Field Session: Final Report

Graphical Data Enhancements

Team: Aaron Schmidt

Michael Schiermeyer

Client: Data Verity

Table of Contents

Abstract

Rich Internet Applications (RIAs) and Software as a Service (SaaS) are rapidly becoming a dominant force in software development, and knowledge of how to develop RIAs is becoming increasingly important. Data Verity is a retail consulting firm that provides its clients with different types of services via the internet. Their Event System Processing (ESP) provides clients with versatile tools to enhance their business practices. These include data mining, reporting, file management, and case management. Data Verity was looking to improve their ability to graphically represent data on their system using development platforms such as Flash, Java Applets.

Project Introduction

Data Verity's current graphing capabilities are limited as they can only represent data in a small, non-scaleable, non-customizable image. Having a tool that allows Data Verity's clients to customize a graph and use it in a report represents significant progress towards better graphical capabilities. Aesthetic appeal is a very valuable part of Data Verity's business model. These changes are a significant step in maintaining the new look and appeal to the web application. Not only will this help Data Verity keep its current clients but will assist in acquiring new ones.

Requirements

Our goal is to create versatile and interactive graphical representations of data that can be easily changed to fit the clients' needs. By using Javascript (AJAX and ExtJS), Java (JFreeChart), PHP, XML, and Flash (amCharts) we will provide support for Column, Bar, Line, Area, Pie, Donut, and Stock charts. Data Verity's clients will customize these graphs in a web interface and then have the ability to export them to one of five image formats (JPEG, GIF, PNG, PDF, and SVG).

Data Verity values the ability to edit charts in every possible way. This is why they requested a settings panel be displayed next to the amCharts flash graph. This side panel was created in ExtJS and contains tools to edit numbers, text, colors, booleans, and selections. The panel can easily display over 100 different settings without appearing crowded and allows the client to edit every setting amCharts allows while permitting Data Verity to block some settings through configuration.

After editing one or many settings on the ExtJs settings panel, a user can click a button to redraw the flash graph. The new graph will reflect the changes made to the settings panel. The graph serves as a preview to the static graph while also allowing for more in depth analysis. The flash chart allows users to mouse over data to pop up a box with details associated with that data, something that is not done with the static images. After deciding on the proper settings, the client user can then save those settings to be used in the static image creation.

The reason why Data Verity wanted so many different formats is because each fills its own niche. Joint Photographic Experts Group (JPEG) is a commonly used method of compression for photographic images. The JPEG compression algorithm is at

its best on photographs and paintings of realistic scenes with smooth variations of tone and color. For web usage, where the bandwidth used by an image is important, JPEG is very popular. JPEG is also the most common format saved by digital cameras. [1]

On the other hand, JPEG is *not* as well suited for line drawings and other textual or iconic graphics, where the sharp contrasts between adjacent pixels cause noticeable artifacts. This is where a lossless data compression image, like Graphics Interchange Format (GIF), comes in. When a JPEG opens it loses some of its data. When a lossless data image opens it shows an exact representation of the data it contains. The disadvantage is that until recently, the GIF format was under a patent. [2]

Portable Network Graphics (PNG) is a bitmapped image format that employs lossless data compression. PNG was created to improve upon and replace GIF as an image-file format not requiring a patent license. PNG is superior in image quality over JPEG and GIF but it doesn't intrinsically support animated images. [3]

Portable Document Format (PDF) is a file format created by Adobe Systems in 1993 for document exchange. PDF is used for representing two-dimensional documents in a manner independent of the application software, hardware, and operating system. The biggest advantage of a PDF is that it can represent its images in vector format. Unlike raster image formats like JPEG, GIF, and PNG, vector images contain not the image itself (pixels) but rather a mathematical description of the image. In this way images can be enlarged or shrunk down without any image degradation. [4]

Scalable Vector Graphics (SVG) is a family of specifications of XML-based file format for describing two-dimensional vector graphics, both static and dynamic (interactive or animated). Since they are XML files, SVG images can be edited with any

text editor. However, viewing them can be difficult as each browser has a different way rendering them. [5]

Each image type has better quality than its predecessors. However, as quality improves, file size increases. Also, the more specialized image formats (ex. SVG) need special readers to render correctly. If a client needs a graph that is tiny and is only used once, JPEG is the way to go. However, if the client needs a high quality image that can be rescaled to any size without pixilation while being editable, they need SVG.

