Israel JBoss User Group
Session 09

Flex
RIAvolutionize your web app

Ilan Avigdor
"Tikal Knowledge"
Ilan@Tikalk.com

Israel JBUG

Agenda

- Introduction
- Architecture
  - Client + Demo
  - Communication
  - Server
- Conclusions

Israel JBUG

RIA Concepts

- Desktop Rich, Internet Reach
- Application vs. Site
- Plugin vs. Browser
- Runs on client machine

FLEX Concepts

- Layer on flash for developers
- ActionScript 3.0
- Compiled for AVM2

Advantages - RIA

- For Users
  - Desktop-like experience
  - Responsiveness
  - Cross-browser
  - Statefull (no unnecessary page reloads)

Advantages - RIA

- For Developers
  - OOP – Web for the non-webbers
  - Statefull
  - Maintenance - One code base for web and desktop
  - Effortless deployment

Disadvantages – RIA

- Plugin dependency
- Initial load and initialization
- Desktop-like
  - Memory leaks
  - CPU bottlenecks

Advantages – Adobe FLEX

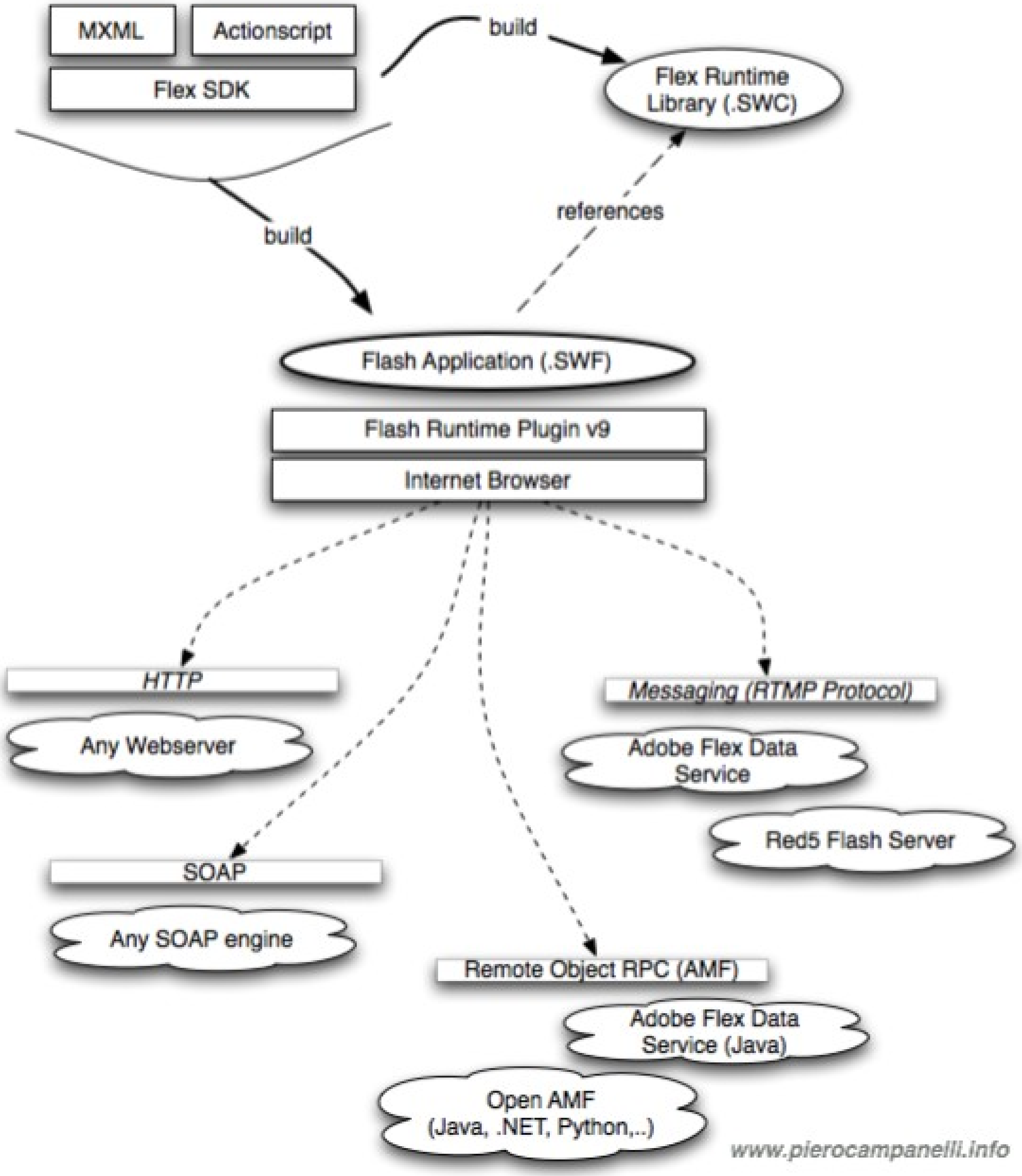
- Capabilities
- Ubiquity
- Deployment Flexibility
- Proven Technology
- Expressiveness
- Openness
- Innovation

Disadvantages – Adobe Flex

- Limited to Flash player abilities
- SWF is proprietary
- PopUp windows limited to browser size
- Multi Threading not supported
- Search engines – Not

Israel JBUG

Architecture



Israel JBUG

Architecture

- Client
- Communication with server
- Server

Flex Client Features

- Rich Component Library
- Display list programming
- MXML
- Event Model
- Effects
- Styling & skinning
- Binding
- Charting (not included)
- Constraint based layout

Flex Client Features cont.

