

SQL Server Reporting Services and Crystal Reports: A Competitive Analysis

by Brian Bischof
www.CrystalReportsBook.com

EXECUTIVE SUMMARY

During 2004, Microsoft grabbed the attention of the Visual Studio .NET community by announcing a new reporting product: SQL Server Reporting Services (SSRS). Not only did they promise to give programmers a new reporting tool, but it was going to be free as well. Suddenly everyone was comparing Reporting Services to Crystal Reports – the report designer that has been bundled with VB since VB 3 and integrated into Visual Studio .Net (and will also be included in the next release of Visual Studio .Net 2005).

The goal of this paper is to illustrate differences between SSRS and the current version of Crystal Reports, Crystal Reports XI (version 11). Each product has its strengths and weaknesses and these are highlighted here. It's important to evaluate each product and consider which one works best for your application's reporting requirements.

After working with both products, I compared them on product offering, report file format, licensing, data connectivity, security, and design capabilities. We'll get into the full details on each subject, but I do want to highlight two things about Reporting Services:

1. Although advertised as being free, Microsoft does not recommend the "free" configuration and additional SQL Server licensing (anywhere from \$5000 to \$25000) will be required just to get started.
2. Reporting Services only handles about 75% of common reporting scenarios in my estimation. Some obvious features expected from a reporting tool are missing and still has bugs that needs to be fixed. This most likely won't be enough for developers who need to write new reports on a regular basis or have users who are critical of the report output.

In one respect this paper might appear biased because I wrote the book “Crystal Reports .NET Programming” and my knowledge of Crystal Reports is much more extensive than my knowledge of SSRS. That being said, I have spent a large amount of time researching and working with SSRS and have tried to keep this paper balanced.

I see many people stating that SSRS is a great reporting tool and they recommend it for every application. While it's great to see people taking to SSRS so eagerly, giving it a blanket recommendation ignores the fact that SSRS is a first attempt at reporting and certain areas need significant work. I also see people saying that they convinced their client/manager to commit to SSRS and now they need help because expected features are missing. It would have been much better for these people to know what to expect before committing to the technology.

Each product is at a different point in development. Crystal Reports XI is at version 11 and it is a very mature product. This version includes numerous bug fixes and performance enhancements over past versions. What is holding it back the most is its past perceptions. Some people won't even consider using it because they are still holding a grudge over problems they had with version 8.5. In fact, Crystal Reports for Visual Studio .NET 2003 uses a “lite” version of Crystal Reports 9 that is now two years old.

SSRS is the new kid on the block. Before starting development Microsoft had the advantage of evaluating all the reporting tools from existing vendors. They can keep the features they like and improve upon areas they see weaknesses in. This is a very interesting position to be in. Unfortunately, there are only so many developer hours in a day and it takes many releases to build a product that is both stable and feature complete. For example, the Windows operating system wasn't even usable till version 3.1 and even then they didn't solve the “Blue Screen of Death” headache until XP came out. You can see how each product's length of time in the market both helps and hurts it.

Before committing your resources to a certain tool you need to be aware of what each tool gives you and where it is lacking. You can't just run a few of the wizards and assume the tool is right for you. This paper exposes weaknesses in each product and also highlights areas where each is strong.

What is the purpose of Reporting Services?

Microsoft is promoting SSRS as the native reporting tool for SQL Server. With every new version of SQL Server Microsoft is trying to push the product further into the Business Intelligence space. Having a built-in reporting tool makes it much easier to promote SQL Server as an all-in-one BI platform. Microsoft also hopes that SSRS will sell more SQL Server licenses. After you build your apps using SSRS as a free add-on, you will most likely have to buy an additional SQL Server license to run Reporting Services in a production environment.

SSRS is targeted primarily at the .NET developer who is comfortable with writing Transact-SQL. This is validated by the fact that it is only available as an add-on to the Visual Studio .NET IDE and the only people who can design reports with SSRS are .NET developers. There is no end-user application that makes it easy for non-developers to create reports.

