1

Crystal Reports .NET Overview

Crystal Reports has enjoyed a long association with Microsoft and has shipped with Visual Basic (and subsequently Visual Studio) as the default report writer since 1993. Developers have traditionally had a love-hate relationship with Crystal Reports; they loved the functionality it provided and the free run-time license, but they hated having to upgrade to the latest version to get the features they required. Another complaint was that reports could not be created or modified programmatically; they could be created only through the user interface (UI) with either the developer UI with Visual Studio or the consumer UI with the Crystal Reports retail package.

Just as the release of Visual Studio .NET 2002 represented a significant leap for the Microsoft development platform, the release of Crystal Reports for Visual Studio .NET was also a milestone for the Crystal Decisions development team. Following the Microsoft .NET strategy, they redeveloped the product to take advantage of the .NET Framework and made it a fully featured product in its own right; developers no longer have to wait to upgrade to the latest release to get the features they need.

When Visual Studio was upgraded recently to Visual Studio .NET 2003, the version of Crystal Reports that ships inside the box was also updated. In this chapter, we are going to take a first look at Crystal Reports for Visual Studio .NET 2003 (Crystal Reports .NET), examining how the product is different from other versions of Crystal Reports, how to find and run the sample applications that are included, and where to find the tutorials that will get you up to speed with the product. We will also take a look at the Crystal Reports .NET architecture and learn how it fits into the .NET Framework.

Whether you are an experienced application developer looking to move to Visual Studio .NET or you are developing your first application and have never heard of Crystal Reports, it all starts here.

Chapter 1

What Is Crystal Reports?

In simplest terms, Crystal Reports is a report design tool that allows you to create reports capable of retrieving and formatting a result set from a database or other data source. In addition to simply reading data from a data source, Crystal Reports has its own formula language for creating calculations and includes a number of features that can be used to turn raw data into presentation-quality reports, with graphs, charts, running totals, and so on.

If you look at all of the different types of reports that can be created using Crystal Reports, shown in Figure 1-1, you will find that they are as varied as the developer or end user who created them.

You can create reports that range from a simple list with only a few columns to a complex management report that shows multiple graphs, tables, and Key Performance Indicators (KPIs). The flexibility of the report designer means that it can be used for many different types of output, depending on your needs.

In addition to a powerful toolset for creating reports, Crystal Reports also features a number of Application Programming Interfaces (APIs) and tools specifically created for developers to allow them to integrate these reports into their own applications. To help understand these features and how they are used, we are going to have a brief look at the history of the products leading up to this release of Crystal Reports .NET.

Figure 1-1

A Brief History

In the beginning, a small company in Vancouver called Crystal Services developed a DOS-based reporting add-on for ACCPAC accounting in 1988. A few years later, in 1992, the company released Crystal Reports, touting it as the "world's first Windows report writer," and it wasn't too long after that Microsoft standardized Crystal Reports as the reporting engine for Visual Basic. The rest is history.

Within a year of that historic partnership between Crystal Services and Microsoft, over a million licenses of Crystal Reports were shipped, giving it a foothold within the developer community and ensuring its long-term success. Since that time, Crystal Reports has evolved alongside the available platforms and development tools, moving from floppy distribution to CDs, from 16- to 32-bit, and from a `.dll` print engine to ActiveX control to embedded designer to Automation Engine to .NET Classes.

Over the years, through the transition of the company from Crystal Services to Seagate Software to Business Objects, the user interface for creating reports hasn't changed much; the basic features are still the same, even though the look and feel of the icons and menu bars may change depending on the UI design standards of the day. What have really changed over the years and releases of Crystal Reports are the functions and features that have been developed, culminating in a product that can easily hold its own with just about every other report writer on the market. To have a look at some of those features, we are going to delve into exactly what you can do with Crystal Reports .NET.

What Can You Do with Crystal Reports .NET?

To start with, Crystal Reports .NET includes an integrated Report Designer available within the Visual Studio IDE (shown in Figure 1-2) that you can use to create report files (`.rpt`) to integrate with your application.

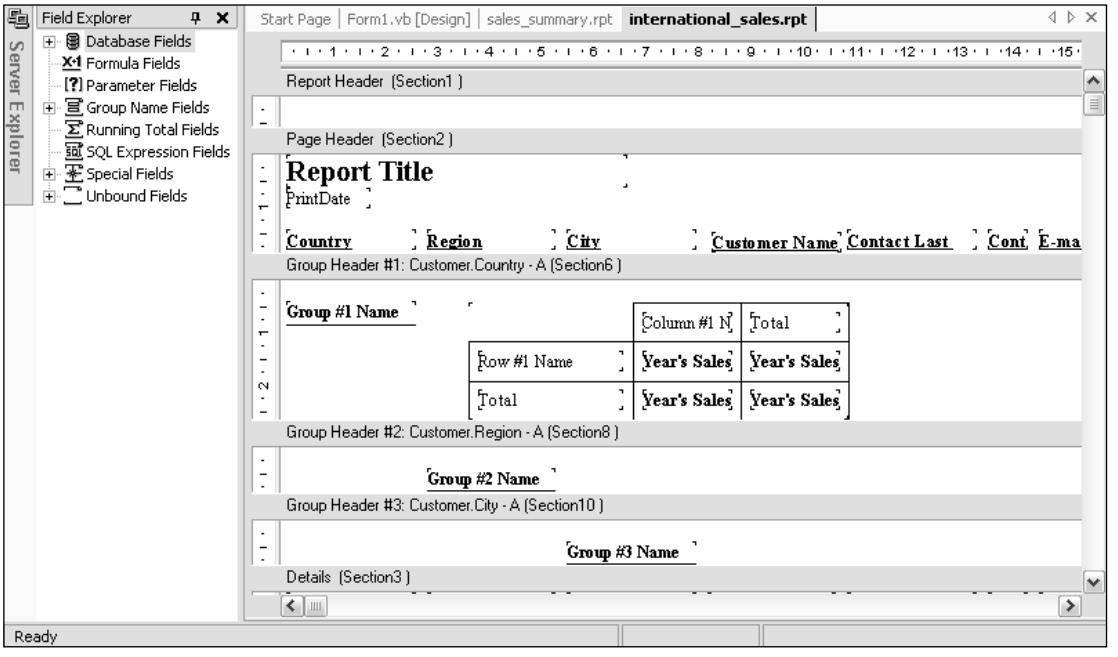

Figure 1-2

This Report Designer (covered in Chapter 3, "Designing Reports") features a number of experts (or wizards) to help you get started creating a report. It will guide you through the report development process, from selecting a data source and the field that will appear on your report, to determining what records should appear.

Once you have a basic report designed, you can then add features like formula fields, running totals, graphs, and so on to make your report design as complex as required. Reports come in all shapes, sizes, and forms. You may want to create a report that can be used to print an invoice from your application, to compile statistics for a management report, or to produce an inventory count sheet.

You don't even have to constrict yourself to a particular size or shape; reports can be created that print shipping labels or address labels and can include bar codes, pictures, graphics, and so on.

> **To get an idea of the types of reports that can be created using Crystal Reports, check out the sample reports available from the Crystal Decisions Web site at http://community.crystaldecisions.com/fix/samplescr.asp.**

After you have created a report, you need some way to display it from your application. Crystal Reports .NET has two different viewers to make this happen. The Windows Forms Viewer (which we look at in Chapter 4, "Report Integration for Windows-Based Applications") can be used with Windows applications to preview any reports you have integrated into your application. It features a rich object model that allows you to control the appearance of the viewer and some aspects of the report at run time.

