

Data Visualization with Adobe Flex

Chris Giametta



<http://appfoundation.com>

<http://appfoundation.com/blogs/giametta>

<http://blog.appfoundation.com>



Adobe Flex Capabilities



Localization

- Static or dynamic resource bundles

Decouple Application Views

- Runtime Shared Libraries

- Components

Build Process

- Command line compiler

- Ant

Unit Testing

- FlexUnit

Communication Protocol Coverage

- HTTPService

- Web Services

- Remote Object

Communication Channels

- AMF (Action Message Format) used for serializing objects

- RTMP (Real Time Messaging protocol) used for messaging

Desktop application framework to run Rich Internet Applications

- Cross OS runtime
- File System I/O API
- Native OS Drag and Drop
- Database API
- System Notifications
- Application Updates
- Clipboard API for Copy and Paste
- Network Monitoring
- Offline data synch
- Local SQL Database (SQLite)

Flex Class Library



UI Controls

- DataGrid, List, Tree, TileList...
- CheckBox, DateField, ComboBox, Sliders...
- TextInput, RichTextEditor

Layout Managers

- HBox, VBox, Panel, Form, Canvas, Grid, ControlBar...
- View States
- Constraint based layouts

Navigation Controls

- Accordion, ButtonBar, LinkBar, TabNavigator, ViewStack

Media

- Audio, Video, Streaming

Data Visualization

- AdvancedDataGrid
- Charting

Expressive Support

- Vector Graphics
- Drawing API
- Skinning and Styling
- Transitions, Effects

Data Access/Manipulation

- HTTPService, WebService, RemoteObject
- Producer, Consumer
- Remote Data Binding
- E4X

ActionScript 3 Language Features



- Runtime exceptions
- Types
- Classes
- Method closures
- Regular expressions
- Namespaces
- Primitive Types

Adobe Flash Platform



Applications, content, and video



Tools to design and develop



Flash CS4
Professional



Flash Catalyst



Flex Builder

Framework



Flex

Clients



AIR



Flash Player

Servers



Flash Media
Server Family



BlazeDS
Data Services

Existing Solution Issues

- Other solutions do not easily support large data sets and require extra effort to implement workarounds.
- Business Intelligence solutions can be costly and not fit organization needs.
- Integration channels between technology can be difficult due to the lack of common communication protocols.

Flex Advantages



Flex is Designed For:

- Form based apps / heavy data entry
- Easy skinning and styling of components
- Layout engine
- Separation of code and controls
- Target audience – Enterprise application developers
- Can easily integrate with most back-end technology to communicate to most databases

