CONCORDIA UNIVERSITY 2015 SUMMER

COMP 6231 Distributed System Design
PM 4 Report

Supply Chain System (SC) using JAX-WS

Project 2 - Software Failure Tolerant and highly available System

Team 5

Members

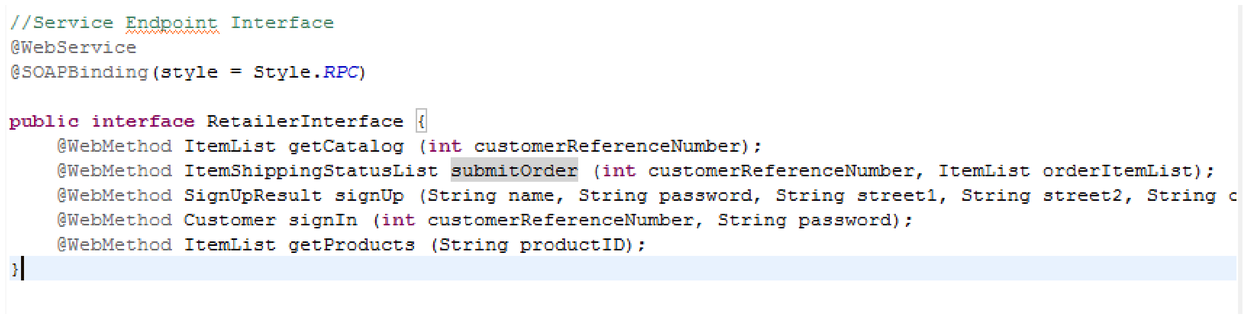
Wanhui Yao - 6422497
Ting Zhang - 6437117
Manne Mahesh Reddy - 27169019
Saravanan Iyyaswamy Srinivasan - 27090838

Contents

1 Outline of the project:

Programming assignment 4 is built upon PM3. In PM3, we did three major modifications.

1) Extracted the Java client-server implementation by abstracting away the CORBA-specific code from PM2.
2) Used Java API for XML Web Services (JAX-WS) technology to build remote Procedure Call-oriented (RPC-oriented) web services.
3) New classed like ItemList, ItemShippingStatusList,ProductList are created as the return types and passing parameters in some of the web service methods. For example, as showed below, the RetailerInterface. The return type of getCatalog() method is ItemList.



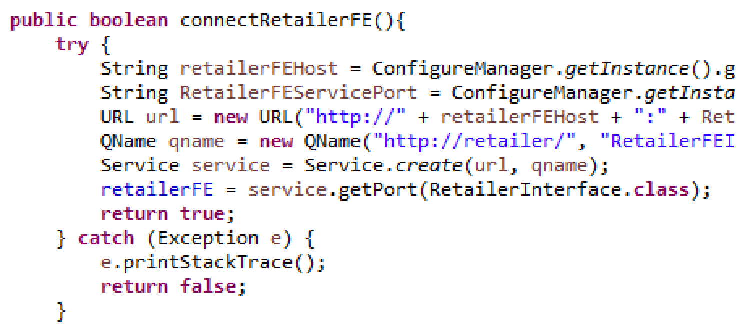
```
//Service Endpoint Interface
@WebService
@SOAPBinding(style = Style.RPC)

public interface RetailerInterface {
    @WebMethod ItemList getCatalog (int customerReferenceNumber);
    @WebMethod ItemShippingStatusList submitOrder (int customerReferenceNumber, ItemList orderItemList);
    @WebMethod SignUpResult signUp (String name, String password, String street1, String street2, String c
    @WebMethod Customer signIn (int customerReferenceNumber, String password);
    @WebMethod ItemList getProducts (String productID);
}
```

Following the changes of the pm3 assignment, project in PM4 is been improvised to support high availability and software failure using the below details.

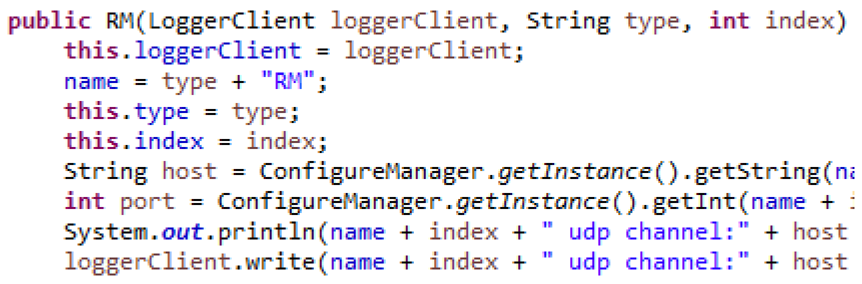
1) Designing the new system using active replication scheme where each server will act as a server replica . And will handle all the client requests using group replication and reliable communication using the front end, sequencer, replica manager and the replica's.
2) Here each replica will receive the client's request with a unique sequence number from the sequencer and there by executes it in a total ordering scheme. Then the result is sent back to the front end of the server.

