White Paper

Crystal Reports 10 for .NET Feature Overview

Authors: Terry Penner, Program Manager – Reporting SDKs
Francis Lui, Reporting .NET Development Lead

Contents

Executive Summary

1.0 Overview

With the new focus on Crystal Enterprise Report Application Server (RAS) as the basis for all report processing, Crystal Reports 10 advances the .NET reporting platform in the most significant way since its debut in Visual Studio .NET 2002. There are four major areas of improvement for the version 10 release of Crystal Reports for .NET (CR.NET).

Stability and functionality for core scenarios. For Version 10, significant focus was placed on product stability and ease-of-use for core developer scenarios. The Crystal Reports product as a whole resolved more than 1000 issues existing in previous versions of Crystal Reports. CR.NET was a major beneficiary of this focus in terms of product quality and stability.

Easy migration of all embedded reporting applications to Report Application Server (RAS). Developers can now add two lines of code to any embedded reporting application to move the application to a Report Application Server solution. There is also an upgrade path to the Crystal Enterprise Page Server, as described later in this overview. An embedded reporting application is any program where reporting is only one part of the larger application.

Full web farm and web garden support. In version 10, all dynamic report modification and creation features are supported in web farm and web garden scenarios.

Improved developer productivity and ease-of-use. In Version 10, the amount of code required for reporting functionality has been greatly reduced. Key enhancements include simplified API calls that match common scenarios, automatic project upgrading, and numerous web form enhancements.

2.0 New Feature Overview

The new features in Crystal Reports 10 for .NET can be divided into two major areas: engine (.NET Reporting Component) improvements and viewer enhancements. Please refer to the new and improved documentation for new walkthroughs, tutorials and enhanced API reference.

2.1 Report Engine Redesign and Improvements

The CR.NET engine was redesigned in version 10, with the goal of providing a common engine and architecture across all Crystal Reports SDKs. This will allow for easy application transition between deployments. This major achievement was accomplished without sacrificing backwards compatibility - all existing .NET engine applications will run with the new CR.NET engine without any code changes.

Applications written with the CR.NET engine shipped in Visual Studio .NET 2002, Crystal Reports 9, and Visual Studio.NET 2003 can easily redirect their existing ReportDocument applications to Report Application Server. The CrystalDecisions.CrystalReports.Engine.dll has been rewritten to call the RAS SDK through the .NET RAS wrapper DLLs, CrystalDecisions.ReportAppServer.*.dll. This new layer replaces the CrystalDecisions.CrystalReports.Engine.dll by default, and is intended to be fully backwards compatible.

The report creation API (RCAPI) is only enabled when using a RAS server. To use the report creation API (RCAPI), developers must call into the .NET RAS SDK layer by retrieving the ReportClientDocument:

```
clientDoc = chart2.ReportClientDocument;
```

Please read the white papers “Migrating your Crystal Reports for .NET application to Report Application Server” and “ Migrating your Crystal Reports for .NET application to the Crystal Enterprise framework” on the Developer Zone to access code projects and other information.

2.2 Web Form Viewer Enhancements

CR.NET 10 contains numerous improvements to the usability, consistency, and customizability of the web form viewers.

2.2.1 Web Form Viewer Parameter and Database Logon Prompting

Instead of throwing MissingParameterValueExceptions and LogOnExceptions, the web form viewer now prompts for missing parameters and database logons. The use of RAS also makes it possible for the web form viewer and report parts viewer to re-prompt only for missing and incorrect parameters and database logons when the user makes a mistake.

2.2.2 Web Form Group Tree Customization

In Version 10, you can now change the font style and colors of the web form group tree levels on a viewer-by-viewer basis in the VS .NET designer. You can also change this on a page-by-page basis using CSS style classes defined in a custom .css file. Finally, you can change the look and feel for an entire web site using the Crystal Reports viewers by modifying http://servername/crystalreportviewers/css/default.css.

2.2.3 Web Form Custom Toolbars

In version 10, you can now easily create a custom toolbar that offers all the functionality of the default web form toolbar. Please refer to the web form sample in the CR.NET samples folder of your Crystal Reports 10 CD to illustrate this.

2.2.4 Thin-Client ActiveX Printing Control

A new CrystalReportViewer.PrintMode property is now available. It contains the following enumerated types: PrintMode.ActiveX and PrintMode.Pdf. When the property is set to PrintMode.ActiveX, an ActiveX printing control allows end-users to print the report directly to any of their local printers. When the property is set to PrintMode.Pdf, it will instead export the report to PDF on the web server, stream the PDF to the browser, and then giving the user the option to print directly to the printer.

