Grails and Flex

Strategies for getting "Flexible" on the client...

What is a Rich UI?

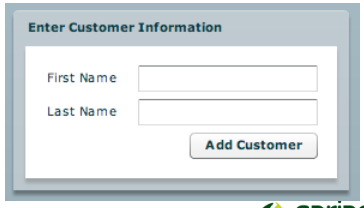
- User interface that updates elements without full page refreshes
  - Offload UI rendering to the client machine
  - Client can be rich, expressive
  - What, Client/Server AGAIN?!

Adobe Flex

- Very popular Rich User Interface
  - XML language (MXML) defines UI
  - ActionScript (ECMAScript-based) as programming language
- Why Flex?
  - Looks great out of the box
  - Languages (MXML and AS) are relatively easy to pick up
  - Easy to network to other data sources (feeds can be files, XML, data streams, or binary remoting)

Sample Flex MXML File

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
                layout="horizontal">
<mx:Panel layout="vertical" title="Enter Customer Information">
   <mx:Form backgroundColor="white">
      <mx:FormItem label="First Name">
         <mx:TextInput id="firstName" width="150" />
      </mx:FormItem>
      <mx:FormItem label="Last Name">
         <mx:TextInput id="lastName" width="150" />
      </mx:FormItem>
      <mx:FormItem horizontalAlign="right" width="100%">
            <mx:Button  label= "Add Customer" />
      </mx:FormItem>
   </mx:Form>
</mx:Panel>
</mx:Application>
```



CHARIOT SOLUTIONS

spring source

Sample Actionscript

Embed in XML??

```
<mx:Script>
    <![CDATA[
        import mx.events.DataGridEvent;
        import model.Person;
        import mx.collections.ArrayCollection;
             ...
        [Bindable]
        private var personList:ArrayCollection = new ArrayCollection();

        private function displayData(event:Event):void{
            var peopleXML = XML(peopleService.lastResult);
            var personXML:XMLList = peopleXML.person;
            for each(var dataItem:XML in personXML) {
                var personData:Person = new Person();
                personData.name = dataItem.firstname.text() +
                        ' ' + dataItem.lastname.text();
                personData.email = dataItem.email.text();
                personData.phone = dataItem.phone.text();
                personList.addItem(personData);
            }
        }

    ]]>
</mx:Script>
```

Look, ma! Imports!!!

Annotations in Brackets??

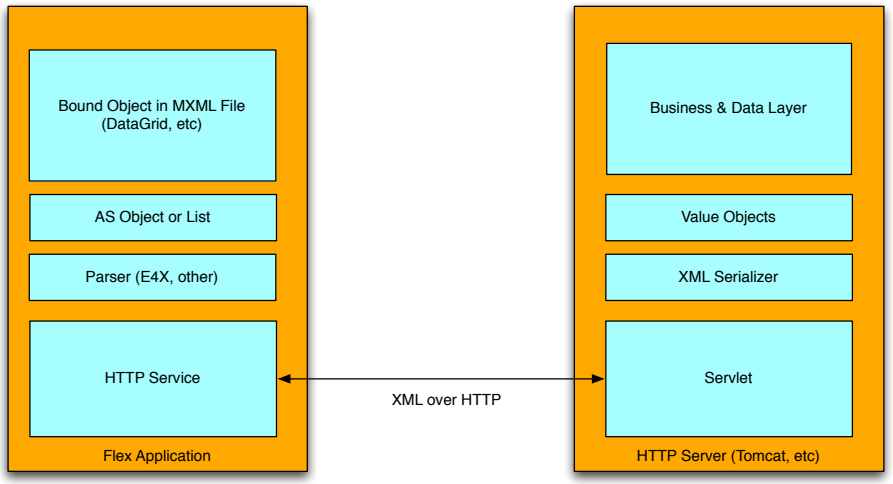
var variable:Type

ECMA Script language can be embedded using <mx:Script> or live as .as files

Looks like a mix of Java, JavaScript, Objective-C, and .NET

Flex Networking Options

HTTP Server Remoting



CHARIOT SOLUTIONS

spring source

HTTP Remoting : Grails Example

```
package org.chariot.confplanner.controller

class FlexHttpDemoController {

   def xmlpeople = {
      render(contentType:"text/xml") {
         people {
           person( name: "Ken Rimple",
            address: "123 Anywhere Street",
            email: "ken.rimple@null.com")
         }
      }
   }
}
```

- Rendering XML in Grails is easy...
- Start by using render to build markup
  - Attributes are parameters
  - Tags are closures