3) Front end is responsible for receiving the client request and it is the one which forwards it to the sequencer for processing. It also receives the processed information from the replica's and sends it back to the client.



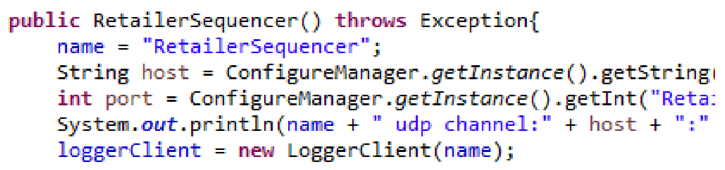
```
public boolean connectRetailerFE(){
    try {
        String retailerFEHost = ConfigureManager.getInstance().g
        String RetailerFEServicePort = ConfigureManager.getInsta
        URL url = new URL("http://" + retailerFEHost + ":" + Ret
        QName qname = new QName("http://retailer/", "RetailerFEI
        Service service = Service.create(url, qname);
        retailerFE = service.getPort(RetailerInterface.class);
        return true;
    } catch (Exception e) {
        e.printStackTrace();
        return false;
    }
```

4) Replica manager is used to maintain the replica's by creating and removing them when needed. It is also responsible for failure detection and recovery of replica's during failure crash or byzantine crash.



```
public RM(LoggerClient loggerClient, String type, int index)
    this.loggerClient = loggerClient;
    name = type + "RM";
    this.type = type;
    this.index = index;
    String host = ConfigureManager.getInstance().getString(na
    int port = ConfigureManager.getInstance().getInt(name + 
    System.out.println(name + index + " udp channel:" + host
    loggerClient.write(name + index + " udp channel:" + host
```

5) Sequencer is used to receive the client request from the FE and assigns a unique sequencer no to it before multicasting it to all the replica's for processing.

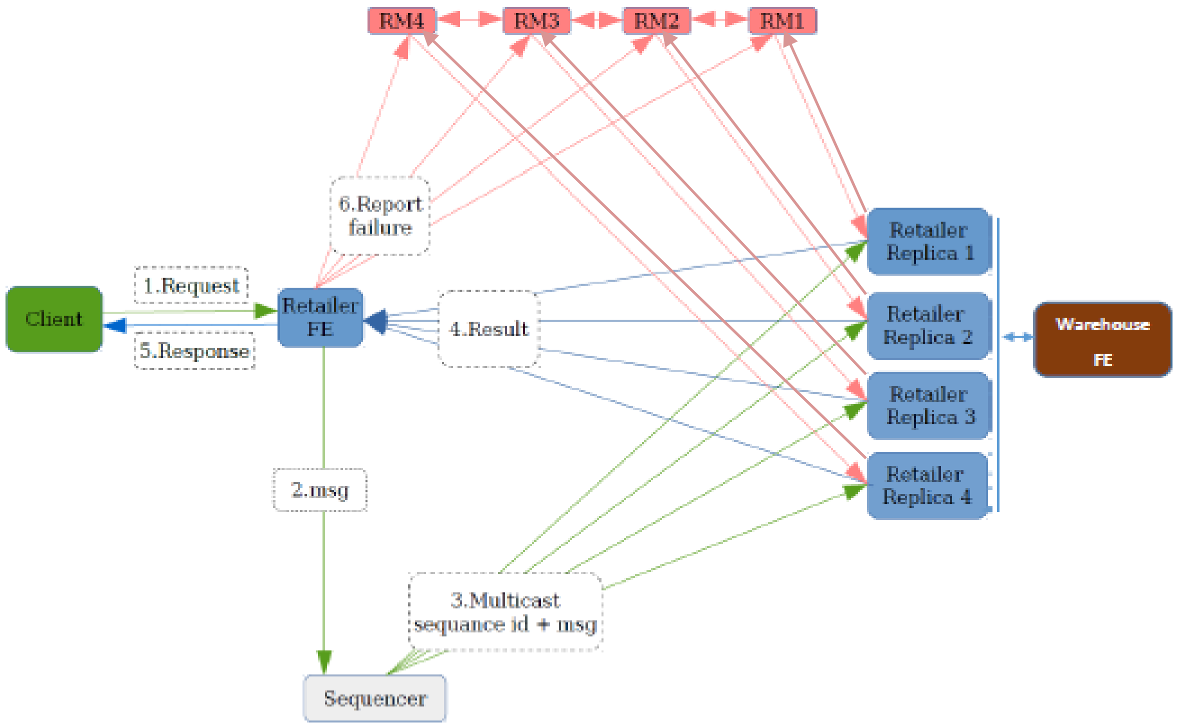


```
public RetailerSequencer() throws Exception{
    name = "RetailerSequencer";
    String host = ConfigureManager.getInstance().getString
    int port = ConfigureManager.getInstance().getInt("Reta
    System.out.println(name + " udp channel:" + host + ":"
    loggerClient = new LoggerClient(name);
```

2 Architecture

Design of the system is developed as below in 3 phases.