- Drag & Drop
- History management
- Printing
- Communication with wrapper
- Shared object
- Highly customizable
- Rich media integration
- Modularity

Israel JBUG

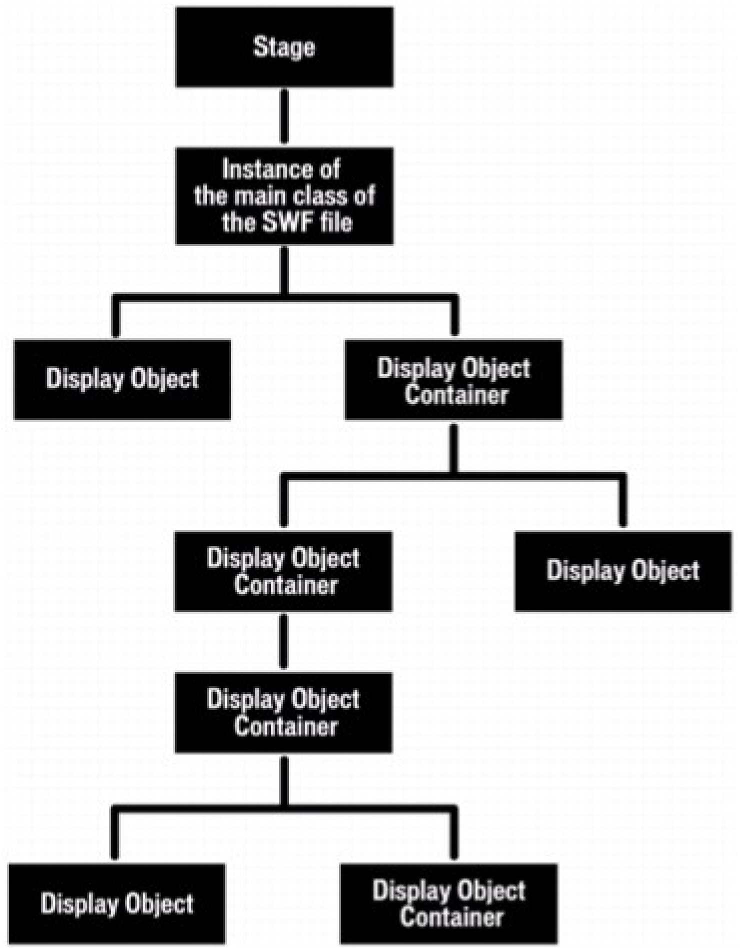
Flex Visual Components

- General
- Buttons
- Date
- Loaders
- Menu
- Text
- Containers
- Repeaters

Flex Components

- Print
- Validators
- Formatters
- Effects
- States
- Transitions
- Data vizualization
  - Charts
  - DataGrid
  - OLAPDataGrid

Display Programming



MXML

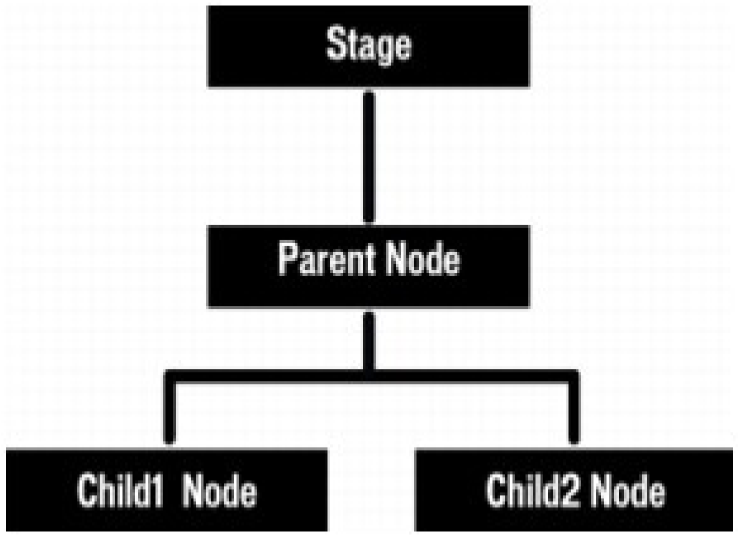
```
<mx:Application >
   <mx:Button id="button" />
/mx:Application>
```