You can add this viewer to any form in your application, either as the sole content of the form or as one of several form components. You can control the viewer's appearance, changing toolbars, and other visual aspects, even creating your own icons and buttons to control the viewer and its actions, like the viewer shown in Figure 1-3.

For Web-based applications, there is also a Web Forms Viewer (Chapter 5, "Report Integration for Web-Based Applications") that has similar functionality and allows you to view reports you have integrated into your Web applications. You can add this viewer to Web pages within your application and show a report either on its own page, in a frameset, or like the report in Figure 1-4, side by side with other application content; it is up to you.

For complete control over your report, regardless of whether you are viewing it through the Windows or Web Forms Viewers, you also have access to the Report Engine (see Chapter 9, "Working with the Crystal Reports Engine"). This will allow you to control even the most minute aspect of your report before you view it using one of the aforementioned viewers. Using the Report Engine, you can control the report's formatting and features, set database credentials, and call direct methods to print, export, and so on.

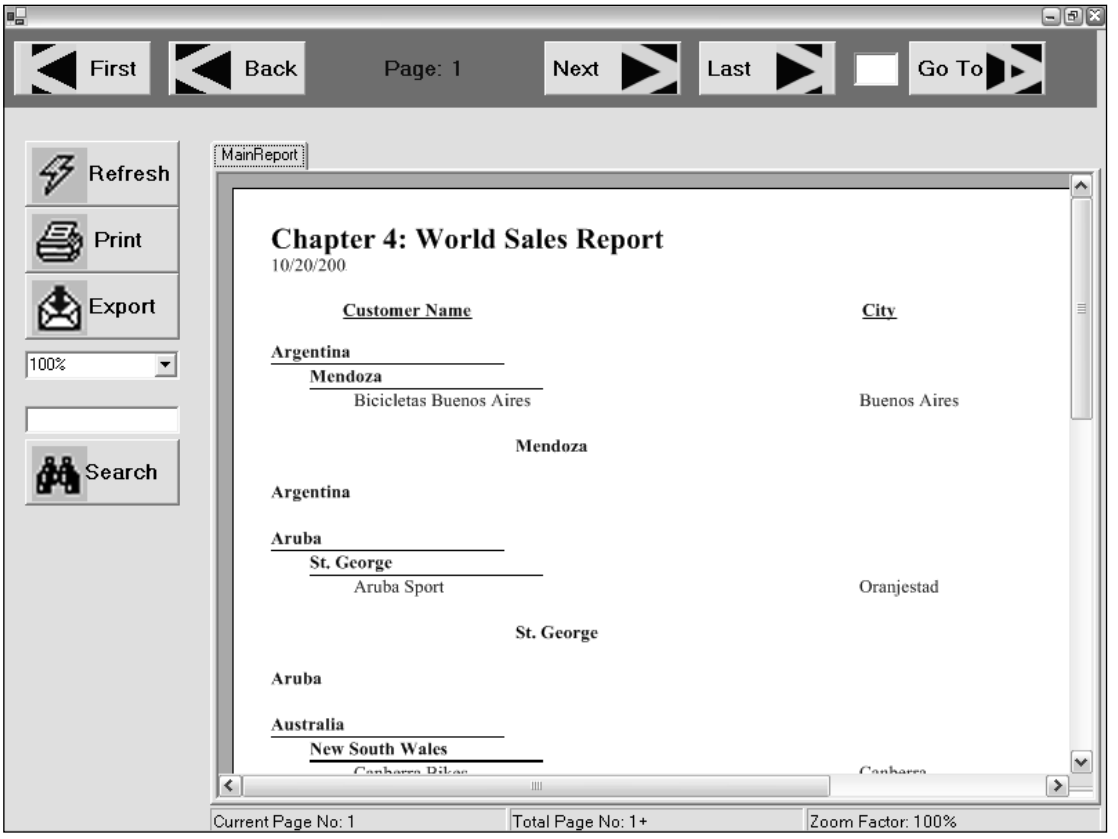

Figure 1-3

For creating distributed applications, Crystal Reports .NET has a number of features specifically designed for creating and consuming XML Report Web Services, either through the generic Web Service that ships with Crystal Reports .NET (which allows you to utilize a report without having to publish it as a Web Service) or by creating your own Web Services from report files, like the one shown in Figure 1-5. In any case, Chapter 6, "Creating XML Report Web Services" will guide you through the process of both creating and consuming XML Report Web Services.

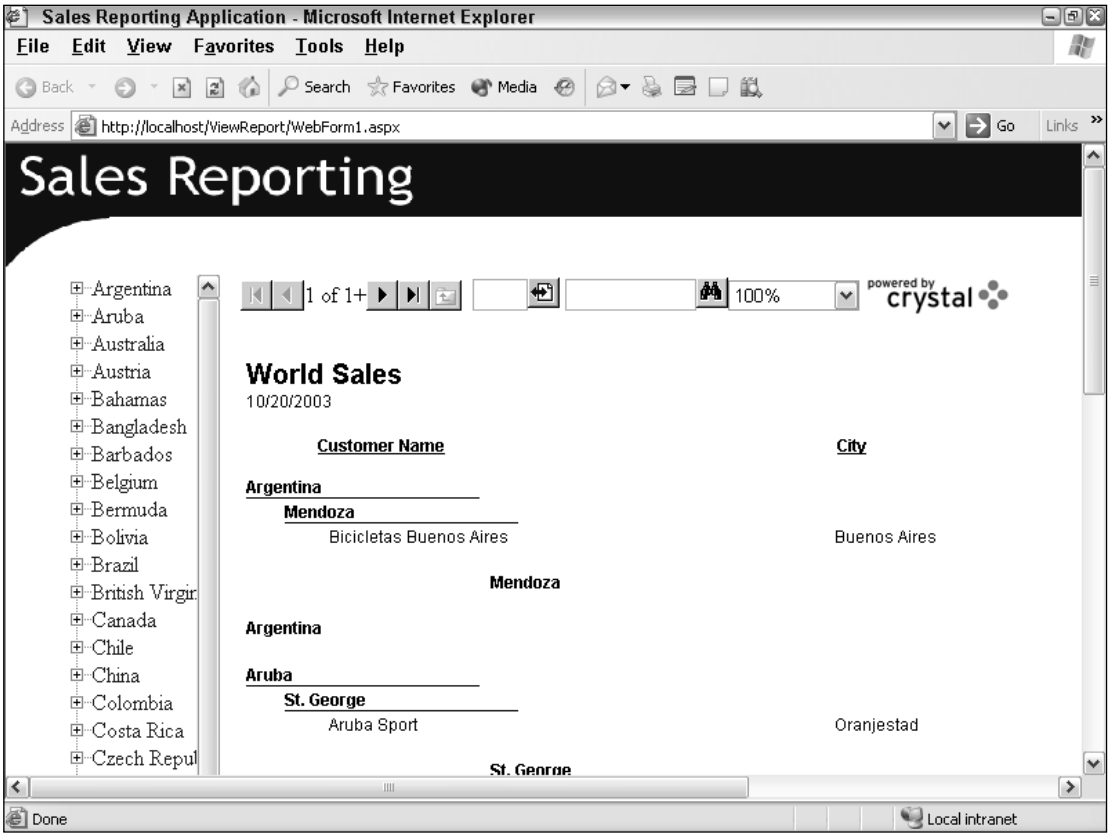

Figure 1-4

Figure 1-5

Crystal Reports .NET is also tightly integrated with Crystal Enterprise, a report scheduling and distribution system that provides a true multi-tier back-end processing platform for reports and allows you to use a scheduling engine and distribution framework to distribute reports to thousands of users.

