EspressReport®

Overview, Technology, & Features

Technological Overview, and Detailed Feature Specifications For the Powerful Java Reporting Tool

Quadbase

Quadbase Systems Inc.

Contents:

I. Overview ... 4

II. Introduction ... 4

III. Designing Reports ... 5

A. Report Layouts ... 6

B. Using Report Designer ... 7

C. Data Sources ... 7

1. Complete SQL Control ... 8

2. Extensive XML Support ... 10

3. Java Object/Array data ... 11

4. SOAP Data Source ... 11

5. Updating Data Sources ... 12

D. Visual Data Representation ... 13

E. Template Re-Use ... 14

1. Extensive Parameterization ... 14

2. Powerful Scripting Features ... 14

3. Template Security ... 15

IV. Running Reports ... 16

A. Output Formats ... 16

B. Using Applets ... 16

1. Report Viewer ... 16

2. Page Viewer ... 16

C. Using the API ... 16

D. Server-Side Features ... 17

1. Parameterized Reports 17

2. Drill Down 18

E. Scheduler 18

V. Technology and Architecture........ 20

A. EspressReport Components 20

B. EspressReport Configuration 21

VI. Conclusion........ 22

I. Overview

EspressReport is a pure Java™ reporting tool from Quadbase Systems, Inc. It allows users to easily develop and deploy sophisticated reports across virtually any platform. It includes a visual design environment as well as a robust API, making it easy to incorporate reporting functionality into servlets, JSPs and applications. With many advanced features it allows users to easily create powerful data presentations, and deliver them in a variety of formats.

II. Introduction

The explosive growth of Web technologies, and the resultant glut of new information, has fundamentally changed the way in which data is presented and delivered. Companies that have rushed to embrace new technologies now find that customers, partners, and suppliers expect Web applications to be vehicles for high-value business information as well. Information is a critical component of solid relationship management, and increasingly that information is expected to be Web delivered.

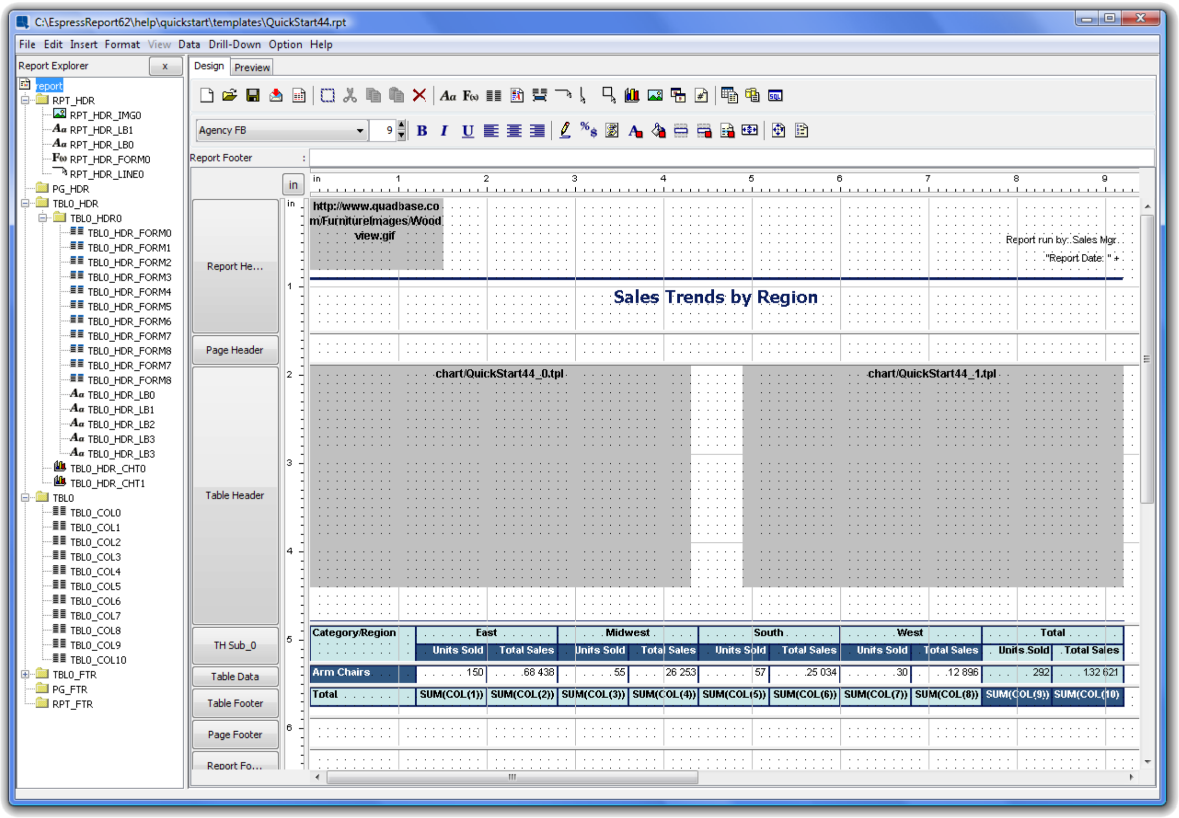
The paradigm is shifting for information analysis and delivery within organizations as well. Thick-client stand-alone reporting tools are making way for thin-client browser based solutions. Information "consumers" increasingly expect to be served custom, interactive information directly rather than designing their own reports in a desktop environment.

In both cases the demand for data presentation, visualization, and delivery is shifting to be more Web focused. Bearing this in mind, Quadbase Systems has developed EspressReport. A pure Java reporting solution designed specifically for integration within Web application environments.

This paper gives a technical background of EspressReport, highlights some of the powerful features of the product, and explains modes of usage.

III. Designing Reports

EspressReport's design tool Report Designer, makes it easy to develop and design reports. It is an easy-to-use graphical user interface that allows reports to be customized in a drag-and-drop style environment. Users, who have used traditional reporting tools, will instantly recognize the banded-style interface of Report Designer. With a similar interface to many traditional reporting tools, it offers a short learning curve for new users.



