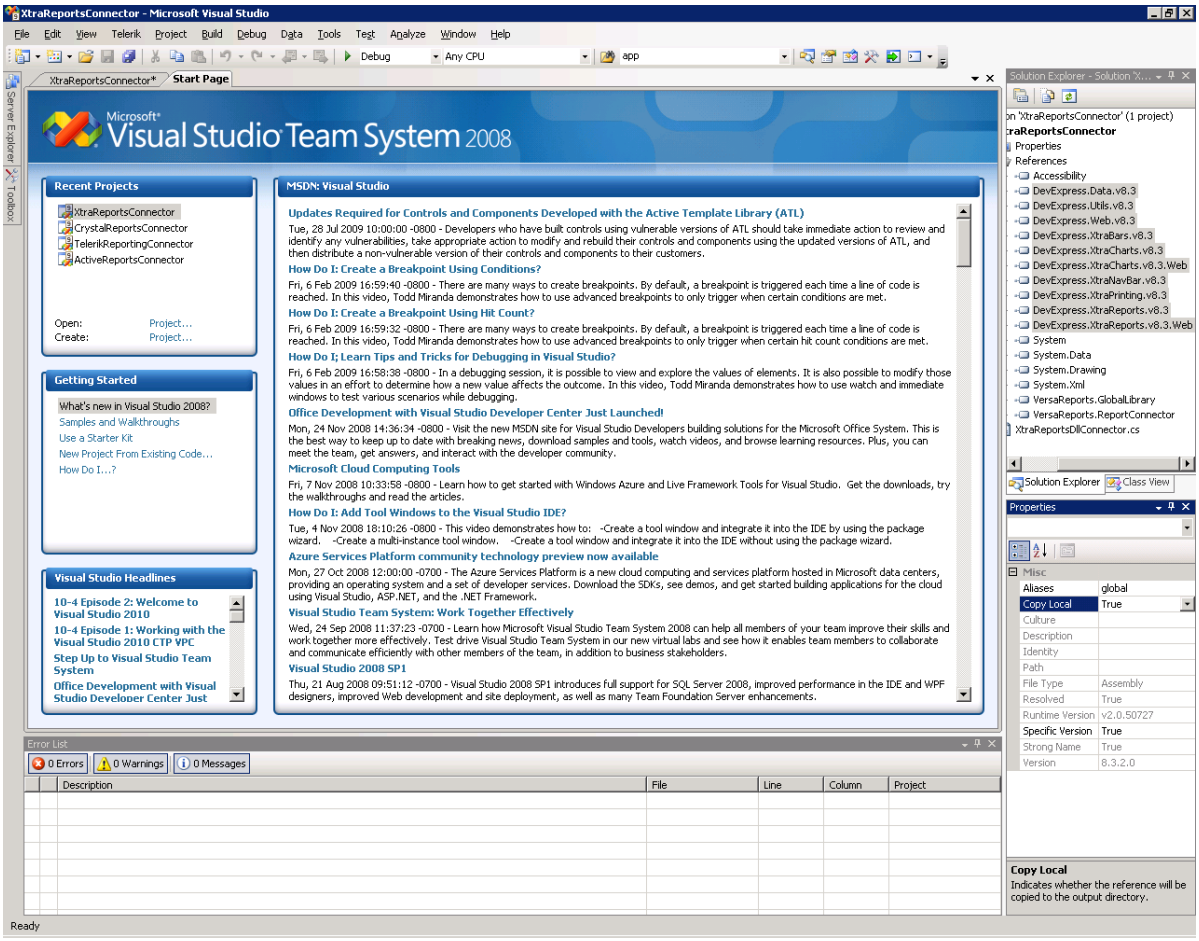
6. Add references to the VersaReports.ReportConnector.dll library (from the *InstallationDirectory*\ReportsRunner folder).

   Do this by right-clicking the project and then clicking **Add Reference**. In the **Add Reference** dialog box, click the **Browse** tab, and then in the **Look In** field navigate to *InstallationDirectory*\ReportsRunner\VersaReports.ReportConnector.dll. Select it and click **OK**.