Working with Large Data Sets

Data Centric Applications Start with the UI

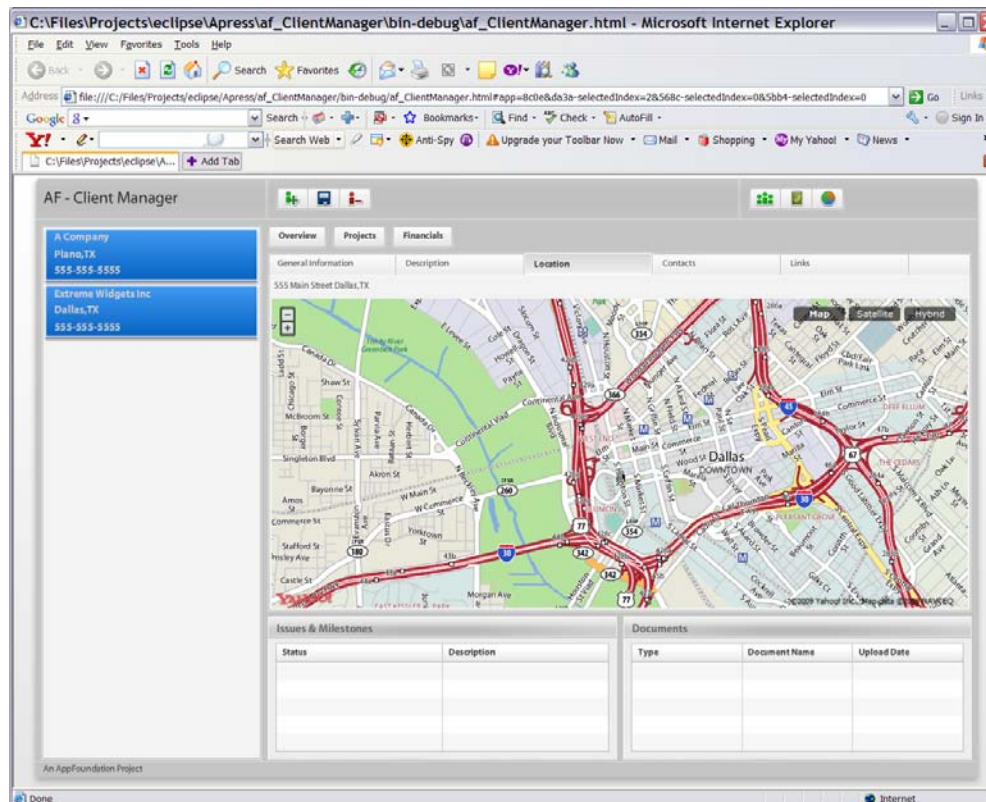


- Great User Interfaces focus on what you need.
- Data is at the center of the app and you now have Rich tools to work with.
- Desktop Application functionality is available in the browser and revolves around Views of the data, not pages.

Desktop Applications are Replacing Web Apps



- Next Generation applications are replacing web applications
- Desktop applications bring better value to business through productivity improvements



Problem Statement



Problem: How do you give your customers large sets of data to reliably work with?

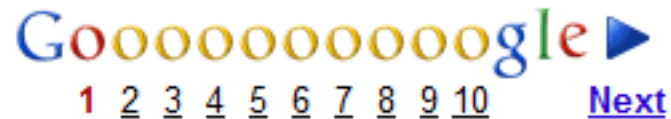
Large Amounts of Data



- Enterprise Applications: Products, Customers, Invoices, Purchase Orders, Documents, News, Users, SKU.
- Many datasets are 10,000 or more records.
- Traditional web applications will not work where rich Internet applications are better suited to support large data sets.

Explicit Paging with Large Data Sets

- Paging allows the user to see specified batches of data; like on popular search engines
- Most users don't look past the first two or three pages of a result set. So returning 3000 results to the client would be a waste in resources.
- Returning massive results to the client will cause it to render slowly



Implicit Paging with Large Data Sets

- With Implicit Paging, we don't want the user worried about viewing finite pieces of data. We want them to get the big picture.
- Rather than downloading all data at once, we will download data on an as-needed basis.
- In this example, when the datagrid is scrolled it will ask for the next data set. Again, we are focused on performance.

Data Aggregation

- Data Aggregation is simply any process in which information is gathered and expressed in summary form.
- Common use is to get specific information about groups such as age, profession, or income.
- In Flex, this information can be used to personalize site views for the user.
- For example, a web site that sells books might take a look at the user's age and present books that would more appeal to them.

Building Dashboards and Rich Data Visualization Applications

Dashboard Considerations

Why Build a Dashboard?

- Help support better decision making.
- Help to make sense of large data sets.
- Give users actionable information.
- Derive decisions based on otherwise unrecognized data patterns.

Considerations for Building a Dashboard:

- Clearly understand the business the dashboard will support.
- Keep it simple.
- Bring the application to life; aesthetics play a crucial role in delivering critical data.
- Usability is crucial! Pay attention to natural user gestures in your applications.