And finally, there are a number of tools that have been included for distributing reports with your application, including updated merge files for this version of Crystal Reports. In Chapter 10, "Distributing Your Application" we will look at how to use these tools to successfully deploy your own applications and how to trouble-shoot installation and setup problems.

How Is Crystal Reports .NET Different from Previous Versions of Crystal Reports?

Crystal Reports .NET 2003 is an updated version of the report writer integrated with Visual Studio .NET 2003 and has been updated from the original version that first shipped with Visual Studio .NET 2002. This version is a special OEM version of Crystal Reports that is available with the Visual Studio .NET suite. It shares some common features with the retail version of Crystal Reports and was built on the Crystal Reports 8.x technology, but components of Crystal Reports .NET have been rewritten using C# and are designed to take full advantage of the .NET Framework.

Integrated Design Environment

Unlike the standalone versions of Crystal Reports, Crystal Reports .NET is part of the Visual Studio .NET Integrated Development Environment (IDE). Using the integrated Report Designer, you can create or modify reports from within the Visual Studio .NET IDE. If you have used the Report Design component from previous versions of Crystal Reports, the concept will be familiar.

Any Language, Any Time

Crystal Reports .NET follows the Visual Studio .NET mantra of "any language, any time" and is not too picky about the language you use to write reporting applications. You can use any of the .NET languages (VB, C#, J#, C++, and so on) to develop reporting applications or integrate reports into your existing applications.

For all .NET languages, the Report Designer remains the same, and the code used to control viewing reports and report engine calls will vary only slightly between languages, due to different syntax rules and conventions. For example, if you were binding a report to a Web Forms Viewer in VB .NET, the syntax would look something like this:

```
crystalReportViewer1.ReportSource = my_Report1
```

The same code can be ported to C#, with only one rather minor syntax change:

```
crystalReportViewer1.ReportSource = my_Report1;
```

So for developers who are creating applications with different languages or deployments, the concepts are the same, regardless of the language used.

Integration Methods

If you are new to Visual Studio .NET in general and have not used Crystal Reports .NET before, another thing you'll note is that the way that we integrate reports into both Windows and Web applications is different. Before Visual Studio .NET, Crystal Reports developers had a number of different integration methods they could choose from for Windows applications, such as an ActiveX control, Automation Server, or direct calls to the Crystal Reports Print Engine. For Web applications, Crystal Reports shipped its own Web component server and report viewers, allowing developers to integrate reporting into their applications.

Although the report integration solution provided for Windows development seemed to make most developers happy, the Web integration provided with Crystal Reports left something to be desired. There were inherent problems with configuration, scalability, and reliability, meaning that the Crystal Reports Web development platform could not be used to create scalable enterprise applications.

With the introduction of Visual Studio .NET, it was possible to bring both Windows and Web development into the same framework. The Crystal Report Engine is now a COM+ object wrapped around an updated version of the Crystal Reports Print Engine you may have worked with in the past. The Report Engine can be used to customize features at run time and also takes care of report processing.

When working with Crystal Reports for Visual Studio .NET, you have a choice of either leaving the report on the local machine (using that machine's resources to process and display the report results using the Windows Forms Viewer), publishing it to a Web server (using the Web Forms viewer), or publishing it as a Report Web Service that can be consumed and viewed by either the Windows or Web Forms Viewer.

> *Each of these integration methods will be covered in its own chapter, starting with Chapter 4, "Report Integration for Windows-Based Applications."*

Ease of Use

Integrating a report into a Windows application is as simple as dragging the Crystal Report Viewer from the toolbar onto a Windows Forms Viewer and binding the viewer to a report file. (You could also create a report using the integrated designer.) The Crystal Report Viewer provides a feature-rich client for viewing reports, and at design or run time you can customize the appearance and options of the viewer, pass parameters, set properties, and so on.

For Web development, there is also the Web Forms Viewer, which communicates with the Report Engine (either on the local machine or on a remote server) to display a report page in DHTML format. This allows you to quickly integrate reporting into your Web applications; there are no runtime files required, and the report processing can be performed on the server.

Building Enterprise Applications

In addition to these enhancements, Crystal Decisions has also released Crystal Enterprise — a scalable, platform-independent report distribution, scheduling, and processing engine that can be used in conjunction with Crystal Reports and Crystal Reports .NET. It provides the back-end muscle to create applications that can support hundreds of users for both real-time and scheduled reports with a clustered, multi-server architecture that can span Windows, Linux, and Unix platforms.

Reports that have been published to the Crystal Enterprise framework can be accessed directly from within Visual Studio .NET and integrated into your application.

In addition to providing a scalable, multi-tier back end for reporting applications, Crystal Enterprise also has its own security layer (which can use Windows NT authentication, LDAP, and so on), internal structures (folders, objects, and rights), and scheduling engine, as well as distribution capabilities that can be used to build complex reporting applications without have to reinvent a solutions architecture just for reporting.

For example, if you needed to create an application that generates a report every week in PDF format and sends it as an e-mail attachment to 10 different users, you could create that functionality within your own application or you could use the inherent scheduling and distribution capabilities within Crystal Enterprise to make a handful of API calls to do this for you.

Crystal Enterprise includes .NET assemblies that give you quick access to all of the properties, methods, and events required to work with the Crystal Enterprise framework. Leveraging the functionality that is included by default with Crystal Enterprise, you can quickly create robust reporting applications in a fraction of the time it would take you to code these features by hand in your own custom application.

Another key area where Crystal Enterprise earns its money is with its clustering technology and multiple-server architecture; imagine in our earlier example that there are now 10 reports that go to 100 different people each day with a copy of the report and a link back to where they can view and search the live report.

The clustering within Crystal Enterprise ensures that these jobs get run regardless of what servers are up or down, and the distributed architecture means that you can add multiple servers to share the processing workload, including servers tasked specifically to run scheduled reports and process on-demand requests.

Although the cost of Crystal Enterprise may be off-putting to some developers, its integration with Crystal Reports .NET and its distributed architecture — which is beyond the scope of this book — will ensure that you have the scalability you need when your reporting application that serves 10 suddenly needs to serve 10,000.

Report Architecture

When you look at Crystal Reports .NET, one of the immediate differences between this version and previous incarnations of the product is its ability to create multi-tier reporting applications. In the past, most Windows applications used a two-tier approach with Crystal Reports, where reports ran on the local machine on which the application was installed.

With the introduction of Crystal Server for version 4.0 of Crystal Reports, a first attempt was made at developing a client-server version of Crystal Reports; but it wasn't until 1994, when Seagate Software acquired Crystal Reports and the corporate scheduling product Ashwin — which could be used to schedule programs processes and so on — was introduced, that multi-tier report applications became a reality.

The combination of the two products was first introduced in 1995 as Crystal Info and later renamed Seagate Info. Through the Seagate Info SDK, an additional processing tier was introduced to developers, with a server-based architecture that allowed reports to be run on a separate server and then returned to the client.

Although the Seagate Info SDK seemed like a good idea, developers were slow to adopt the technology and looked for other ways to create multi-tiered applications.

This led Crystal Decisions to rethink their product roadmap, and using the basic technology and architecture from Seagate Info, they created Crystal Enterprise, which was initially released in 2001. Two of the core features of Crystal Enterprise were an open architecture and a powerful SDK that allowed developers to integrate Crystal Enterprise functionality (scheduling, multiple-servers, security, and so on) into their own applications. Since that initial release, Crystal Enterprise has been grown from strength-to-strength to become a robust, scalable platform for delivering Crystal Reports.

