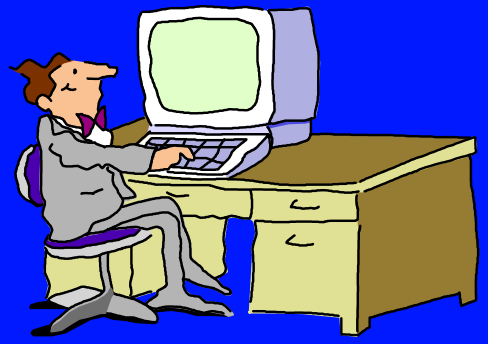
OpenSRF via Java

What we learned doing OpenSRF via Java

Alpha-G Consulting



- Data Migration
- System Administration
- Support
- Hosting
- Reporting/Data Transfer & Interchange
- Contact john@alphagconsulting.com
- Slides: http://alphagconsulting.com/EvgConf2011/

Content of Presentation

- Target Audience
  - Potential/Current Java OpenSRF programmers
- Focus is on OpenSRF interaction via Java
  - Hoping to help others get off to better start by providing some basic patterns
  - Outline of some work we'd like to do to make Java development simpler
- Presentation excludes adding to OpenSRF
  - Basic patterns for interacting w/ existing methods via Java

Caveats and Considerations

- There's a lot to know; we're still learning
- Our project works, but it is not:
  - A sterling example of best practices
  - Polished/complete
- Our plans include:
  - Creating Java classes to simplify future development
  - Rewriting this project
- Interested in Java? Get in touch, please.