Category/Region	East		Midwest		South		West		Total	
	Units Sold	Total Sales	Units Sold	Total Sales	Units Sold	Total Sales	Units Sold	Total Sales	Units Sold	Total Sales
Arm Chairs	150	68 438	55	26 253	57	25 034	30	12 896	292	132 621
Total	SUM(COL(1))	SUM(COL(2))	SUM(COL(3))	SUM(COL(4))	SUM(COL(5))	SUM(COL(6))	SUM(COL(7))	SUM(COL(8))	SUM(COL(9))	SUM(COL(10))

Although, Report Designer may resemble older reporting tools, it is very much designed for Web-based reporting. As a pure Java component it can easily run on any platform. It can also be installed on a Web server, and run through a client browser with zero install on the client. For maximum flexibility API hooks are provided for the Designer, allowing it to be launched programmatically in a number of different configurations.

Code Sample: Launch Report Designer

```
import java.awt.*;
import javax.swing.*;
import java.io.*;
import quadbase.reportdesigner.designer.*;

public class ReportDesigner {

        public static void main(java.lang.String[] args) {

                try {

                        ReportDesigner doReport = new ReportDesigner();
                } catch (Exception ex) {

                        ex.printStackTrace();
                }
        }

        public ReportDesigner() {

                // Begin Code : Start Designer in default mode and show the Designer
                QbReportDesigner designer = new QbReportDesigner(null);
                designer.setVisible(true);
                // End Code : Start Designer in default mode and show the Designer
        }

}
```

A. Report Layouts

For creating reports, EspressReport provides five basic layouts for the report data. Each of the layouts allows users to map the tabular input data into different grouped and/or summarized formats. Mapping options are set using an intuitive wizard. Once the initial report is populated with data, users are free to customize nearly every element in the report.

Simple columnar layout: This is the most basic of all the report types. It presents data in a single table without any grouping or breaks.

Summary break layout: Like the simple columnar report, the summary break report takes columnar data and presents it in a tabular form. However, it allows users to break the data into sections, and insert summary fields.

Crosstab layout: A crosstab report is a report format that shows and summarizes columnar data in a matrix-like form. Crosstab reports often resemble spreadsheets. Both rows and columns are summarized, allowing multi-dimensional data to be displayed in a 2 dimensional format.

Master & details layout: A master & details report is a set of tabular data that is grouped according to a master field. The master field usually has a one-to-many relationship with data in the data table.

Mailing label layout: A mailing label report allows data to be pre-grouped in clusters, making it easy to format data to generate mailing/address labels.

In addition to selecting the data and layout options, the report wizard also allows users to add titles, page numbers, and other elements to the report, as well as select a pre-defined style to format the report. These features allow users to generate finished ad-hoc reports quickly without spending a lot of time formatting report elements.

Users who wish to customize reports have a myriad of options available to them. Virtually every report element can be customized allowing users to create just about any type of data presentation they can imagine.

B. Using Report Designer

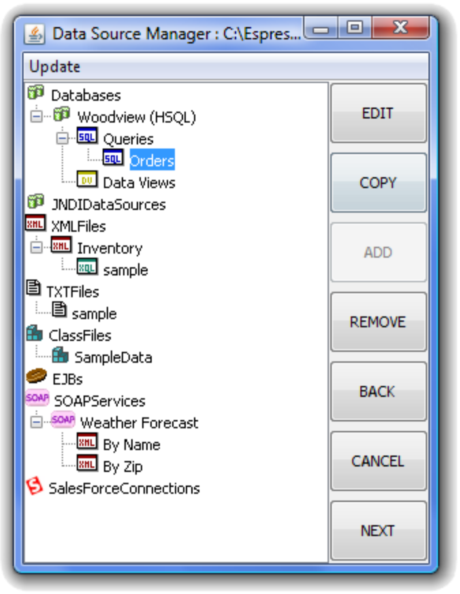
EspressReport takes an innovative approach to report design. EspressReport uses a banded-style interface, where each band in the design window represents a section of the report. Objects can be inserted and manipulated into each section in free form. EspressReport also adds a unique manipulation feature that allows users to move and format groups of objects at once. This feature allows users to manipulate objects in a columnar format as well. Powerful positioning features like snap to grid, and guidelines make it easy to align and position the various elements in a report. The integrated report explorer interface makes it easy to navigate through reports.

Users can also manipulate object formats globally. This feature allows users to assign object properties for each type of report object (label, formula, column field, etc.), at a report-wide level. This allows the specific properties of those objects to be changed, and also changes the default attributes of objects. The global formats can also be exported to an XML file, and passed between reports.

The Designer also allows users to easily implement many advanced reporting features including parameterized reports, scripting, drill down, and sub-reports. EspressReport provides simple interfaces for all these features, allowing end-users to easily integrate powerful features into reports. These unique design features, make it easy for users to design professional-looking reports without a large investment of time.

C. Data Sources

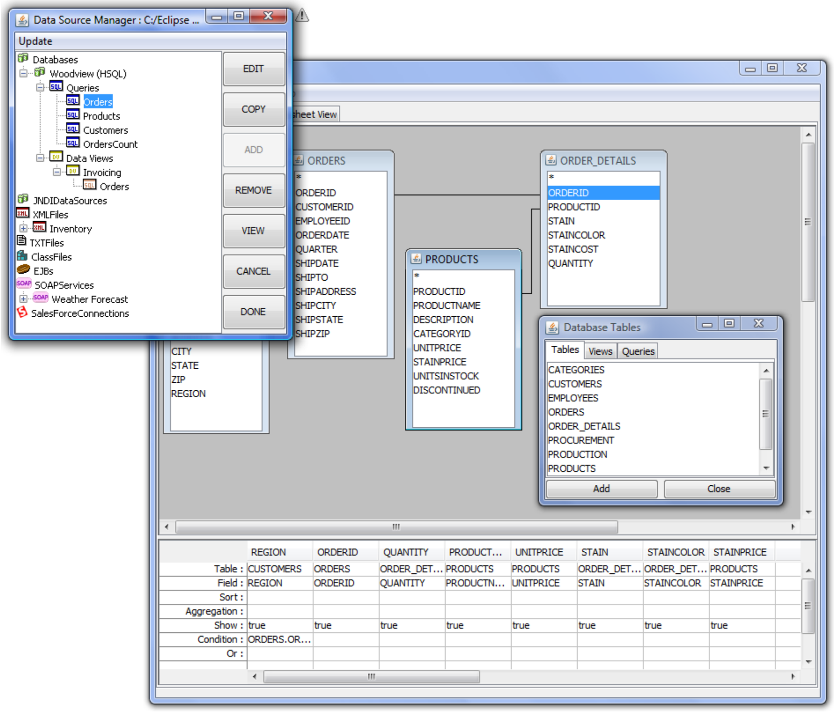
EspressReport provides a unique system for handling data with a built-in data source manager, query builder, and local data view feature. Report Designer can draw data directly from relational databases via JDBC or JNDI data sources, XML files, text files as well as SOAP data sources (WSDL and Salesforce). Application object/array data can be drawn through the API or through Java classes and EJBs. The data source manager allows users to keep track of all of the various data sources used to build reports.



