

Message

From: Naveen Lakkur [naveen.lakkur@compassites.net]
Sent: 4/6/2007 6:08:56 PM
To: 'Seymour Duncker' [sduncker@gmail.com]
Subject: iCharts Technical Overview
Attachments: iCharts Technical Overview.pdf

Dear Seymour,

Please find attached with this mail the Technical Overview document. This should help us to have the discussion on the iCharts possible architecture.

We confirm to participate in the conference call on Thursday to discuss this. My skype id is naveen_lakkur and if it is fine for you, we could have the conference call using skype.

Thanks and have a great day.

Regards,
Naveen Lakkur
President & CEO
Compassites Software Solutions Pvt., Ltd.,
T: +91-80-41538721
F: +91-80-41723623
C: +91-9880013830
E: naveen.lakkur@compassites.net
U: www.compassites.net
B: www.start2lead.com

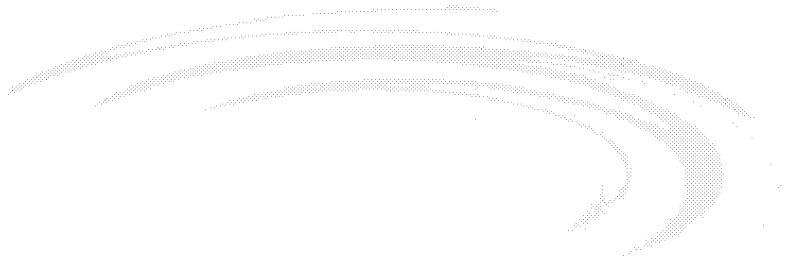
The information contained in this communication is intended solely for the use of the individual or entity to whom it is addressed and others authorized to receive it. It may contain confidential or legally privileged information. If you are not the intended recipient you are hereby notified that any disclosure, copying, distribution or taking any action in reliance on the contents of the information is strictly prohibited and may be unlawful. If you have received this communication in error, please notify me immediately by responding to this mail and then delete it from your system. Compassites Software Solutions Private Limited is neither liable for the proper and complete transmission of the information contained in this communication nor for any delay in its receipt.

--

No virus found in this outgoing message.

Checked by AVG Free Edition.

Version: 7.1.413 / Virus Database: 268.18.24/742 - Release Date: 4/1/2007



iCharts – Technical Overview

Prepared By
Ramesh Rajamani

Technical Architect

Compassites Software Solutions P Ltd
#100/3, Bull Temple Road
Bangalore-560 019
India

P: +91-80-41538721
F: +91-80-41723623
E:Ramesh.rajamani@compassites.net

Table of Contents

Introduction 3

High Level Architecture.....3

 Technology Choices.....5

 Service and Persistence Layer (Java/J2EE frameworks) 5

 Presentation Layer (Adobe Flex Framework) 6

Why Adobe Flex? 7

 Advantages of Flex + Flash.....7

 Disadvantages of Flex.....8

Summary 8

Appendix – References.....9

Introduction

The purpose of this document is to present a high level architecture for iCharts application. The architecture is based on the understanding from the high level business overview presentation given by Seymore. The document explains briefly the components of the architecture. More importance is given to the selection of the presentation layer component which is Adobe Flex and why it is chosen.

High Level Architecture

iCharts is a web application that aims in producing the best, easiest and fastest way to share, update and absorb facts & figures using charts on the net. The key requirements of iCharts are,

