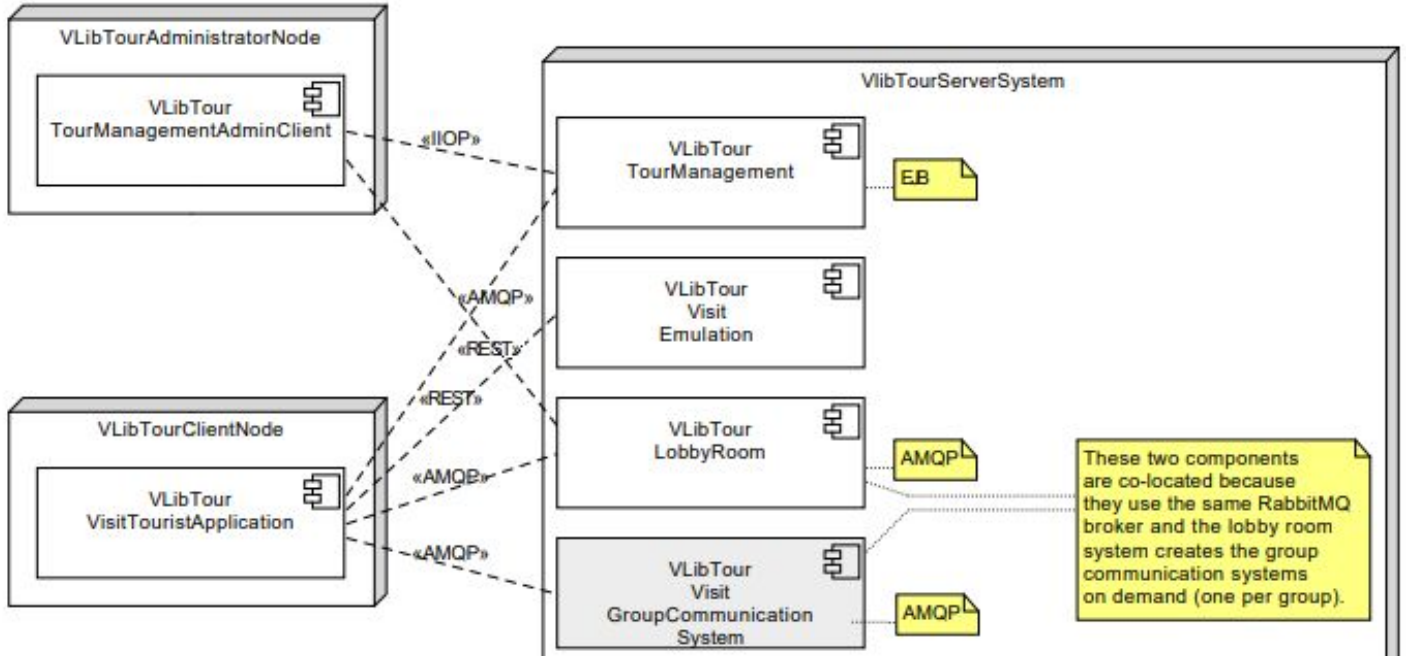
Samuel GUILLEMET, Nathan FERET
10 Novembre 2023