The data source manager window works like a hierarchical file system allowing users to store database connection information, queries, and file locations for text, XML and Java class files. Data source registries are stored in XML format, allowing them to be modified outside of Report Designer.

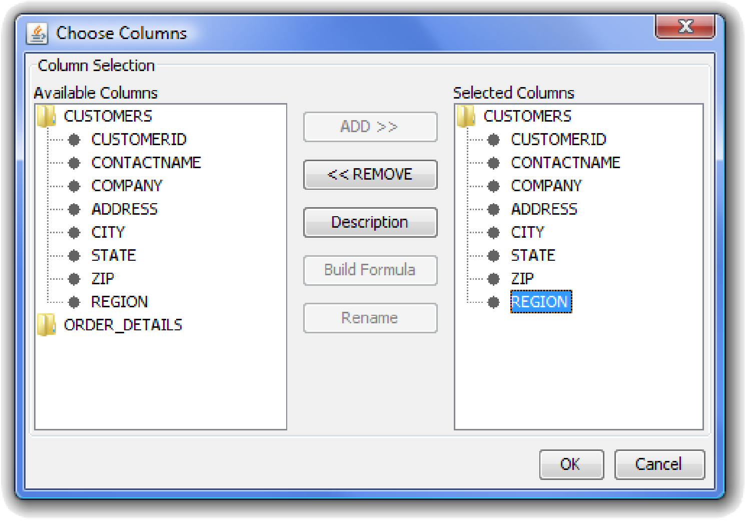
1. Complete SQL Control

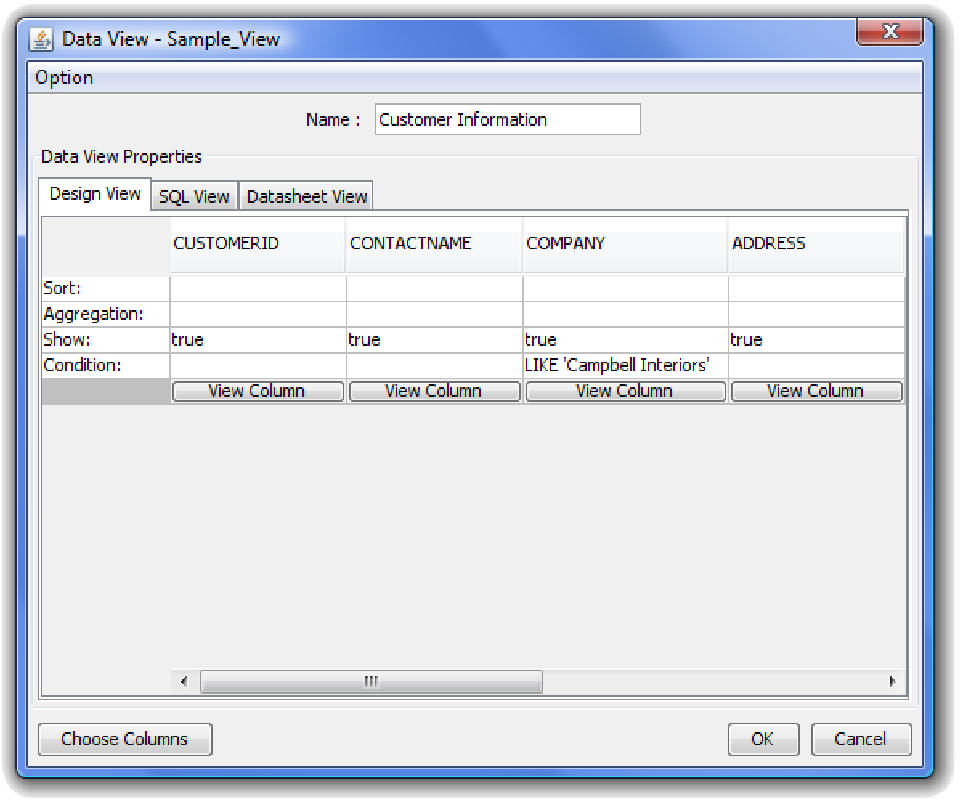
Unlike many report writers, EspressReport gives users full control over the SQL that is generated to retrieve the report data. In addition, three different interfaces are provided for users to query databases. For advanced users, EspressReport allows you to simply enter or import a SQL statement. The full SQL statement can be imported and run intact to generate report data. For intermediate users, EspressReport provides a graphical query builder. The query builder allows users to design queries in a QBE style format that allows them to select fields, conditions, joins, aggregation, and even build complex expressions using imported and custom database functions.



	REGION	ORDERID	QUANTITY	PRODUCT...	UNITPRICE	STAIN	STAINCOLOR	STAINPRICE
Table :	CUSTOMERS	ORDERS	ORDER_DET...	PRODUCTS	PRODUCTS	ORDER_DET...	ORDER_DET...	PRODUCTS
Field :	REGION	ORDERID	QUANTITY	PRODUCTN...	UNITPRICE	STAIN	STAINCOLOR	STAINPRICE
Sort :								
Aggregation :								
Show :	true	true	true	true	true	true	true	true
Condition :	ORDERS.OR...							
Or :								

For users with limited knowledge of the underlying database names or structure, EspressReport provides the data view interface. In building a data view an administrator can pre-define tables, joins, fields, and even filters creating in effect a local schema for the user to select from. End users can then select (typically aliased) fields, and filter results without any knowledge of the underlying database structures.