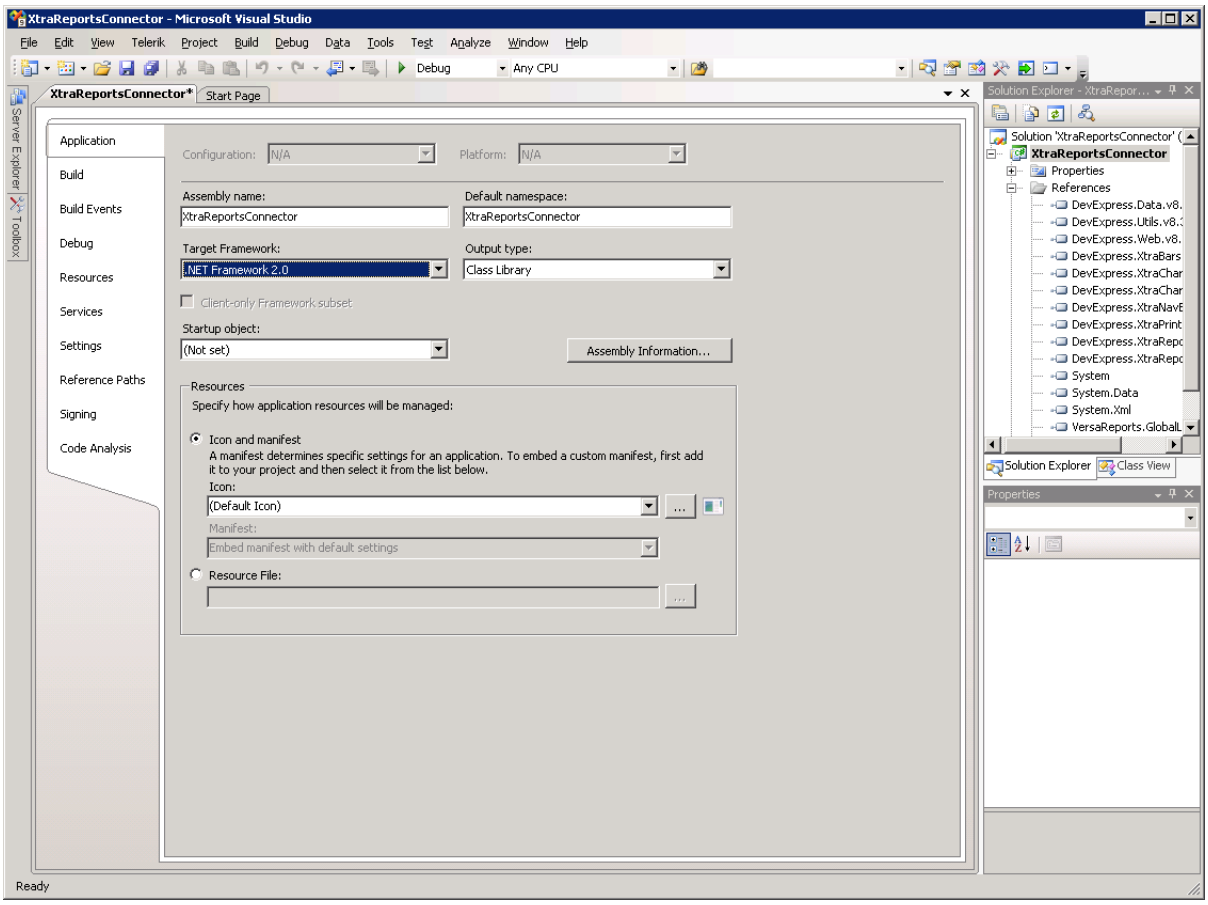


7. Add references to all distributable libraries for the report designer.

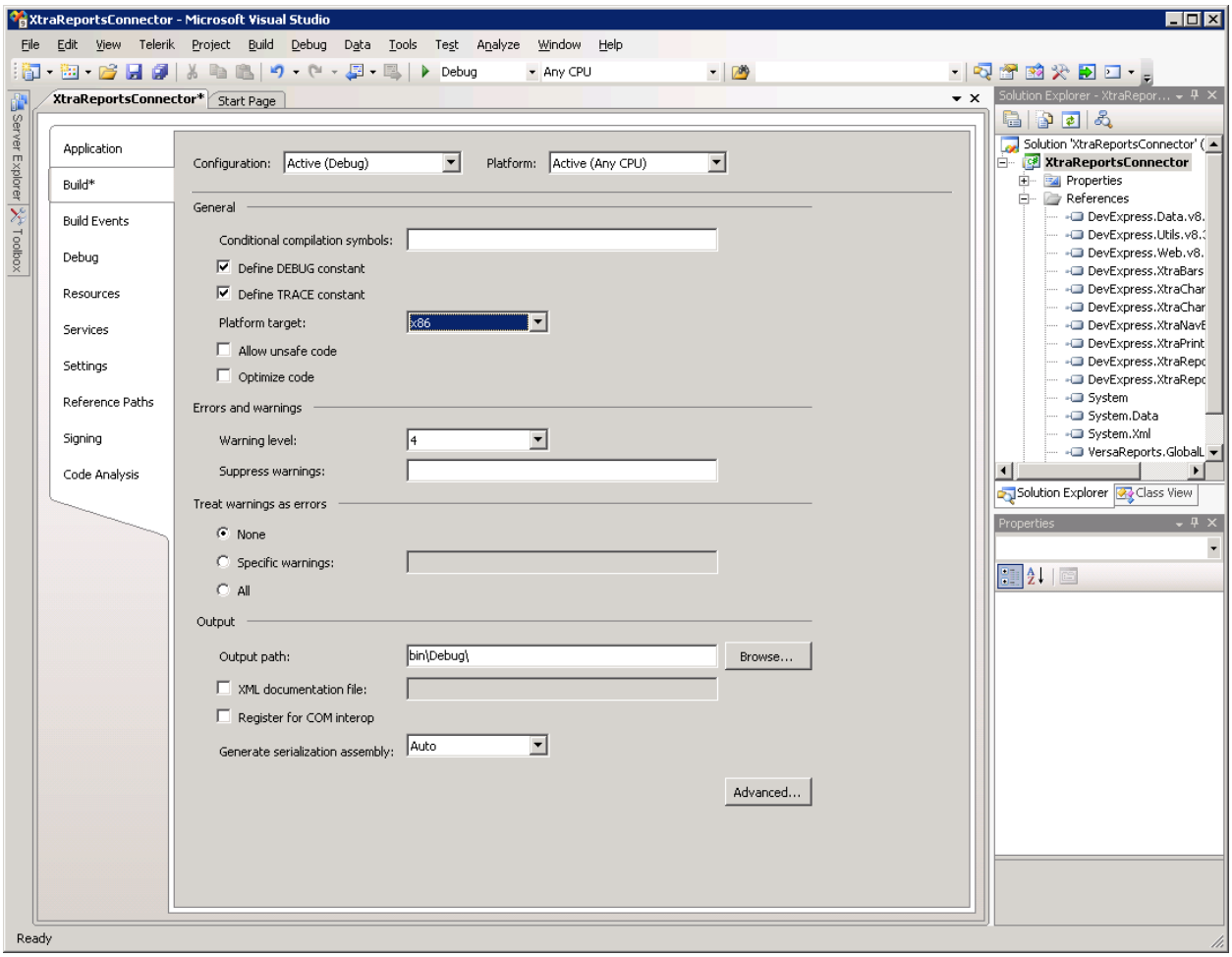
   Do this by right-clicking the project and then clicking **Add Reference**. In the **Add Reference** dialog box, click the **.NET** tab and select the appropriate references. Refer to the report designer documentation for the .NET references to be added. Multi-select them (Ctrl-click each) and click **OK**.