```
<people>
  <person name="Ken Rimple"
       address="123 Anywhere Street" email="ken.rimple@null.com"/>
</people>
```

CHARIOT SOLUTIONS

spring source

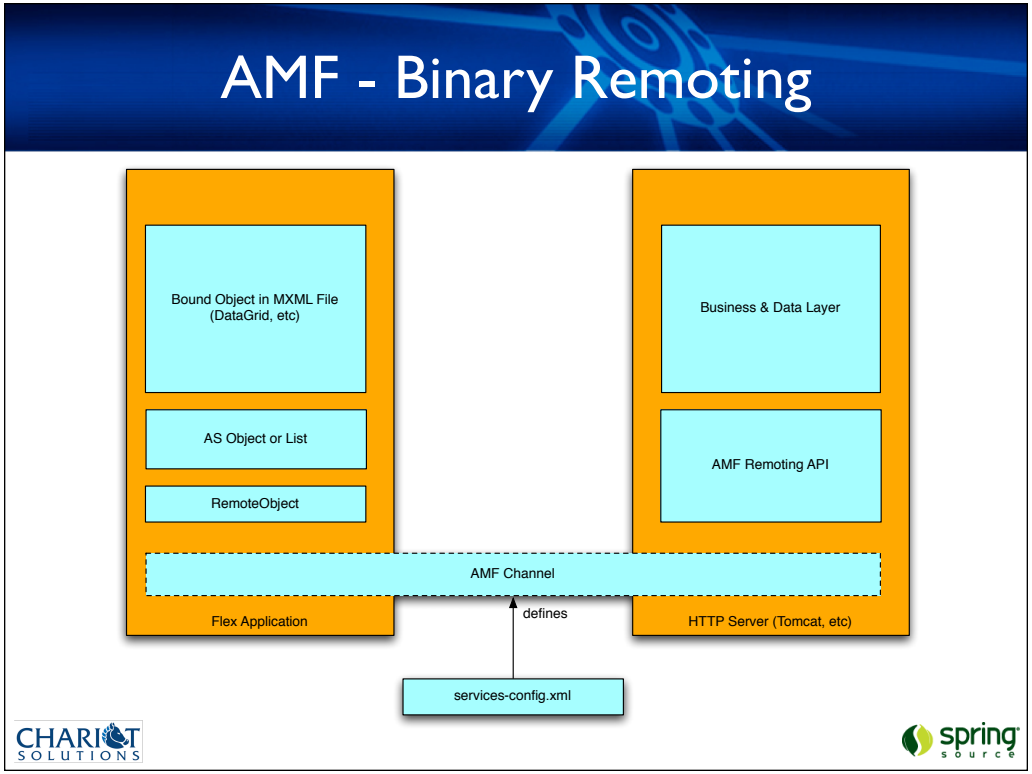
HTTP Remoting : Flex Example

- The Flex Application

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
   layout="vertical">

   <mx:HTTPService id="peopleService"
      url="http://localhost:8080/timesheet/flexDemo/xmlpeople" />

   <mx:Button click="{peopleService.send()}" label="Get People" />

   <mx:DataGrid id="myDataGrid"
      dataProvider="{peopleService.lastResult.people.person}">
      <mx:columns>
         <mx:DataGridColumn dataField="name"/>
         <mx:DataGridColumn dataField="address" />
         <mx:DataGridColumn dataField="email" />
      </mx:columns>
   </mx:DataGrid>

</mx:Application>
```

CHARIOT SOLUTIONS

spring source



AMF – Binary protocol that can serialize across HTTP
Provides super-fast remoting mechanism for passing object graphs from client to server

AMF -vs- HTTP/XML

	AMF/Blaze DS	HTTP
Pro	Fast, Easy Serialization	Open - Can remote to anything
Con	More complex setup, harder to debug	Slower to process, larger network packets

AMF-based Remoting Frameworks

- Many available
  - Java: **BlazeDS**, GraniteDS, Cinnamon
  - Ruby/Rails: Ruby AMF
- Vary from simple enablement APIs to remoting of ORM objects
  - With great power...