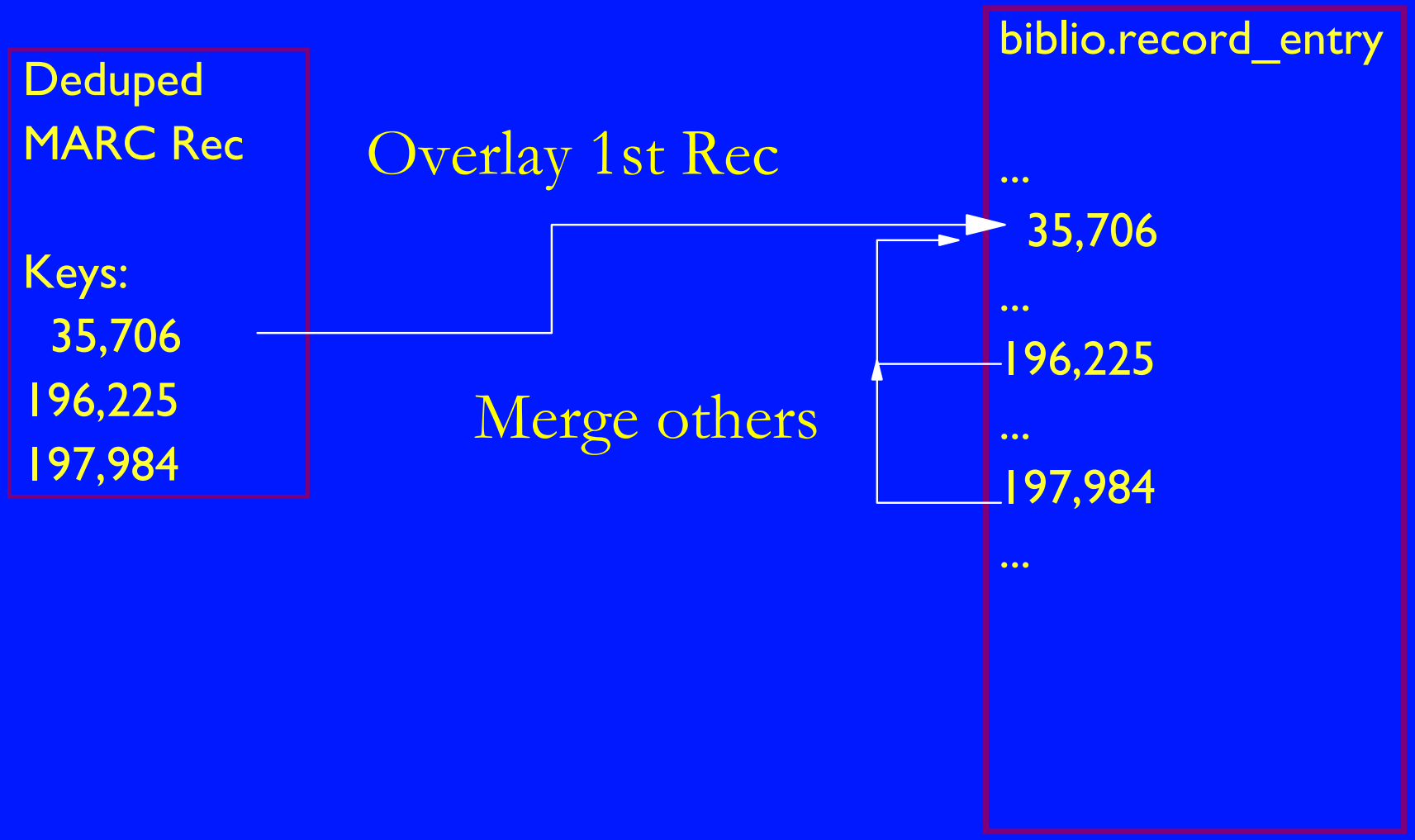
The Problem: Duplicate Bibs

- Georgia PINES Library System
- Each member library's bibs loaded
  - No de-duping done as libraries added
  - Many duplicate bib records
  - Hand de-duping impractical
- Client contracted with Backstage Library Works to have MARC file de-duped
- How to reload de-duped MARC file into Evergreen?

The Project: Overlay/Merge

- MARC file from Backstage Library Works
- Create Overlay/Merge Tool
- Overlay/Merge utility reads file:
  - Overlays first matched record
  - Merges additional matched records
  - Logs & continues on error

Overlay/Merge Tool



α-g Consulting

Why Java?

- Experience: Lots of it
  - Used with other library automation systems
  - Extensive library-automation tool set
- Write [with] What You Know
  - Familiar Conventions & Patterns
  - Familiar Tools
- Diving into OpenSRF
  - One thing at a time
  - Trying to keep it simple

What we Learned

- We're Still Learning
- Dealing with OpenSRF
  - Basic Patterns
  - Using the Tools as Intended
- Some Specific Issues
  - Version Incompatibilities
  - Direct Database access vs. OpenSRF
  - Overloading the Database Server
  - Evergreen Versions Again
  - License Complications

Cartoon Cavemen: Mechanic & Assistant

Mechanic

Working on stone wheel

"Hand me a crescent wrench."

"That's not a crescent wrench! It's a ball-peen hammer!"

[Looks down at rock in hand.]

"Oh, I don't know, maybe it is a crescent wrench...."

"Damn stone tools!"

Assistant

Has a box of rocks

Hands over a rock.

Peers at rocks in box.

Evergreen Java Tools

- When you don't know what the tools are for, they all look like rocks (whether they are or not).
- If your assistant does not know the difference between one tool and another, they'll likely pick the wrong one.
- Learning the tool set is the main challenge.

org.opensrf packages

- net.xmpp
  - Stripped-down XMPP (Jabber) communication
- test
  - Testing various aspects of OpenSRF interaction
- util
  - Config info
  - JSON manipulation
  - OSRF objects
  - Logging

org.open-ils packages

- idl
  - Handling of fm_IDL.xml object definitions
- test
  - TestLogin
  - TestIDL
- util
  - Logging in
  - MD5 hash calculation

3rd-Party Packages

- Woodstox: com.ctc.wstx
  - "High-performance" XML processor
  - http://woodstox.codehaus.org/Download
- JSON: org.json
  - JSON parsing & packing
  - https://github.com/douglascrockford/JSON-java
- Memcached: java_memcached
  - OpenSRF Caching interaction
  - http://www.docjar.com/jar_detail/java_memcached-release_2.0.1.jar.html

Learning to Use What's There

- Basic Understanding of the OpenSRF model
  Read Dan Scott's "Easing Gently into OpenSRF"
  http://journal.code4lib.org/articles/3284
- Double handful of Test/Example Programs
  - Start with LoginTest and ClientTest
  - Get the basic patterns clearly in mind
  - Extend them for simple prototypes
- We Started with Attempting to get an initialized ClientSession (logging in)

