Today's Objectives

- Remote Procedure Calls
- Remote Method Invocation
- Naming intro

Review: Web Server

- How does a web server work?
- What was easy? What was hard? What are your takeaways?

Review

- What is RPC?
  - What is its purpose?
  - How does it work?
  - What implementation of RPC are we going to use?

Case Study: XML-RPC

- XML-RPC is an RPC protocol
  - XML to encode its calls
  - HTTP as transport mechanism
- Apache's Java implementation
  - `http://ws.apache.org/xmlrpc`

Answer to Why Objects Instead of Primitives

```
public Integer[] sumAndDifference(int x, int y)
```

```
org.apache.xmlrpc.XmlRpcException: Failed to create input stream: Unexpected end of file from server
        at org.apache.xmlrpc.client.XmlRpcSunHttpTransport.getInputStream(XmlRpcSunHttpTransport.java:99)
        at org.apache.xmlrpc.client.XmlRpcStreamTransport.sendRequest(XmlRpcStreamTransport.java:152)
        at org.apache.xmlrpc.client.XmlRpcHttpTransport.sendRequest(XmlRpcHttpTransport.java:143)
        at org.apache.xmlrpc.client.XmlRpcSunHttpTransport.sendRequest(XmlRpcSunHttpTransport.java:69)
        at org.apache.xmlrpc.client.XmlRpcClientWorker.execute(XmlRpcClientWorker.java:56)
        at org.apache.xmlrpc.client.XmlRpcClient.execute(XmlRpcClient.java:167)
        at org.apache.xmlrpc.client.XmlRpcClient.execute(XmlRpcClient.java:137)
        at org.apache.xmlrpc.client.XmlRpcClient.execute(XmlRpcClient.java:126)
        at examples.Client.mainClient.java:43)
Caused by: java.net.SocketException: Unexpected end of file from server
        at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:793)
        at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:652)
        at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:790)
        at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:652)
        at sun.net.www.protocol.http.HttpURLConnection.getInputStream(HttpURLConnection.java:1218)
        at java.net.HttpURLConnection.getResponseCode(HttpURLConnection.java:379)
        at org.apache.xmlrpc.client.XmlRpcSunHttpTransport.getInputStream(XmlRpcSunHttpTransport.java:92)
        ... 8 more
Caused by: java.net.SocketException: Unexpected end of file from server
        at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:793)
        at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:652)
        at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:790)
        at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:652)
        at sun.net.www.protocol.http.HttpURLConnection.getInputStream(HttpURLConnection.java:1218)
        at java.net.HttpURLConnection.getResponseCode(HttpURLConnection.java:379)
        at org.apache.xmlrpc.client.XmlRpcSunHttpTransport.getInputStream(XmlRpcSunHttpTransport.java:92)
        at org.apache.xmlrpc.client.XmlRpcStreamTransport.sendRequest(XmlRpcStreamTransport.java:152)
        at org.apache.xmlrpc.client.XmlRpcHttpTransport.sendRequest(XmlRpcHttpTransport.java:143)
        at org.apache.xmlrpc.client.XmlRpcSunHttpTransport.sendRequest(XmlRpcSunHttpTransport.java:69)
        at org.apache.xmlrpc.client.XmlRpcClientWorker.execute(XmlRpcClientWorker.java:56)
        at org.apache.xmlrpc.client.XmlRpcClient.execute(XmlRpcClient.java:167)
        at org.apache.xmlrpc.client.XmlRpcClient.execute(XmlRpcClient.java:137)
        at org.apache.xmlrpc.client.XmlRpcClient.execute(XmlRpcClient.java:126)
        at examples.Client.mainClient.java:43)
```


PropertyHandlerMapping

- A handler mapping based on a property file
- The property file contains a set of properties
  - The property *key* is the handler name
  - The property *value* is the name of a class being instantiated
- For every non-void, non-static, and public method in the class, an entry in the handler map is generated
- A typical use would be to specify interface names as the property keys and implementations as the values

Case Study: XML-RPC

- XML is a standard for describing structured documents
  - Uses tags to define structure:
    - <tag> … </tag> demarcates an element
    - Tags have no predefined semantics …
  - Elements can have attributes
    - Encoded as name-value pairs
    - A well-formed XML document corresponds to an element tree

```
<?xml version="1.0"?>
<methodCall>
    <methodName>SumAndDifference</methodName>
      <params>
          <param><value><i4>40</i4></value></param>
          <param><value><i4>10</i4></value></param>
      </params>
</methodCall>
```


(thanks to Vijay Karamcheti)

XML-RPC Response

- HTTP Response
  - Lower-level error returned as an HTTP error code
  - Application-level errors returned as a <fault> element (next slide)

```
HTTP/1.1 200 OK
Date: Mon, 22 Sep 2003 21:52:34 GMT
Server: Microsoft-IIS/6.0
Content-Type: text/xml
Content-Length: 467

<?xml version="1.0"?>
<methodResponse>
  <params><param>
  <value><struct>
   <member><name>sum</name><value><i4>50</i4></value>
   </member>
   <member><name>diff</name><value><i4>30</i4></value>
   </member>
  </struct></value>
  </param></params>
</methodResponse>
```

XML-RPC Fault Handling

- Another kind of MethodResponse

```
<?xml version="1.0"?>
<methodResponse>
<fault>
<value><struct>
<member>
<name>faultCode</name>
<value><i4>500</i4></value>
</member>
<member>
<name>faultString</name>
<value><string>Arg 'a' out of range</string></value>
</member>
</struct></value>
</fault>
</methodResponse>
```

Added example of exceptions in repo.