Data View - Sample_View

Option

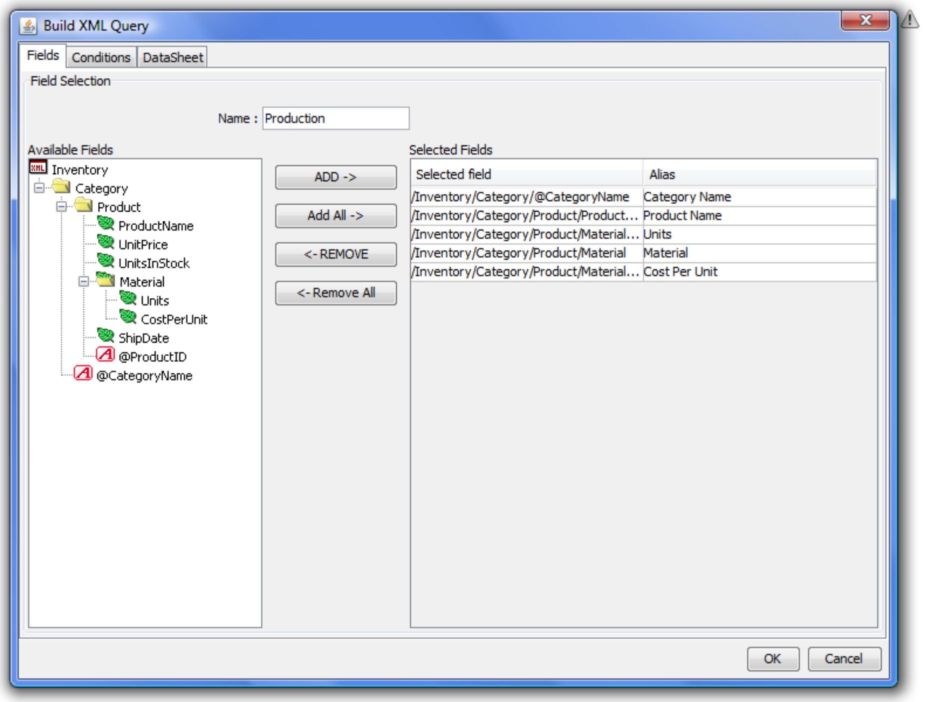
Name : Customer Information

Data View Properties

	CUSTOMERID	CONTACTNAME	COMPANY	ADDRESS
Sort:				
Aggregation:				
Show:	true	true	true	true
Condition:			LIKE 'Campbell Interiors'	
	View Column	View Column	View Column	View Column

2. Extensive XML Support

Many application models rely on XML to encode and transfer data between application components. To report on this type of data EspressReport includes extensive interfaces to set up XML data definitions and query them. Based on a DTD file, users can then generate queries from an XML file, or an XML output stream from a servlet/JSP.



Name : Production

Available Fields

```
Inventory
  Category
    Product
      ProductName
      UnitPrice
      UnitsInStock
      Material
        Units
        CostPerUnit
      ShipDate
      @ProductID
    @CategoryName
```

Selected Fields

Selected field	Alias
/Inventory/Category/@CategoryName	Category Name
/Inventory/Category/Product/Product...	Product Name
/Inventory/Category/Product/Material...	Units
/Inventory/Category/Product/Material	Material
/Inventory/Category/Product/Material...	Cost Per Unit

3. Java Object/Array data

In addition to directly retrieving data from relational databases or XML sources, EspressReport lets you pass data directly into a report from Java objects or arrays, or by connecting to EJBs. At run-time data can be passed directly into the report, and at design-time object/array data can be pulled from a Java class file. EspressReport provides several different interfaces for users to bring their application data into EspressReport.

4. SOAP Data Source

EDAB allows the user to retrieve data using SOAP (Service Oriented Architecture Protocol). Two types of SOAP data source are supported, namely WSDL compliant and Salesforce. To connect to a SOAP data source using WSDL, the user does not have to know URLs for the services, SOAP actions, operation names and parameters. All that is required is the location of WSDL file, which contains all the necessary information. A couple of dialogs in Data Source Manager facilitate setting a WSDL SOAP data source easily.

Likewise for existing Salesforce users, Salesforce SOAP data sources can be set up in a couple of simple dialogs in Data Source Manager. The user must have a valid Salesforce account and have access to a Salesforce account from a trusted network. A SOQL (Salesforce Object Query Language) needs to be included to complete the data source set up. Query parameters can be added to the query in the same manner as database queries.

Code Sample: Class to return a simple data array for a report

```
package help.examples.DataSources.classes;

import java.awt.*;
import quadbase.reportdesigner.ReportAPI.*;
import quadbase.reportdesigner.util.*;

public class SampleData implements IDataSource {

        // Setting DbData for passing data as arguments
        String dataType[] = {"string", "string", "double", "double"};
        String fieldName[] = {"OrderID",  "Product", "Price", "Quantity"};
        String records[][] = {{"1001",  "Chair", "325.00", "4"},
                              {"1001", "Table", "1211.00", "1"},
                              {"1020", "Dresser", "2214.00", "1"},
                              {"1020", "Table", "1211.00", "1"},
                              {"1020", "Cabinet", "4021.00", "2"},
                              {"1031", "Chair", "325.00", "2"},
                              {"1044", "Table", "1211.00", "1"},
                              {"1044", "Dresser", "2214.00", "1"}};
        DbData data = new DbData(dataType, fieldName, records);

        // create a empty constructor
        public SampleData() {};

        // implement getResultSet
        public IResultSet getResultSet() { return data; }

}
```