Connecting: org.open-ils.test.TestLogin

```
org.opensrf.Sys
    .bootstrapClient( args[0] // opensrf_core.xml
                    , "/config/opensrf");
Map<String,String> params
                = new HashMap<String,String>();
// Evg usrname/passwd (as for Staff Client login)
params.put( "username", args[1] );
params.put( "password", args[2] );
org.open_ils.Event evt
          = org.open_ils.util.Utils
                                .login(params);
```

Internals of Utils.login Method

```
Object resp
   = ClientSession.atomicRequest
   ( "open-ils.auth"
   , "open-ils.auth.authenticate.init"
   , new Object [] {init}
   );
... // a REALLY bad place for a breakpoint
resp = ClientSession.atomicRequest
      ( "open-ils.auth"
      , "open-ils.auth.authenticate.complete"
      , new Object[]{params}
      );
```

Found & Fixed Minor Bugs

- Xpath version differences
  - */domains/domain* should be */domain*
  - org.opensrf class sources
    - ClientSession.java & Sys.java
  - org.opensrf.util class source
    - Config.java
- Also in Config.java
  - Changed getInt method to assume get method returns an Integer for "/port" Xpath

Why Still Could Not Log In

Java Box's Config File

192.168.1.179
5222
opensrf
opensrfPasswd

Not in Config

Evergreen Host Config

private.localhost
5222
opensrf
opensrfPasswd

A Kludge to Get Logged In

- Change Java box's config to *private.localhost*
- Add an entry to the Java box's *hosts* file

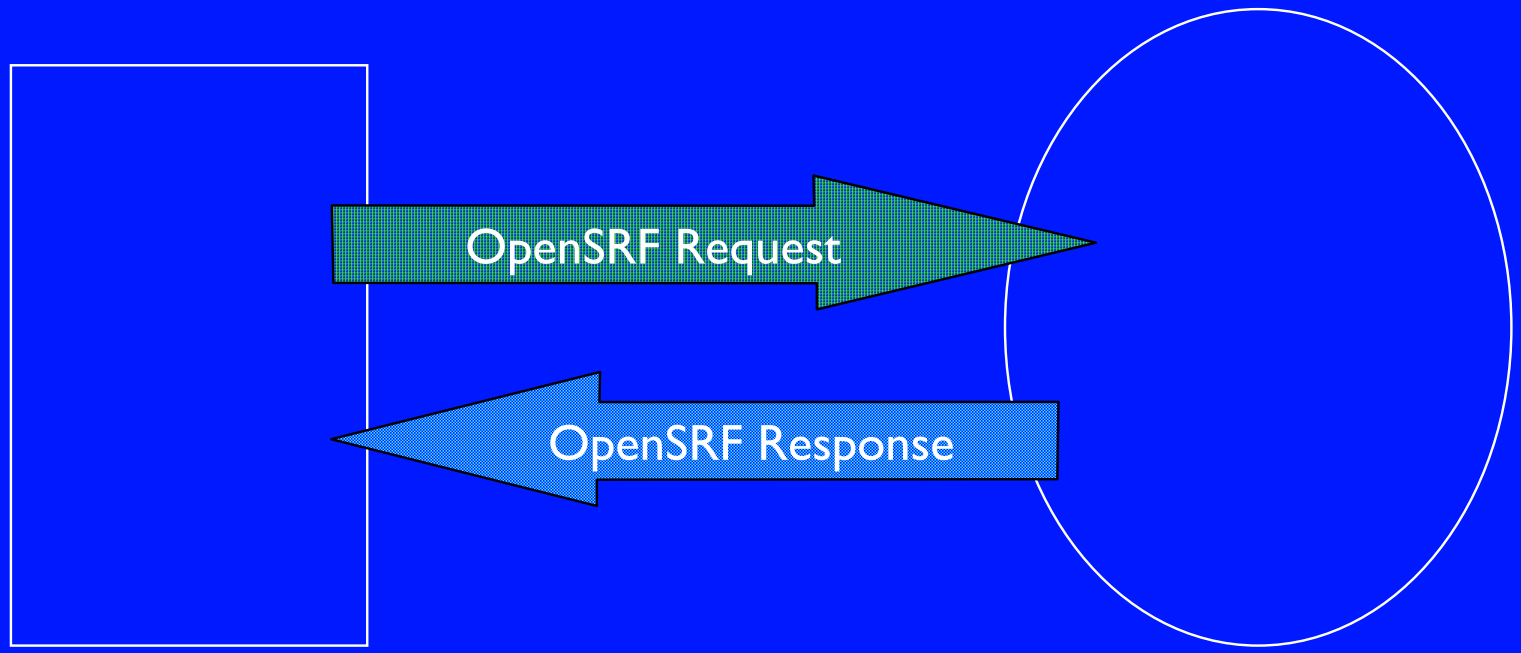
```
127.0.0.1          localhost
192.168.1.179      private.localhost
```