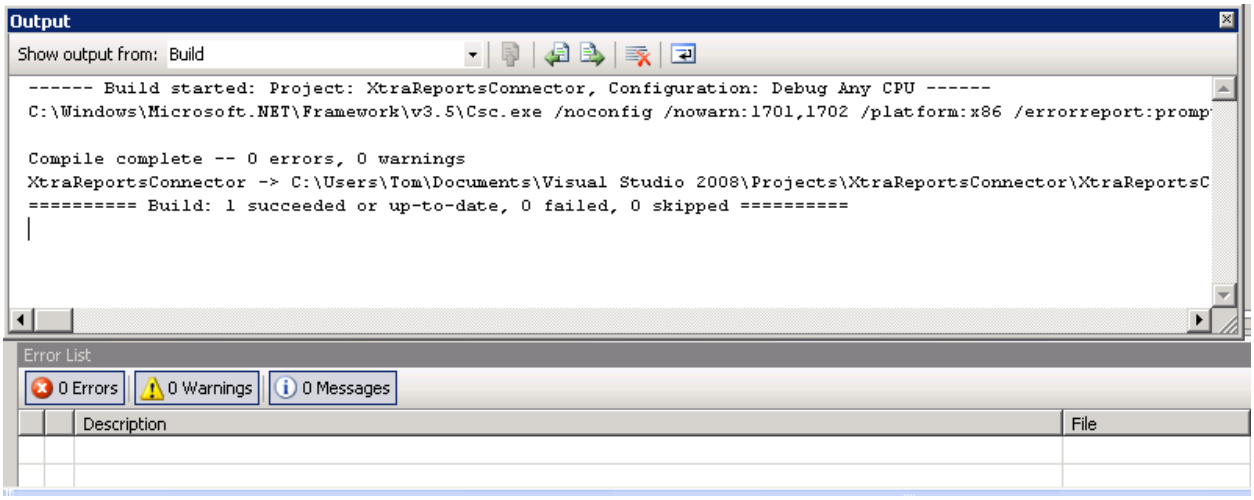
8. Change **Copy Local** to **True** for all report designer libraries you added.



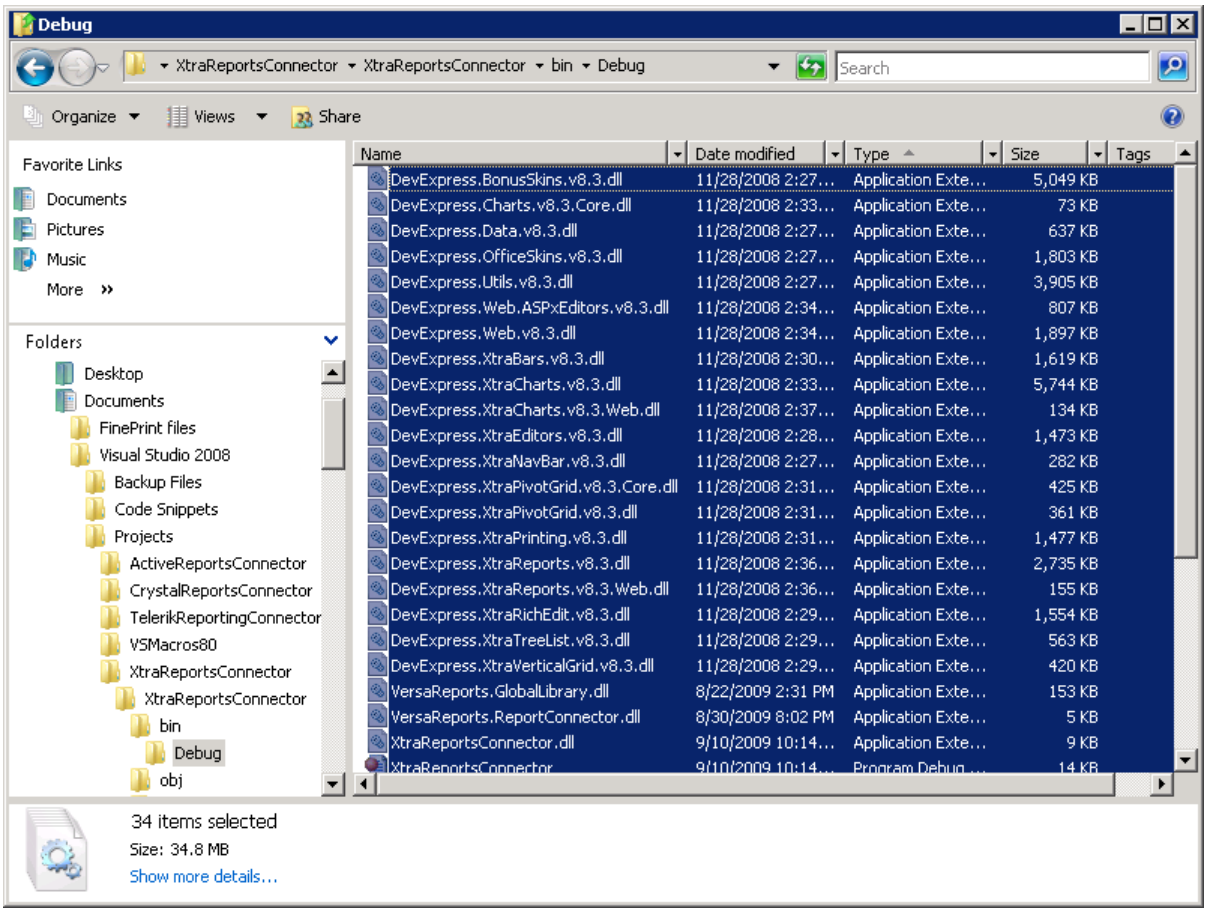
9. From the **Project** menu, click **Properties** and then click the **Applications** tab. Verify **.NET Framework 2.0** is selected from the **Target Framework** list.