5. Updating Data Sources

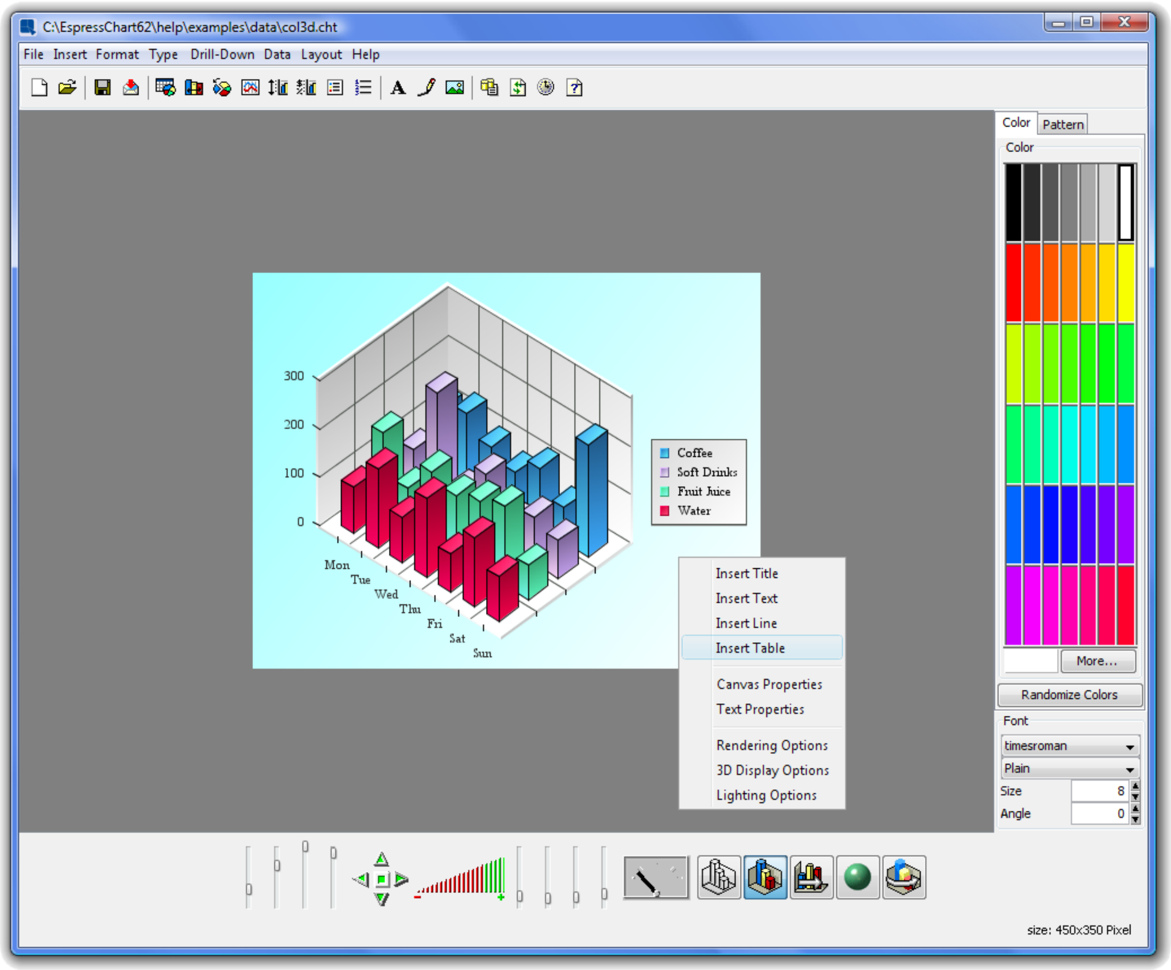
EspressReport stores the data source information both in the template and in the data registry. This unique approach allows reports to be deployed independently of the registry files. At the same time, modifications to the data source performed at the report level can be saved back to the registry, and modifications to the registry (like database connection information) can be easily propagated to groups of templates.

This feature makes it easy to migrate installations with many templates (say from a development to a production environment). Only the connection information in the registry needs to be changed, and templates can be updated automatically.

D. Visual Data Representation

EspressReport offers many different methods for visual data representation. The incorporated chart designer utility allows users to select from over 30 different two and three-dimensional chart types with which to plot report data. Nearly every aspect of the chart can be customized within the chart designer window before a chart is entered into the report. EspressReport can draw charts with true 3D rendering, allowing pan/zoom, rotation, and light source modification. Chart templates are saved separately from reports, allowing chart designs to be used many times without re-designing the charts.

Advanced charting features include time-series zooming, data drill-down, run-time text substitution, and mouse-over pop-up labels. Statistical features allow users to draw trend lines, normal curves, histograms, Pareto charts, box charts, and control lines and areas. Charts can have their own data sources, and even be deployed independently from reports.



E. Template Re-Use

One of the important needs of a reporting system is to keep the number of report templates to a minimum. EspressReport provides a number of important customization and security features that allow users to cut down on the number of templates required for an implementation or application. These features allow users and developers a short turn-around time to create and implement EspressReport as an application/system component.

1. Extensive Parameterization

EspressReport provides a number of features making it easy to parameterize reports. Users can easily add a number of dynamic filters to database and XML queries. Also, users can associate report parameters with Java class file data sources, and implement their own filtering criteria on the report data. EspressReport also allows users to map parameter value to select statement in addition to database field and function. Formula parameters allow users to do run-time text/value substitution in calculations or labels in the report itself.

At run-time parameter values can be supplied programmatically, or submitted by the user. EspressReport includes a unique feature that generates an HTML form with parameter options automatically, and returns the report in the desired format using a servlet.

A very powerful feature that EspressReport supports is “cascading parameter”. By default, the user is prompted to enter all of the report parameters at once in the prompt dialog. This configuration, however, may not be the best approach if some parameters are mapped to database columns with a significant number of distinct values. It can be difficult to select from a very large list, and depending on the parameter combination, users may be able to select sets that don't return any data.

To assist with these problems, EspressReport provides a feature that allows the user to configure the order in which the parameters should be entered. With this feature enabled, the user enters parameters in the dialog, in a pre-defined order. As such each selection will be applied as a filter to the next parameter prompt(s). Using cascading parameters can limit the number of distinct values presented to the user, and can prevent the user from selecting invalid parameter combinations.