- Two facts of note:
  - It is inelegant, ugly, & poor practice; but
  - It does work.

Logged In *(finally)*: Now What?

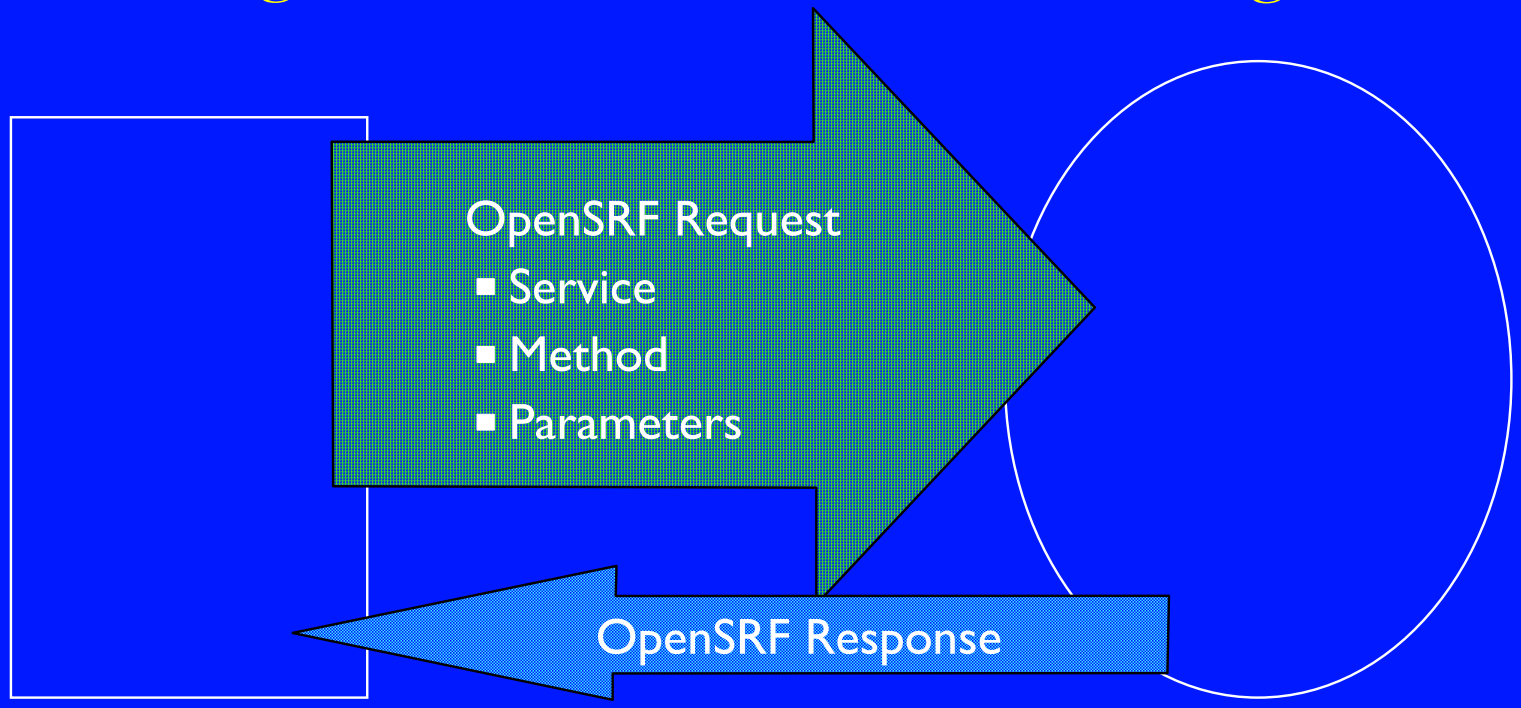
Java Program

OpenSRF/
Evergreen



α-g Consulting

OpenSRF Request



Central Issues: Calling Methods

- What is the method to do the job?
- Did you establish ClientSession to the method's service?
- How do you set up the parameters?
- How do you invoke the method?