10. Click the **Build** tab and in the **Platform target** list, click **x86**.



11. Build the project by clicking **Build *ReportConnectorProjectName*** on the **Build** menu. In our example you would click **Build XtraReportsConnector**.



12. If the build is successful, copy the .dll files from the *ProjectFolder*\Bin\Debug (in our example, XtraReportsConnector\Bin\Debug) to the *InstallationDirectory*\ReportsRunner and *InstallationDirectory*\web\Bin folders.

Appendix B
The report_types.xml File

For all report types that Universal Report Server will support at your installation, an XML block must be created in the *InstallationDirectory*\web\App_Data\report_types.xml file. The basic structure of this file is:

```
<?xml version="1.0" encoding="utf-8" ?>
<report_types>
  <report_type ...>
    <connector ... />
    <report_file ... />
    <supporting_assemblies>
      ...
    </supporting_assemblies>
    <output_formats>
      <format ... />
    </output_formats>
  </report_type>
<report_types>
```

The **report_types** node can contain multiple **report_type** nodes, each one describing a unique report type that Universal Report Server can handle.

The **report_type** node has the following attributes:

- **id** - the unique identifier for this report type. The identifier cannot be changed once reports are registered for this report type.
- **name** - the descriptive name for this report type: for example, "Crystal Reports RPT File".

Each **report_type** node also contains exactly one each of the following nodes:

- **connector** - The **connector** node describes the information about the Report Connector DLL, using the following attributes:
  - **assembly** - the location of the DLL on the server's file system.
  - **class** - the fully-formed (including namespace) class name for the connector. This class must be an implementation of the ReportConnectorInterface class.
- **report_file** - The **report_file** node describes the report file that gets registered for each report. Its attributes are
  - **extensions** - the comma-separated file suffixes (for example, .rpt) for the acceptable types of report files for this connector.
  - **storage** - the location where the report file will be stored. If the value for this attribute is "database", then the report file must be uploaded to the Universal Report Server database for storing between uses. If the value for this attribute is "disk", then the report resides on disk and the person registering the report will need to provide the path on the physical disk to the report file.