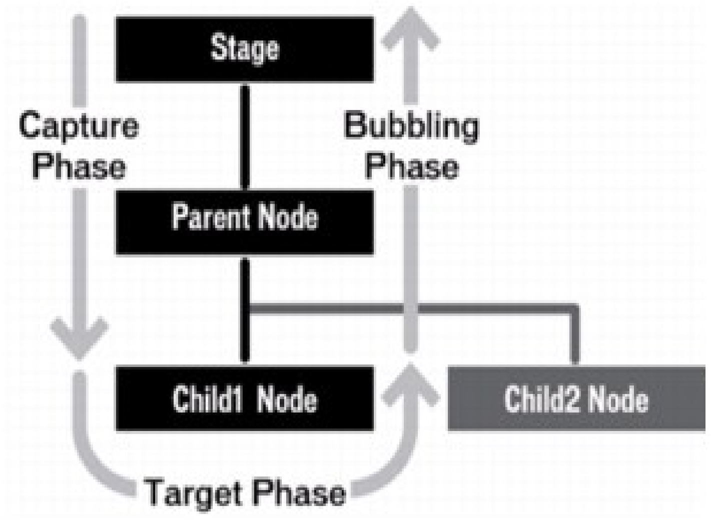
Equals

```
public class Example extends Application
{
  internal var button:Button;

  public function Example() {
    super();
    button = new Button();
    addChild(button);
}
```

Flex Event Model



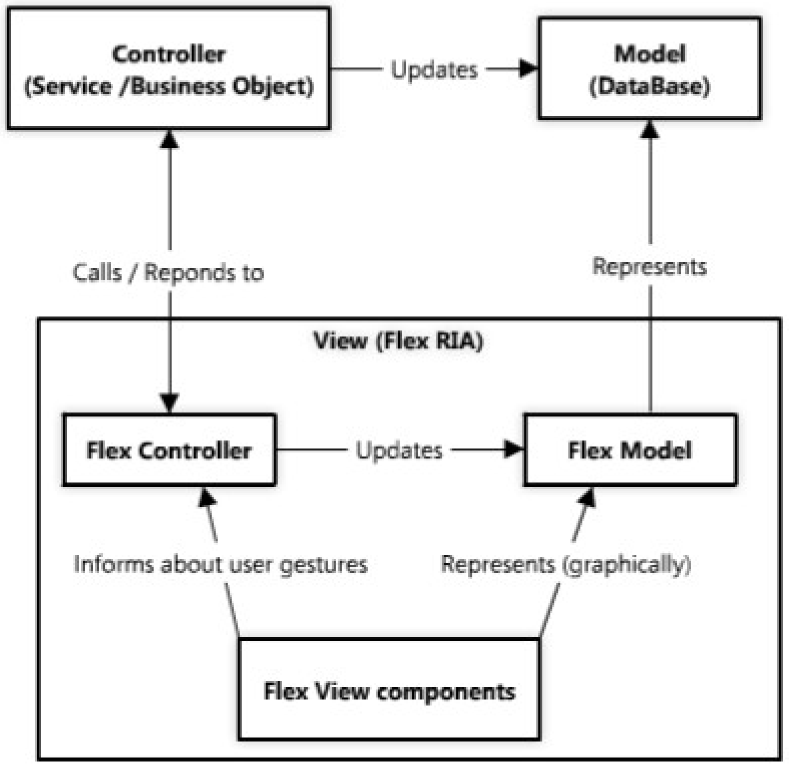


Israel
JBUG

Flex Binding

```
<mx:Application >
    <mx:TextInput id="src"/>
    <mx:Text id="dest"
            text="{src.text.toUpperCase()}"/>
</mx:Application>
```

Architecture
Client – MVC Architecture



Architecture
Flex Frameworks

- Why?
  - Out-Of-The-Box MVC framework
  - Cross projects
  - Better scalability
- Which?
  - Cairngorm
  - PureMVC

