Web Service (SOAP) Methodology (suggested) – Where to Start?

Existing Java Class that performs the function of the service = Bottom Up

Bottom up:
XML: + Add annotations by hand for JAXB – but if you have or can create an XSD – xjc -> annotated Java classes
Deploy - > WLS generate assets - XSD & WSDL & Annotated Java Classes (JAXB to Marshall – Unmarshsall)
Exposed an existing Java class as WS
All methods by default are exposed (manually choose what to un-expose)
- Tight coupling between the service interface the implementation – changes in the method signature – result in change of interface = “Brittle”

No existing Java Classes - Start with WSDL and XSD = Top Down

Create XSD(s) – tools and editors – or given to you
Run xjc (JAXB) to generate annotated Java Classes
Create WSDL – by hand, tool, or wsgen (from Java source)
JAXB to generate annotated Java classes

Creating a Web Service:
Requirements:

- Customer Databased-WS:
  - CRUD
- WS Interface SOAP/WSDL or RESTFul?
- WS Implementation – Java

- What do you have to start with?

XML

JSON

XML Schema
WSDL
Top Down
Use JAXB xjc
to gen annotated Java Classes:

Existing Java Impl
POJO, EJB,
+ Annotate w/ @WebService

+ input args
+ return types
+ Service Interface and class to impl service

DB – Where, tech used?
Customer java
getCust
newCust
updateCust
deleteCust

Cust ID

Server to Run On?
Java Impl.