What Method?

- Documentation is terse
- Used OpenSRF logging with Staff Client
  - Set OpenSRF to logging level 4
  - Edited and then saved bib record
- Asked for confirmation on #evergreen IRC channel
  - Interestingly, there was some discussion back and forth
- Method determines ClientSession's Service

Methods: Overlaying and Merging

- Overlay/Update biblio.record_entry
  - Service : open-ils.cat
  - Method: biblio.record.xml.update
  - Params : auth-token, record-id, record-in-XML
- Merge Records
  - Service : open-ils.cat
  - Method: biblio.records.merge
  - Params : auth-token, survivor-id, merge-ids

OpenSRF Method call in Java

```
String oSRFmethod
      = "open-ils.cat.biblio.record.xml.update";

List< Object > params = /* add params to List */;

try {
  org.opensrf.Request req
        = org.opensrf.ClientSession
                      .request( oSRFmethod
                              , params );

} catch ( org.opensrf.SessionException sessX ) {
  /* handle exception */

} // try / catch
```

Building Parameters

```
String xmlUpdateMethod
     = "open-ils.cat.biblio.record.xml.update";

List< Object > params = new ArrayList< Object >();
params.add( authKeyStr );
params.add( new Integer( recordID ) );
params.add( marcXMLStr );

try {
  org.opensrf.Request req
       = org.opensrf.ClientSession
                    .request( oSRFmethod
                            , params );

} catch ( org.opensrf.SessionException sessX ) {
...
```