CSC 5002

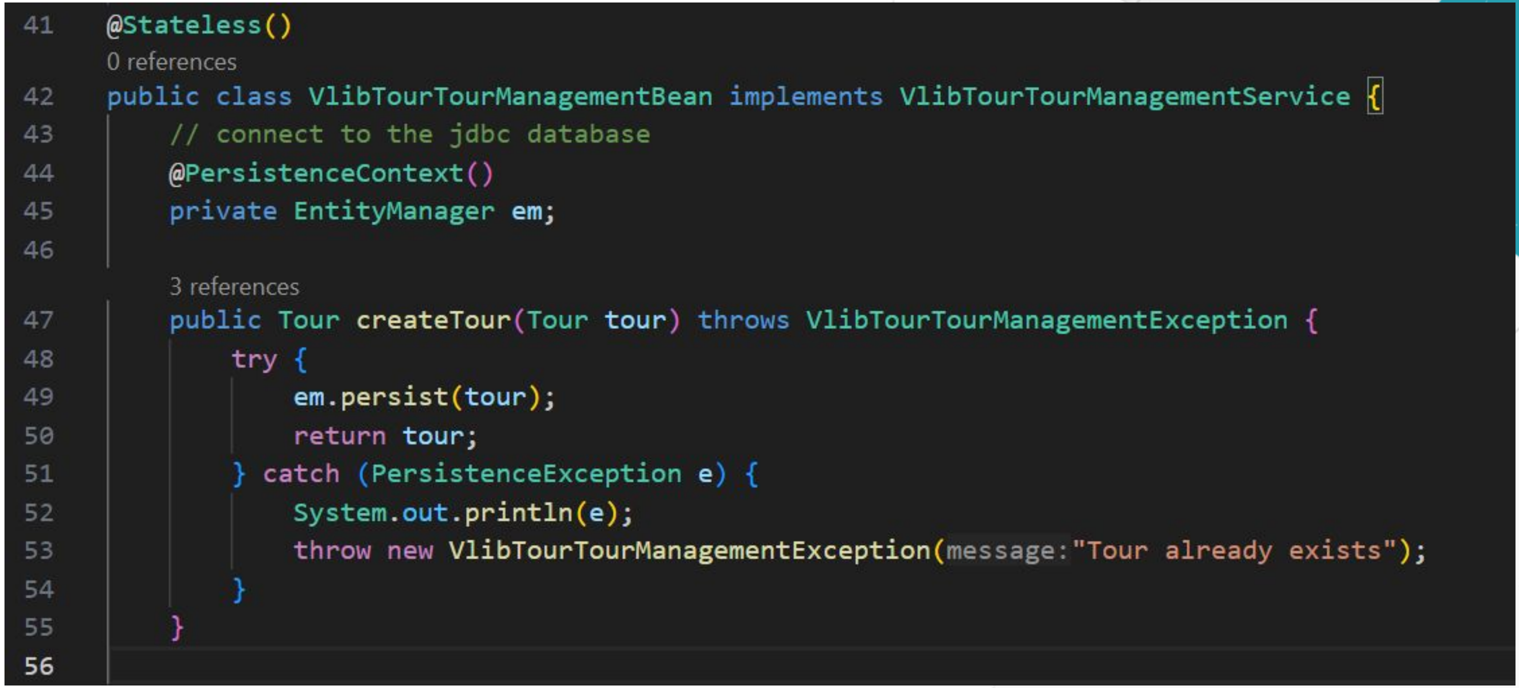
Middleware and software architecture for distributed applications

Architecture de l'application


VLibTourAdministratorNode
VLibTour TourManagementAdminClient
«IIOP»
VLibTourClientNode
VLibTour VisitTouristApplication
«AMQP»
«REST»
«REST»
«AMQP»
«AMQP»
VlibTourServerSystem
VLibTour TourManagement
EJB
VLibTour Visit Emulation
VLibTour LobbyRoom
AMQP
VLibTour Visit GroupCommunication System
AMQP
These two components are co-located because they use the same RabbitMQ broker and the lobby room system creates the group communication systems on demand (one per group).