1) **Client to Retailer**

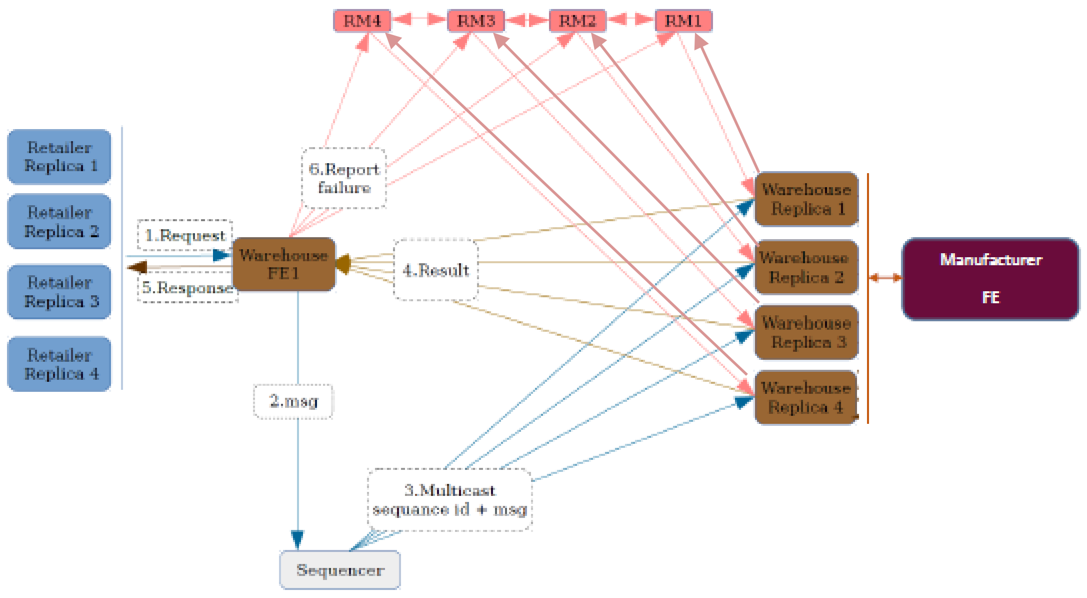


Description :

- Client sends requests to retailer FRONT END by RPC, retailer FRONT END send back responses.
- Retailer FRONT END send the message to sequencer through UDP (must be guaranteed to arrive to sequencer).
- Sequencer multicasts the message from retailer FRONT END plus a unique sequence id to all the retailer replicas.
- Once results are delivered from retailer replicas to retailer FRONT END, it will take the major result and responds to client.
- If some failures happen, the retailer FRONT END will report them the replica managers.

- The replica manager communicates with each other by heartbeat to make decision when retailer replica is not valid and to re launch the replica.
- The replica managers also receive failure reports from the retailer's FRONT END.