Israel JBUG

DEMO

Architecture
Communication With Server

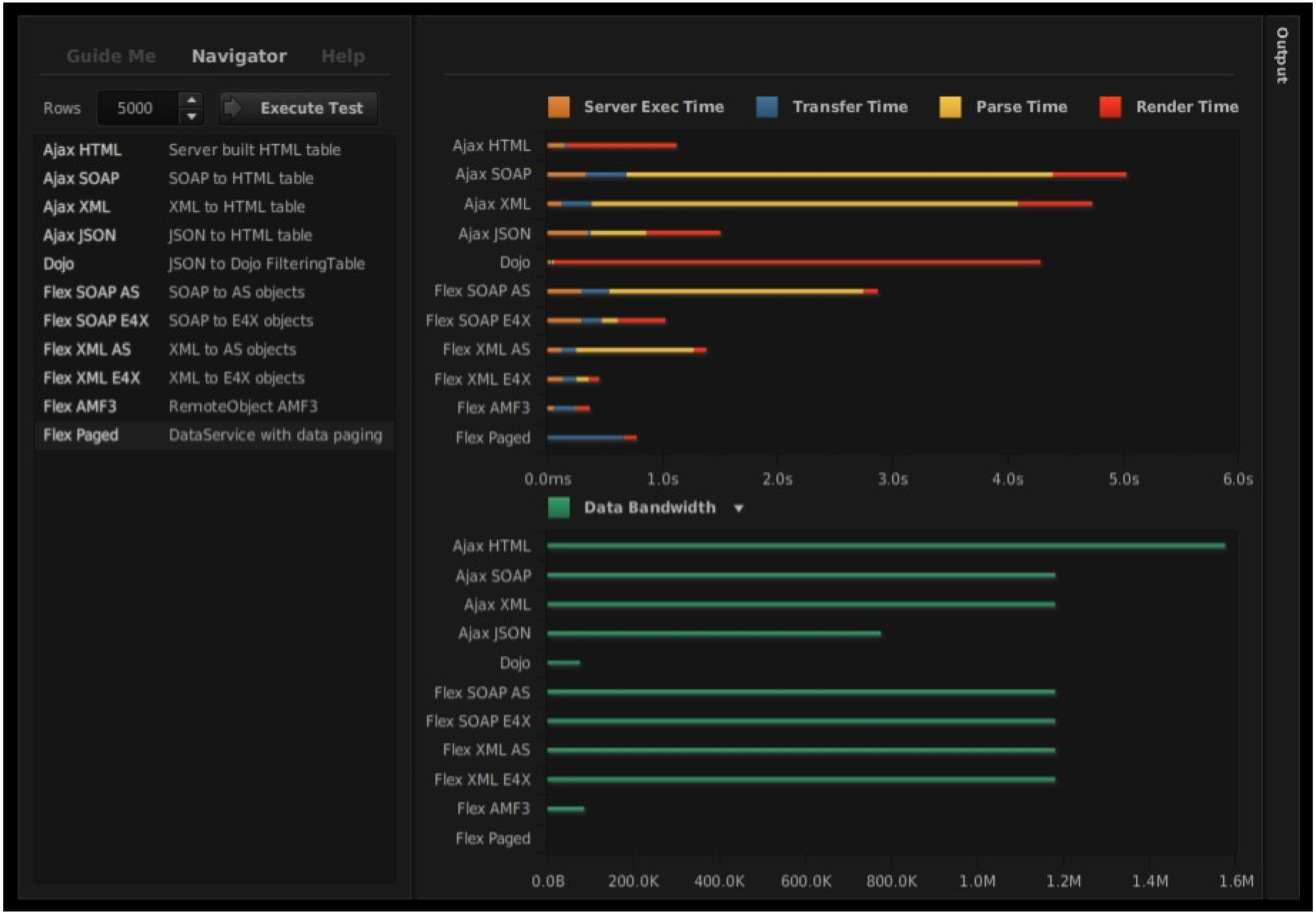
- Protocols
- Communication Patterns
  - RPC
  - Data Push

Israel
JBUG

Communication With Server Protocols

- HTTP
- SOAP
- XML
- Binary (POP3, SMTP, IMAP, and NNTP)
- RTMP
- JSON
- AMF3

Communication with Server Protocols benchmark



Ajax HTML	Server built HTML table
Ajax SOAP	SOAP to HTML table
Ajax XML	XML to HTML table
Ajax JSON	JSON to HTML table
Dojo	JSON to Dojo FilteringTable
Flex SOAP AS	SOAP to AS objects
Flex SOAP E4X	SOAP to E4X objects
Flex XML AS	XML to AS objects
Flex XML E4X	XML to E4X objects
Flex AMF3	RemoteObject AMF3
Flex Paged	DataService with data paging

Communication With Server
AMF3 Implementations

- Fluorine -.Net
- Red5 – java
- Cinnamon – java
- SabreAMF – PHP5
- Rubyamf – Rails
- PyAMF - Python

Communication With Server

Communication Patterns

- Request Response
  - HTTP
  - Web Services
  - Remoting
- Data Push
  - Sockets - Bidirectional connection
  - Messaging - Pub/Sub

Communication with Server HTTP

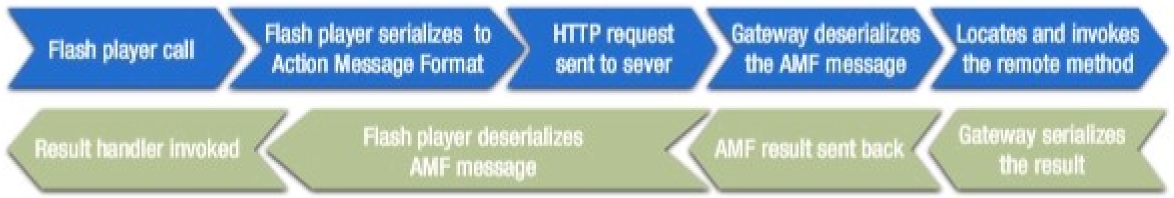
```
<mx:HTTPService
    url="http://localhost/req.php"
    method="POST">
  <mx:request>
    <username>UserName</username>
    <address>Address</address>
  </mx:request>
</mx:HTTPService>
```