So for Visual Studio .NET developers, the introduction of Crystal Reports .NET provided a wealth of tools that could be used to build scalable applications, from simple applications integrating basic reporting, to functionality, to complex reporting applications that serve thousands of users. With the update for Visual Studio .NET 2003, Crystal Reports .NET provides an even more stable platform for a wide variety of reporting applications. These generally fall into one of the following two categories: single-tier and two-tier.

Single-Tier

Crystal Reports integrated with applications created in previous versions of Visual Basic were usually deployed as single-tier applications. In a single-tier application, a developer would use one of the various integration methods to combine Crystal Reports within their application and would then distribute the report file and all of the Crystal Reports `.dll` and runtime files required to make the application work. When a report was run, it ran locally as a thick-client application, using the resources of the machine where the application was installed.

With Crystal Reports .NET, you can still create single-tier (sometimes called *fat*) applications and distribute the runtime files required to run and view a report. Some of the limitations found in applications created with previous versions of Visual Studio tools will still apply, including the need to redistribute the report file if any changes are required. A much better solution is to consider applications with two or more tiers.

Two-Tier

Most Web applications created with Crystal Reports .NET are considered two-tier applications, as shown in Figure 1-6. In the first tier, a Web application makes a request for a report and the report is processed on the Web server that hosts the application.

This architecture provides definite advantages over a single-tier application, including off-loading of the report processing and viewing to a server and a publish-once mentality for publishing a single copy of a report to a Web server that can be accessed by multiple users. However, with this type of two-tier architecture, your application will be limited by the number of users that can physically connect to a single Web server, and report processing will add a definite increase to this server's work load if used heavily for viewing reports.

Now, let's move on to an even better solution with an even thinner client.

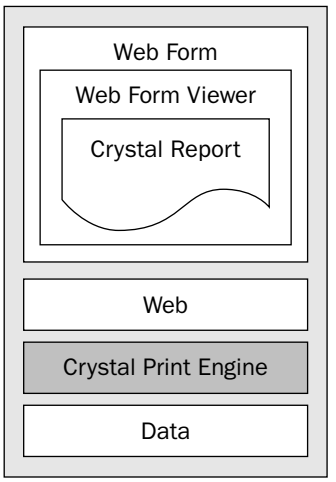

Figure 1-6

Three-Tier

A true three-tier reporting application, like the one shown in Figure 1-7, can be (but doesn't have to be) created using XML Report Web Services (covered in Chapter 6, "Creating XML Report Web Services"). A Report Web Service is a Crystal Report that has been exposed as a Web Service to be used (or consumed) by an application. Applications can connect to a Report Web Service, and the underlying report can be viewed using either the Web or Windows Report Viewer. This provides all of the functionality (view, drill-down, and export) found when integrating reports into a single or two-tier application, but with the report running on a server behind the scenes, the lightest client resources are required for actually viewing a report.

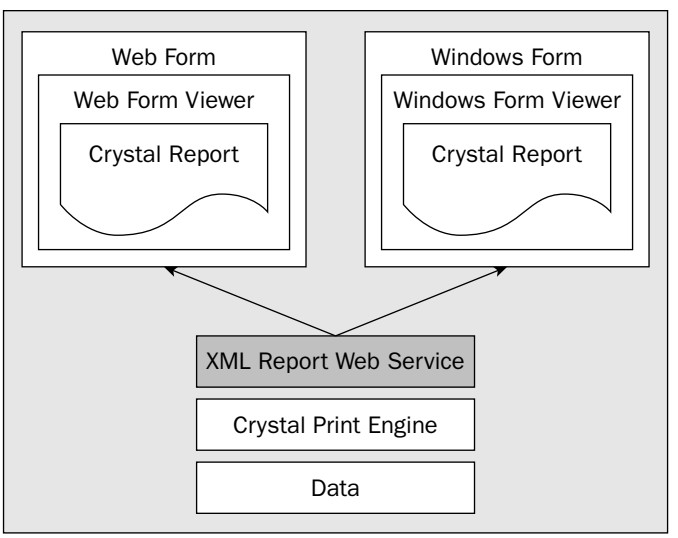

Figure 1-7

In addition to being able to expose reports as Web Services for internal users, you can also publish Report Web Services to users external to your organization, providing a method for external users to access data held within your own data sources.

Multi-Tier Applications

When working with applications that are to be deployed to large numbers of users, you will probably want to consider moving to a multi-tier architecture, which is just a generalization of the three-tier concept and is shown in Figure 1-8, where components can be added as the application user base grows.

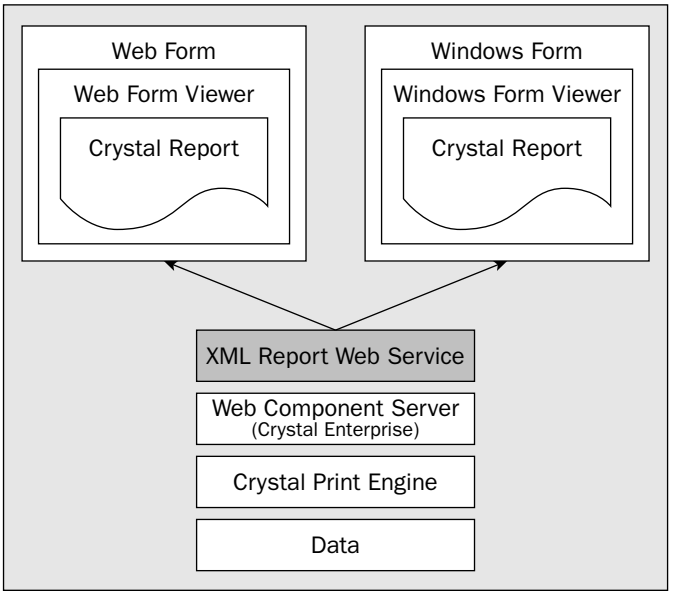

Figure 1-8

Crystal Enterprise is a Web-based, standalone solution for secure report delivery and distribution that can be integrated with Crystal Reports .NET. From within the Visual Studio .NET environment, you have access to the reports stored in the Crystal Enterprise framework and to a rich object model that exposes all of the Crystal Enterprise features and functionality (scheduling, security, and e-mail distribution) for use in your own application.

> **For more information on Crystal Enterprise, check out www.businessobjects.com/products/reporting/crystalenterprise**

Report Designer

You can use the Crystal Reports Designer, shown in Figure 1-9, to create a report from scratch, or you can use a number of experts (similar to wizards) to help you get started. The interface is similar to the retail versions of Crystal Reports, and it shares enough similarity that existing report developers should have no problems making the transition to the .NET version. With that said, there are some specific options and features that are unique to this version.