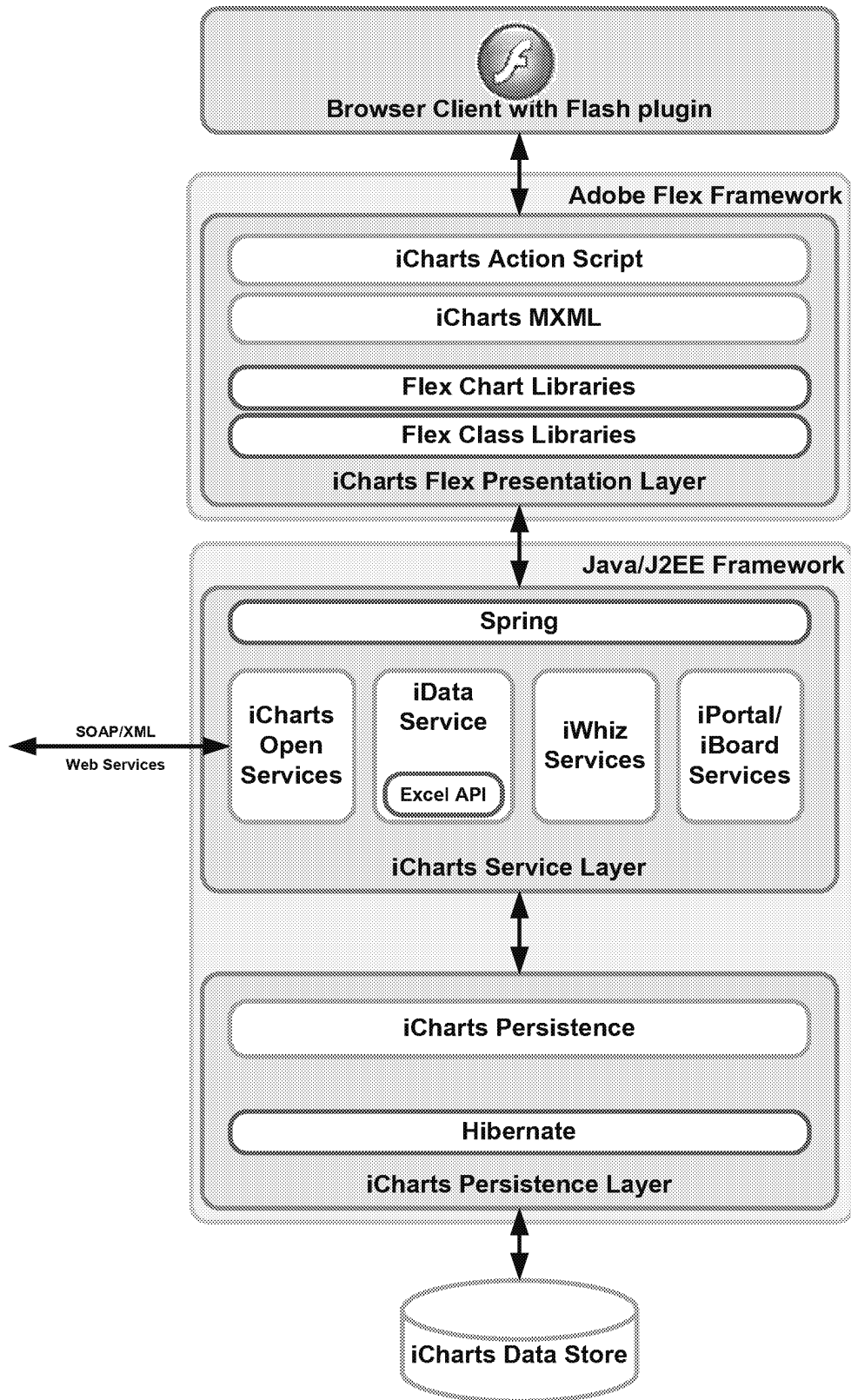
- ability for a user to upload an excel document that contains numerical data that has to be represented as a chart
- ability to select and customize a chart to represent the data that is uploaded
- ability to present the chart in the iCharts portal as well as iBoard page
- ability to embed the chart produced in any other website through a simple code snippet
- asynchronously update the chart when the data used to represent the chart changes in the iCharts repository
- provide a rich user interface to design and represent the charts that excels graphical representation of Microsoft Excel or any other chart tools.

There are four major components of iCharts which are briefly explained below.

- **iData** – This component is responsible for enabling uploading of data through excel files and enable certain data manipulations and calculations based on it.
- **iWhiz** – This component helps the user to select different charts and customize the way the chart has to be displayed based on the data loaded through iData.
- **iPortal** – This component displays a portal home page for the users to view different charts he/she has uploaded and manage drag and drop charts to iBoard.
- **iBoard** – This component helps in grouping multiple iCharts into one user-defined context and allows to associate documents and/or ancillary information, shared with others for comments either publicly or privately.