<http://demos.appfoundation.com/hp/dashboard/>

<http://demos.appfoundation.com/gis/>

Building Enterprise Flex Applications

Project Team Structure

- A typical project team structure to deliver RIAs would consist of:
 - Project Manager
 - Interactive Designer – Wireframes, screen mocks, artwork
 - Flex Architect and Developers
 - Java Architect and Developers
- Large Interactive Project Team
 - Front-end Development (Developers, designers, information architects, producers)
 - Back-end Development (System Architects, Java developers, technical managers)
 - Business Management
 - Legal
 - Public Relations
 - Project Managers
 - Media Team

Tech Stack

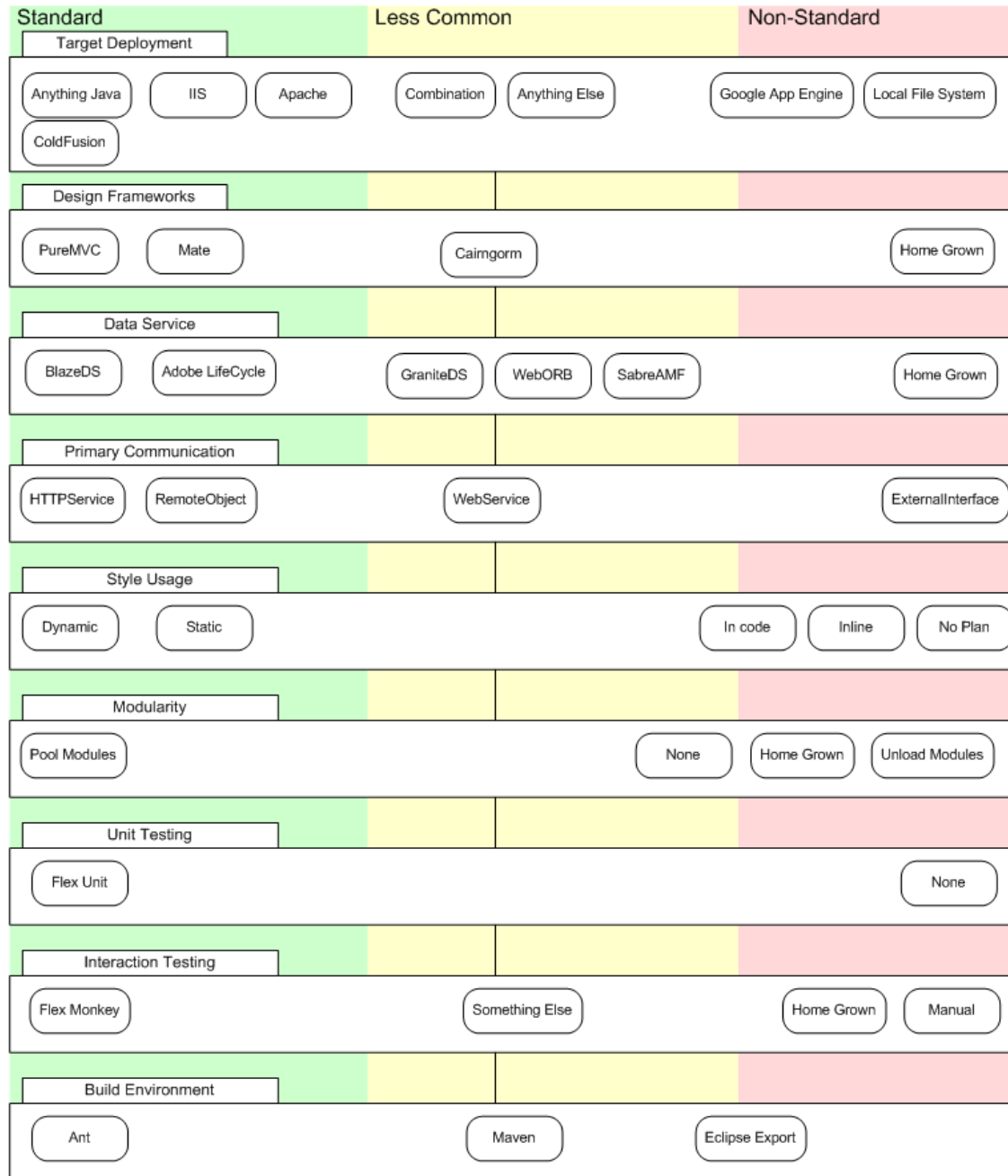


Flex applications, like most apps, have many different twists and turns within the technology stack.