2. Powerful Scripting Features

For run-time and conditional formatting, EspressReport includes a powerful internal scripting library that allows users to get a handle to, and modify most report elements using simple, intuitive syntax. Using scripts, users can set conditions, format cells, get a handle to specific rows of data, write loops for running calculations, and even modify whole report sections.
These powerful scripting features allow a great deal of dynamic formatting to be pushed to the template level. This allows for greater template re-use, and cuts down on the amount of code developers need to write to customize reports.

3. Template Security

Security features within report templates allow report content to be modified or controlled depending on who is viewing the template. EspressReport supports cell-level and row-level security, making it easy to re-use the same templates for different classes of users. At design-time, developers can define any number of different security levels within a template. For each security level the behavior of specific elements or columns of data can be modified. In addition, filters/parameter values can be assigned to specific security levels. At run-time only the security level needs to be supplied to the template to customize the content.

IV. Running Reports

EspressReport has many different options when it comes to running reports. Users can deploy reports in an applet downloaded onto the client, or incorporate reporting functionality into servlets, JSPs and applications using the Report API. The built-in scheduling interface allows users to run reports at a specified time or interval.

A. Output Formats

EspressReport can generate reports in several different formats. For basic reports, users can generate reports in standard HTML, compatible with any browser. For more sophisticated designs, users can generate reports in DHTML style sheets to handle the more complicated formatting requirements. To print reports, or for paginated output, EspressReport can generate high-quality PDF output. For MS Office users, EspressReport can generate reports as Excel spreadsheets, or Rich Text files (readable by MS Word). Other outputs include CSV, and XML data files, and delimited text.

B. Using Applets

EspressReport provides several applet viewers that allow users to display and interact with reports and charts.

1. Report Viewer

The Report Viewer is an applet that loads the report on the client machine via a Web browser. There are two versions of Report Viewer, an AWT and a Swing version. The AWT version allows users to run Report Viewer without downloading any plug-ins. The applet can view any EspressReport template, and allows reports to be hyperlinked to other reports directly, as well as Web pages. Reports are displayed in a paginated format. Users can generate an HTML page with the applet imbedded directly from Report Designer. The applet displays reports in a paginated format. Users navigate between pages and linked reports, using a customizable pop-up menu.

2. Page Viewer

For large reports, loading the entire report on the client isn't generally feasible. For these scenarios, EspressReport provides the Page Viewer component. Page Viewer works much in the same manner as Report Viewer, but incorporates a page serving technology that sends report pages to the client only as they are requested. With Page Viewer, users can load/preview large reports on the client, without running out of memory.

C. Using the API

The included API (or application programming interface) extends the most flexibility to EspressReport users. Using Report API, developers can incorporate reporting functionality into servlets, JSPs, and applications. Report API can generate reports on the fly based on user parameters, and stream them directly to the client browser, or cache the generated reports on the server. Nearly every report object can be controlled using the API, and reports can be designed purely programmatically. The API can also run templates built in Report Designer, while modifying report properties at run-time. Developers can also embed the Report Designer directly in their application environment, giving end users the ultimate flexibility in ad-hoc reporting.

Code Sample: Open and stream a report to the client as html using a servlet

```
import java.awt.*;
import java.applet.*;
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import quadbase.reportdesigner.ReportAPI.*;
import quadbase.reportdesigner.ReportElements.*;
import quadbase.reportdesigner.ReportViewer.*;
import quadbase.reportdesigner.util.*;
import quadbase.reportdesigner.lang.*;

public class SampleServlet extends HttpServlet implements SingleThreadModel {

        public void doGet(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException {

                // Set the "content type" header of the response
                res.setContentType("text/html");

                // Get the response's OutputStream to return content to the client.
                OutputStream toClient = res.getOutputStream();

                try {
                        // Open up specified Report
                        QbReport report = new QbReport((Applet)null,
"templates/SampleReport.rpt");

                        // Export (Stream) the report to DHTML
                        report.export(QbReport.DHTML, toClient)
                } catch(Exception e) {
                        e.printStackTrace();
                }

                // Flush  and close the outputStream
                toClient.flush();
                toClient.close();
        }

        public String getServletInfo()        {
                return "SampleServlet servlet for EspressReport";
        }
}
```

D. Server-Side Features

Included with EspressReport are several key server-side features that allow users to cut development time while implementing some of the most common reporting features. These servlets can run as is, or users can modify the source code to suit their needs.