    If reports will be stored in-place on disk, the administrator must edit *InstallationDirectory*\web\web.config file and assign the appropriate value to the RootDirectories key in the <appSettings> node. The value is a comma-separated list of directories where report files can be located. IIS must have permission to read the files in these directories and all their folders. An example of a RootDirectories entry in web.config would be:

    ```
    <add key="RootDirectories" value="c:\reports,c:\projects\user_reports"/>
    ```

- **extra_info_prompt** -if the person registering the report needs to provide extra info about the report file, this attribute needs the value of the prompt string that will be shown. In other words, if the report being registered is a DLL file, the extra_info_prompt might contain the string "Enter the Class Name for the Report". If this attribute is not set or set to a blank string, the extra info prompt is not displayed to the person registering reports of this type.
- **pass_connection** - some report types support the ability to allow the registering or scheduling user to change the data connection information. If this report type supports that change – and the Report Connector for the report type must support it – then this attribute is set to "true", otherwise, set it to "false".

❖ **supporting_assemblies** - The **supporting_assemblies** node contains zero or more **add** nodes to enumerate the Report Connector's supporting DLLs that are not loaded into the GAC or into the directory where the VersaReports.ReportConnector.dll file is located. In other words, if your report designer's deployment files are not installed in the GAC and you don't want to copy them from their installation location into both the *InstallationDirectory*\web\Bin and the *InstallationDirectory*\ReportsRunner folders, create **add** nodes in the **supporting_assemblies** node to list those files. For example:

```
    <supporting_assemblies>
      <add assembly="C:\Program Files\Developer Express .NET
v8.3\Sources\DevExpress.DLL\DevExpress.Charts.v8.3.Core.dll" />
      <add assembly="C:\Program Files\Developer Express .NET
v8.3\Sources\DevExpress.DLL\DevExpress.Data.v8.3.dll" />
      <add assembly="C:\Program Files\Developer Express .NET
v8.3\Sources\DevExpress.DLL\DevExpress.Utils.v8.3.dll" />
      <add assembly="C:\Program Files\Developer Express .NET
v8.3\Sources\DevExpress.DLL\DevExpress.Web.v8.3.dll" />
      <add assembly="C:\Program Files\Developer Express .NET
v8.3\Sources\DevExpress.DLL\DevExpress.XtraCharts.v8.3.dll" />
      <add assembly="C:\Program Files\Developer Express .NET
v8.3\Sources\DevExpress.DLL\DevExpress.XtraPrinting.v8.3.dll" />
      <add assembly="C:\Program Files\Developer Express .NET
v8.3\Sources\DevExpress.DLL\DevExpress.XtraReports.v8.3.dll" />
      <add assembly="C:\Program Files\Developer Express .NET
v8.3\Sources\DevExpress.DLL\DevExpress.XtraReports.v8.3.Web.dll" />
    </supporting_assemblies>
```

Note that the **add** node has a single attribute, **assembly**, which contains an absolute path location to a DLL file. You cannot use relative path locations in the **assembly** attribute.

❖ **output_formats** - the **output_formats** node contains a list of **format** nodes that describe the type of report result files that can be created with the Report Connector. A sample **output_formats** block looks like this:

```
    <output_formats>
      <format extension="pdf" name="Acrobat Reader" mime_type="application/pdf"
native="false" icon="images/pdf.png" />
      <format extension="rtf" name="Rich Text File" mime_type="application/rtf"
native="false" icon="images/rtf.png"/>
      <format extension="xls" name="Excel 2003" mime_type="application/vnd.ms-excel"
native="false" icon="images/xls.png" />
      <format extension="prnx" name="Preview-Only Format" native="true"
display_page="XtraReportsViewer.aspx?id={0}" icon="images/report.png" />
    </output_formats>
```

Each **format** node has a number of attributes:

- **extension** - the suffix to add to this type of report file output. This extension is also stored in the database along with the scheduled report definition, so do not change these extensions once you start scheduling reports with this type of output format.
- **name** - the visible name shown to the user in the Report Scheduling Wizard for this output format.