Communication with Server
Web Service

```
<mx:WebService id="userReq"
wsdl="http://Localhost/users.cfc?wsdl">
  <mx:operation name="returnRecords"
    result="returnRecordsHandler()" />
  <mx:operation name="insertRec"
    result="insertCFCHandler()" />
 </mx:WebService>
```

```
private function clickHandler():void
{
  userReq.insertRec(n.text,address.text);
}
```

Communication with Server Remoting


Flash player call
Flash player serializes to Action Message Format
HTTP request sent to sever
Gateway deserializes the AMF message
Locates and invokes the remote method
Gateway serializes the result
AMF result sent back
Flash player deserializes AMF message
Result handler invoked

```
<mx:RemoteObject id="userReq"
  destination="ColdFusion"
  source="flexapp.returnusers">
<mx:method name="insertRecord"
  result="insertHandler()"/>
</mx:RemoteObject>

private function Onclick():void
{
userReq.insertRecord(   name.text,
                     address.text);
}
```

Communication with Server Sockets

```
<mx:Application >

private function **Onconnect**(event:Event):void
{
    trace("Connected");}

private function **Ondata**(event:DataEvent){
    trace ("data arrived" + event.data);}

  <net:XMLSocket id="sock"
    connect="**Onconnect**()" data="**OnData**()"/>

  <mx:Button
  click="{sock.connect('localhost',4444)}"/>

</mx:Application>
```

Communication with Server
Messaging Registration

```
<mx:Producer id="chat"
  destination="MyTransientTopic" />
<mx:Consumer id="chatSubscriber"
  destination="MyTransientTopic"
  message="receive(event)" />
```

Communication with Server
Messaging Send/Receive

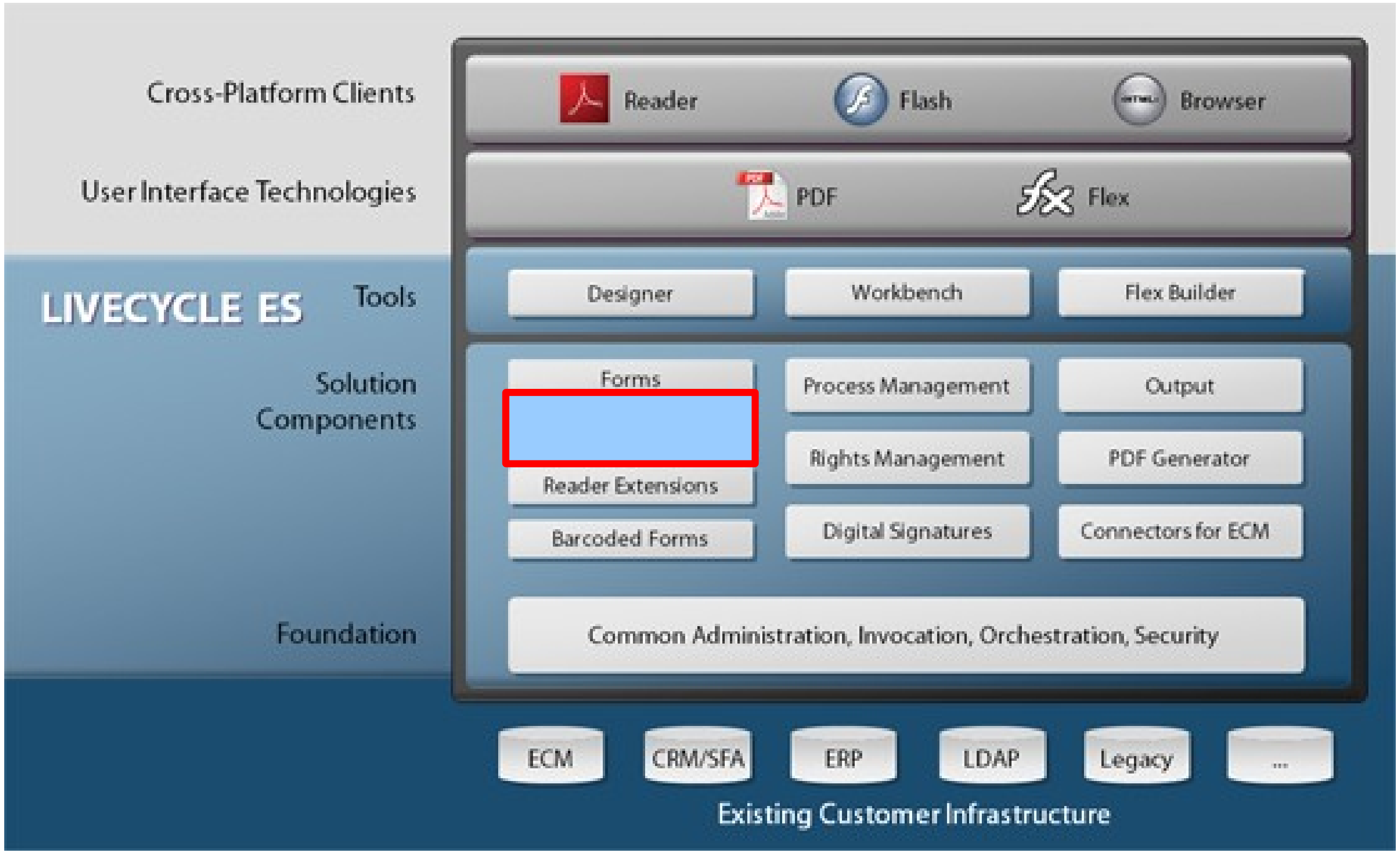
```
private function sendChatMessage():void {
  msg = new AsyncMessage();
  msg.body = input.text;
  chat.send(msg);
}
receive(event:MessageEvent):void{
var msg:AsyncMessage =   event.message as
AsyncMessage; output.text += msg.body ;
 }
```