1. Parameterized Reports

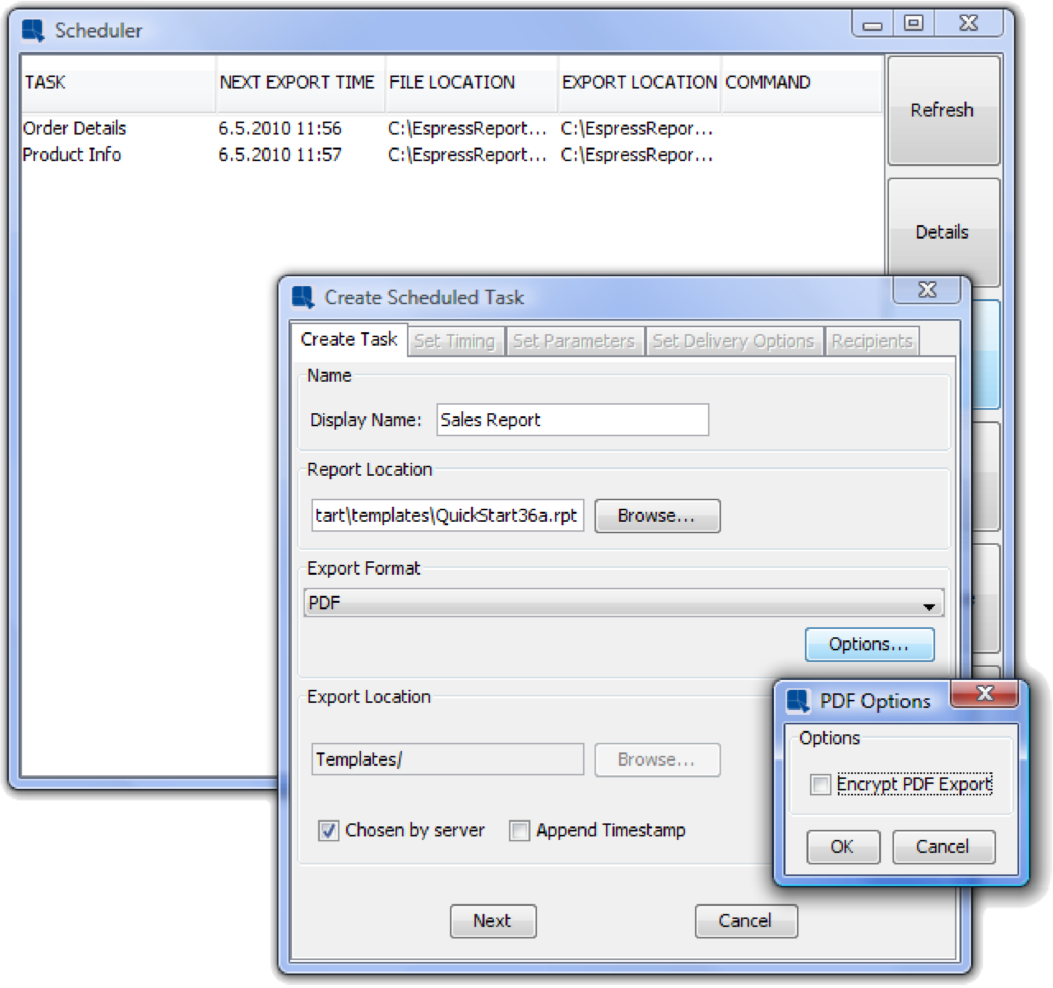
For parameterized reports, EspressReport includes a unique feature that allows an HTML form containing the report parameters to be generated on the fly. Using this feature, developers can simply open the report, and call the HTML form, rather then getting the parameter information from the report template, and then constructing their own form to collect parameter values from the user. HTML forms generated using this feature call an included servlet, which streams the report to the client in the selected format.

2. Drill Down

One of the most common reporting needs is the ability to drill into top-level reports and charts to pull underlying information. EspressReport supports this functionality, and provides servlet tools to make it easy to deploy reports and charts in this configuration. Using the drill down features in EspressReport, users only have to design one report or chart template for each level of drill down. Linking between layers is handled automatically, and an included servlet handles deployment. Using these features makes creating and running drill down reports no more complicated than any other report type.

E. Scheduler

Included with EspressReport is a scheduling application that allows users to schedule reports or processes. Using the provided interface, users can schedule either a report template, or a command line process to run at a set time, or at a specified time interval. Reports can be exported to the file system, or sent via email. Schedules can be set to run with different parameter values, and security levels. API hooks allow user's to access and modify schedule jobs programmatically.



TASK	NEXT EXPORT TIME	FILE LOCATION	EXPORT LOCATION	COMMAND
Order Details	6.5.2010 11:56	C:\EspressReport...	C:\EspressRepor...	
Product Info	6.5.2010 11:57	C:\EspressReport...	C:\EspressRepor...	

Name
Display Name: Sales Report

Report Location
tart\templates\QuickStart36a.rpt

Export Format
PDF

Export Location
Templates/
[✓] Chosen by server
[ ] Append Timestamp

Options
[ ] Encrypt PDF Export

Code Sample: Create a schedule job

```
import java.io.*;
import java.util.*;
import quadbase.scheduler.*;
import quadbase.reportdesigner.util.IExportConstants;

public class MySchedule {

        public static void main(java.lang.String[] args) {

                try {
                        MySchedule ScheduleReport = new MySchedule();
                        ScheduleReport.start();
                } catch (Exception ex) {
                        ex.printStackTrace();
                }
        }

        public void start() throws Exception {

                // Create a schedule object to run a report template
                ScheduleObject sObj = new ScheduleObject("SCH_OBJ1", ScheduleObject.REPORTOBJ);

                // Specify which template to use
                sObj.setFileLocation("Templates/SAMPLE.rpt");

                // Specify the export format
                sObj.setReportType(IExportConstants.DHTML);

                // Specify the periodicity of the schedule (here run once) and start time
                sObj.setTaskOption(ScheduleObject.ONE_TIME);
                Calendar calendar = Calendar.getInstance();
                calendar.add(Calendar.MINUTE, 5);
                sObj.setStartDate(calendar.getTimeInMillis());

                // Don't send email notification
                sObj.setSendEmail(false);

                // Add new schedule to jobs list
                ScheduleModifier.addScheduleTask(sObj);
        }
}
```

In addition, the scheduler brings many other useful features e.g. support for scheduling multiple reports/charts in a package, daily frequency scheduling, and distributing to different users based on parameter values.

V. Technology and Architecture

EspressReport is a pure Java product, giving it the flexibility to run on nearly any platform. It runs on most JVM's equivalent to JDK 1.4 and higher. EspressReport is compatible with most application servers, and servlet runners such as WebLogic™, WebSphere™, and Apache Tomcat™. It can directly connect to any database using JDBC, or using a JNDI lookup name for deployed J2EE data sources.

A. EspressReport Components

EspressReport has seven associated components:

Report Designer: Report Designer is a GUI tool that allows users to build reports in a point-and-click environment. Included with the Report Designer are interfaces to access data, query databases, and design charts. The Report Designer can run as a client application or in a client-server configuration with the Designer loaded as an applet through a Web browser.

Chart Designer: Chart Designer is a GUI tool that allows users to build and design charts in a point-and-click environment. Launched from within Report Designer it allows users to create charts to be embedded within report, or stand-alone charts to be run using the Report API.

Report API: Report API is an easy-to-use application programming interface that allows users to imbed reporting functionality into their applications, servlets, or JSPs, either on the server-side or client-side. Because it is pure Java it can run on most platforms with few or no changes. Any and every part of the report is customizable using the API giving users full control over report formatting at run-time. Reports can be created and deployed with just a few lines of code.