The listing here shows the difference in common vs. non-standard Flex technology standards.

Flex Technology Standards (Subjective)

Less Standard = More Potential Workarounds



Flex Enterprise Architecture



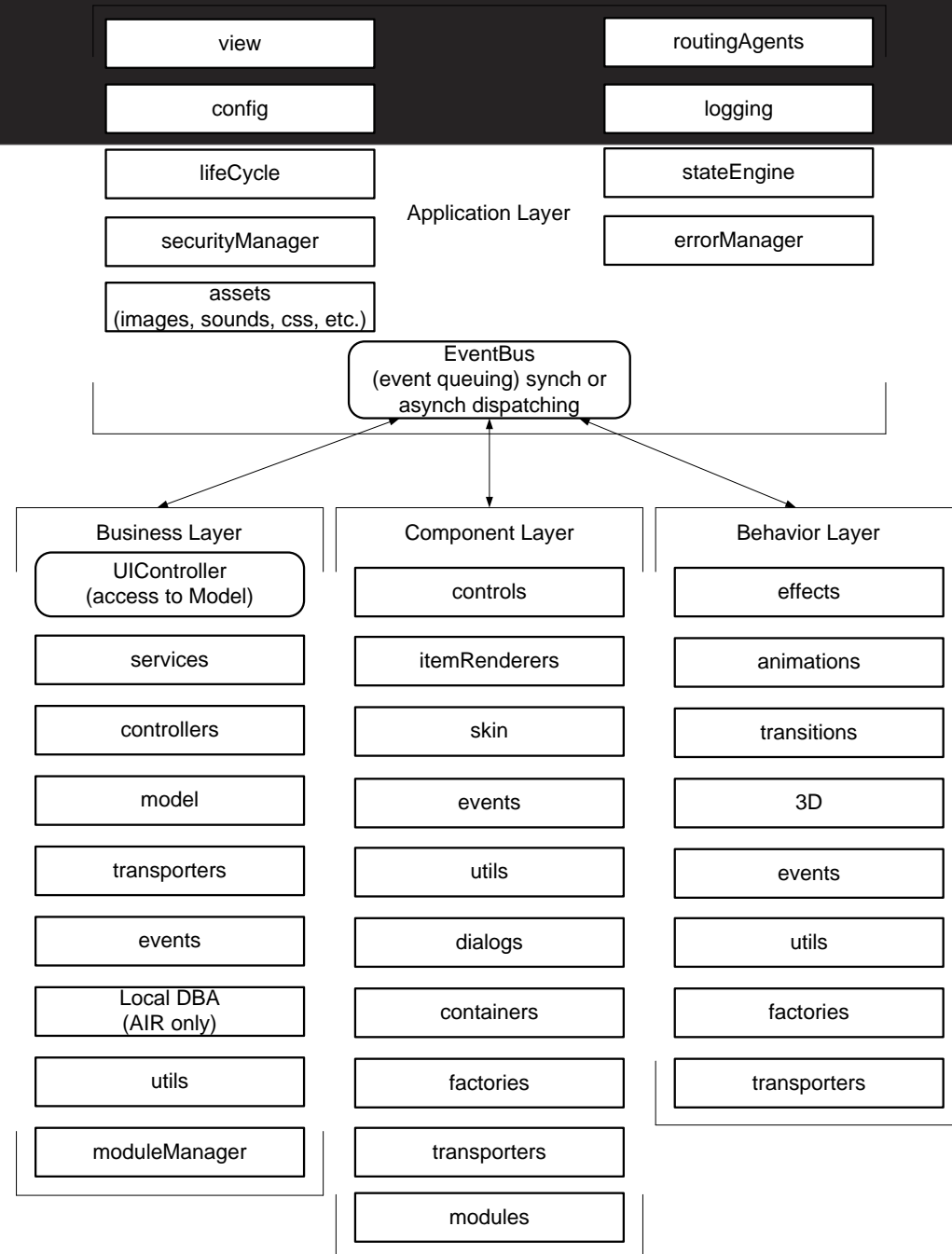
Most Enterprise Flex Applications are using custom component. How are you handling:

- Dynamic assets and CSS content
- Localization
- SWF size (modules)
- Development Framework
- Automation Testing
- Build Process
- Resource Bundles
- Object Graphs

Flex Ent Framework

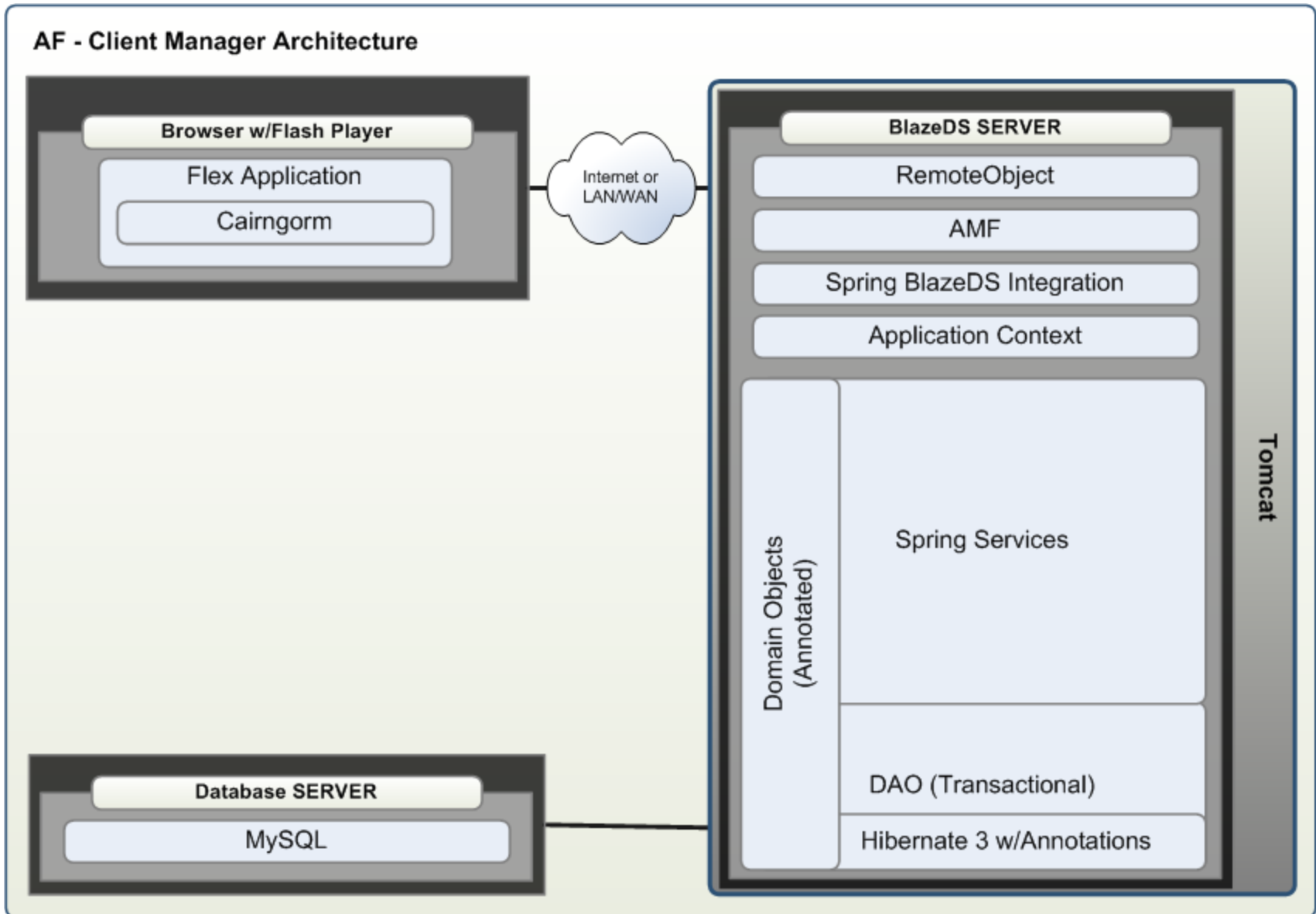


To better examine the anatomy of an Enterprise Flex application, take a look at the application architecture for a large scale Flex application.