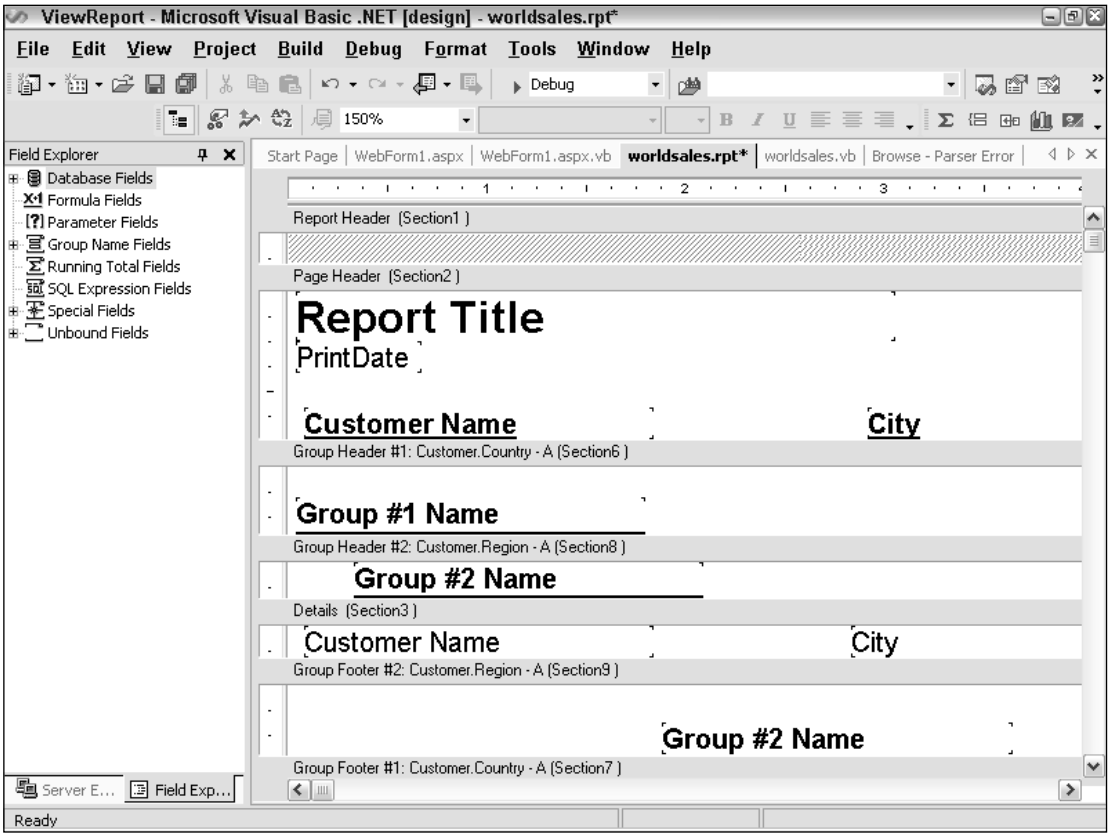

Figure 1-9

To start, Crystal Reports .NET has extended support for a number of data sources, including ADO .NET, OLE DB (ADO), ODBC (RDO), Access/Excel files (DAO), Crystal Field Definition files (from previous versions of Crystal Reports), and XML. When working with these data sources, Crystal Reports .NET can utilize either a *pull* or *push* mode of data retrieval.

To create a report that pulls the required data, you can create a report from a data source just as you normally would and let Crystal Reports handle writing the SQL statement, submitting the statement to the database, retrieving the records, formatting and displaying the records, and so on. This is the most common mode of integrating reports into applications and does not require any additional coding.

In push mode, a report can be created from a data source and used within your application, but it is the application itself that is handling the hard work of connecting to the database, populating an ADO .NET (or other) recordset and then pushing that recordset to the report. From that point, Crystal Reports will format and display the records it has received.

This method of integration requires more manual coding, but it provides more control over the `dataset` and report processing. Using the push mode to retrieve the data for your report means that you can use optimized SQL and stored procedures via ADO .NET to share database connections with other transactions that occur within your application, for example.

Incompatibilities

When using the Crystal Reports Designer available in Visual Studio.NET, you'll notice that there are a number of features that are available in the retail versions of Crystal Reports but are not supported here. A list of these features has been included for your reference:

- Geographic mapping is not supported in Crystal Reports for Visual Studio .NET. Map objects in Crystal Reports are implemented through third-party technology provided by MapInfo, and this has not yet been ported over to the .NET Report Designer. If you want to use existing reports that have maps with Crystal Reports for Visual Studio .NET, you can still do so, but the map objects will appear blank.
- OLAP (Online Analytical Data Processing) data sources and the grids that display OLAP information within a report are also not supported. If you are using an existing report that displays an OLAP grid, this area will be shown as a blank.
- Crystal Dictionaries, Crystal Queries, and Seagate Info Views are not supported. If you need to use an existing report that is based on any of these file formats, you will need to recreate the report directly from the database or data source itself.

Up until now we have looked only at previous versions of Crystal Reports. In August 2002, Crystal Decisions released Crystal Reports 9.0, which shares the same file format as Crystal Reports .NET and recently updated the product to version 10.0. It includes a standalone Report Designer, which does not require Visual Studio, as well as an updated Report Designer for use within the Visual Studio .NET environment. Thus you could have someone else create reports for your application without having to train them on how to use Visual Studio .NET.

Also included are new components for use with .NET — including increased data access, more productivity features, and a mobile viewer with associated tools that works with the .NET Mobile Internet Toolkit. For more information on Crystal Reports 10.0, visit www.businessobjects.com/products/reporting/crystalreports/.

Crystal Reports .NET Benefits

Now that we have looked at some of the differences among versions of Crystal Reports and at some of their uses and limitations, we need to have a look at some of the reasons you should be excited about this version and how your applications can benefit from the features we talked about earlier.

Leverage Existing Development and Skills

Crystal Reports can leverage the existing reports you have created, regardless of version. If you already have a suite of reports created in version 7.0, for example, you can quickly import them into Crystal Reports .NET, and they will be ready to be integrated in your application. In addition, the report design process remains the same, with a number of experts to guide you through report design and the same familiar design concepts, formula languages, and features you have used in previous versions. A word of warning: You can import reports from older versions of Crystal Reports .Net, but not the other way around; once you have opened or edited a report in Crystal Reports .NET, it uses a Unicode file format that is incompatible with previous versions.

Tight Visual Studio .NET Integration

From within Visual Studio, accessing a new report is as easy as selecting Project → Add New Item and then selecting Crystal Report. There is no need to open a separate application to design reports, and all of the reporting features are available to you, allowing you to programmatically control the look and feel of a report, how it is processed and viewed, and so on.

Windows and Web Report Viewers

For a feature-rich report viewing experience, Crystal Reports .NET includes a report viewer for Windows Forms, which has been built using the Windows Forms Classes and provides all of the functionality users have come to expect from Crystal Reports, including drill-down, search, exporting, and so on. In addition to a robust report viewer for Windows Forms, Crystal Reports .NET also includes a thin-client report viewing control for ASP .NET, providing most of the functionality found in the Windows viewer in a zero-client (meaning no client is downloaded or installed) DHTML environment, with no additional plug-in or viewer to download.

Easy Deployment

Crystal Reports .NET includes a number of merge modules to make creating setup projects easier. Instead of manually determining the required `.dll`s and other Crystal-related components, you can simply add one of the merge modules listed here to a setup project:

- `Crystal_Managed2003.MSM`

  For installing the Crystal Reports .NET managed components, including:

  - `CrystalDecisions.CrystalReports.Engine.dll`
  - `CrystalDecisions.Web.dll`
  - `CrystalDecisions.Windows.Forms.dll`

- `Crystal_Database_Access2003.MSM`

  For installing all of the database drivers and all non-managed components (Charting, Export Formats)

- `Crystal_Database_Access2003_enu.MSM`

  For installing select language-specific components

- `REGWIZ.MSM`

  For tracking registration details and license keys

One of the most common errors when deploying a Crystal Reports .NET application is forgetting to change the `LicenseKey` property in your setup project. This must be set or an error involving `keycodev2.dll` may occur.