Architecture
Server side Platforms

- LiveCycle ES – J2EE
- LiveCycle DS – J2EE
- BlazeDS(open source) – J2EE
- Granite(open source) – J2EE
- WebOrb(open source)
  - .NET, J2EE, ROR, PHP

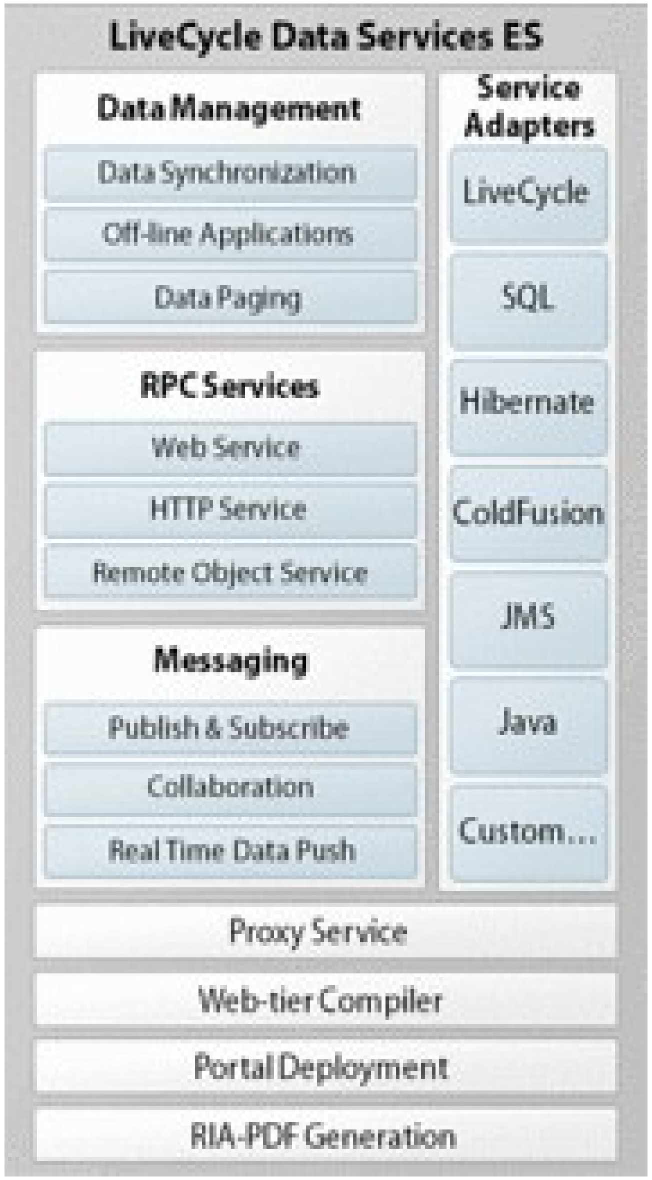
Israel
JBUG

Server side Platforms
LiveCycle ES



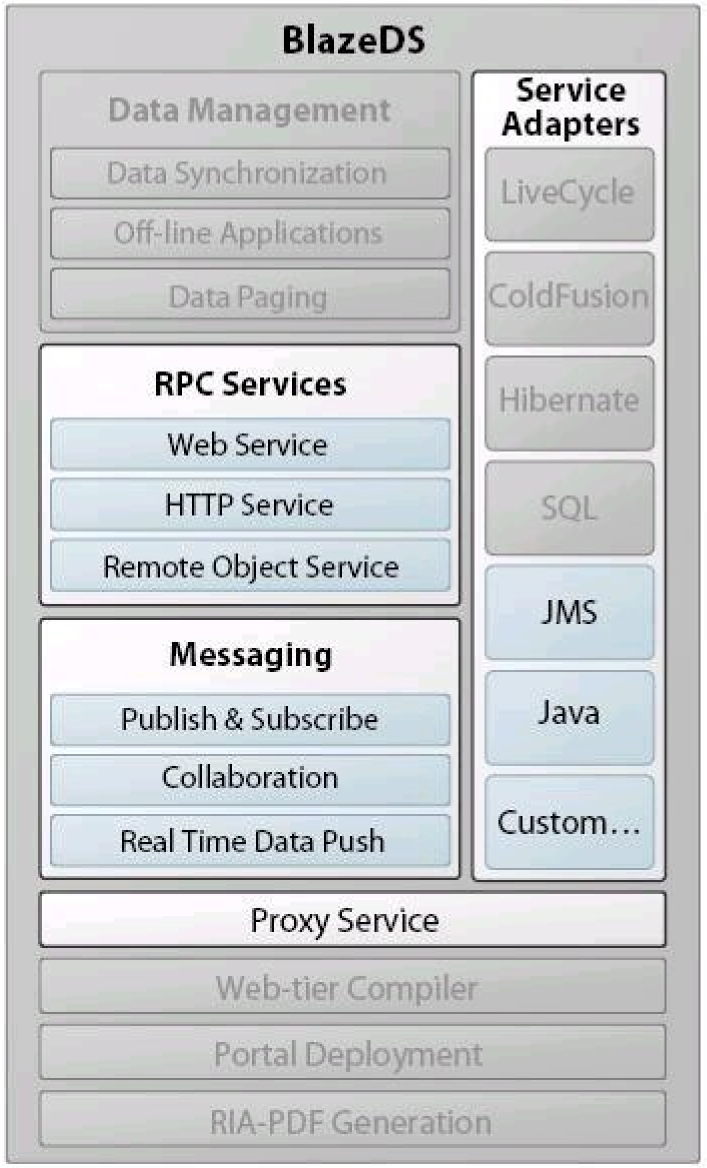
Israel JBUG

Server side Platforms LiveCycle DS vs. BlazeDS



LiveCycle Data Services ES

- **Data Management**
  - Data Synchronization
  - Off-line Applications
  - Data Paging
- **RPC Services**
  - Web Service
  - HTTP Service
  - Remote Object Service
- **Messaging**
  - Publish & Subscribe
  - Collaboration
  - Real Time Data Push
- **Service Adapters**
  - LiveCycle
  - SQL
  - Hibernate
  - ColdFusion
  - JMS
  - Java
  - Custom...
- Proxy Service
- Web-tier Compiler
- Portal Deployment
- RIA-PDF Generation



BlazeDS

- **Data Management**
  - Data Synchronization
  - Off-line Applications
  - Data Paging
- **RPC Services**
  - Web Service
  - HTTP Service
  - Remote Object Service
- **Messaging**
  - Publish & Subscribe
  - Collaboration
  - Real Time Data Push
- **Service Adapters**
  - LiveCycle
  - ColdFusion
  - Hibernate
  - SQL
  - JMS
  - Java
  - Custom...
- Proxy Service
- Web-tier Compiler
- Portal Deployment
- RIA-PDF Generation

LiveCycle DS Express **LiveCycle DS Community**

Israel
JBUG

Server side Platforms
WebOrb

- .Net
  - » Remoting, Data Management Messaging,RTMP,AMF
- Java
  - » Remoting, Data Management Messaging
- Rubi
  - » Remoting
- PHP
  - » Remoting

Server side Platforms Granite

- Stable (Production ready)
  - » AMF3
  - » Ejb3 services (session beans)
  - » Ejb3 persistence (Hibernate) with lazy-loading support
  - » Spring services with Acegi security
  - » Pojo services
- Experimental (beta)
  - » Data push (*Gravity*)
  - » Seam services
  - » Guice/Warp services