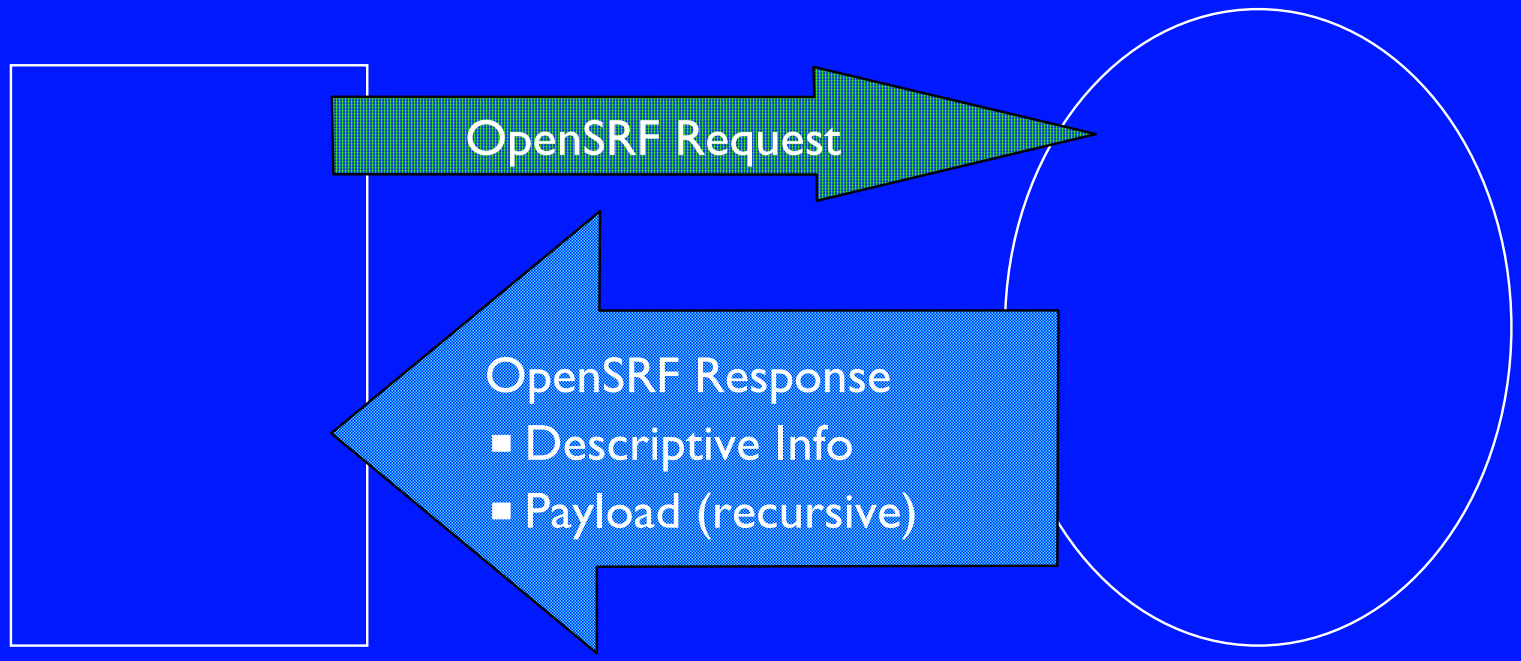
Parameter Notes

- Auth Token
  - Often required first parameter
- Pass Parameters in either:
  - List< Object >
  - Object[ ]
- JSON encoding handled by:
  - ClientSession.request method
- Parameter Order is Critical

OpenSRF Responses

Java Program

OpenSRF/
Evergreen



α-g Consulting

LoginTest's JSON Response

authtoken indicates successful authentication; required parameter for many methods you would use.

```
{ desc=Success
, payload={ authtoken=321c3593bb1cee5f81f8e4ce2d583e63
          , authtime=28800 }
, pid=7930
, stacktrace=oils_auth.c:444
, textcode=SUCCESS
, ilsevent=0
}
```

It worked!

A Typical Raw JSON Response


```
[{"__c":"osrfMessage"
 ,"__p":{"threadTrace":"0"
        ,"locale":"en-US"
        ,"type":"RESULT"
        ,"payload":{"__c":"osrfResult"
                   ,"__p":{"status":"OK"
                          ,"statusCode":"200"
                          ,"content":{"__c":"bre"
                                     ,"__p":[ ...
```


Name/Value Pairs: HashMap< String, Object >

Array Elements: Object[ ] or List< Object >

JSON Raw Response *(continued)*

α-g Consulting

Decoding an OpenSRF Response

- org.opensrf.*Response*
  - getter methods for *status*, *statusCode*, & *content*
  - *getStatusCode* returns HTTP-like values (e.g. 200 == OK)
  - *getContent* returns the payload's content
- org.open_ils.*Event*
  - Extends HashMap
  - Direct lookup
- Payload Objects must be in OSRFRegistry