In addition to the merge modules listed earlier, you may need to include the `VC_CRT` and `VC_STL` modules if you are reporting from ADO recordsets, as the Crystal Reports database driver `crdb_adoplus.dll` relies on the files within these modules.

For more information on deploying your Crystal Reports .NET application, go to Chapter 10, "Distributing Your Application."

ADO .NET

With the introduction of ADO .NET, data access has become much easier, and Crystal Reports .NET can take advantage of ADO and the ADO .NET dataset. Instead of having to work out how to access various data sources, Crystal Reports .NET can simply access the ADO .NET dataset as the source for any report you may create.

XML Report Web Services

For sharing reports and creating tiered applications, XML Report Web Services are invaluable. Within the Visual Studio IDE, you can create a Web Service from a report file with two clicks. From that point, Report Web Services can be exposed to users inside and outside of your organization and can be consumed using one of the new viewers included with the product. To optimize the report pages coming over the wire, XML is used to send the report a page at a time to either the Windows or Web Report Viewer, which makes reports viewed from Web Services quick and responsive.

Installing Crystal Reports .NET

Crystal Reports .NET ships as a component of Visual Studio .NET and can be installed from the common Visual Studio .NET setup utility. If you are installing Visual Studio .NET for the first time, you may need to complete the Windows Component Update shown in Figure 1-10 before you can begin. The setup utility will look at your current configuration and determine whether you need to update any files or applications. If required, setup will guide you through the update process.

After you have completed the component update, you can install Visual Studio .NET. The option to install Crystal Reports for Visual Studio .NET can be found under the Enterprise Development Tools options when selecting installation components. By default, when you select the Crystal Reports option, all of the related components will be installed as well, as shown in Figure 1-11:

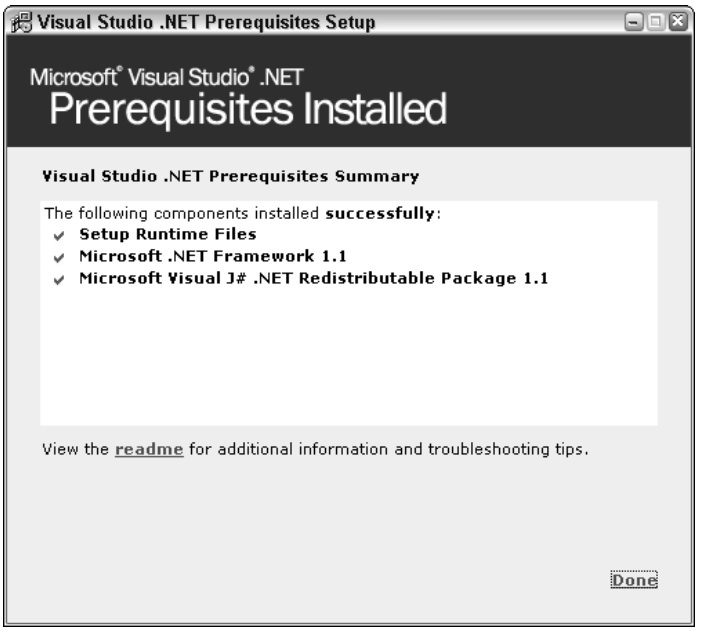

Figure 1-10

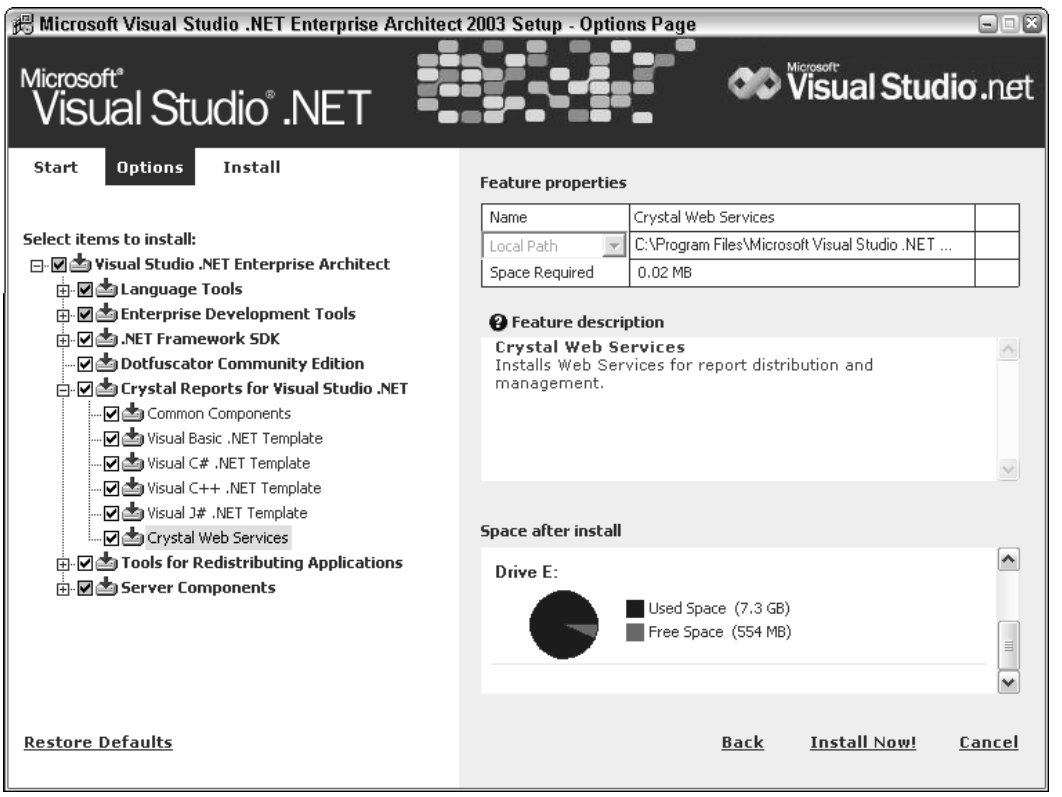

Figure 1-11

Once it is installed, the Crystal Reports icon will appear on the Visual Studio .NET splash screen, and from the Add New Item menu, you should see the option to add a new Crystal Report, as shown in Figure 1-12.

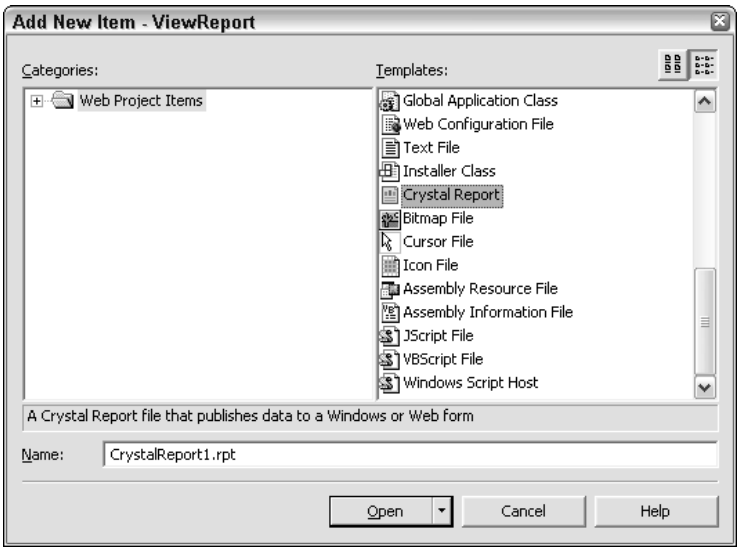

Figure 1-12

With the installation complete, we can jump right into looking at the samples that are installed with the product and learning about Crystal Reports .NET.