Reporting Services Overall Impressions

My impression of Reporting Services is that when Microsoft designed it, they realized that it wouldn't be possible to implement a fully functional report writer for a 1.0 release. So they designed it to have two goals: 1) Cover the largest spectrum of reporting needs in the easiest possible way and 2) Make it extensible so that either the developer or third parties can fill in the gaps. I would estimate that it handles 75% of the common reporting needs. This may not be enough for developers who need to write new reports on a regular basis or have users who are critical of the report output.

For the features that SSRS doesn't provide, you can write custom extensions to fill in the missing pieces. The biggest question this evokes is how much code will have to be written? For example, SSRS doesn't report from DataSets. Although this is a serious limitation, it isn't much of an obstacle to research how to build a custom data extension and pass the DataSet to the report. At the other end of the spectrum is the fact that SSRS doesn't export to Microsoft Word. Although I've seen people say that you can build a custom rendering extension to export to Word, considering that Microsoft's team of programmers haven't been able to do this yet then I think the assumption that a single programmer could do it is a pretty unrealistic statement.

Considering that SSRS helps you produce 75% of your reporting needs, you have to examine your reporting requirements and determine how much functionality is included and how much are you are willing to do without.

Let's look at different aspects of Crystal Reports and SSRS and see how they compare. One thing to note throughout this paper is that even though Crystal Reports and SSRS differ in respect to the report design architecture, each has its own pros and cons. You have to evaluate how each aspect impacts your business.

Product Offerings

Reporting Services consists of three main components: The Report Designer lets developers build reports within the Visual Studio .NET IDE. The Report Server processes the reports and renders them across the network/internet. The Report Manager manages reports and deployment subscriptions.

Crystal Reports has many product offerings. The primary ones that correlate to Reporting Services are: A .NET embedded reporting component for developers(CR Dev edition and in the integration in VS.Net) as well as a stand-alone report designer for business users(CR Pro and Dev has the stand-alone designer). Crystal Reports Server processes reports and renders them across the network/internet. It also provides management and subscription based deployment. Crystal Reports was purchased by Business Objects over a year ago and it was quickly integrated into the products line. In the new XI release all of the Crystal products are now compatible with the Business Objects products, which is a nice bonus for customers.

The thing to note when discussing Crystal Reports is the fact that there are so many versions available and people are still using versions that were released over five years ago. It's important to realize that the current Crystal Reports XI release (version 11) is a far superior product to previous editions. Numerous bugs have been fixed and new rendering optimizations have been added. This paper specifically addresses the report designer and .NET developer capabilities of Crystal Reports XI Developer edition. There is some feature crossover with Crystal Reports .NET 2003, but since that product is two years old and is based on a limited edition of Crystal Reports 9 it has become dated. Upgrading to the XI Developer edition costs around \$400 for existing .NET users.

Report File Formats

Both Crystal Reports and SSRS save report files in different formats and each has different goals. Crystal Reports has been in the market for over ten years and it uses a proprietary format that has been continuously modified with each new release. The goal is to maintain compatibility with different versions as well as sharing reports across the entire Crystal Reports product line. SSRS is a new tool and uses XML to store the report definition. The XML schema is called the Report Definition Language, or RDL for short.

Since the RDL file is text based, anyone could open the file with Notepad, examine it and make changes to it. The benefit of RDL is that it opens the door for third-parties to write tools that work with the RDL file. Some examples of what might come out over the next year are a report documenter and possibly competitive stand-alone tools for creating reports. The down side is that unless you have specific file permissions set, the report can be modified by anyone which can be a security and data integrity risk. Suddenly two people could have the same report with different numbers.

RDL also allows you to do dynamic report customization to a report within an application. A potential pitfall here is that this requires a thorough understanding of the RDL specification. Making changes to the report during runtimes requires parsing and modifying XML files. There is a steep learning curve and it requires advanced programming skills. In fact, Microsoft recently posted a tutorial on how to modify part of the RDL during runtime. The tutorial was almost 50 pages long and required writing hundreds of lines of custom code.

Crystal Reports uses a binary format for saving reports. Report objects are accessible through their APIs. There are different APIs based upon the product you use and each one gives a different level of customization over the report. Existing third-party tools exist for doing other tasks such as documenting reports and performing batch processes.