2.2.5 ReportSource Design-Time Property in the Property Browser

The integrated Crystal Reports designer in Visual Studio.NET now presents a filtered list of compatible report sources to make it significantly easier to databind objects to the web form page, report part, and mobile report part viewers.

2.2.6 Automatic DataBinding

With automatic DataBinding, you no longer need to add a DataBind() call to the code-behind class to databind objects to a web form's Page, Report Part, and Mobile Report Part viewers. You can disable this automatic feature by setting "AutoDataBind = false" in your application.

2.2.7 Look-and-Feel Consistency With the COM and Java Server Viewers

The look and feel has been made consistent with the DHTML viewers running on the COM and Java platforms. The toolbar now spans above the group tree, and there is a view list of open drill-down views. All report part formatting improvements in version 10 in the COM and Java server viewers have also been implemented in the .NET viewers.

2.3 ReportSource Enhancements

2.3.1 Page Server Support

In version 10, you can set any of the viewers to use an ISCRReportSource from the Page Server or RAS server.

```
EnterpriseSession entSession = myIdentity.EnterpriseSession;
Object pageServerFactoryObj = entSession.GetService("", "PSReportFactory")
PSReportFactory pageServerFactory = pageServerFactoryObj.Interface;
ISCRReportSource reportSource = (ISCRReportSource)
        pageServerReportFactory.OpenReportSource(151);
Viewer1.ReportSource = reportSource;
```

Alternatively, you can use the Crystal Enterprise for .NET EnterpriseItem object to specify whether the ReportSource originates from the Report Application Server or Crystal Enterprise Page Server.

```
MyViewer.ReportSource = myEnterpriseItem.GetReportSource(Server.PageServer);
```

or

```
MyViewer.ReportSource =
myEnterpriseItem.GetReportSource(Server.ReportApplicationServer);
```

2.4 Multilingual Client Support in Web Applications

In order to use localized strings and date, time, and number formats in VS .NET 2002 and VS .NET 2003, you needed to run the aspnet_wp.exe worker process under an account with the user default locale set to the desired language. With the new multilingual client support for webform viewer and web service applications in version 10, it is now easier to create a single reporting web application on a single server that returns localized viewer strings, error messages, and the appropriate formats for multiple languages simultaneously.

For example, a French user can have a French viewer with French tooltips, and a Chinese user will get the corresponding Chinese view, as long as the French and Chinese .NET language resource DLLs are installed on the server machine.

The "Accept-Languages" HTTP header from the web request is used to determine the locale of the strings returned to the user.

2.5 Deprecation of EnterpriseReportDocument

With the new CR.NET engine, it is now possible to access RAS SDK methods and properties via the ReportDocument object. Therefore, CR.NET has standardized on ReportDocument in the VS .NET Server Explorer and toolbox. EnterpriseReportDocument is now hidden from the IDE, although existing applications using EnterpriseReportDocument will still run.

2.6 Automatic Project Migration

In CR.NET 9 and VS .NET 2003, opening, building and running an old VS .NET 2002 project would cause an "Invalid Report Source" exception. You would then need to run ProjectMigrator.exe to upgrade the application.

In CR.NET 10, solutions opened in VS.NET are checked for old project references, web.config assembly references, and .rpt CustomTool properties. If an item in need of upgrading is found, the user is prompted to upgrade the project.

Clicking "Yes" upgrades the project to CR.NET 10 files and settings. Clicking "No" fixes the project so that you can build and run it using the components it originally used. Choosing "No" causes some design-time features to be disabled, since the older components in the project are incompatible with the newer 10 components running within the IDE.

2.7 Improved APIs for Database Logons, Parameters, and Export

In version 10, new APIs for Database logons, setting parameters and exporting have been added. These new APIs reduce the amount of code required to solve the most common reporting scenarios.

2.8 Deployment Improvements

In version 10, the number of merge modules required to deploy your application has been reduced. For more information, please refer to the separate white paper entitled "Deploying your CR.NET application".

2.9 Improved Licensing and Upgrade Options

In embedded applications using the Developer edition of CR.NET 10, the user is restricted by keycode to three concurrent processing licenses with Unlimited Retry turned off. When the processing limit is reached, a fail code is returned after two retries of two seconds each.

In embedded applications using the Advanced edition of CR.NET 10, the user is restricted by keycode to three concurrent processing licenses with Unlimited Retry turned on. When the processing limit is reached, the engine will retry an unlimited number of times until the request can be processed.