2) Retailer to Warehouses

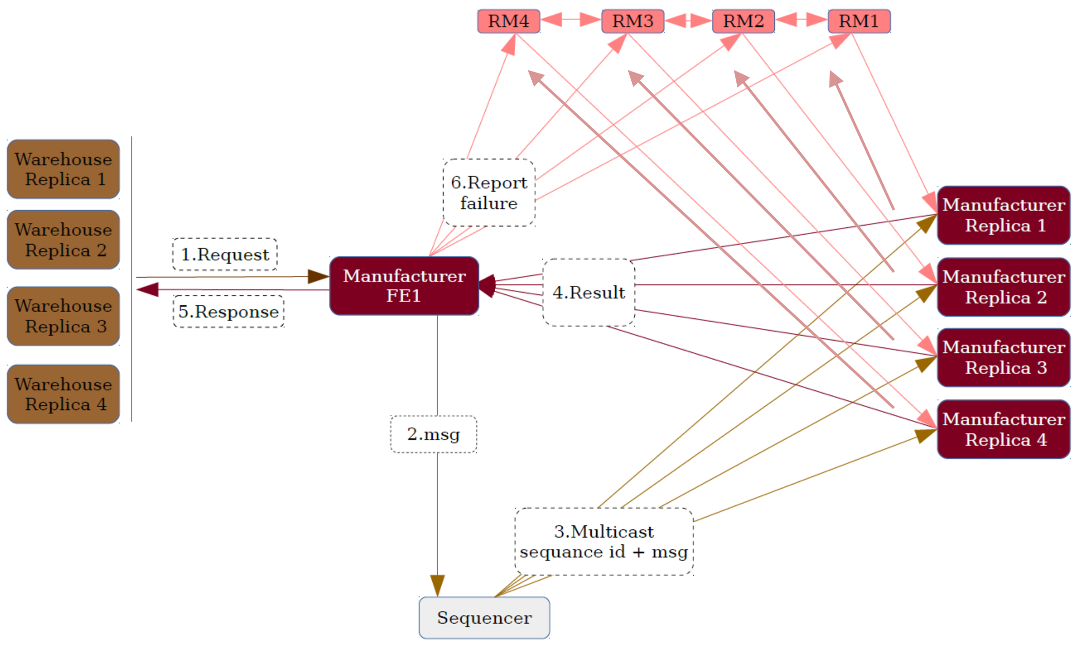


Description :

- Retailer Replicas send requests to warehouse FRONT END by RPC, warehouse FRONT END sends back responses.
- Warehouse FRONT END sends the message to sequencer through UDP (must be guaranteed to arrive to sequencer).
- Sequencer multicasts the message from warehouse FRONT END plus a unique sequence id to all the warehouse replicas.
- Once results are delivered from warehouse replicas to warehouse FRONT END, it will take the major result and responds to retailer replica.

- If some failures happen, the warehouse FRONT END will report them to the warehouse managers.
- The warehouse managers communicate with each other by heartbeat to make decision when warehouse replica is not valid and to relaunch the replica.
- The replica managers also receive failure reports from the warehouse FRONT END.

3) **Warehouse to Manufacturers**



Description :

- Warehouse Replicas send requests to manufacturer FRONT END by RPC, manufacturer FRONT END sends back responses.
- Manufacturer FRONT END sends the message to sequencer through UDP (must be guaranteed to arrive to sequencer).
- Sequencer multicasts the message from manufacturer FRONT END plus a unique sequence id to all the manufacturer replicas.
- Once results are delivered from manufacturer replicas to manufacturer FRONT END, it will take the major result and responds to warehouse replica.
- If some failures happen, the manufacturer FRONT END will report them to the manufacturer managers.
- The manufacturer managers communicate with each other by heartbeat to make decision when manufacturer replica is not valid and to relaunch the replica.
- The replica managers also receive failure reports from the manufacturer FRONT END.

2.1 Directory structure

The DSCS system is organized as below modules:

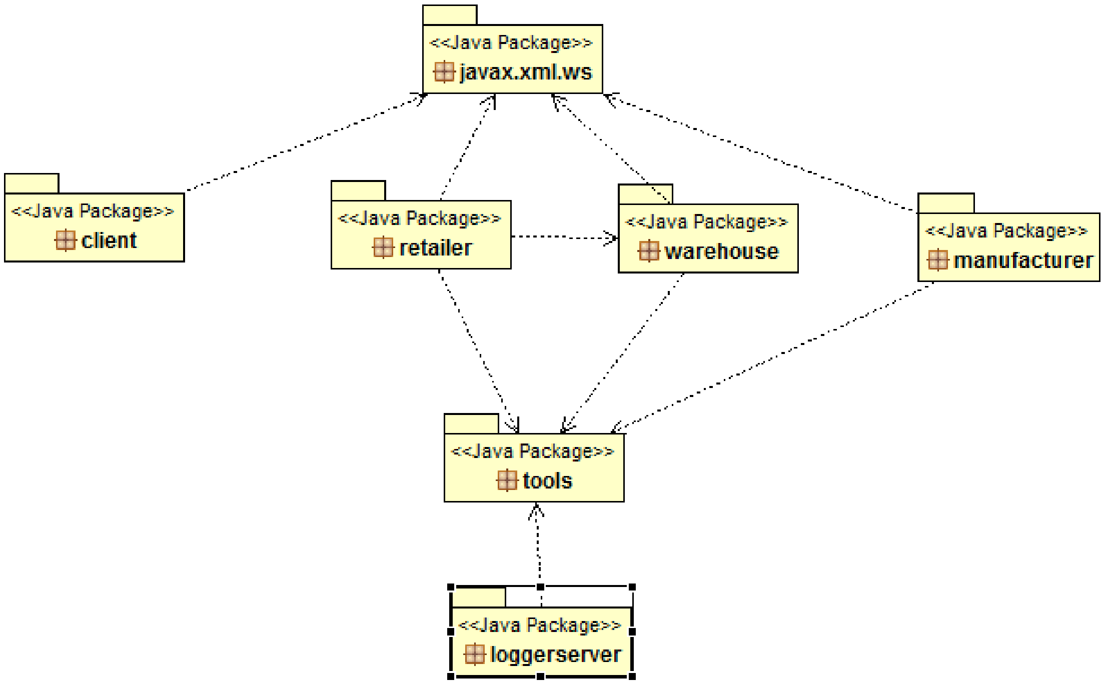
NO.	**Module**	**Classes**
1	Client	Client.java : client side App
2	Retailer	RetailerInterface.java : server endpoint interface RetailerImpl.java : implementation class RetailerServer.java : the App of retailer CustomerManager.java : help method to manage customer
3.	Warehouse	InventoryManager.java WarehouseFE.java WarehouseFEImpl.java WarehouseFEMessageProcesser.java