Licensing

Discussing licensing is probably one of the hardest topics to get a solid grasp of. Each tool comes with a detailed list of licensing requirements depending upon your situation. The best way to understand it

would be to call a sales rep and discuss your needs with them. I'll do my best here to give you a high level overview.

SSRS, on the surface, appears to have a simpler license model: it's free with the purchase of a SQL Server license. Of course, when you look at it in more detail, things are not so simple. If you want to install SSRS on a standalone server, you must buy an additional license of SQL Server for each computer. The reason this is important is because SSRS is very resource intensive. Rendering reports consumes additional CPU resources and memory and these are the same resources that SQL Server is competing for. Many companies find that it isn't practical to keep SSRS on the same server as SQL Server. On a humorous note, I read a newsgroup post where the person complained that SSRS "eats memory like candy." There is no way to allocate memory between SSRS and SQL Server. This means when processing reports, memory that SQL Server could be using will instead be consumed by SSRS.

There are also security and configuration issues that must be addressed by the database administrator before installing SSRS on the server. Installing SSRS creates additional tables in the database for managing the reports. When installing service packs it modifies table structures and it is recommended that you back up your SQL Server databases. Many DBAs are very protective of their servers and won't let anyone install any additional software on them. Opening the database to additional security risks isn't an option.

Overall, saying that SSRS is free only works when you talk to the marketing department. In fact, it seems that Microsoft has been toning down the "free" aspect of SSRS recently. In interviews with Microsoft product management they are now stating that SSRS is best run on a standalone server with the purchase of an additional SQL Server license. Of course, every company runs different software and has different loads on their server. It's not possible for a simple whitepaper to make recommendations about how your application will perform. Before using SSRS on the same computer as SQL Server in a production environment, you should perform sufficient testing and analysis to insure that SSRS doesn't degrade database performance or expose it to security risks.

Another potential licensing expense is the upgrade to the Enterprise edition of SQL Server. There are two reasons why you would consider upgrading to the Enterprise edition. As mentioned in a previous paragraph, SSRS can consume large amounts of the server's memory. The Standard edition limits you to 2GB of RAM. Depending on how many users are running reports, you might need to expand the server's memory to more than 2 GB. This requires upgrading to SQL Server Enterprise. Secondly, for printing reports that integrate Forms Authentication with existing ASP.NET websites, you must upgrade to the Enterprise edition of SQL Server. Only the Enterprise edition provides a custom API for using Forms Authentication. The SQL Server 2000 Enterprise license costs \$20,000. When you take into account that many companies will host SSRS on a standalone server, then you will need a \$5,000 license for the Standard edition (SQL Server) and a \$20,000 license for Enterprise edition (SSRS). Total license fees are \$25,000.

Crystal Reports has a similar product offering as Reporting Services. If you want to build reports that run over the internet as a web service, Crystal Reports for .NET is free and comes with Visual Studio. As a bonus, web delivery is fully compatible with ASP.NET Forms Authentication at no cost. Crystal Reports XI Developer is compatible with Forms Authentication as well.

If you want the additional functionality of centralized report management, scheduled deployments and integrated security then you can upgrade to Crystal Reports Server. The good news for Crystal Reports users is that the licensing model has been greatly simplified with the latest release of Crystal Reports XI. Business Objects has introduced Crystal Reports Server for \$7,500 (retail). They also provide unlimited technical support for a year. That's a pretty sweet deal!

The goal is that if a company has to spend a minimum of \$5,000 for a Reporting Services license that still requires many programming resources to fill in the missing functionality and it isn't compatible with Forms Authentication, then it isn't much of a jump to spend a little more for a reporting solution that has all the features already built in and it comes with unlimited tech support.

Data Connectivity

How a report connects to a data source is a critical aspect of the reporting architecture. Both Crystal Reports and SSRS have two different approaches for this and each has a different impact.