The license keycode is checked at the following locations:

```
"HKEY_LOCAL_MACHINE\SOFTWARE\Crystal Decisions\10.0\Crystal Reports\Keycodes\CR Ent"
"HKEY_LOCAL_MACHINE\SOFTWARE\Crystal Decisions\10.0\Crystal Reports\Keycodes\CR Dev"
```

If Report Application Server is used as the processing engine (ie. the ReportAppServer property of ReportDocument is set to the name of your Report Application Server instance), licensing is controlled by the RAS Server. The 3 CPL and retry mechanisms are not applied in this case.

2.10 RAS Upgrade

With only a few lines of code, you can now easily migrate all your ReportDocument-based embedded reporting applications to use RAS.

To redirect your ReportDocument applications to RAS, add one line of code:

```
chartRpt.ReportAppServer = "myRASServer1";
```

This causes the entire .rpt file to be streamed to the RAS server when you call ReportDocument.Load(), and is equivalent to opening the file using the "rassdk://" prefix.

For better performance, it is best to avoid streaming the file from your client SDK machine to the RAS server. To achieve this, change your code to use the "ras://" prefix and copy your report file to a location accessible to RAS Server machine. This is supported for all ReportDocument-based code, including strongly-typed classes, e.g.:

```
Chart  chartRpt = new Chart();
chartRpt.ReportAppServer = "myRASServer1";
chartRpt.FileName = @"ras://d:\reports\sales reports\japan\chart.rpt";
```

2.11 Enterprise Upgrade

With only a few lines of code per report, you can now easily migrate all your ReportDocument-based embedded reporting applications to use Enterprise-managed Report Application Servers.

The first step is to add each .rpt file to the Enterprise system using the Enterprise Report Publishing Wizard.

To redirect your ReportDocument applications to Enterprise, add two lines of code to point your application to a managed RAS server:

```
CrystalDecisions.Enterprise.EnterpriseSession entSession = LogonToEnterprise();
chartRpt.EnterpriseSession = entSession;
chartRpt.FileName = "cecuid:// 4158314F4D4E376B6E3439486853312E56726E7263334D";
```

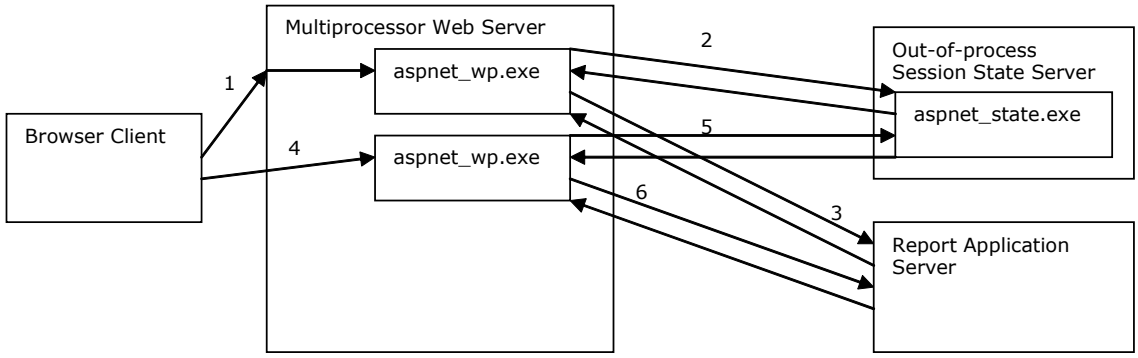
Alternatively, you can open managed RAS reports using InfoObjects using ReportDocument.Load():

```
CrystalDecisions.Enterprise.EnterpriseSession entSession = LogonToEnterprise();
CrystalDecisions.Enterprise.InfoObject infoObject = GetInfoObject( 123, entSession );
chart2.Load( infoObject, entSession );
```

2.12 Web Farm and Web Garden Support

Network Load Balancing (NLB) "web farms" and multiprocessor "web gardens" are the recommended scalable implementation for the .NET platform. It relies on the web servers being stateless, with all state stored in either an out-of-process session state server (ASP.NET's "StateServer" session state mode), or a SQL Server database ("SqlStateServer" mode).

2.12.1 Web Garden Support



In the case of a web garden, a symmetric multiprocessing (SMP) server runs one ASP.NET worker process (aspnet_wp.exe/w3wp.exe) per CPU. However, when a web client connects, ASP.NET will choose an arbitrary worker process to run the request. Since the last request

may have been processed by a different worker process, the ReportDocument may not exist in the in-memory ASP.NET cache.