		WarehouseInterface.java WarehouseReplica.java WarehouseReplicaMessageProcesser.java
4.	Manufacturer	ManufacturerFE.java ManufacturerFEImpl.java ManufacturerFEMessageProcesser.java ManufacturerInterface.java ManufacturerReplica.java ManufacturerReplicaMessageProcesser.java PurchaseOrderManager.java
5.	RM	ReplicaStatus.java RM.java RMMessageProcesser.java
6.	LoggerServer	LoggerServer.java LogerWriter.java Connections.java
7.	tools	ConfigurerManager.java LoggerClient.java Customer.java Item.java ItemList.java Product.java ProductList.java (And other common classes shared by the above modules above)
8.	Channel	Channel.java ReadThread.java WriteThread.java NetworkIO.java ChannelManager.java ReplicaChannelManager.java (And other common classes shared by the above modules above)
9.	Message	AckMessage.java Action.java Message.java Packet.java ReplicaResultMessage.java

		HeartBeatMessage.java RMSyncMessage.java (And other common classes shared by the above modules above)

2.2 Basic architecture

UML graphs:



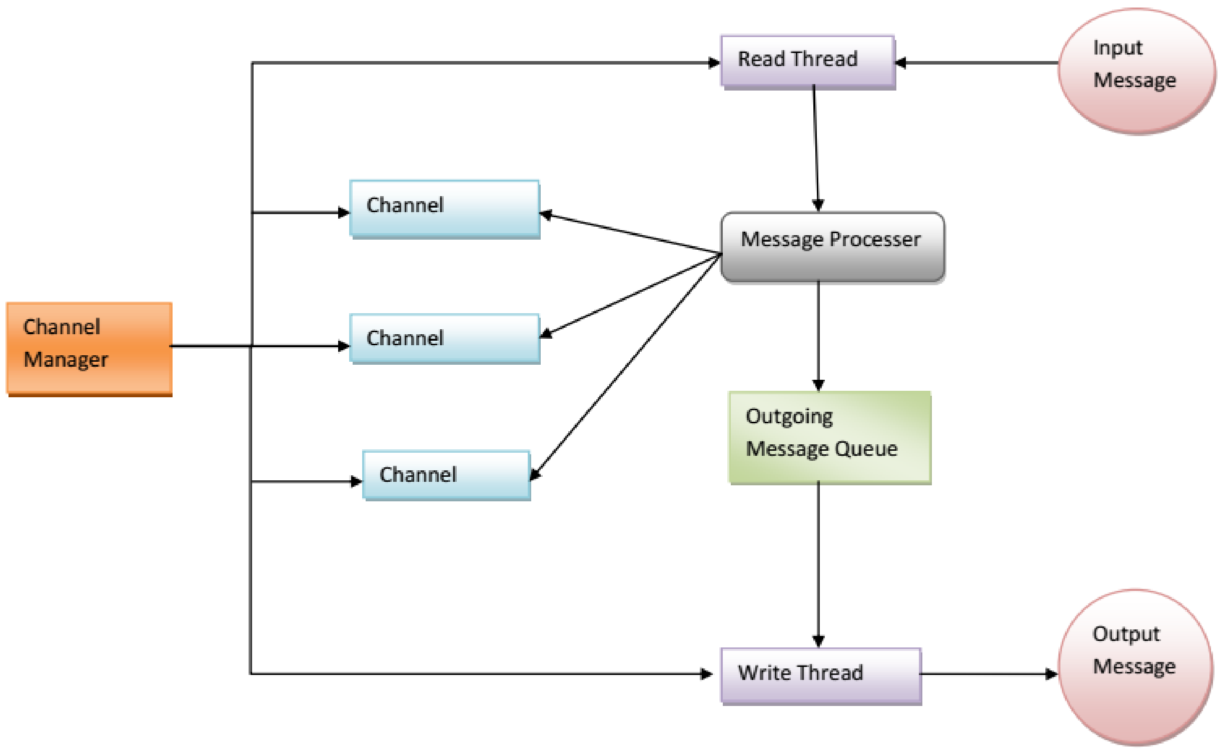
This system uses 3 JAX_WS architectures, Client-Retailer, Retailer-Warehouse and Warehouse-Manufacturer.