SSRS is designed so that reports can share a common data source. Switching from a development server to a production server is fairly straightforward and all reports will immediately reference the new server. This can be done by either modifying the Data Source properties within the report project or using

the Reporting Manager to set the Data Source location. By using a shared data source that is managed by a single tool, the deployment process is greatly simplified. The downside is that the individual reports don't have the flexibility to have the data source modified during runtime. For example, if a report defaults to using a certain server and database and occasionally needs to switch to a different server for reporting on historical data, this can't be done. By sharing a common data connection, you lose the granularity of being able to control individual reports.

Crystal Reports stores the data source properties within each report. Each data source can also have its own connection information. This gives you great flexibility for using different servers to generate data from, however when you want to deploy your reports from a development server to a production server you will have to remap the reports to the new server, unless you are using Crystal Reports Server. With Crystal Reports Server, you can use Business Views to share data sources. Multiple reports can be based on a single business view. The repository allows storing Command Objects to be shared among reports. This means changing to a production server is just as simple as with SSRS.

Although Microsoft created DataSet objects as the preferred method of connecting to data sources with a .NET application, SSRS doesn't support them. You have to write a custom data extension to connect the DataSet to the report. SSRS also doesn't support reporting from XML files unless you want to write more custom data extensions.

Crystal Reports supports connecting to DataSets and reporting from XML files without writing any additional code.

Another aspect to consider is how the report pulls data for the data source. Crystal Reports uses a "single point of entry" architecture. It can connect to a myriad of data types (SQL Server, Oracle, My SQL, etc.) and join them together to form a single resultset. This single resultset is used by the report to render the output. Each data source must be related to the other so that the tables can be joined on their relevant fields. If you want to report on an independent data source you have to incorporate sub-reports to do so. The benefit is that there is less overhead on the database servers because the data is queried more efficiently. The drawback is that it can require writing more complex SQL statements to join all the different tables so that they form a single resultset.

SSRS can have multiple unrelated data sources for a single report, but cannot link or join them to a single resultset. Each object on the report gets assigned its own data source and reports on it independently of the other objects. Thus, a report can have multiple points of entry using multiple resultsets. This means it is easier to conceptualize the data that a report is printing because the resultsets are more compartmentalized and this also virtually eliminates the need for sub-reports. The drawback is that there can be crossover between the different resultsets and the database server has more overhead as it processes multiple requests for similar information.

Securing ASP.NET Sites with Forms Authentication

The most common method of securing ASP.NET sites is using Forms Authentication. A website has certain pages flagged as public access and other pages require secure access. When the user first attempts to access a secure page they are prompted to enter their login credentials. After successfully logging in a cookie is dropped in the user's browser and they can access all pages from that point forward with no additional logins.

Crystal Reports is fully compatible with ASP.NET Forms Authentication. Using reports within a secure website is a seamless experience for the user.

SSRS is not compatible with Forms Authentication. SSRS security is implemented by SSRS server using Windows authentication. Each user should have their own User Id and Password for the server. The Reporting Services server is located on a separate server from the application and requires separate authentication. This requires the user to log in prior to accessing reports even if they already logged in before. This scenario is OK for intranets where each user has their own login credentials and they don't mind hitting a different server to print reports. It isn't ideal for public websites that have a single login page for the entire site. It also isn't ideal if you want reports to appear as a seamless part of the web application.

The reason that SSRS reports can't be included within the web app is because the request is being generated from the client side and not within the report server. The report server tracks user authentication separately from the main application and the Forms Authentication cookie can't be shared between the two. It is possible to get around this limitation. You have to buy a license for the Enterprise edition of SQL Server (\$20,000) and then learn the custom security API model. This is a very code intensive process and

requires extensive debugging. Doing this means that you are replacing the built-in security model with your own custom built security model. You will also lose other features like workgroup management (unless you plan on rewriting that functionality as well). In a Microsoft webinar it was even suggested that if you want to get around the Forms Authentication problem then you could install SSRS outside your firewall.

Designing Reports

The report designer provides a canvas for creating new reports and adding report objects in the appropriate places. This is where the majority of your time is spent and feeling comfortable with the designer plays a big part in your choice of reporting tools. Crystal Reports and SSRS use different approaches to designing reports.