To solve data synchronization problems between nodes in a web garden or web farm, ASP.NET provides an out-of-process session state server that can be deployed on a separate server (or on one of the machines in the web farm). The CR.NET web garden and web farm solution leverages this by using the ASP.NET session state server to store the serialized ReportDocument object. The act of storing the ReportDocument in out-of-process session state causes the RAS server to keep the .rpt file open. When a different aspnet_wp.exe worker process in the web garden processes a request and deserializes the ReportDocument, it reconnects to the same live report job in the same RAS server as before.

Here is the application code that accomplishes this:

```
<script language="C#" runat=server>
   private const string    REPORTKEY = "myreport";

   public object oRpt = null;

   protected void Page_Init( object sender, EventArgs e )
   {
      oRpt = Session[ REPORTKEY ];
      if ( oRpt == null )
      {
         string  sReportPath = "ras://" + Server.MapPath( "world sales report.rpt" );

         ReportDocument  report = new ReportDocument();
         report.Load( sReportPath, Server.MachineName );
         Session[ REPORTKEY ] = report;
         oRpt = report;
      }

      DataBind();
   }
</script>
```

2.12.2 Web Farm Support

The only difference between the web farm scenario and the web garden scenario is that the individual nodes of the web farm are on different machines. Therefore, you have to configure each machine in the web farm to write images to the same network share and grant

permission to the ASPNET account of each machine to write to the share. Granting additional network permissions may not be necessary for Windows Server 2003, because the w3wp.exe process already runs under an account with network permissions.

For example, you would manually set the following registry key:

```
[HKLM\SOFTWARE\Crystal Decisions\10.0\Report App\Viewer]
"CrystalImageDir"="\\fileserver\images"
```

The dynamic images would then be stored in \\fileserver\images.

2.13 DataSet Processing Improvements

2.13.1 Faster DataSet Processing

In Crystal Reports 10, architectural changes have dramatically improved DataSet performance. Testing on a 6.8MB sample DataSet showed that the new driver is approximately 2.2 times faster than previous versions, uses 74% less private memory, and uses 63% less working set memory. Results may vary for your specific implementation.

2.14 Backward Compatibility

With the exception of apps that called the early Crystal*Lib.dll Enterprise wrapper DLLs, with CR.NET 10 you are able to take a compiled app that was built against VS .NET 2002, 2003, or CR.NET 9, and run the application on a Crystal Reports 10 runtime install without recompiling.

To successfully run your application with CR.NET 10 assemblies, you will would need to modify your web.config (for web applications) or *app*.exe.config (for Windows Form applications) to redirect certain assembly versions to the version 10 ones.

For more .NET samples, tutorials and walkthroughs, please visit the Developer Zone at: www.businessobjects.com/devzone

Americas
Business Objects Americas Inc
Tel : +1 408 953 6000
+1 800 527 0580

Australia
Business Objects Australia Pty Ltd
Tel : +612 9922 3049

Belgium
Business Objects BeLux SA/NV
Tel : +32 2 713 0777

Canada
Business Objects Canada Inc
Tel : +1 416 203 6055

France
Business Objects SA
Tel : +33 1 41 25 21 21

Germany
Business Objects Deutschland GmbH
Tel : +49 2203 91 52 0

Italy
Business Objects Italia SpA
Tel : +39 06 518 691

Japan
Business Objects Nihon BV
Tel : +81 3 5720 3570

Netherlands
Business Objects Nederland BV
Tel : +31 30 225 9000

Singapore
Business Objects Asia Pacific Pte Ltd
Tel : +65 6887 4228

Spain
Business Objects Ibérica SL
Tel : +34 91 766 87 43

Sweden
Business Objects Nordic AB
Tel : +46 8 508 962 00

Switzerland
Business Objects Switzerland SA
Tel : +41 56 483 40 50

United Kingdom
Business Objects (UK) Ltd
Tel : +44 1628 764 600

www.businessobjects.com

Distributed in:
Albania
Argentina
Austria
Bahrain
Brazil
Cameroon
Chile
China
Colombia
Costa Rica
Croatia
Czech Republic
Denmark
Ecuador
Egypt
Estonia
Finland
Gabon
Greece
Hong Kong SAR
Hungary
Iceland
India
Israel
Ivory Coast
Korea
Kuwait
Latvia
Lithuania
Luxembourg
Malaysia
Mexico
Morocco
Netherlands Antilles
New Zealand
Nigeria
Norway
Oman
Pakistan
Peru
Philippines
Poland
Portugal
Puerto Rico
Qatar
Republic of Panama
Romania
Russia
Saudi Arabia
Slovak Republic
Slovenia
South Africa
Taiwan
Thailand
Tunisia
Turkey
UAE
Venezuela