

Compte Rendu RMI

Lu Chenxin, Xu Jitao, Zhu Zhenghui

Cas d'utilisation:

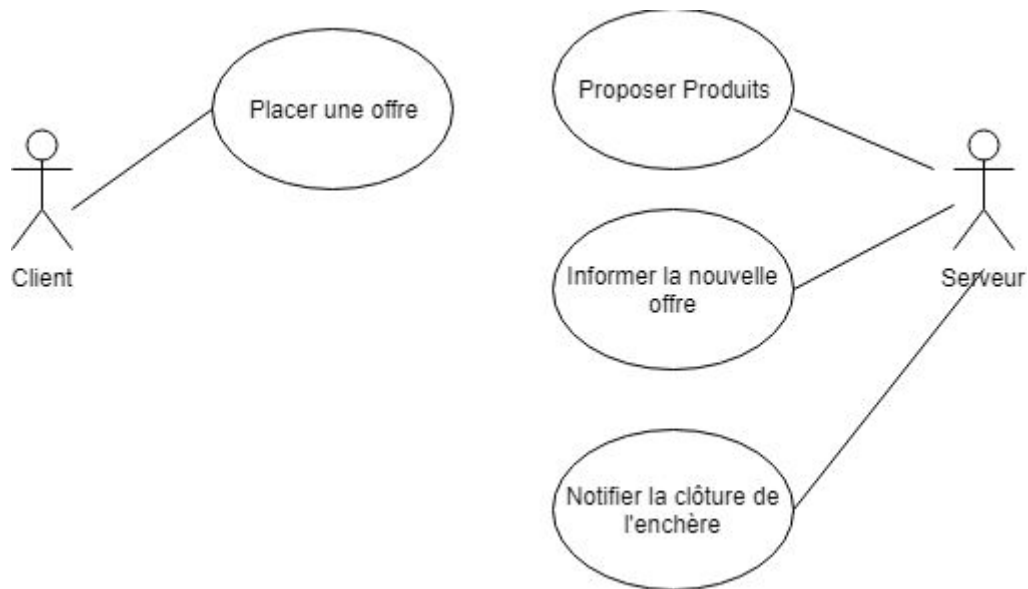


Diagramme de sequence:

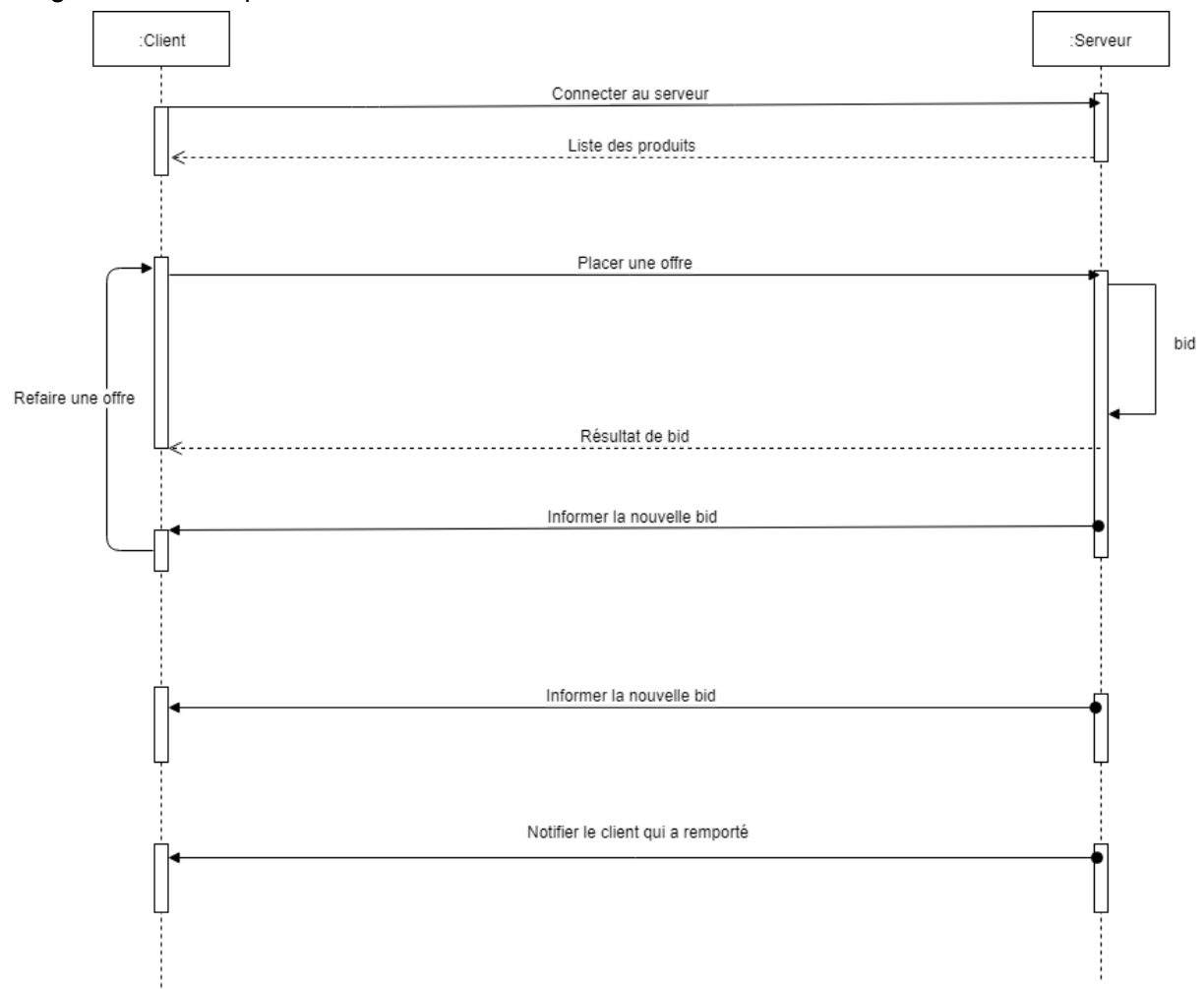
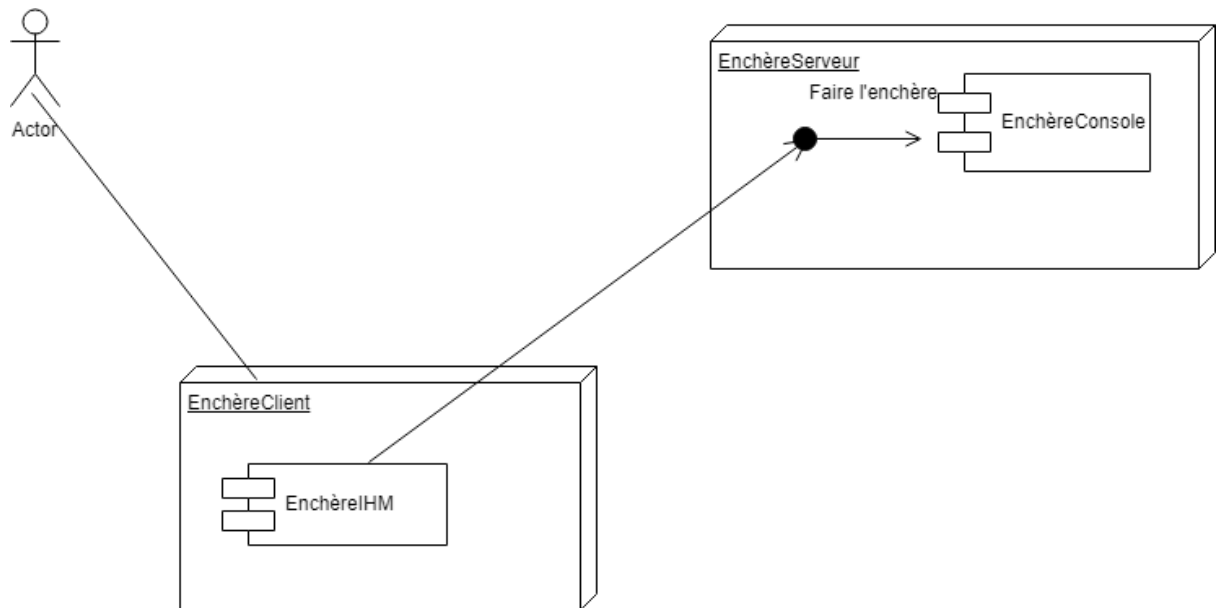


Diagramme de déploiement:



Code Serveur:

```
Server.java — src
UNREGISTERED

FOLDERS
src
  Bidder.java
  BidderImpl.java
  Client.java
  Notification.java
  NotificationImpl.java
  Product.java
  ProductImpl.java
  Server.java

Server.java
1 import java.rmi.registry.*;
2 import java.rmi.server.UnicastRemoteObject;
3 import java.util.*;
4
5 public class Serveur {
6     public static void main(String args[]) {
7         int port = 1099;
8         if(args.length == 1)
9             port = Integer.parseInt(args[0]);
10        try {
11            ArrayList<Product> productList = new ArrayList<>();
12            Product stubIphone = (Product)UnicastRemoteObject.exportObject(new ProductImpl("IphoneX", 500, 900), 0);
13            Product stubPixel2 = (Product)UnicastRemoteObject.exportObject(new ProductImpl("Pixel2", 400, 700), 0);
14            productList.add(stubIphone);
15            productList.add(stubPixel2);
16            Registry registry = LocateRegistry.getRegistry(port);
17            if(!Arrays.asList(registry.list()).contains("IphoneXCallback")) {
18                registry.bind("IphoneXCallback", stubIphone);
19            }
20            else {
21                registry.rebind("IphoneXCallback", stubIphone);
22            }
23            if(!Arrays.asList(registry.list()).contains("Pixel2Callback")) {
24                registry.bind("Pixel2Callback", stubPixel2);
25            }
26            else {
27                registry.rebind("Pixel2Callback", stubPixel2);
28            }
29            System.out.println("Service IphoneXCallback et Pixel2Callback lient au registre.");
30
31            ArrayList<Notification> notificationList = new ArrayList<>();
32
33            String[] callbackNameList = registry.list();
34            int nbCallback = callbackNameList.length;
35            for (int i = 0; i < callbackNameList.length; i++) {
36                if (callbackNameList[i].startsWith("NotificationCallback")) {
37                    notificationList.add((Notification)registry.lookup(callbackNameList[i]));
38                }
39            }
40
41            for (Notification n : notificationList) {
42                n.setProductList(productList);
43                n.productList();
44            }
45
46            ArrayList<Integer> oldPrices = new ArrayList<>();
47            for(int i = 0; i < productList.size(); i++) {
```

```

46     ArrayList<Integer> oldPrices = new ArrayList<>();
47     for(int i = 0; i < productList.size(); i++) {
48         oldPrices.add(productList.get(i).getPrice());
49     }
50     while (true) {
51         callbackNameList = registry.list();
52         if (nbCallback < callbackNameList.length) {
53             System.out.println("New product added.");
54             int prodSize = productList.size();
55             int registrySize = callbackNameList.length;
56             int nb = registrySize - prodSize - 1;
57             if (callbackNameList[nb].startsWith("NotificationCallback")) {
58                 notificationList.add((Notification)registry.lookup(callbackNameList[nb]));
59             }
60             nbCallback = callbackNameList.length;
61             notificationList.get(nb).setProductList(productList);
62             notificationList.get(nb).productList();
63         }
64     }
65 }
66 catch (Exception e) {
67     System.out.println(e);
68 }
69 }
70 }
71

```

Code Client:

```

1  import java.rmi.registry.*;
2  import java.rmi.server.UnicastRemoteObject;
3  import java.util.*;
4
5  public class Client {
6      public static void main(String args[]) {
7          String machine = "localhost";
8          int port = 1099;
9          Bidder stubBidder, bidder = new BidderImpl();
10         Notification stubNotification, notification = new NotificationImpl();
11         Scanner sc;
12         if(args.length == 3) {
13             machine = args[0];
14             port = Integer.parseInt(args[1]);
15         }
16         else if(args.length == 2) {
17             machine = args[0];
18         }
19         try {
20             Registry registry = LocateRegistry.getRegistry(port);
21             stubBidder = (Bidder)UnicastRemoteObject.exportObject(bidder, 0);
22             stubNotification = (Notification)UnicastRemoteObject.exportObject(notification, 0);
23             String notificationCallbackName = "NotificationCallback";
24             int nbNotif = 1;
25             while (true) {
26                 String temp = notificationCallbackName + nbNotif;
27                 if(!Arrays.asList(registry.list()).contains(temp)) {
28                     registry.bind(temp, stubNotification);
29                     break;
30                 }
31                 else {
32                     nbNotif++;
33                 }
34                 System.out.println(temp);
35             }
36             System.out.println("Service NotificationCallback lie au registre.");
37
38             while (true) {
39                 System.out.println("Saisir le nom de produit et le prix d'enchère, SVP. Exit pour ne pas faire enchère.");
40                 sc = new Scanner(System.in);
41                 String bid = sc.nextLine();
42                 if (bid.equalsIgnoreCase("exit")) {
43                     UnicastRemoteObject.unexportObject(notification, true);
44                     break;
45                 }
46                 String[] parts = bid.split(" ");
47                 String productName = parts[0];
48                 int bidPrice = Integer.parseInt(parts[1]);
49                 ArrayList<Product> products = notification.getProductList();
50                 for (Product p : products) {
51                     if (p.getName().equalsIgnoreCase(productName)) {
52                         p.bid(bidPrice, stubNotification);
53                         break;
54                     }
55                 }
56             }
57         }
58         catch (Exception e) {
59             System.out.println("Client exception: " + e);
60         }
61     }
62 }
63

```