Adobe BlazeDS

- Adobe's "Reference Implementation" of AMF on Java
  - Typically enabled via a Messaging Service "gateway servlet"
  - Turns Java objects into ActionScript objects and vice-versa
  - Provides access to datastores, queues, resources via "Service Adapters"

BlazeDS in Spring

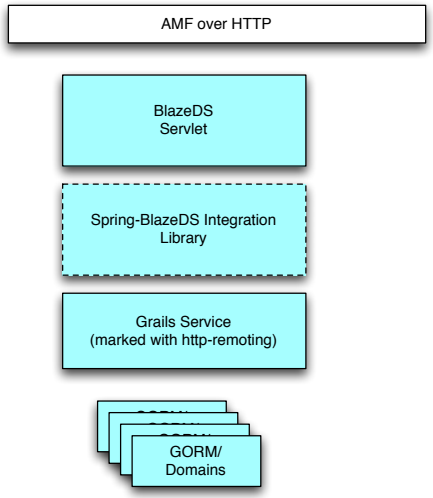
- You can use BlazeDS directly
  - Just Java libraries and Flex configuration
  - But tedious to set up
- Spring BlazeDS Integration Project - integrates Spring with Flex using BlazeDS
  - Allows for annotation-driven or Xml-driven bean discovery

BlazeDS and Grails

- BlazeDS plugins for Grails
  - Original flex plugin (proof-of-concept)
  - Grails Flex Scaffold
  - * not developed yet - Grails plugin based on Spring Blaze DS Integration project
- You can roll your own...

The Grails Flex Plugin

- Installed with: grails install-plugin flex
- Sets up BlazeDS and a remoting channel
- Automatically exposes services that are marked with
  static expose = [`blazeds-remoting']
- On Flex-side, use mx:RemoteObject and directly reference service by name



CHARIOT SOLUTIONS

spring source

Grails Flex Plugin DEMO

Spring Flex Example

- Grails-side

```
class PeopleService {

    static expose = ['flex-remoting']
    def fnames = ['John', 'Joe', 'Zeke']
    def lnames = ['Smith', 'Jones', 'Cramden']


    boolean transactional = true

    def list() {
       def people = []
       fnames.each { fname ->
           lnames.each { lname ->
               people << new Person(name: fname + " " + lname,
                                    address:"10 Main Street",
                                    email:"$fname.$lname@null.com")
           }
       }
       people
    }
}
```

```
class Person {

    String name
    String address
    String email
}
```

CHARIOT SOLUTIONS

spring source

Spring Flex Example

- Flex-side

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
    layout="vertical">

    <mx:RemoteObject id="ro" destination="peopleService" />

    <mx:Button click="{ro.list()}" label="Get People" />

    <mx:DataGrid id="myDataGrid" dataProvider="{ro.list.lastResult}">
        <mx:columns>
            <mx:DataGridColumn dataField="name"/>
            <mx:DataGridColumn dataField="address" />
            <mx:DataGridColumn dataField="email" />
        </mx:columns>
    </mx:DataGrid>

</mx:Application>
```

Benefits

- No definition of URL, just endpoint (service)
- Automatic serialization/deserialization of Groovy / ActionScript classes
- Binary format improves performance

Flex Remoting Plugin Issues

- Currently a proof of concept
- Not heavily tested in production
- Webtier compilation - issues
- Will be obsoleted with the Spring/ BlazeDS plugin in a future release
- Does not generate or scaffold ActionScript classes or Flex forms

Working with Flex/Grails

- Use Flex Builder for editing Flex projects
  - Be aware of the visual editing tools - will change code and sometimes for the worst
- Place your flex project inside of the Grails project
  - For HTTP Services, use relative paths from the flex main class
  - For Binary AMF / BlazeDS services, use service name, and set up your project settings for remoting objects, etc.
  - Consider using Flex Mojos Project if on Maven (demo if time)

Alternative Approaches

Manually Installing Spring Blaze DS Integration in Grails

- High-level overview
  - Install libraries, configuration
  - Use either XML namespace flex: or @RemotingDestination annotation from Spring-Flex API
  - Can set up JMS channels and publish/subscribe for async messaging
  - Can enable any Grails service with the above methods

Grails Spring BlazeDS

- Demonstration

gdsflex plugin

- Wraps the GraniteDS API
- Provides deep linking to Hibernate / JPA data from the Flex tier
- Allows exposing Services and Controllers
- Uses AMF binary protocol
- Tide extends Spring to client side
- Claims to support Spring Security

Thank you!