Contact Manager Example Application

Contact Example Application Stack



Example App Required Downloads

- Spring and Spring BlazeDS Integration – Both with dependences: <http://www.springsource.org/download>
- Download Hibernate with annotations from: <http://www.hibernate.org>
- Get Cairngorm 2.2.1 from: <http://opensource.adobe.com/wiki/display/cairngorm/Downloads>
- Get BlazeDS (Binary Distribution) from: <http://opensource.adobe.com/wiki/display/blazeds/BlazeDS>
- Download Apache Tomcat from: <http://tomcat.apache.org/download-60.cgi>

Contact Table Definition



- Create a simple table in your favorite database called jm_Contact.

For MySQL execute:

```
CREATE TABLE `javamug`.`jm_contact` (  
  `contactid` BIGINT NOT NULL,  
  `name` VARCHAR(120),  
  `email` VARCHAR(120),  
  `phone` VARCHAR(45),  
  `location` VARCHAR(120),  
  PRIMARY KEY(`contact_id`)  
)  
ENGINE = InnoDB;
```

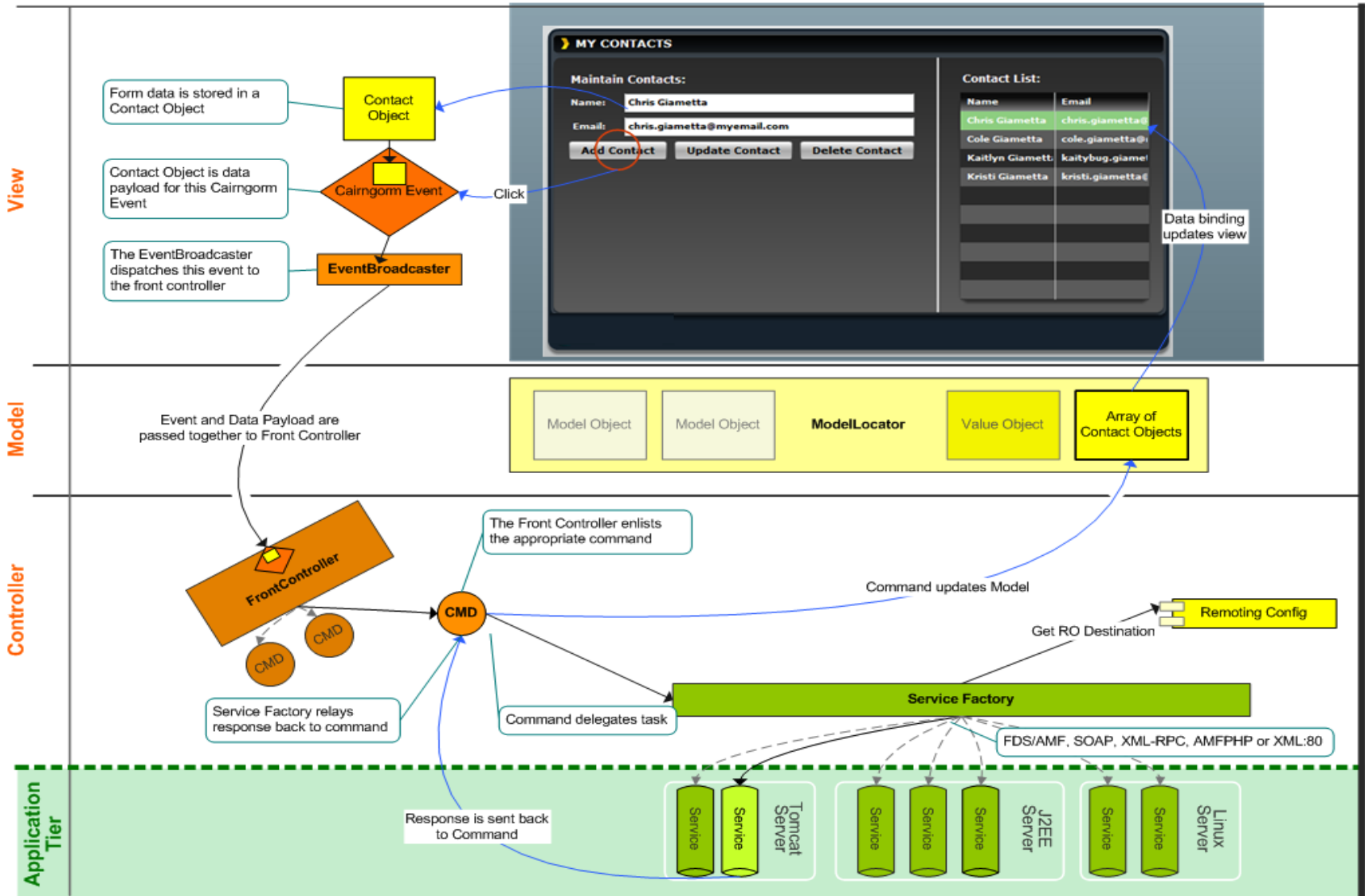
Contact Service in Spring

To fully implement the contact service, you need to create a Spring Service and DAO.

- Create an interface for the service and bean.
com.af.javamug.services.ContactService
com.af.javamug.services.ContactServiceImpl
- Create an interface for the DAO and bean.
com.af.javamug.dao.ContactDAO