Design

Our project is split into two sections: (1) Design a tool to help Data Verity's clients customize a graph and (2) Create a static image from that graph in the appropriate format (JPEG, PNG, etc.) that which can then be used in a report. We have separated each section into several steps.

(Section 1) To create the customization tool we:

- Build an ExtJS settings webpage from a JavaScript Object Notation (JSON) object input. (Fig. 1)
- Write the input settings to an XML file and store it on Data Verity's server.
- Allow the XML settings to communicate with amChart inputs.
- Build a customized chart and display it on Data Verity's webpage. (Fig. 2)

(Section 2) To create the static images we:

- Retrieve the input settings from saved XML file created earlier. (Fig. 1)

- Create a static image from data using Java library JFreeChart. (Fig. 3)
- Customize image format for client needs.
- Compare the customized graph with the static image in order to ensure similarity.

Implementation

There were many technical challenges; everything from data legality to extending the functionality of libraries. The most notable were:

- making the XML file writer more customizable
- importing the chart settings into JFreeChart
- saving files in SVG format

All XML file writers that we found were limited in the way you could define attributes of certain elements (ex. <chart id=”pie”> wasn’t possible). In order to get around this we decided to write our own XML writer. We implemented it by converting everything into character strings instead of just writing data straight to the XML file.

Importing the chart settings from amCharts into JFreeChart was complicated because they run on different standards. For example, amCharts doesn’t specify a chart type in its XML file. We had to figure out how to extract that information without being overly restrictive. The solution ended up being a combination of XPath, a Java XML parsing tool, and regular expressions.

Saving a chart as an SVG image is tricky. It involves not only customizing things such as color and size, but also translating the image to vector format. Unlike pixel based

images, vector graphs don't distort when zooming in. We ended up using a collection of libraries called Batik. Batik is a Java-based toolkit for applications or applets that want to use make use of scalable images. Batik is maintained by the Apache Software Foundation, a community of open-source software projects. [6]

Other implementation decisions, such as the use of JFreeCharts were, heavily debated. We make a static image through JFreeCharts using an XML file created by amCharts (Fig. 1). However, it is possible to export an amCharts straight to an image without using JFreeCharts. This simplifies code, but it requires a considerable amount of bandwidth and puts a substantial strain on the server. Additionally, the image formats you can save in are limited. JFreeCharts takes most of the strain off the server and puts it on the client side without taking any bandwidth. Considering that JFreeCharts has potential for reducing server lag, using client processing power, and generating a larger variety of image types, it can clearly be deduced that JFreeCharts outperforms amCharts in every way needed.

Conclusions

Data Verity plans to extend the functionality of our projects in a variety of ways in the future. The design and flow of the amCharts settings editor worked out to be very generic. It can be used to edit any XML file, not just a chart settings file. Data Verity expects to have other uses for editing XML files in this user-friendly fashion.

In addition, the design is generic enough to easily add new chart types in the future. Data Verity may implement scatter plots and/or stock charts in the future, although they have little use for them at this time. Data Verity would very much like to

support funnel charts, however, amCharts and JFreeCharts do not support this type. In the event an update or addition becomes available it would be very easy for them to support funnel charts.

Throughout this project we've learned that many of our assumptions about the programming world are wrong. For example, our project was very open ended. We thought a lack of barriers would maximize the quality of the product we produce. This theory worked extremely well through most of the project. However, when week five rolled around, our client's expectations were different than our own. In the future we will define our workload clearly from the start. In the end the software was working properly on Data Verity's server and the client considered the project a success.

In addition to the insight we've gained into the programming world, we've gained various practical skills. We learned how ExtJS through amCharts can create aesthetically pleasing graphs. We've seen how JFreeCharts can combine a data file and XML to export an image to more formats than we will ever need. But, most importantly, we've learned how to work as a team.