Crystal Reports uses a banded report designer. Each “band” stretches across the width of the page and represents a different section of the report. For example, there are sections for the page header/footer, group header/footer and for page details. The banded report design has been around for many years and is used in many products.

SSRS uses an object based report designer. This is a relatively new type of report design that allows free-form placement of objects on the report. Rather than dropping the objects in certain sections, there is one large Body section and the objects are placed appropriately within the Body. The properties of each object are set to specify where in the report the data should appear (page header, page detail, etc.). This allows you to look at the Body section of the report and get an idea of what the report will look like without having to preview it. The location of each object in relation to the other objects is easy to determine. The banded report design in Crystal Reports uses a different paradigm in design mode. If you want to place a chart along side the detail records, you place the chart in another section and tell the sections to overlap. In SSRS you would put the two objects next to each other.

The type of report designer is purely a personal preference. People who have used reporting tools for years are familiar with the banded report designer and are comfortable with it. People who are new to reporting can probably learn either type of designer just as quickly as the other and become comfortable. It's my theory that the reason for the appeal of SSRS to developers is that it has something called the Table object. The Table object is a simple grid where you can drag and drop objects onto it. Since it is a grid, the objects always line up with each other. The impact of this object is that so many reports are column based and the Table object is perfect for this type of report. It is easy to add new columns to the Table. When you add a new object into the middle of the table all the columns automatically shift to make room for the new object. Crystal Reports doesn't have a Table object and adding objects to a column based report can take some extra steps to get everything to line up. Business Objects would be wise to build a similar Table object for Crystal Reports in a future release.

The SSRS report designer has limitations that you should be aware of. First is that it is not a true WYSIWYG designer. The way a report appears in preview mode can be different than how the report is printed. It's my theory that getting a true WYSIWYG experience is a technically complex thing to do. Crystal Reports is very good at it because it is on its eleventh release. SSRS isn't as accurate because it's a 1.0 product. This is a big problem for users who require precise formatting of report objects. For example, the scientific and financial industry is very exact about where data should appear and how columns should line up. An object that isn't aligned properly could go unnoticed by a layman, but will be a huge faux-pas for an accountant or a business analyst. Crystal Reports has various tools for making sure that even the pickiest user is satisfied. For example, the report preview mode lets you make changes to the report layout while previewing the report. This makes it easy to see how your changes affect the final report. There is also a built-in HTML preview mode for seeing how the HTML output looks without exiting the designer. Crystal Reports even has a zoom control in the designer for insuring pixel perfect accuracy. When designing reports for the layman this might seem like overkill. But to satisfy the financial user these features are critical.

Passing Parameters

Passing parameters lets the user filter data output for their specific needs. Parameters are also used to customize the formatting of the report. Both Crystal Reports and SSRS allow users to specify parameters prior to printing a report.

Crystal Reports provides extensive parameter support. The types of parameters supported are: single value, multi-value, and range value (e.g. Start date to End date), or a combination of all three.

SSRS parameters only support entry of a single value. It doesn't support multi-value parameters or range parameters. For example, the user can't be presented with a list of Employees and select more than one for reporting on. Doing so requires writing custom code and writing more complex SQL queries.

Dynamic Cascading Parameters in Crystal Reports allow you to connect each parameter to a live data connection. Parameters can also be linked together so that what the user selects in one parameter will filter the selection list of another parameter. Both products support Dynamic Cascading Parameters.

One thing that I see a lot of people commenting on is the lack of a date-time picker for parameters. SSRS makes you enter date parameters as regular text and doesn't have a built-in calendar component for selecting dates. Although this is more of a benefit than a requirement, it is high on the list of features that people would like to have. Crystal Reports does have a built-in calendar control.

Calculating Formula Values

Creating formulas in a report is an essential aspect of providing relevant data to the reader. Databases consist of raw data and you use formulas to apply business logic to the data so that it is more meaningful to the reader. Formulas are also useful for providing custom formatting of data during runtime (e.g. when inventory is below the minimal level, change the color to red).

Both Crystal Reports and SSRS have a formula editor that lets you write functions and perform runtime formatting. The Crystal Reports editor is more advanced than the SSRS editor. The Crystal Reports editor provides a better user interface for determining what functionality is available and helping with the syntax. SSRS only gives you an empty box and lets you type. There is no assistance provided at all.