A service endpoint that declares the methods that a client can invoke on the service and the corresponding implementation part are created respectively on Manufacturer, Warehouse and Retailer.

2.3 Process Flow

Channel Message

Data flows between each module of the system in a separate channel. Each instance of a module is connected to another using a channel where message is exchanged following the UDP Protocol. And all the channels are maintained using the Channel Manager.



Read and write Thread's are used to read and write the incoming and outgoing message packets between modules. All the requests are processed in a message queue which is controlled a message processor over the channels.

Action

All the processes in the system (eg. submitOrder etc...) are termed as a action and is categorised . And it is used in process communication between modules according to each sub system.

HeartBeat

Heartbeat messages are been used between the replica's and the replica managers to communicate during failure of the replica. Using the heartbeat message the replica manager is used to decide the failure of the replica and restarts it.

3 Techniques Used

The application use Java API for XML Web Services (JAX-WS) technology together with TCP /IP protocol to implements all the actions. Remote Procedure Call-oriented (RPC-oriented) web services are implemented in client-retailer, retailer-warehouse, and warehouse-manufacturer connections. Logger server to other ends' communication takes places in TCP/IP protocol.

UDP protocol is been used between all the front end - sequencer, sequencer-replica and replica-replica managers. And the protocol is made reliable using ack message and time out specifications.

4 How to run

Compile the project:

In the bin folder, all classes have been created. If the bin folder does not contain the class files, please open the project with eclipse or use command line to compile it.

Copy the settings and libs folder into bin:

Copy the settings folder and libs folder resident in the root folder of the project.

Run logger server:

Open a Prompt in Windows or a Terminal in Linux, change directory to bin, then run: java loggerserver/LoggerServer

Run manufacturer(s):

Run manufacturer front end with parameter 1 to 4

```
   java -cp ".:./libs/dom4j-2.0.0-ALPHA-2.jar:./libs/swingx-1.6.jar:"
manufacturer/ManufacturerFE $1
```

Run manufacturer sequencer with parameter 1 to 4

```
   java -cp ".:./libs/dom4j-2.0.0-ALPHA-2.jar:./libs/swingx-1.6.jar:"
sequencer/manufacturerSequencer/ManufacturerSequencer $1
```

Run warehouse(s):

Run warehouse front end with parameter 1 to 4

```
   java -cp ".:../libs/dom4j-2.0.0-ALPHA-2.jar:../libs/swingx-1.6.jar:"
warehouse.WarehouseFE $1
```

Run warehouse sequencer with parameter 1 to 4

```
   java -cp ".:../libs/dom4j-2.0.0-ALPHA-2.jar:../libs/swingx-1.6.jar:"
sequencer/warehouseSequencer/WarehouseSequencer $1
```

Run retailer(s):

Run retailer front end

```
   java -cp ".:../libs/dom4j-2.0.0-ALPHA-2.jar:../libs/swingx-1.6.jar:"
retailer.RetailerFE
```

Run retailer sequencer

```
   java -cp ".:../libs/dom4j-2.0.0-ALPHA-2.jar:../libs/swingx-1.6.jar:"
sequencer.retailerSequencer.RetailerSequencer
```

Run client:

```
 java -cp ".:../libs/dom4j-2.0.0-ALPHA-2.jar:../libs/swingx-1.6.jar:" client.Client
```

Run replica manager:

```
 java -cp ".:../libs/dom4j-2.0.0-ALPHA-2.jar:../libs/swingx-1.6.jar:" rm.RM $1 $2
   example: java -cp ".:../libs/dom4j-2.0.0-ALPHA-2.jar:../libs/swingx-1.6.jar:"
rm.RM Retailer 1
         java -cp ".:../libs/dom4j-2.0.0-ALPHA-2.jar:../libs/swingx-1.6.jar:" rm.RM
Warehouse1 4
```