Based on the requirement and the components mentioned above, below is a high level architecture diagram that will enable the development of the iCharts application.

iCharts High Level Architecture



The architecture follows MVC (Model View Controller) architecture. Each of the components of the architecture is a layer by itself that is separated from each other and loosely coupled. The backbone of the architecture is the service and the persistent layer which provide support to store, retrieve data and manipulate the data based on the business requirements. The architecture components are explained briefly below.

- **Presentation Layer** – The presentation layer is responsible for receiving the requests from the client and calls the appropriate component in the service layer for processing and gets back the response. Once the response comes back from the service layer, the response is appropriately rendered at the server or rendered at the client end for display to the user.
- **Service Layer** – The Service layer is the back bone for the entire system. The business logic is placed in the service layer. It acts as an interface between the presentation layer and the persistence layer. Each component given in the business requirement of iCharts is available as a service in the service layer. There is an additional component the iCharts Open Services, which is an interface to the outside world to access the services of iCharts programmatically through web service calls.
- **Persistence Layer** – The persistence layer is responsible for receiving requests from the service layer and translating them to requests to the physical store which is a database.

Technology Choices

The below section describes the technology choices suggested for the architecture given above.

Service and Persistence Layer (Java/J2EE frameworks)

The service and persistence layer will be developed using Java/J2EE based technologies. The Java 2 Platform, Enterprise Edition (J2EE) defines the standard for developing multi-tier enterprise applications. The J2EE platform simplifies enterprise applications by basing them on standardized, modular components, by providing a complete set of services to those components, and by handling many details of application behavior automatically, without complex programming.

The J2EE platform takes advantage of many features of the Java 2 Platform, Standard Edition (J2SE), such as "Write Once, Run Anywhere" portability, JDBC API for database access and a security model that protects data even in internet applications. The J2EE standard includes complete specifications and compliance tests to ensure portability of applications across the wide range of existing enterprise systems capable of supporting the J2EE platform.

The service layer and the persistence layer are pure POJO (Plain Old Java Objects) that doesn't involve any other framework or technology's code polluted

in it. This makes the code clear and more focused on the business functions. But what binds these layers? How does the presentation layer talk to service layer? The answer is the Spring Framework which is a lightweight component container that implements the Inversion of Control (IoC) pattern.

Using an IoC container, components don't instantiate or even look up their dependencies (the objects they work with). The container is responsible for injecting those dependencies when it creates the components (hence the term "Dependency Injection" also used to describe this pattern). The result is looser coupling between components. The Spring IoC container has proven to be a solid foundation for building robust enterprise applications. The components managed by the Spring IoC container are called Spring beans. The presentation layer communicates with the services of the service layer through the Spring container which injects the dependencies during the runtime.

The persistence layer uses Data Access Objects which help in interacting with a framework called Hibernate. This framework takes the responsibility of communicating with the database and returning back response in the form of objects that can be directly used by the application. This eases and increases the development time of this layer as most of the low level database functions are handled by the Hibernate framework. Hibernate also persists the results such that repetitive requests that require the same result are retrieved quickly through the persisted objects rather than going till the database. The power of Hibernate comes from its capability as a Object Relational Mapping (ORM) tool to map the tables and relations between tables in the database in the form of objects and object relationships.

The data store could be handled by any relational database management system. Hibernate supports most of the popular relational database management system and can be portable any point of time without any code change to the supported relational database management system.

Presentation Layer (Adobe Flex Framework)

The rich presentation requirement for representing charts and manipulating them dynamically by the user requires a technology that can produce such an effect. Adobe Flash player is a cutting edge technology that supports rich and powerful graphic features. A technology that can produce an output that can be played by Flash will be the right combination that can meet the rich user interface requirements of iCharts. Adobe Flex is a framework that can sit on a presentation layer that coordinate with the business tier of an application and produce a rich user interface output that can rendered by Flash player. Flex supports integration with high end technologies such as Java/J2EE and .Net.