The maturity of Crystal Reports shows in its extensive library of built-in functions. This isn't matched in SSRS. For example in the financial reporting area SSRS provides only 13 built-in functions and Crystal Reports gives you over 50. For working with dates and times SSRS gives you a few simplistic functions such as DateDiff() and WeekDayName(). Crystal Reports provides over four dozen date related functions of which the most significant let you group records by financial quarter and perform account aging.

Another problem with SSRS is that it doesn't support sharing variables between sub-reports and the main report. Nor does it encourage sharing data between objects. A frequent report requirement is to take data derived from a sub-report and display it on the main report (e.g. subtotals). Implementing this functionality in SSRS requires more complex SQL programming.

A major downfall of SSRS formulas is that they don't have an order of execution property. For example, in Crystal Reports you can set whether a formula can be calculated before another formula is calculated or you can set it to execute prior to a report printing or while records are being printed. This prevents a lot of problems with incorrect results and allows formulas to work with complex grouping use cases. With SSRS, this isn't possible and problems will result. For example, a running total can appear fine in the details section of a report but show up as zero within the group header.

SSRS aggregate functions are limited to being calculated within a single recordset or within a single area. This limitation is due to the architecture of object based reporting because report objects don't share data. Complex reports require using a combination of objects and nesting objects but SSRS doesn't provide a means of letting these objects pass values to each other.

SSRS has a limitation with performing aggregate functions. The first limitation is that aggregate functions can't be performed on values in a textbox. The second limitation is that you can't perform aggregates of aggregates. For example, the function Sum(First(Fields!Inventory.Price, "Group")) doesn't work in SSRS.

Exporting Reports

It's a fact of life that users want to do more than just read their data on a report. They want to export it to different formats for viewing within other applications. They also want to take that data and customize it using tools they are already familiar with (e.g. Excel). Both Crystal Reports and SSRS provide export functionality to different formats. The Crystal Reports export formats are much more extensive than SSRS; SSRS just gives you the essentials.

When exporting to PDF, the resulting file size of an SSRS generated report can be much larger than the size of a PDF generated by Crystal Reports. For example, a Crystal Reports PDF that is 300K in size can be over 1MB when generated by Reporting Services.

There are still many bugs in SSRS with exporting to Excel. Various problems occur for exporting multi-column reports, sub-reports, hidden fields, etc. Crystal Reports has improved dramatically over

previous versions in this area. This is clearly a very complex process and just takes time for developers to keep working out the bugs (yes, even for Microsoft).

SSRS also has a useful exporting feature. SSRS gives you granular control over the export options of each report object. Within the SSRS designer there are DataOutput properties for toggling visibility, defining each value's name, and setting whether it is an element or attribute. This gives you full XML customization and affects other export types as well. Crystal Reports doesn't have this level of granularity for its XML output.

The following table maps out the export capabilities of each reporting tool.

Format	Crystal Reports	Reporting Services
PDF	Yes	Yes
Excel Spreadsheet	Yes	Yes
MS Mail	Yes	Yes
XML	Yes	Yes
CSV	Yes	Yes
Excel Data Only	Yes	No
Word	Yes	No
Basic Text	Yes	No
Rich Text	Yes	No
Lotus Notes	Yes	No
Microsoft Smart Tags	Yes	No

Sub-Reports

Crystal Reports uses sub-reports to show data that wouldn't be possible with a regular report. As mentioned earlier, Crystal Reports has a single point of entry and can't report off unlinked data sources. Sub-reports get around this problem by letting you include a separate report, with its own data source, within a parent report. The biggest complaint about sub-reports with Crystal Reports is that they only allow one level of nesting.

SSRS supports sub-reports and it does it to 20 levels deep. I would think that if you needed this degree of nesting then it might be time to re-think your report design.

One problem with sub-reports in SSRS is that it has trouble exporting them to Excel. The Excel export only supports nested List objects.

