Student’s name: Nguyễn Đức Dũng

Class: ICT.02-K61

**Class Exercises**

**Module: Distributed Systems**

**Chapter 4: Communication**

**Practical Exercises:**

**Question 1:** What is the code part that shows that the Server assigns the correlation ID to the response?

DeliverCallback deliverCallback = (consumerTag, delivery) -> {  
 AMQP.BasicProperties replyProps = new AMQP.BasicProperties  
 .Builder()  
 **.correlationId(delivery.getProperties().getCorrelationId())** .build();

**Question 2:** You base on both code of Client and Server program to explain which code shows that the Client sends request to Server through *rpc\_queue* and create a new queue to wait for the reply of the Server.

* In Client side:

public String call(String message) throws IOException, InterruptedException {  
 final String corrId = UUID.*randomUUID*().toString();  
 String replyQueueName = channel.queueDeclare().getQueue();  
 AMQP.BasicProperties props = new AMQP.BasicProperties  
 .Builder()  
 .correlationId(corrId)  
 .replyTo(replyQueueName)  
 .build();  
 channel.basicPublish("", requestQueueName, props, message.getBytes("UTF-8"));  
 final BlockingQueue<String> response = new ArrayBlockingQueue<>(1);  
 String ctag = channel.basicConsume(replyQueueName, true, (consumerTag, delivery) -> {  
 if (delivery.getProperties().getCorrelationId().equals(corrId)) {  
 response.offer(new String(delivery.getBody(), "UTF-8"));  
 }  
 }, consumerTag -> {  
 });  
 String result = response.take();  
  
 channel.basicCancel(ctag);  
 return result;  
}

We establish a connection and channel.

Our call method makes the actual RPC request.

Here, we first generate a unique correlationId number and save it - our consumer callback will use this value to match the appropriate response.

Then, we create a dedicated exclusive queue for the reply and subscribe to it.

Next, we publish the request message, with two properties: replyTo and correlationId.

At this point we can sit back and wait until the proper response arrives.

Since our consumer delivery handling is happening in a separate thread, we're going to need something to suspend the main thread before the response arrives. Usage of BlockingQueue is one possible solutions to do so. Here we are creating ArrayBlockingQueue with capacity set to 1 as we need to wait for only one response.

The consumer is doing a very simple job, for every consumed response message it checks if the correlationId is the one we're looking for. If so, it puts the response to BlockingQueue.

At the same time main thread is waiting for response to take it from BlockingQueue.

Finally we return the response back to the user.

* And in server side:

DeliverCallback deliverCallback = (consumerTag, delivery) -> {  
 AMQP.BasicProperties replyProps = new AMQP.BasicProperties  
 .Builder()  
 .correlationId(delivery.getProperties().getCorrelationId())  
 .build();  
 String response = "";  
 try {  
 String message = new String(delivery.getBody(), "UTF-8");  
  
 int n = Integer.*parseInt*(message);  
  
 System.*out*.println(" [.] fib(" + message + ")");  
  
 response += *fib*(n);  
 } catch (RuntimeException e) {  
 System.*out*.println(" [.] " + e.toString());  
 } finally {  
 channel.basicPublish("", delivery.getProperties().getReplyTo(), replyProps,  
 response.getBytes("UTF-8"));  
 channel.basicAck(delivery.getEnvelope().getDeliveryTag(), false);  
 // RabbitMq consumer worker thread notifies the RPC server owner thread  
 synchronized (monitor) {  
 monitor.notify();  
 }  
 }  
};  
channel.basicConsume(*RPC\_QUEUE\_NAME*, false, deliverCallback, (consumerTag -> {  
}));

We use basicConsume to access the queue, where we provide a callback in the form of an object (DeliverCallback) that will do the work and send the response back.

|  |
| --- |
| **Question 3:** Now, you try to add the delay to the Server program in inserting this code below the line: response += fib(n);  try { Thread.sleep(2000);  } catch (InterruptedException \_ignored) { Thread.currentThread().interrupt();  }  The Server program will sleep 2 seconds for each request. Rebuild the Server and launch it. Open several other command window and launch the Client on them simultaneously. Open a separated command window and launch the command below: |
| >rabbitmqctl.bat list\_queues name messages\_ready  messages\_unacknowledged |
| What is the result you obtain? Explain it.  Timeout: 60.0 seconds ...  Listing queues for vhost / ...   |  |  |  | | --- | --- | --- | | Name | Messages\_ready | Messages\_unacknowledged | | amq.gen-7rH7KU1PXH6hb-NxSTQmWg | 0 | 0 | | amq.gen-wBIDPKnNatLn-xpeFWXlvQ | 0 | 0 | | rpc\_queue | 3 | 1 | | amq.gen-yZSFSJw3hgeOSygNAEgAuw | 0 | 0 | | amq.gen-5leAe1ynrf7ufRAWe10m5g | 0 | 0 |   The comment show list queues are running, with the number of message\_ready and number of messages\_unacknowledged. |

**Question 4:** What is the IP address of your 2 machines? How to ping each other?

We setting Networking in setting of machine to Bridged Adapter

And then open terminal, type: ifconfig to get IP adress !

Now, for my practice, I get 2 IP address on 2 virtual machine:  
192.168.1.153 and 192.168.1.16  
To ping each other, I use command: ping 192.168.1.16 or ping 191.168.1.153

**Question 5:** Can you watch the video in the client machine? Evaluate the quality of the video streaming service.

I can watch the video in client machine, And quality of video streaming is the same between client and server.

**Question 6:** What is the result of the ping test? Can you see an increase of 100 milliseconds?

Yes, the time field increase of 100 ms !

**Question 7:** Disable the buffering function of VLC in Client machine. Then, evaluate the video quality at the Client machine. How can you conclude the impact of fix delay on video streaming service?

No impact of delay on video streamming service compare to quality of video streaming without delay.

**Question 8:** Evaluate the video quality at the Client machine. How can you conclude the impact of delay variation on video streaming service?

The variation in video is caused with the variable delay introduced in combination with fixed delay. Fixed delay introduces constant delay on video sequences and has no significant impact on the video quality. Variable delay, for instance 100ms±10ms introduces delay between 90 and 110 which has higher impact on video quality.

And, for testing on client machine, I saw some noise sometime on video streamming.

**Question 9:** Evaluate the video quality at the Client machine. How can you conclude the impact of fix loss rate on video streaming service? Try to increase the value of loss rate to see the impact more clear.

When amount of packet lost increases, result shows that MOS suffers an exponential decay. Many people were able to observe that the videos were poor as the percentages of losses peaks to the maximum allowable loss percentage for my testing.

**Question 10:** Evaluate the video quality at the Client machine. How can you conclude the impact of loss rate variation on video streaming service? Try to increase this value to see the impact more clear.

Low rate variances of packet losses are observed to be low for all videos and packet sizes. When increase loss rate varition, these results could be translated as how difficult is for viewers to make decision on the videos they watch in the range between 0.3% and 1% of packets. But, low values variance for a range between 3% and 10% shows that viewers were easily able to decide on whether the video is bad or good.

**Question 11:** Evaluate the video quality at the Client machine. How can you conclude the impact of packet duplication on video streaming service? Try to increase this value to see the impact more clear.

In my test, I cannot see clearly effect of packet duplication.

**Question 12:** Evaluate the video quality at the Client machine. How can you conclude the impact of packet corruption on video streaming service?

In my test, I cannot see clearly effect of packet corruption.