com.af.javamug.dao.hibernate.ContactDAOImpl
- Create a Contact domain object to be used throughout the Spring application.

Cairngorm Microarchitecture

Flex UI Microarchitecture – Basic Server RPC Using Cairngorm 2.0



Updated: 01/23/2007 - Diagram updated by Chris Giametta with special thanks to Evan Gifford, Steven Webster, Alex Uhlmann, Russell Munro, Jesse Warden, Dan Nielsen, Tom Chiverton

Application Transport Architecture using Cairngorm



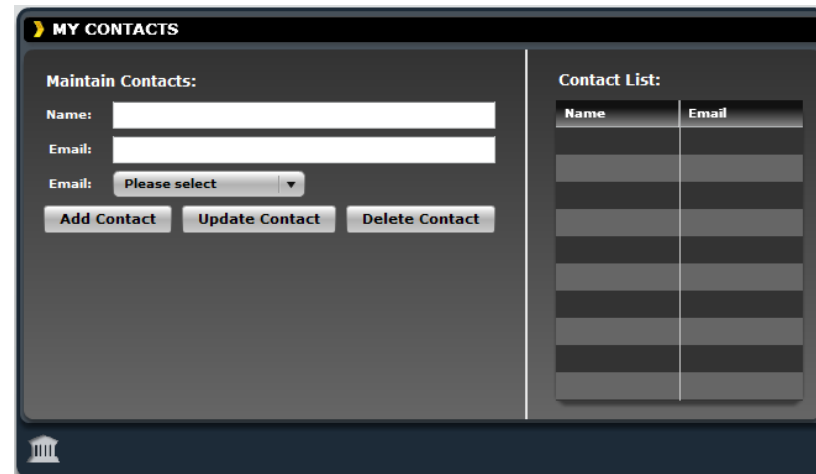
- Create the RemoteObject Services with an endpoint
- Set the Services and ContactController in the main application tag
- Create the CRUD contact Events
- Create the CRUD contact Commands
- Create the FrontController and reference the commands to the events
- Create the Contact Delegate

Running the Application

- Build the Flex and Spring application
- Deploy the WAR files to the Tomcat server. In my example's case:

C:\Tomcat 6.0\webapps
- Start the Tomcat server instance and hit the following local URL:

http://localhost:8080/jm_Flex/jm_Flex.html





AppFoundation
Design.Refine.Deliver

Data Visualization with Adobe Flex

Chris Giametta



<http://appfoundation.com>

<http://appfoundation.com/blogs/giametta>

<http://blog.appfoundation.com>