Payloads and Object Registry

Payload

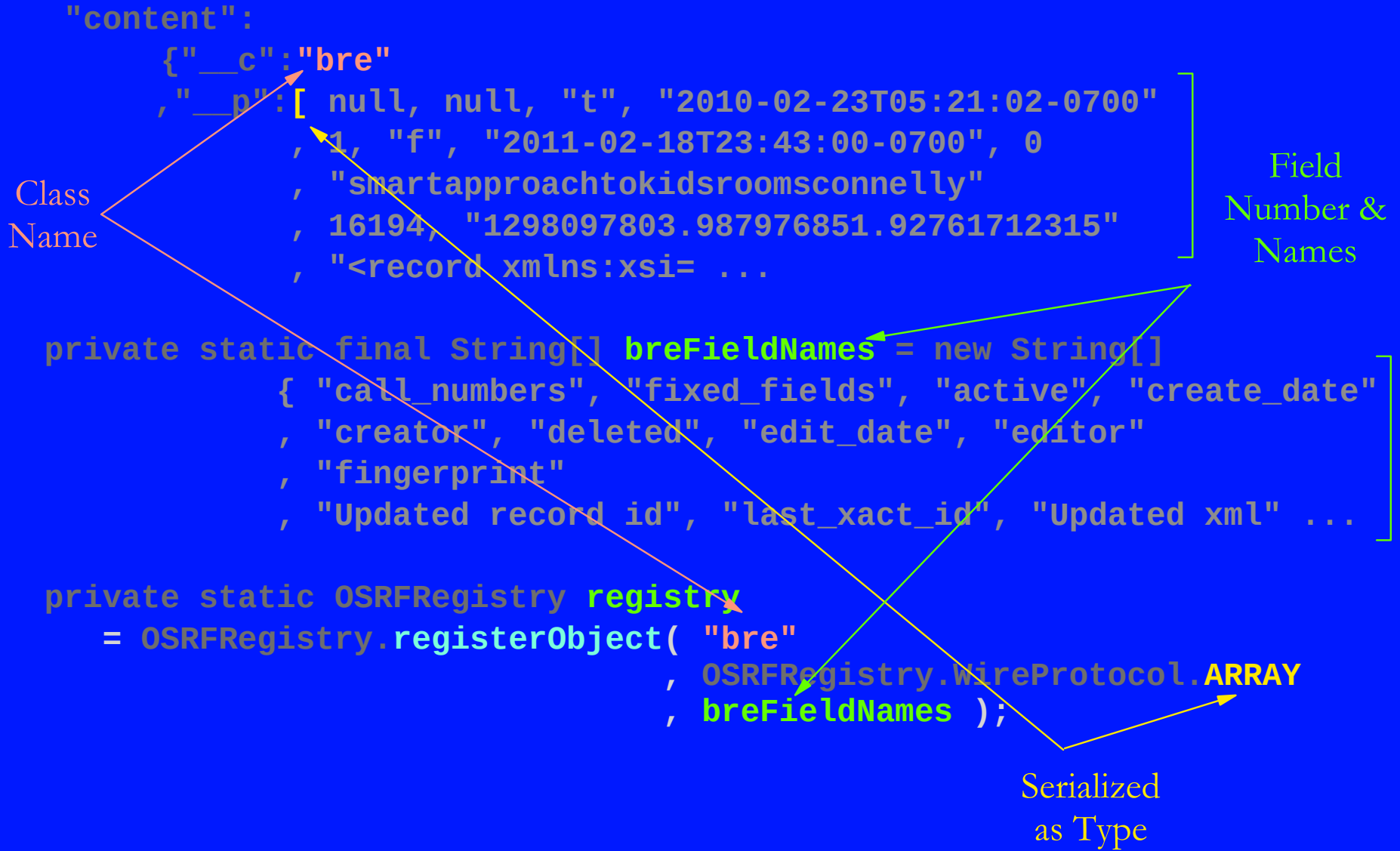
```
"content":
     {"__c":"bre"
     ,"__p":[ null, null, "t", "2010-02-23T05:21:02-0700"
            , 1, "f", "2011-02-18T23:43:00-0700", 0
            , "smartapproachtokidsroomsconnelly"
            , 16194, "1298097803.987976851.92761712315"
            , "<record xmlns:xsi= ...
```

Java

```
private static final String[] breFieldNames = new String[]
          { "call_numbers", "fixed_fields", "active", "create_date"
          , "creator", "deleted", "edit_date", "editor"
          , "fingerprint"
          , "Updated record id", "last_xact_id", "Updated xml" ...};

private static OSRFRegistry registry
   = OSRFRegistry.registerObject( "bre"
                                , OSRFRegistry.WireProtocol.ARRAY
                                , breFieldNames );
```

OSRFRegistry Correspondences



α-g Consulting

With Object in Registry

```
OSRFRegistry.registerObject( "bre"
                           , OSRFRegistry.WireProtocol.ARRAY
                           , breFieldNames );

// create instances of OSRFOjects with registered field names
org.opensrf.util.OSRFObject osrfBreObj = new OSRFObject( "bre" );
osrfBreObj.put( "deleted", "false" );
osrfBreObj.put( "creator", new Integer( 189 ) );

// Get data from response as OSRFObject
org.opensrf.Sys.bootstrapClient( "opensrf_core.xml"
                               , "/config/opensrf" );
String service = "open-ils.cstore";
String method  = "open-ils.cstore.direct.biblio.record_entry.retrieve";
org.opensrf.ClientSession session = new ClientSession( service );
org.opensrf.Request req = session.request( method, params );
org.opensrf.Result  res = req.recv( 10000 ); // waits up to 10 secs
org.opensrf.util.OSRFObject osrfBreObj = res.getContent();
Boolean isDeleted = new Boolean( (String)
                                 osrfBreObj.get( "deleted" ) );
```

OSRFRegistry from fm_IDL.xml

```
// Load the registry from the fm_IDL.xml file

org.open_ils.idl.IDLParser parser
                = new IDLParser( "fm_IDL.xml" );
parser.parse();

// create instances of any OSRFOject defined in fm_IDL.xml
org.opensrf.util.OSRFObject osrfUsrObj
                           = new OSRFObject( "au" );
osrfUsrObj.put( "alert_message"
              , "Email address is invalid." );
osrfUsrObj.put( "email"         , null );
```

At this point, you can call the appropriate method to update the actor.usr table with the data in osrfUsrObj.