Learning from Sample Applications

Crystal Reports for Visual Studio .NET ships with a number of sample applications to help you get started. The majority of sample applications are simple, but each demonstrates some aspect of report integration, features, or functionality and provides a good learning resource if you are just starting out with Crystal Reports .NET or are new to this version.

Installing Sample Applications

The Crystal Reports .NET sample applications are installed by default and can be found in the Crystal Reports directory where you have installed Visual Studio .NET. These samples are in self-extracting files that you will need to run before you can open the samples within Visual Studio .NET. These

samples, shown in Figure 1-13, are located at `X:\Program Files\Microsoft Visual Studio .NET 2003\Crystal Reports\Samples\Code` (where `X:` is the drive where you have installed Visual Studio .NET).

There are two sets of sample applications available, both of which are bundled in self-extracting files—`WebForms.exe` and `Winforms.exe`. The first, which will extract to a folder marked WebForms, is written using ASP .NET and demonstrates the use of the Crystal Reports Web Forms Viewer. To view these samples, you will need to use IIS Internet Services Manager to create a virtual directory called `CRSamples` that points to the directory where you extracted the sample files. From that point, you should be able to access these samples from `http://localhost/CRSamples`.

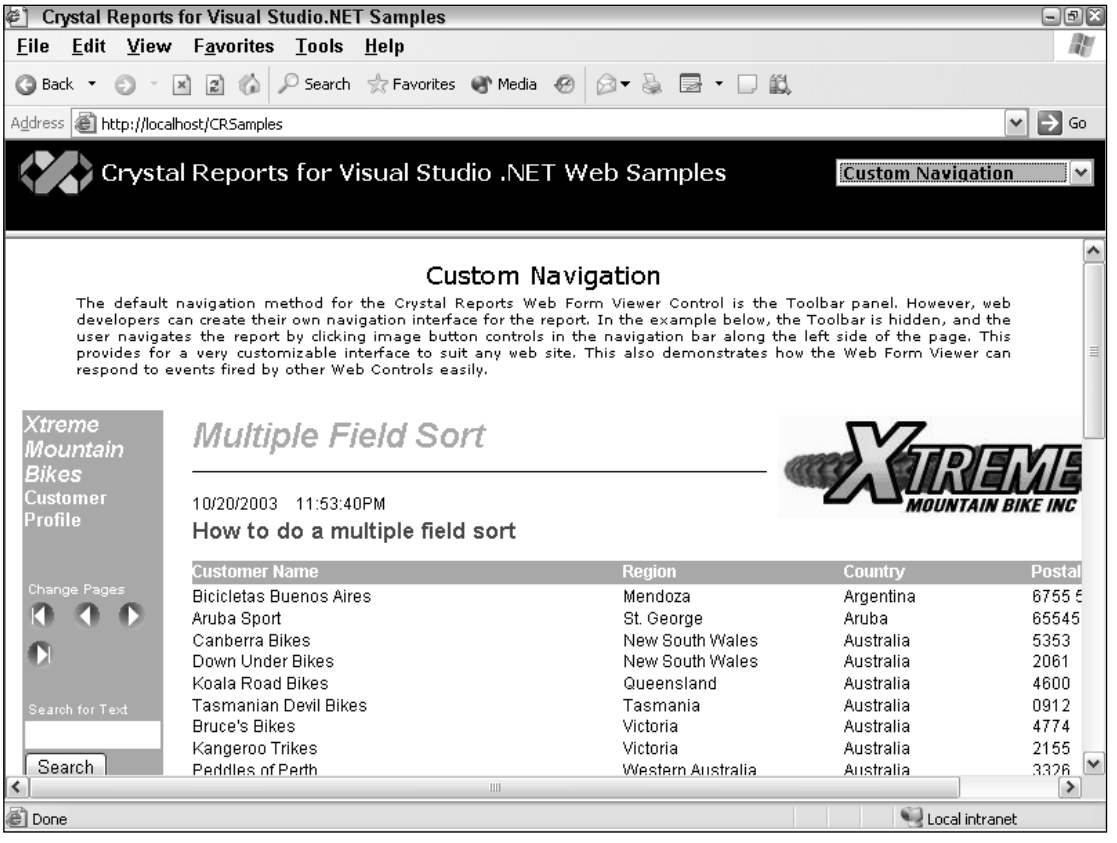

Figure 1-13

The following samples are included in this set:

Sample	Description
Simple Page	The Simple Page sample application demonstrates using the Web Forms Viewer with one of the sample reports (World Sales). This sample demonstrates drilling down into the details of the report and some of the other features of the report viewer, including the report group tree, the viewer toolbar, and the page control options.
Custom Navigation	This sample demonstrates the amount of customization that can occur within the Web Forms Viewer. The viewer has a set toolbar by default, but developers who wish to control the look and feel of the entire page can control the toolbar or even create their own with minimal coding.
Interactivity	This shows how events can be fired when different areas of a report are clicked, which will change the text of the textbox in the upper right-hand corner. It also gives some insight into the events supported by the Web Forms Viewer.

There is also a More Samples link that will take you back to the Crystal Decisions Web site.

The second set of sample applications, for Windows Forms, is extracted from a self-extracting file and can be found in the WinForms folder. These samples demonstrate the use of Crystal Reports .NET with Windows applications and include separate projects for Visual Basic and Visual C#. Both of these projects demonstrate a simple Preview implementation of the Windows Forms Viewer and allow you to select a report to view. Once you have selected a report file (there are a couple located in the `Reports` directory of the Samples folder), the report is bound to the viewer; the Print Engine runs the report and uses the viewer to display the results.

Sample Reports

In addition to sample applications, there are also sample report files available for you to use in your testing and development. There are two different sets of reports available in the Reports directory of the Samples folder: Feature Examples demonstrate different features and functionality within Crystal Reports .NET (Charting, Embedded Hyperlinks, Sorting, and so on), and General Business reports are typical of reports that may be created and used in business (Income Statement and World Sales Report, for example).

All of these reports have been created using the sample Access database that ships with Crystal Reports .NET and are indispensable to use when debugging. If you are having difficulty integrating your report and can't determine whether it is your code, the viewer, or the report designer itself that is not working, you can substitute your report with one of the sample reports and at least eliminate one option!

Sample Data

A sample database has been included with Crystal Reports .NET, and the sample reports listed previously are based on this database. The Xtreme Mountain Bike Company database (`xtreme.mdb`) shown in Figure 1-14 is an Access database that contains tables for Customers, Orders, Products, Suppliers, and Employees. It does not require a copy of Access to be installed or loaded on to your machine to be used.

Tutorials

Crystal Decisions has its own Web site dedicated to developing reporting applications with Visual Studio .NET, and it includes a number of tutorials or walkthroughs that can be used to get up to speed quickly with the product. The Web site is located at `www.businessobjects.com/products/dev_zone/`. You will need to register before you can download the tutorials or other materials.