Formatting Pages

Formatting page content lets you customize the look of the report to give it a professional image. Both Crystal Reports and SSRS have extensive formatting features to enhance the look of your reports. Although SSRS gives you a multitude of formatting options, however there are some advanced formatting features that it's missing. These might be important to certain users and it may effect your report requirements.

SSRS doesn't support conditional page breaks. Page breaks always occur in the same location and you can't reference data within the report to conditionally turn them on or off. Crystal Reports supports conditional page breaks.

Web sites use CSS files to streamline the development process. The CSS file stores the formatting rules so that the web site presents a consistent corporate image. By making a change to the CSS file it automatically propagates the changes across the entire website. It also allows you to "brand" reports for use by different clients. Crystal Reports supports CSS files for all report objects. SSRS doesn't support CSS within the body of the report. You have to manually format the report content to match the corporate image and changes have to be repeated for every single report. The ability to "brand" reports for clients isn't possible.

Both Crystal Reports and SSRS support displaying text objects. But only Crystal Reports supports displaying Rich Text Format (RTF) objects.

SSRS doesn't support restarting page numbers when a group changes. This is required for reports that are broken apart for individual distribution (customer invoices, monthly sales reports, etc.). Crystal Reports supports fully customizing the page numbering.

Labels are tough to work with. Labels require exact placement on the page and conditionally hiding different fields. SSRS makes this unnecessarily complicated.

Crystal Reports also gives you the ability to specify different formatting within objects. SSRS formatting must be done to the entire object. For example, a text box in Crystal Reports can have formatting and fonts applied to the individual characters within a string. With SSRS you would have to create multiple objects, each with their own formatting, and place them next to each other.

Crystal Reports lets you add descriptions to each report object so that when the user's mouse hovers over the object the description pop up. This isn't supported with SSRS.

Report alerts give the user a custom message when certain data is printed. For example you can alert the user when inventory levels fall below a certain level. Crystal Reports has this capability but SSRS doesn't.

Conclusion

When deciding which tool to standardize on, it's important to look at the architecture of each tool. Within this paper I listed the differences between the two products and also mentioned how each design decision comes with certain advantages and disadvantages. Remember to test each reporting tool against the functionality that is critical to your reporting requirements. Determine which product best fits your needs before making your choice.

Overall, both tools are an excellent choice for implementing reporting capabilities into your applications, but they cater to different audiences. Crystal Reports has the benefit of being the veteran in the field and supports almost every conceivable reporting requirement there is. It has versions for developers as well as business users. SSRS is the new kid on the block and it is focused only on the developer market. SSRS is a 1.0 release and still has some growing pains to go through. If your company's users have a broad range of reporting requirements and expect their reports to look a certain way, then Crystal Reports could be your best bet. If your developers want to use the latest tools and you have the resources to write the code for the missing functionality, then SSRS could be your choice.

Reporting Tools Comparative Snapshot

Feature	Crystal Reports XI Dev	SSRS Standard
Version	11	1.0
File Format	Binary (.RPT)	XML (.RDL)
Target User	Business User and Developer	Developer only
Designer Interface	Banded	Zone based
License Cost	CR.NET – Free Dev Edition - \$300 Upgrade CR Server - \$7,500	Included with SQL Server 2000 Additional Server - \$5,000 Enterprise License - \$20,000
Data Connectivity	Joins multiple sources into a single result set	Multiple sources with unlinked multiple result sets.
DataSet Reporting	Supported	Not Supported
Forms Authentication	Supported	Not supported Enterprise license support
WYSIWYG formatting	Very precise	HTML focused
Formula Library	50+ financial functions 48 date functions	13 financial functions 6 date functions
Sub-reports	One level deep	20 levels deep
CSS	Supported	Minimal support

About the author

Brian Bischof, CPA, MCSD, is the President of Dot Net Tech, Inc. providing Microsoft .NET custom software development in Southern California. He is the author of the best-selling book “Crystal Reports .NET Programming” (ISBN 0974953652), “The .NET Languages” (ISBN 1893115488) and co-author of “Professional Visual Studio .NET” (ISBN 1590593685). Please send your comments about this article to Comments@CrystalReportsBook.com.