The Flex programming model is made of

- **ActionScript** – It is an ECMAScript compliant, object-oriented programming model. With some syntactical differences, ActionScript looks and feels similar to Java, and supports the same object-oriented constructs: packages, classes, inheritance, interfaces, strong (but also dynamic) typing etc.
- **MXML** – It is an XML-based language that provides an abstraction on top of ActionScript, and allows parts of an application (typically the View) to be built declaratively.
- **Flex Libraries** – An extensive set of class libraries. The online API documentation is available here in a Javadoc-like format.

The Flex source code (.mxml and .as files) is compiled into Flash bytecode (.swf) that is executed at the client-side by the Flash virtual machine using a Just-In-Time compiler.

The Flex runtime architecture is closely aligned with the just-in-time deployment model of web applications. The client portion of a Flex application is deployed as a binary file that contains the compiled byte code for the application. Users then deploy this file to a web server just as they would an HTML file or an image. When the file is requested by a browser, it is downloaded and the bytecode is executed by the Flash Player runtime.

Why Adobe Flex?

As mentioned earlier, this document is also intended to give some insights on why Flex is a better choice to fit the presentation layer. The previous section described the power of Flash in delivering a rich user interface that is on par and most of the times exceed the visual effects of a desktop application. Such an effect is a primary and critical requirement for iCharts to deliver crystal quality chart outputs. Hence the response to the browser should be a Flash player executable. If the output is something that can be played only by Flash then there are hardly few technologies that can fit in the presentation layer and deliver such an output. The primary technology that stands up here is Adobe Flex which is from the producers of the same Adobe Flash.

Note – Please refer to eWeek’s evaluation of Flex 2 given in the Appendix – References section.

Advantages of Flex + Flash

The following are the advantages of Flex.

- Enhance the user experience with richer, more effective applications
- Display complex data in a simple, visual way through charts and graphs
- Lead to more informed decision making based on a dynamic data interface
- Simplify complex data display

- Allow customers to create their own customized versions of a product
- Visually represent the customized item, complete with personalized elements
- Support co-browsing, enabling two customers in different locations to shop online together in the same application
- Connect to inventory systems to provide real-time availability status
- Increase customer loyalty
- Improve customer retention through ease of use and anytime, anywhere access
- Reduce operational costs through decreased paperwork and lower call center activity
- Increase opportunities for intelligent upselling and cross-selling
- Decrease data errors
- Extend the power Ajax using Flex + Flash combination
- The Flex developer tools provide a Rapid Application Development increasing the productivity and ease of development and maintainability. The tools integrate with popular IDE's such as Eclipse
- Good documentation and help manuals available online with community forums.
- Customer support from Adobe

Disadvantages of Flex

The following are the disadvantages of using Flex

- Flash is ubiquitous and supported by all major environments as well as popular browsers. The survey of the number of users that have Flash installed as a plugin to the browser is 98%. This suggests that the 2% of the users who are not having Flash player will not be able to use the services of iCharts.
- Flash is proprietary and is commercial from Adobe inc. If Adobe for any reason stops supporting Flex and/or Flash then the migration of the presentation layer to any other technology would be time consuming and complex.
- Even though the Flex development kit is free the Flex builder tools is commercial and a cost per user license is implied.
- There is a learning curve involved if the resources are new to Flash and Flex components.

Summary

With the right combination of Flex and J2EE the requirements of iCharts application can be met and would give the primary requirement of delivering a rich graph application on the net. Even though there are minor disadvantages of using Flex as the presentation layer technology, the advantages outnumber the disadvantages and the disadvantages has minimal or no impact at the time of this writing.

Appendix – References

Adobe Flex Home - <http://www.adobe.com/products/flex/>

Adobe Flex Whitepapers - <http://www.adobe.com/products/flex/whitepapers/>

Java/J2EE - <http://java.sun.com/j2ee/overview.html>

Spring Framework - <http://www.springframework.org/>

Hibernate Framework - <http://www.hibernate.org/>

eWeek's Flex 2 Evaluation

<http://www.eweek.com/article2/0,1895,2009659,00.asp>

Adobe Flash Player Statistics -

http://www.adobe.com/products/player_census/flashplayer/