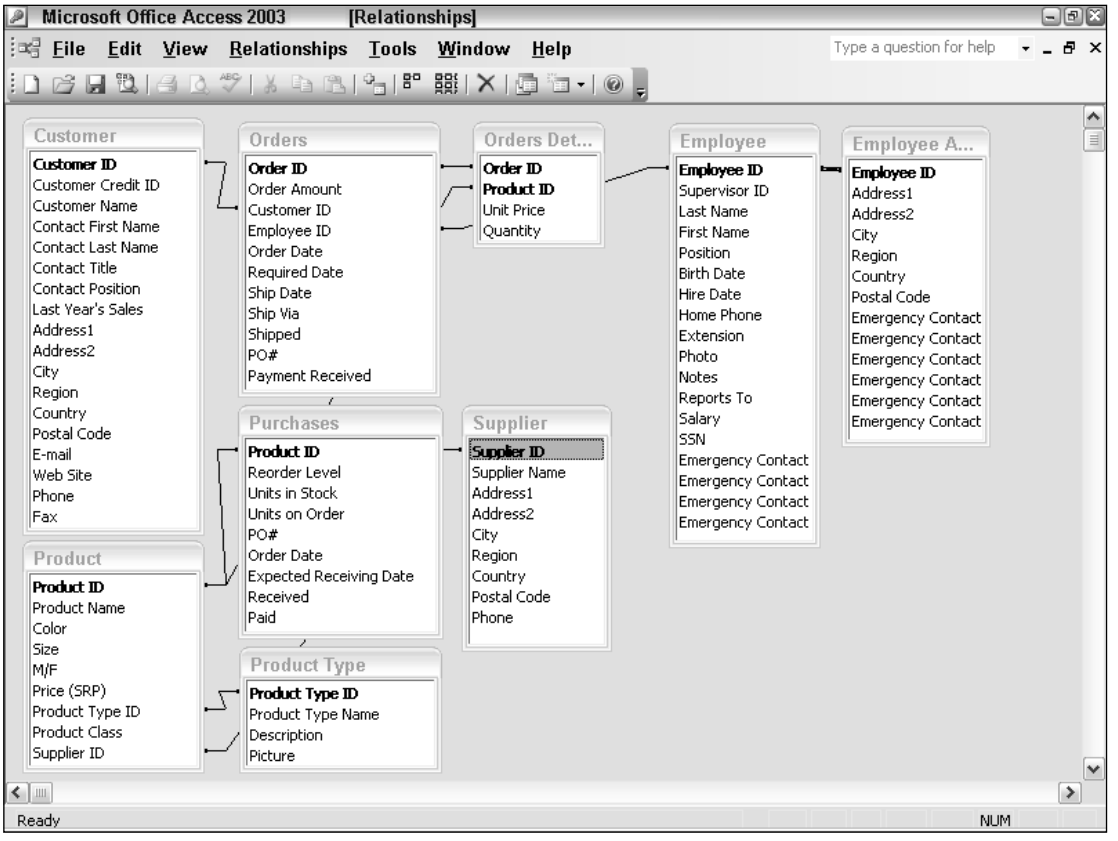

Figure 1-14

There are a number of tutorials available:

- Reporting off ADO .NET Datasets
- Viewing a Report in a Web Application
- Designing and Viewing a Report in a Windows Application
- Exposing Reports as Web Services
- Interactivity and Reports in Web Applications

Most of these tutorials can be completed using the sample database and reports that ship with the product, or you can go through the tutorial using your own reports and data source.

You will also find a number of sample reports, applications, tutorials and walkthroughs on the Crystal Developers Journal Web site, available at `www.crystaldevelopersjournal.com` (shown in Figure 1-15). In addition to their own articles and content, the site serves as a clearing house with links to other Crystal-related content on the Web.

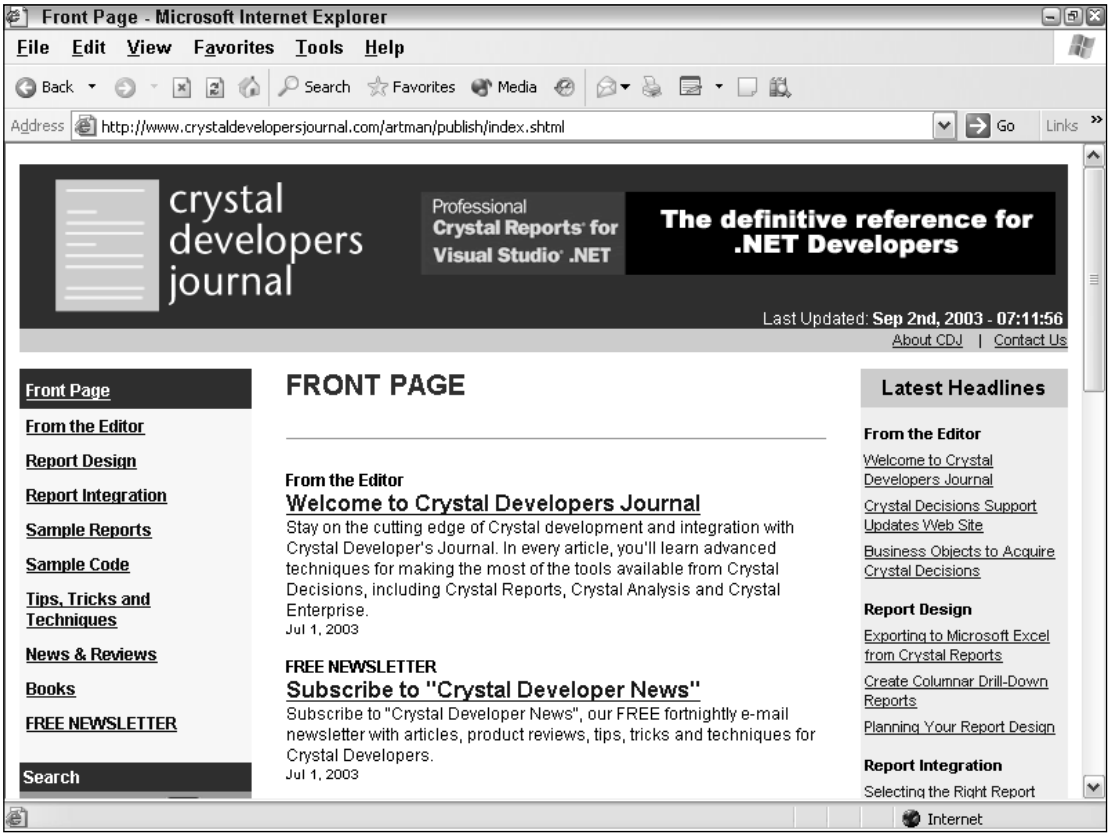

Figure 1-15

You can also find more information (and post a question if need be) on Microsoft's public newsgroups. The majority of Crystal-related questions, regardless of version or language, get posted to microsoft.public.vb.crystal, but there are always a few questions posted in the general dotnet newsgroup microsoft.public.dotnet.general.

> *You will need a newsgroup reader, such as Outlook Express, to access these newsgroups. You can also visit Microsoft's HTML version of the newsgroups on their Web site.*

Finally, Crystal Decisions maintains its own Web-based forums at `http://community.businessobjects.com`, where you can post questions and get some answers. Crystal Decisions does not monitor these forums, but generally the advice is good, and you can always find someone who is willing to help. While you are on the site, make sure you register your copy of Crystal Reports .NET for free access to technical support and updates (and the requisite marketing e-mail or two).

Summary

Crystal Reports for Visual Studio .NET 2003 builds on the reporting technology found in Visual Studio .NET 2003 and is a powerful addition to the .NET toolset, designed to take advantage of the new .NET development framework. Using the Crystal Reports Designer, you can quickly create or modify reports without having to leave the Visual Studio IDE. When it is time to integrate your report into either a Windows or Web application, Crystal Reports includes a number of viewers that you can quickly integrate into your application. With a scalable back-end processing architecture, Crystal Reports for Visual Studio .NET should be the only tool you need to integrate reporting into your enterprise applications.

In the next chapter, we'll move beyond the product overview to actually start creating reports and integrating them into our own development.