Advanced Software Engineering

Homework #3

Patrick O’Brien

EJB:

For testing and development I used Spring Beans with jaxws spring implementation (1.9) and CXF for the client (and wsdl2java) (2.7.6). Please see ivy.xml for more specifics. There were no other dependencies like corba or ejb in the last two assignments.

DEVELOPMENT ENVIORNMENT:

I am using Eclipse IDE Juno Service Release 2. Dependency management is being handled by IVY.

SYSTEM ARCHITECTURE/DEPENDENCIES

The majority of the code from assignment #1 was ported over with no changes. The only change was to the wrapper classes. Instead of using corba, webservices were implemented as the communication layer of choice. The initial auction house code was designed that way to allow quick swapping of the communications layer.

I used the jaxws spring libraries to implement the web service piece. I believe the underlying implementation for that is the jetty server. I also used the standard javax annotations to designate my webservice class and the methods as well as the data classes.

I used the cxf libraries for the client code as well as the wsdl2java generation. In this assignment I only ended up using one generated class in the client implementation: Auction. The other calls were so simple that using standard java classed was sufficient. In a production system wsdl2java would be used in an ant or maven script to generate the client class libraries.

One thing of note is that both the customer and the auction house are both webservices. The customer takes in a port number so as to allow multiple customers on the same machine. The customer passes in its ws url to the auction house upon registering. When an auction is finished the auction house calls that customers ws to inform them they have won that auction. The other way this is sometimes done is via continual polling of the server by the client. Esp in cases where you cannot use a ws on the client side.

I am using the built in java TreeMap data structures, this is due to no requirements for persistence another data source can easily be plugged in, including a database/cache, no-sql or map-reduce implementation. All variables accessible via multi-threading are using the java concurrency package.

Logging is handled via slf and logback. The logback.xml file included in the project is very standard for a basic setup to write to the console.

Initialization of the auction house is done via xtreme library; parsing a json file. Invalid data is included as a testing of the loading software validation to ensure that the auction house does not go into an invalid state. I. E.. Auctions for items that do not exist.

PROBLEMS:

I had no problems setting this up. This took very little time to complete because I had abstracted out the communication layer in the first assignment.