Asynchronous RPCs

- executeAsync()
- org.apache.xmlrpc.AsyncCallback
  - handleResult
  - handleError

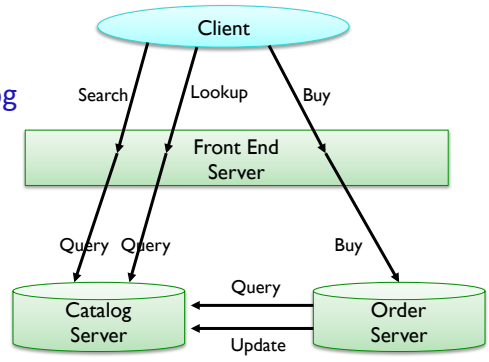
Thrift: RPC Framework

- "The Apache Thrift software framework, for scalable cross-language services development, combines a software stack with a code generation engine to build services that work efficiently and seamlessly between C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, JavaScript, Node.js, Smalltalk, OCaml and Delphi and other languages."

- Originally developed by Facebook
- Open-source on Apache in 2007
- http://thrift.apache.org/

Project 2 Overview: Tiny Bookstore

- Implement 3 servers and a client
- All communicate using XML-RPC
  - Apache XMLRPC for Java
  - Client → FrontEnd
  - FrontEnd → Order/Catalog
  - Order → Catalog
- Concurrent queries
- One purchase at a time
- Create a Python client



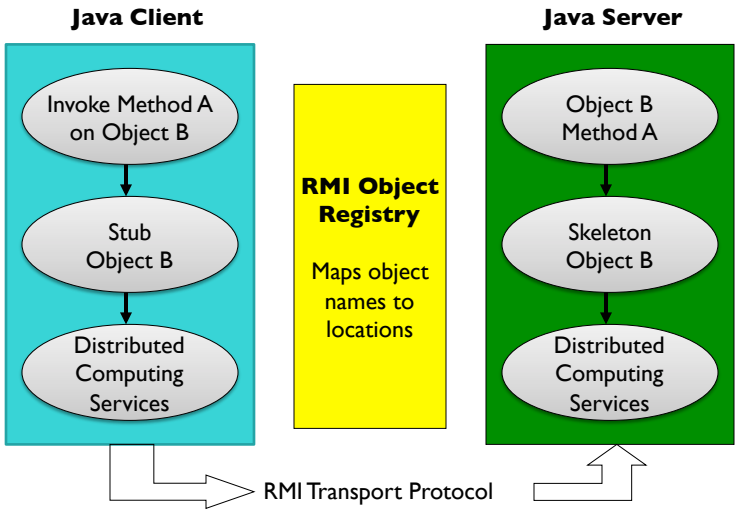
Form teams and check out project by Wednesday

REMOTE METHOD INVOCATION

Remote Method Invocation (RMI)

- Popular variant on RPC
- Obtain handle to remote object, invoke method on object
- Transparency goal: remote object appears as local object

Case Study: Java RMI



(thanks to David Del Vecchio)

NAMING

Introduction to Naming

- Naming is a fundamental issue that is often overlooked in distributed system design
- Names refer to a variety of resources
  - Computers, services, files, remote objects, people (email addresses), etc
- Names are required to locate desired data
  - For example, consider the hostname in a URL

What are some goals with naming?
What are some challenges?

Name Services

- Name services store collections of *naming contexts*
- Main operation is to support *name resolution*
  - Look up attributes from a given name
- Also need to support creating new ***bindings***, deleting bindings, and listing bound names
- Goals:
  - Unification - resources managed by different services use same naming scheme
  - Integration - enable sharing and naming of resources across administrative domains

Example RMI Code

Java RMI vs. XMLRPC

- Java RMI is arguably simpler
  - Programs look a bit more "normal"
  - Can serialize (by implementing `Serializable`) and return different objects
- XMLRPC is more *flexible*
  - Can interact (easily) with other XMLRPC clients written in different languages
  - But procedure calls are somewhat limiting
  - Difficult to send non-standard objects

Take-Away Messages?

- What conclusions are you drawing about RPC and RMI?

RPC Take-Away Messages

- RPCs handle a lot automatically
  - Marshaling and unmarshaling of data
  - Threading, message formatting, socket creation, etc.
- Designed to simplify network programming through familiar programmatic abstractions
  - (You should appreciate all of this after building your web servers!)
- Variations
  - RMI, SOAP
- Build on top of RPC: e.g., NSF, some Web Services

My Very First Paper

- "Investigating JavaRMI for a Computer Science Curriculum." Proceedings of the 14th annual Eastern Small Colleges Computing Conference (ESCCC), Marist, NY, October 1998.

Looking Ahead

- Work periods Wed and Fri
  - Project
  - Article on Perusall