5 Test scenarios:

5.1 Test cases for system : Manufacturer

ID	1
Title	processPurchaseOrder
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Manufacturer system should be running after processing the purchase order.
Priority	High
Test Steps	• Warehouse calls the manufacturer to process the order item using the product type and other information's. • Replenish process of the warehouse does the call. • Manufacturer system checks the product type and unit price for the item ordered. • Manufacturer system produces the ordered item if all conditions satisfy and returns the order number to the warehouse.
Expected Results	• Order Number should be received in the warehouse after successful production of the ordered item.
ID	2
Title	getProductInfo
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Manufacturer system should be running after processing

	the product information
Priority	Medium
Test Steps	• Warehouse calls the manufacturer to retrieve the product information. • Uses the Product type to fetch the information. • Manufacturer system checks the product type and sends the product information.
Expected Results	• Manufacturer system should deliver the product information if the product type matches anyone of its products.
ID	3
Title	receivePayment
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Manufacturer system should be running after receiving the payment from the warehouse
Priority	Medium
Test Steps	• Warehouse calls the manufacturer to send the payment • Uses the order number and total price as the input • Manufacturer system checks the order number and sends the payment received confirmation
Expected Results	• Manufacturer system should deliver the payment confirmation after checking the order number and should save the ordered items.
ID	4
Title	getProductList
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published.

	• Client should be published in a specific port number and must be connected with a retailer published.
ID	5
Title	getName
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Manufacturer system should be running after successfully sending its name.
Priority	Medium
Test Steps	• Warehouse calls the manufacturer to get its name.
Expected Results	• Manufacturer system should send its name to the warehouse system when this function is called.

5.2 Test cases for system : Warehouse

ID	6
Title	getProductsByID
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Warehouse system should be running after sending the product information.
Priority	High
Test Steps	• Retailer calls the warehouse to retrieve the product information using its product ID • getCatalog and getProducts functions of the retailersystem calls this function. • Warehouse system checks the product ID and sends the

	correct product information. • If the product ID is empty the system returns the whole product list from its inventory.
Expected Results	• Warehouse should reply the product list according to the product ID from its inventory.
ID	7
Title	getProductsByType
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Warehouse system should be running after sending the product information.
Priority	High
Test Steps	• Retailer calls the warehouse to retrieve the product list using product type. • Warehouse system checks the product type and sends the correct product lists from its inventory.
Expected Results	• Warehouse should reply the product list according to the product type from its inventory.
ID	8
Title	getProductsByRegisteredManufacturers
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Warehouse system should be running after sending the product list using the manufactured name.
Priority	High
Test Steps	• Retailer calls the warehouse to retrieve the product list

	using the manufacture name. • Warehouse checks whether the manufacturer is registered with it. • Warehouse then retrieves the Product list using the manufacture name from its inventory.
Expected Results	• Warehouse should send the product list according to the manufacture name received.
ID	9
Title	getProducts
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Warehouse system should be running after sending the product list using the manufacture name and product ID.
Priority	High
Test Steps	• Retailer calls the warehouse to retrieve the product list from a specific manufacture. • warehouse checks for the product id and manufacture name. • System then retrieves the product list information from the manufacture's inventory using the specific product id
Expected Results	• Warehouse should reply the product list according to the product ID and manufacture name from the manufacturer's inventory
ID	10
Title	shippingGoods
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and

	must be connected with a retailer published.
Post - Conditions	• Warehouse system should be running after returning the items ordered by the retailer.
Priority	High
Test Steps	• Retailer calls the warehouse to retrieve the products ordered. • warehouse checks the retailer name and order quantity. • System then supplies the ordered goods to the retailer. • If the item quantity is less it orders the manufacturer to complete the order. • Uses the replenish function for the above process.
Expected Results	• Warehouse should ship the item list to the retailer successfully as per the retailer's order.
ID	11
Title	registerRetailer
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Warehouse system should be running after registering the retailer using its name
Priority	Medium
Test Steps	• Retailer calls the warehouse to register itself . • Warehouse checks the retailer name. • System then adds the retailer to its list and sends confirmation information.
Expected Results	• Warehouse should reply the conformation after registering the retailer into its list.
ID	12
Title	unregisterRetailer
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer.

	• Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Warehouse system should be running after un-registering the retailer using its name
Priority	Medium
Test Steps	• Retailer calls the warehouse to un-register itself . • Warehouse checks the retailer name in its list. • System then deletes the retailer from its list and sends confirmation information.
Expected Results	• Warehouse should reply the conformation after un-registering the retailer from its list.
ID	13
Title	getName
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Warehouse system should be running after successfully sending its name.
Priority	Medium
Test Steps	• Retailer calls the warehouse to get its name.
Expected Results	• Warehouse system should send its name to the retailer system when this function is called.