Report Viewer: Report Viewer is an integrated applet that allows users to view and interact with reports. The applet shows reports in a paginated format, giving users the opportunity to navigate around reports using a pop-up menu. Also report templates can be directly hyperlinked together using the applet. EspressReport can generate HTML pages with the applet imbedded without requiring any coding.

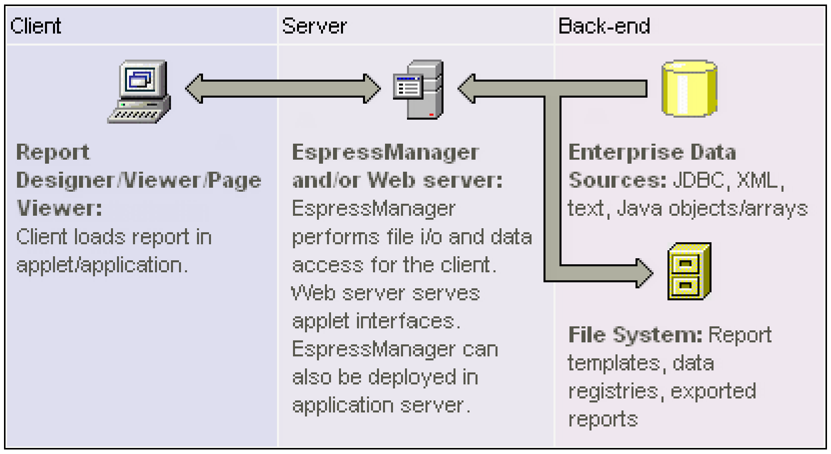
Page Viewer: Page Viewer is an integrated applet like Report Viewer that uses page serving technology. With Page Viewer, pages in the report are only sent to the client when requested. This allows users to view/preview large reports.

EspressManager: EspressManager serves as the “back end” to the Report Designer and scheduler interfaces. It supports Report Designer running as an applet EspressManager handles the data access and file I/O activities on the server-side. In addition, EspressManager provides database connection and data buffering. EspressManager also runs the scheduling process on the server-side, executing the reports according to the user-defined jobs.

Scheduler: The scheduler is a small interface that runs in conjunction with EspressManager. It allows users to schedule reports, and other events, as well as manage scheduled tasks.

B. EspressReport Configuration

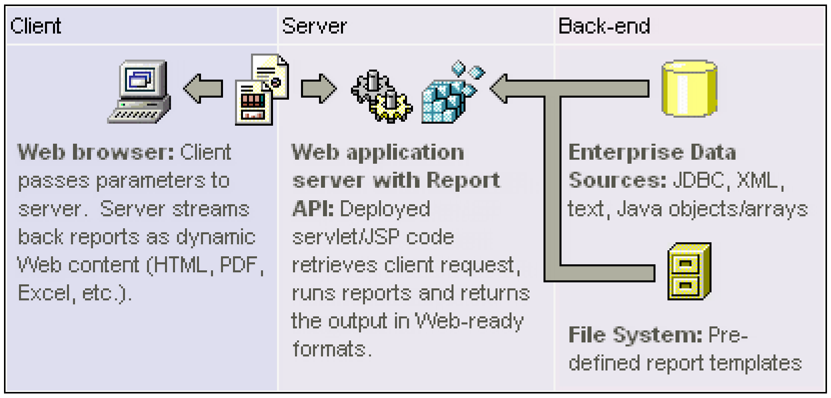
EspressReport has a number of different configurations in which it can run both at design-time, and at run-time. At design-time the Report Designer GUI interface, which includes data access tools and charting tools can be loaded as an application on a client machine, or as an applet in a client server architecture. The following diagram illustrates EspressReport running with the Report Designer at design-time.


Client
Server
Back-end
Report Designer/Viewer/Page Viewer: Client loads report in applet/application.
EspressManager and/or Web server: EspressManager performs file i/o and data access for the client. Web server serves applet interfaces. EspressManager can also be deployed in application server.
Enterprise Data Sources: JDBC, XML, text, Java objects/arrays
File System: Report templates, data registries, exported reports

When Report Designer is running, the EspressManager component runs on the server-side. The EspressManager performs the data access and the file I/O which is prevented by the client applet due to security restrictions. The EspressManager also provides connection and data buffering for database connections, as well as concurrency control for a multi-user development environment. The EspressManager must be run in conjunction with the Report Designer.

EspressManager can run as a server-side application, a background process, or as a Windows service. In addition, EspressManager can be deployed as a servlet collection within an application server, or servlet container. This allows users to deploy the Report Designer, and/or Scheduler interfaces without having any separate server-side processes.

At run-time, the EspressManager does not need to be running. EspressReport is designed to run embedded within other application environments, and can have a minimal deployment of the API classes, and report template files. The following diagram illustrates EspressReport running in a servlet/JSP environment.


Client
Server
Back-end
Web browser: Client passes parameters to server. Server streams back reports as dynamic Web content (HTML, PDF, Excel, etc.).
Web application server with Report API: Deployed servlet/JSP code retrieves client request, runs reports and returns the output in Web-ready formats.
Enterprise Data Sources: JDBC, XML, text, Java objects/arrays
File System: Pre-defined report templates

When running in an application server, the Report API can be used to generate reports within servlets and JSPs and stream the generated report to the client browser. Clients can also view reports, by loading the Report Viewer applet or the Page Viewer applet.

Report generation can be handled on demand, triggered by application processes, or scheduled. A scheduler interface is also provided with EspressReport. In order to run the scheduler the EspressManager must also be running.

Memory optimization/paging features allow users to configure the amount of memory that EspressReport uses. These features allow users to run even extremely large reports quickly without over-taxing system resources.

VI. Conclusion

EspressReport is an innovative product, built to address the changing needs of the reporting market. It is simple enough for power users to build and develop reports, yet sophisticated and powerful enough to satisfy the most demanding developers.

With its pure Java design, in easily runs on virtually any platform, and offers the scalability to handle most enterprise-level reporting needs. With it's powerful feature set, flexible architecture, and intuitive, easy-to-use design environment, EspressReport sets a new standard for next-generation Web reporting.