Some Specific Issues

- Version Incompatibilities
- Direct Database Access vs. OpenSRF
- Evergreen Versions Again
- Overloading the Database Server
- License Issues

Version Incompatibilities

- Target of initial implementation was 1.6.x
- First occasion to use it was with 2.0.x
- What we did:
  - Hard-coded the fields for *bre* OSRFObject type
- What happened:
  - Program crashed/would not run with 2.0.x
- What should have done:
  - Should have loaded the definition from fm_IDL.xml

Direct DB Access vs. OpenSRF

- Have frequently used JDBC /w Evergreen
- Batch program
- Target version 1.6.x
- What we did:
  - Used JDBC to read record IDs
  - Used OpenSRF to update biblio.record_entry.marc
- What happened:
  - Worked just fine

Evergreen Versions Again

- MARC "ingest" processing differs radically
  - 1.6.x handles in Perl code (OSRF method)
  - 2.0.x handles at the database level
- What we did:
  - Just let the underlying layers handle it
- What happened:
  - 1.6.x worked okay
  - 2.0.x major DB thrashing; died repeatedly

Evergreen Versions Again *(continued)*

- What we should have done:
  - Change Session ConnectState?
  - Heart of the matter was apparently asynchronous database operations
- What we did do:
  - Added more CPUs to the VM running DB
  - Added delay parameter
  - Delay is based on how long prior operation took to dispatch and return result
  - Belatedly learned about config options

Overloading the DB Server

- What was happening:
  - OpenSRF Router dispatches each request to a (new) service instance
  - OpenSRF response returned while DB operations finish asynchronously
- Maybe next time:
  - Disable many/all indexing (ingest) functions
  - Design multi-threaded program
    - One (or more) threads update MARC XML
    - Other threads poll queues of indexing tasks

License Issues

- GPL v. 2 is unclear about use of GPL JARs in non-GPL programs
- GPL is generally considered a *viral* license
- According to conversations on IRC channel
  - Intent of committers is not to require GPL'ing all work that calls Java API
  - Such discussions are not, of course, binding
- Recommend explicit *Classpath Exception*

Some Final Thoughts

- Java & OpenSRF
  - Can play well together
  - We will definitely continue to use Java
- Version changes in Evg/OpenSRF can bite
- Going Forward
  - Rewrite program to take advantage of what we learned
  - Create additional Java programs
  - Anxious to collaborate with other Java users

Some Final Thoughts *(continued)*

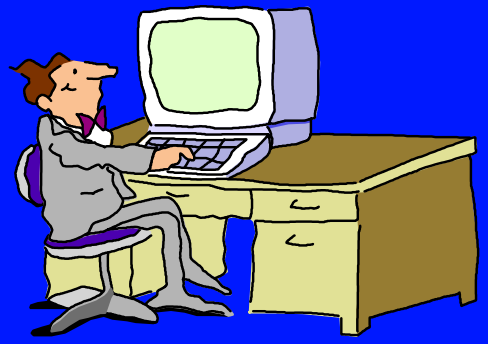
- Things to do differently, include:
  - Learn more about OpenSRF before coding.
  - Experiment more extensively.
  - Should have simplified logging (write processed record IDs to the DB instead of a log file).
- To add to the Java API:
  - Create classes for extensively used objects.
  - Build Java source for objects from fm_IDL.

Acknowledgements

- Bryan Kingsford, Alpha-G
  Did 95% of coding of initial version
- Dan Wells, Calvin College
  Resolved killer problem in 901-maintenance regex
- Everyone who answered our questions on the IRC channel or via email
- Backstage Library Works
- Georgia PINES Library System

Questions and Discussion

Alpha-G Consulting



- Data Migration
- System Administration
- Support
- Hosting
- Reporting/Data Transfer & Interchange
- Contact john@alphagconsulting.com
- Slides: http://alphagconsulting.com/EvgConf2011/