5.3 Test cases for system : Retailer

ID	14
Title	getCatalog
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number

	and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Retailer system should be running after sending the catalog information of all the products to the client.
Priority	High
Test Steps	• Client calls the retailer to get all the information of various products. • retailer checks all the warehouse connected to it. • retrieves the product list from all the warehouses. • Sends the whole list to the client.
Expected Results	• Retailer should reply the catalog of product information to the client.
ID	15
Title	submitOrder
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Retailer system should be running after sending the ordered list of products to the client.
Priority	High
Test Steps	• Client calls the retailer to place an order. • Retailer contacts the warehouses connected it to produce those orders. • Orders have information about product type and the quantity needed. • Once the order is prepared it returns the goods with the shipping status
Expected Results	• Retailer should reply the shipping status and the ordered items submitted by the client.
ID	16
Title	signUp
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific

	port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Retailer system should be running after sending the confirmation about the signUp process
Priority	High
Test Steps	• Client calls the retailer to get register as a valid customer in to the system. • Client passes the address , name and password to create its account information with the retailer.
Expected Results	• Retailer should reply signup confirmation result to the client, if the information provided is valid.
ID	17
Title	signIn
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Retailer system should be running after successful sign in process by the customer.
Priority	High
Test Steps	• Client calls the retailer to get logged into the system using reference number and password.
Expected Results	• Retailer should reply signin confirmation result to the client, if the information provided is valid.
ID	18
Title	getProducts
Pre-Conditions	• Logger server should be running • Manufacturer system should be published using a specific port number. • Warehouse should be published in a specific port number

	and must be connected with the manufacturer. • Retailer should be published in a specific port number and must be connected with a warehouse published. • Client should be published in a specific port number and must be connected with a retailer published.
Post - Conditions	• Retailer system should be running after successfully sending the product list from the warehouses.
Priority	High
Test Steps	• Client calls the retailer to get the product list from all its warehouses. • Retailer find the list of all the products from the warehouses which it is connected too. • Retailer then sends the list to the client.
Expected Results	• Retailer should reply the list of all the products available with it and also with the warehouses connected with it.

6 Extra works and improvements

- Registration of Retailer:

  Supply chain system consists of many retailers and warehouses. So any retailer can receive the information about list of products in a warehouse but only registered retailer only can make an order for the products. Warehouse maintains the list of the registered retailers and verify the retailer when asks for good. Retailer should provide the name of retailer and list of items to ship the goods.

- Amazon online service:

  Amazon online service is used to generate a query to retrieve initial prices of products. An account is created in Amazon to retrieve the API to receive items. If not, default prices are generated with random functions.

- Retailer Service:

  Retailer connects with the warehouse to receive goods. Retailer can register with any number of warehouses and can ask to ship the good. We have two retailer classes one sequentially access the list of warehouses and other, in which the retailer can ask for good from two warehouses.

- Replica Implementation:

  Implementation of replica's for each module is used for the high availability of the system during software failure or process crash. 4 replica's are created for system module. By which the system can handle 3F+1 byzantine failure.

- Sequencer Implementation:

  Sequencer is added for each module of the system, where each process is assigned a unique id before it gets processed. By using the sequencer, the total ordering of the process is done.

7 Problem faced

- Web service method which returns ArrayList :

  We encountered problems when implementing a web service method which returns an ArrayList or has an ArrayList as the parameter. Finally we solved it by wrapping the ArrayList into a newly created class, like ItemList, ProductList.

- How to design Web Service which is highly available :

  We did a lot of research before choosing JAX_WS as the option to implement web service. Eclipse dynamic Web Object seems easy, but there are not much examples about how to write the client code, and it has been proved that using JAX_WS has a big advantage. It is flexible and platform independent.

- Replica Management is very difficult. Acknowledgment and heart beat messages between replica's and replica managers were too challenging. Also replica crash handling and restarting of the replica with good data is very drastic improvement needed to be done.
- UDP protocol management was very difficult too . Making the protocol into a reliable structure was the most challenging phase in the project.
- Hard to debug.

  Not like traditional project, distributed system runs by multiple processes even on different machines. Java debug tools do not work. To guarantee the system runs correctly, we implemented lots of unit tests. Meanwhile, log is very useful for debugging.
- Synchronization Management

  Due to higher dependency between modules and resource sharing, the task of maintaining synchronization was challenging, especially between the warehouse and retailer modules.
- Hard to work in team.

  It is very hard to divide a project into several separated tasks.

8 References

- Java EE 6 web services http://docs.oracle.com/javaee/6/tutorial/doc/bnayl.html
- Object AID explorer plugin for eclipse - http://www.objectaid.com/
- WEb services and replication https://en.wikipedia.org/wiki/Session_(computer_science)