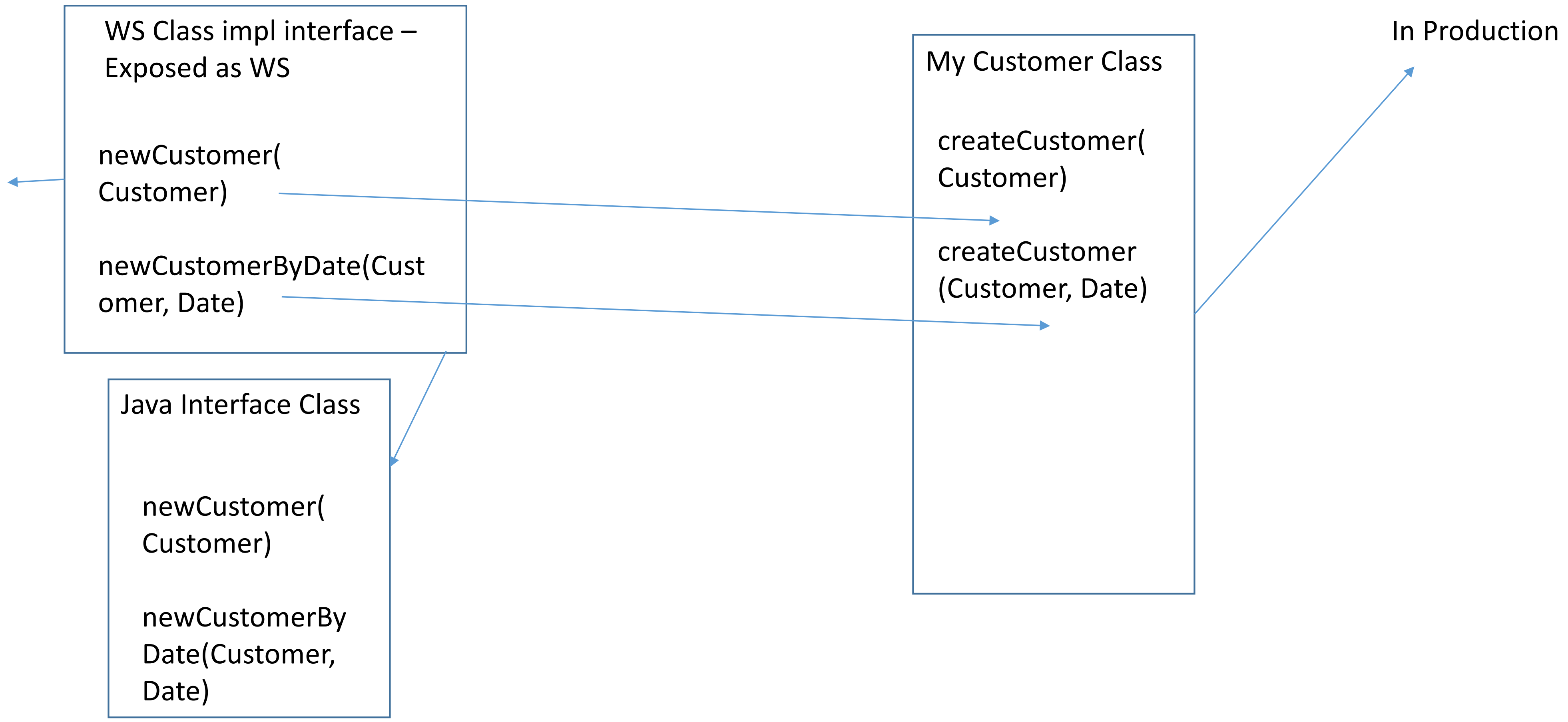
XML Schema

Need a Client Tester
SOAPUI

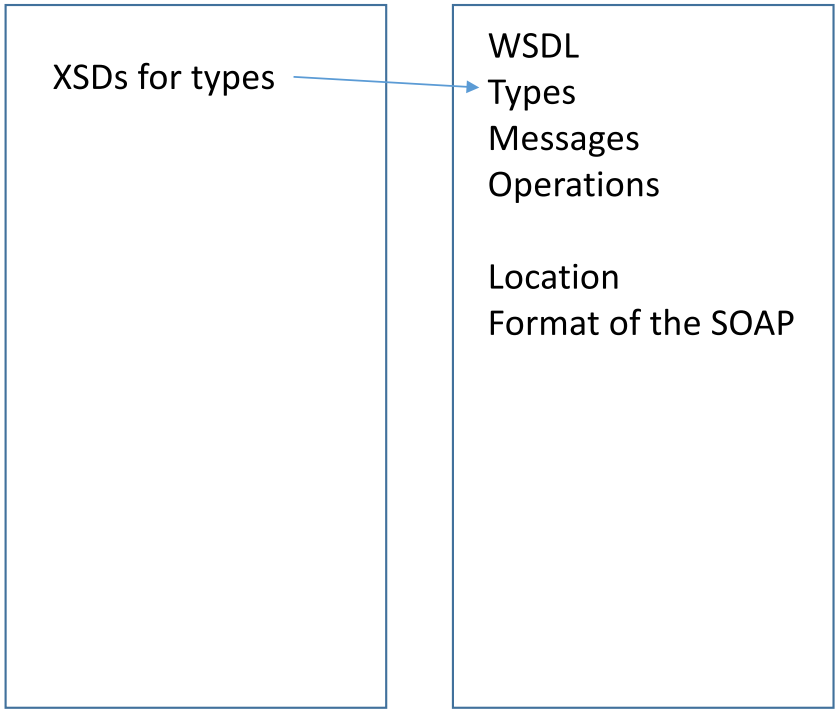
Sample Java Class Exposed as WS

```
public class Customer {

public String createCustomer(Customer aCust) {
        createCustomerByDate(aCust, currentDate();
}

public String createCustomerByDate(Customer aCust, Date aDate){

}
```

Encapsulate WS Interface for Existing Java Class


WS Class impl interface – Exposed as WS
newCustomer(Customer)
newCustomerByDate(Customer, Date)
Java Interface Class
newCustomer(Customer)
newCustomerByDate(Customer, Date)
My Customer Class
createCustomer(Customer)
createCustomer (Customer, Date)
In Production

Start with XSD and WSDL



Design for REST

1. Determine objects (Customer, Order, Item…) to be part of the RESTful service (these will appear as resources)
2. Define URIs - Resources, Parameters
3. Define media types supported by operations & resources = Data format
4. Choose HTTP operations your service will support