- **mime_type** - the MIME type of the output file, so that it can be properly displayed in the user's browser (this instructs the browser how to display this type of report output). For "native" report outputs, do not include a mime_type attribute.
- **native** - set to "false" if the report output format does not require a special web page to display it; set to "true" if the report designer requires a special web page to display the report output. If set to "true", the display_page attribute must be set.
- **display_page** - for native report formats, specify the URL for the page name along with ?id={0} and Universal Report Server will load that page and pass it an encoded ID for the report to be displayed. See the Configuration Notes for Report Connectors for the appropriate URL to assign to this attribute.
- **icon** - the URL for a small icon to use to represent the type of report in the Report Archives page.

The file report_type.sample.xml is included with the basic Universal Report Server installation and you should use this as a guide to creating the specific report_type.xml that you need for your installation, based on the Report Connectors and features you need for your report administrators and schedulers.

Appendix C
Upgrading URS from a Previous Version

If you are upgrading your Universal Report Server software, this appendix addresses specific steps you must take in addition to or instead of the standard installation steps detailed in the chapters of this Installation Guide.

Since you are performing an upgrade, you may skip the instructions in Chapter 3 since they were done during the original installation.

Follow the steps below to upgrade the Universal Report Server software:

1. Back up your *InstallationDirectory*. When you originally installed Universal Report Server, the URS Setup Wizard suggested a location for the software installation. An example is C:\Program Files\VersaReports\Universal Report Server. If you did not change this folder location, make a backup of the VersaReports folder. If you did change the location, back up that folder.
2. Stop ReportsRunner. If ReportsRunner was started in interactive mode (without the -install option), then from the ReportsRunner window, press Q to stop the program from running. (Make sure the "Q" is capitalized.) If ReportsRunner was started as service (that is, using the -install option), then you do not have to do anything – the upgrade installation program will stop the process automatically.
3. Tell all users on the URS website they must log off and stay off during the installation.
4. Backup your Universal Report Server database on the SQL Server using the SQL Server Management Console program. The installation package for this release of URS will change the database structure slightly.
5. If you have made any changes to the *InstallationDirectory*\web\Default.master file, you must make a backup copy of it. For example, copy it to the same *InstallationDirectory*\web folder and rename it to Default.master.back.
6. Run the Universal Report Server Setup Wizard. Refer to Section 4.2 Installing Universal Report Server on page 10 for instructions.

   Important: Be sure to specify the same installation options and database as the original installation.
7. When the installation wizard is finished, do the following post-installation steps:

   a. The *InstallationDirectory*\web**web.config** has been backed up and replaced. You must edit the settings in the newly installed web.config file to match any post-installation settings you might have made to the file from the earlier release of URS. The backup file is in the same directory as the new one and can be identified by the name web.config.back. (Ignore any other .back extension file names; for example, web.config.1.back.)

   b. The *InstallationDirectory*\web\App_Data**report_types.xml** has been backed up and replaced. You must edit the new report_types.xml file to include any post-installation settings you might have made to the file from the earlier release of URS. The backup file is in the same directory as the new one and can be identified by the name new report.types.xml.back. (Ignore any other .back extension file names; for example, report.types.xml.1.back.).

   c. This release includes new sample report viewer web pages located in the *InstallationDirectory*\Sample Report Viewer folder. For those report designers you are using, copy the appropriate report viewer pages to the *InstallationDirectory*\web folder. The previous version of these viewer pages will not operate with this version of the software. Any post-installation changes you might have made to your report viewer pages might also need to be made again to these new pages.
8. Re-start ReportsRunner. If you were running it as a service, you will need to re-create the service by opening a Command Prompt window, navigating to the *InstallationDirectory*\ReportsRunner folder,

and entering the `ReportsRunner -install` command, as described in Section 5.7 Start ReportsRunner on page 35.

9. Verify that the site is running as described in Section 5.8.3 Run Universal Report Server on page 39. If everything is correct for this upgrade, you should be able to view any reports created by the earlier version of URS.
10. Notify users they may start using the site again.