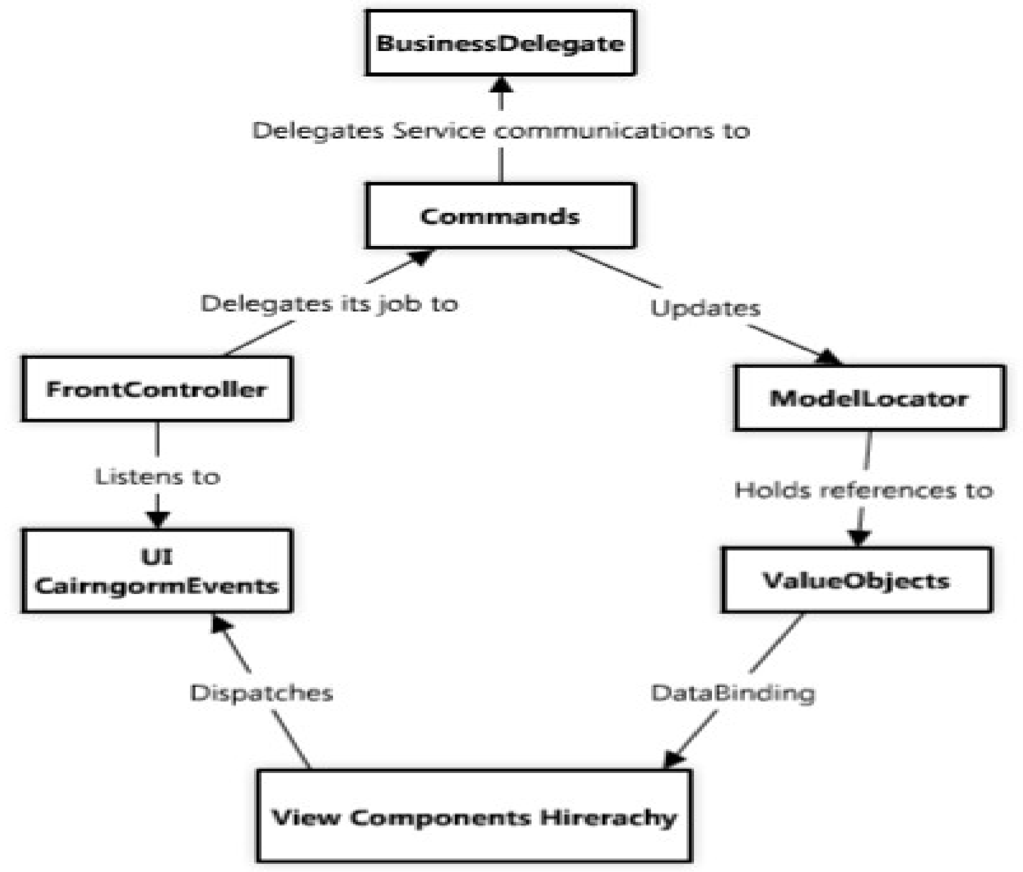
Israel JBUG

QA?

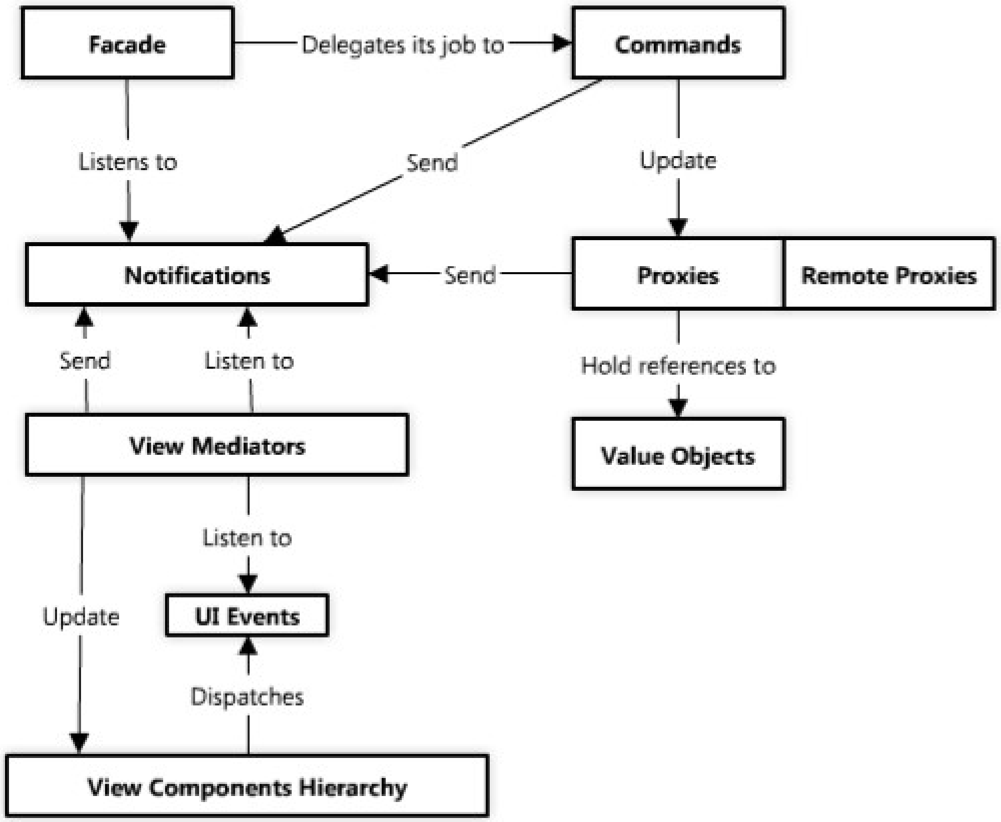
Israel JBUG

Appendix

Architecture
Client – Cairngorm Architecture



Architecture
Client – PureMVC Architecture



Flex frameworks Cairngorm

- Developed and used by Adobe Consulting
- Pros
  - defacto-standard for most organizations
  - Easy to learn

Flex frameworks Cairngorm

- **Cons**
  - » Extensive use of singletons
  - » Does not offer an elegant way for its controller to communicate back to its views(workaround exists)
  - » Problematic when using modules.

Flex frameworks
PureMVC

- No Flex dependency or awareness
- Pros
  - Cleaner separation of view and Vos
  - Considered "better"
- Cons
  - No ServiceLocator-like tool
  - No Model-View Data Binding
  - Requires more work
  - More complex