Figure 1

Interaction between amCharts and JFreeChart

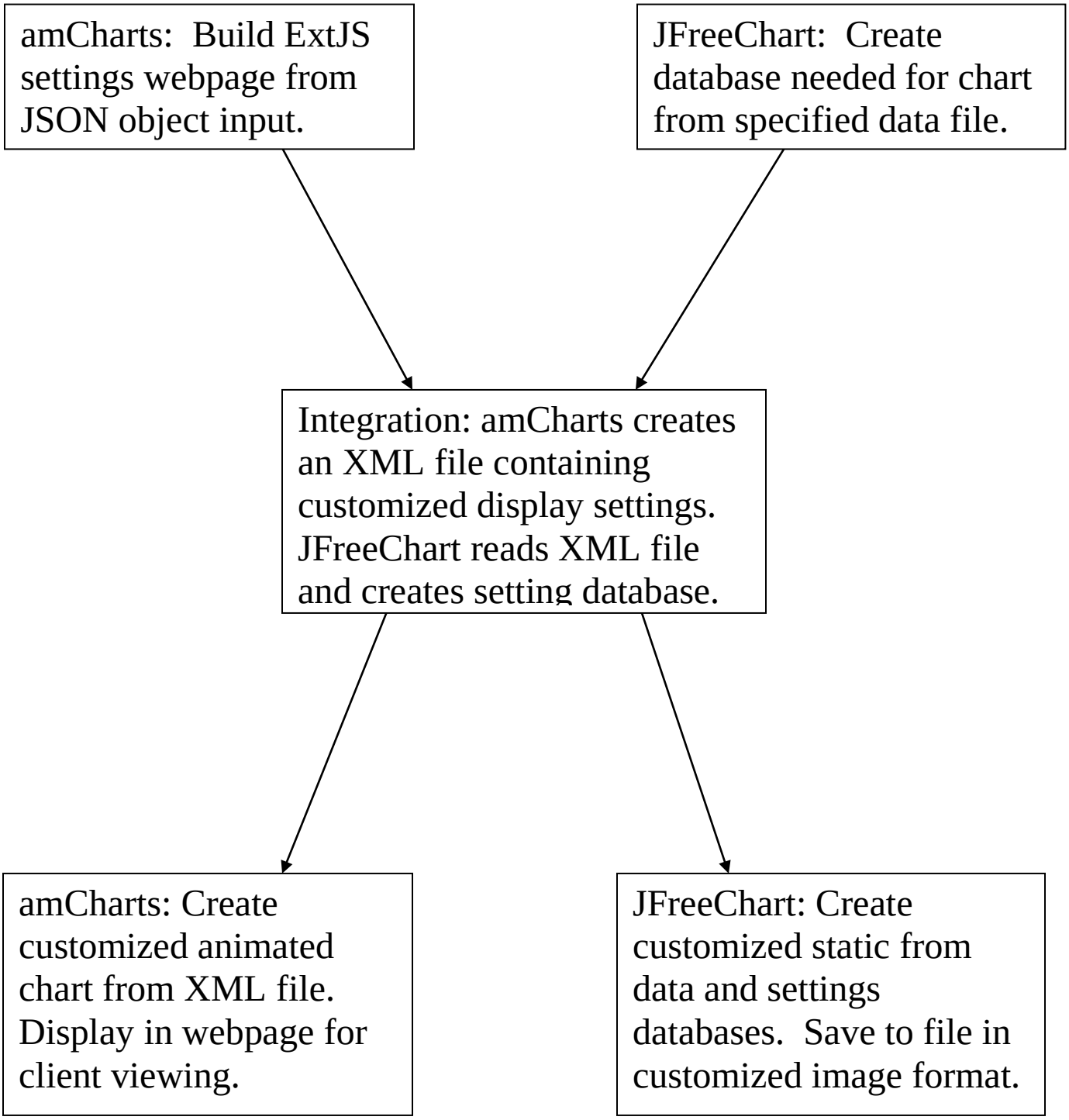


Figure 2
amCharts: Overlaid Example

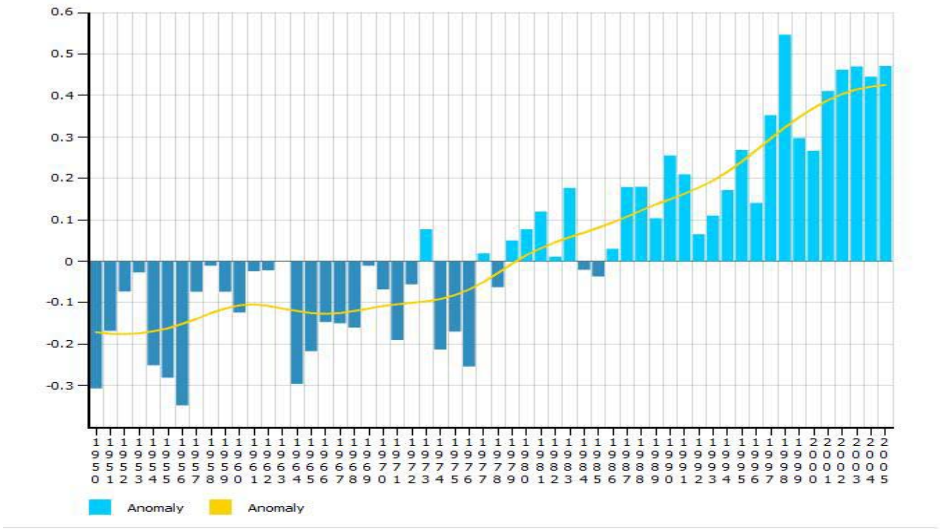
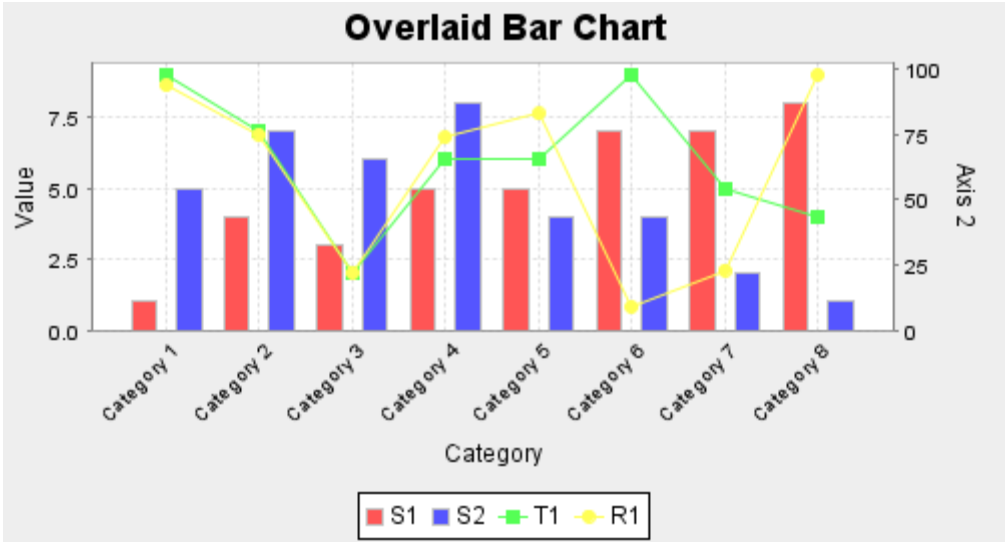


Figure 3
JFreeChart: Overlaid Example



Works Cited

[1] ://en.wikipedia.org/wiki/JPEG

[2] ://en.wikipedia.org/wiki/Graphics_Interchange_Format

[3] ://en.wikipedia.org/wiki/Portable_Network_Graphics

[4] ://en.wikipedia.org/wiki/PDF

[5] http://en.wikipedia.org/wiki/Scalable_Vector_Graphics

[6] http://xmlgraphics.apache.org/batik/