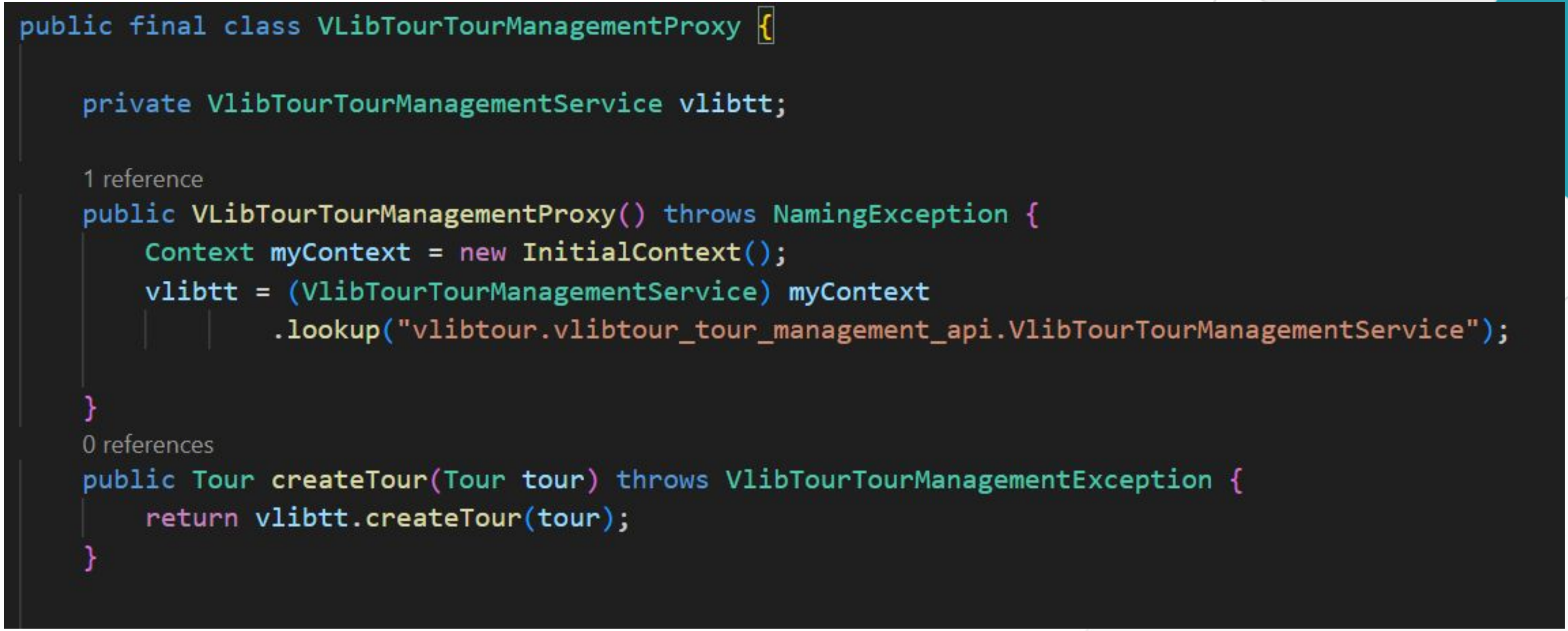
TourManagement

TourManagementBeans

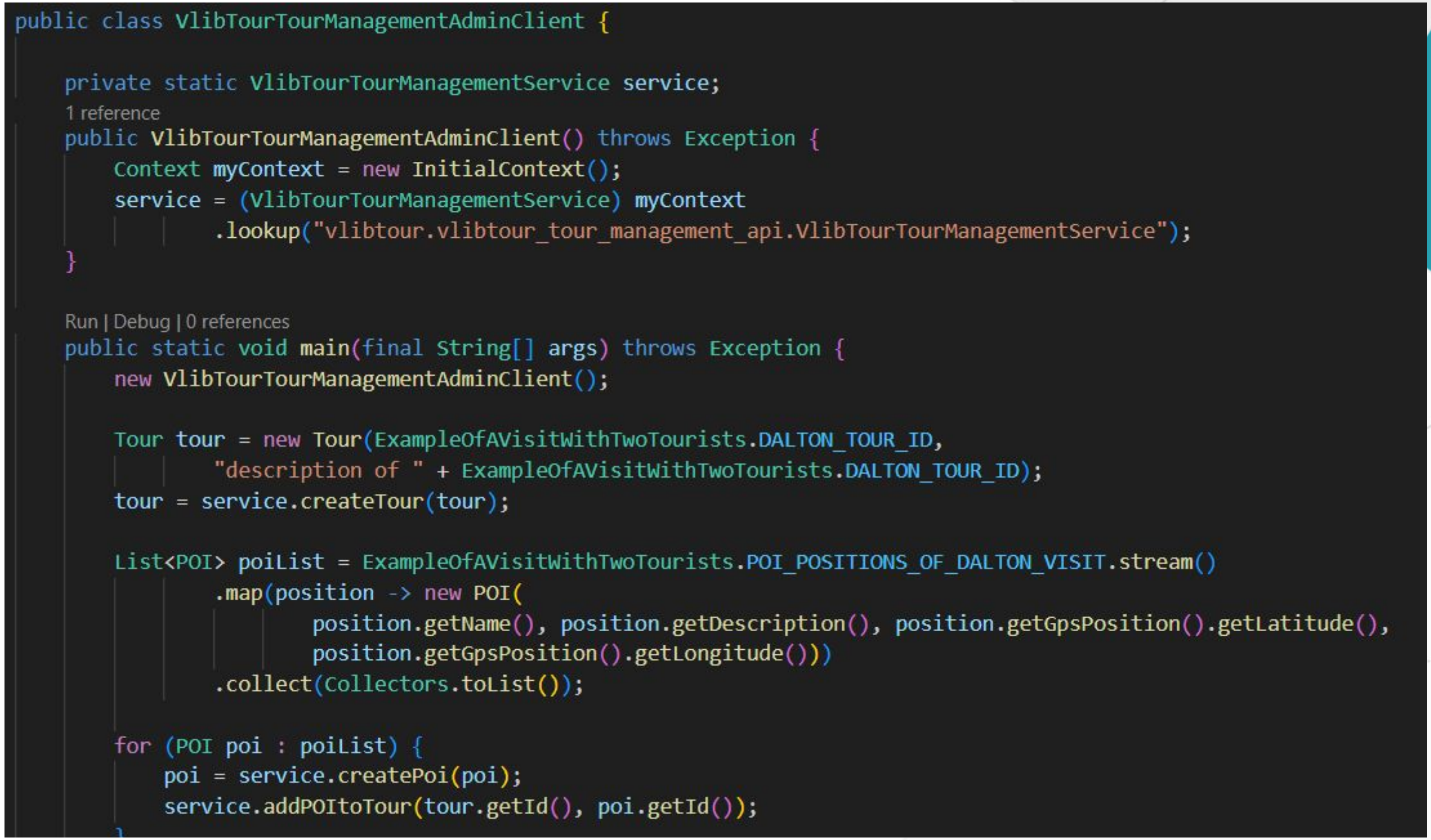


```
@Stateless()
0 references
public class VlibTourTourManagementBean implements VlibTourTourManagementService {
    // connect to the jdbc database
    @PersistenceContext()
    private EntityManager em;

    3 references
    public Tour createTour(Tour tour) throws VlibTourTourManagementException {
        try {
            em.persist(tour);
            return tour;
        } catch (PersistenceException e) {
            System.out.println(e);
            throw new VlibTourTourManagementException(message:"Tour already exists");
        }
    }

```

TourManagement : Le Proxy


```
public final class VLibTourTourManagementProxy {

    private VlibTourTourManagementService vlibtt;

    1 reference
    public VLibTourTourManagementProxy() throws NamingException {
        Context myContext = new InitialContext();
        vlibtt = (VlibTourTourManagementService) myContext
                .lookup("vlibtour.vlibtour_tour_management_api.VlibTourTourManagementService");

    }
    0 references
    public Tour createTour(Tour tour) throws VlibTourTourManagementException {
        return vlibtt.createTour(tour);
    }
```

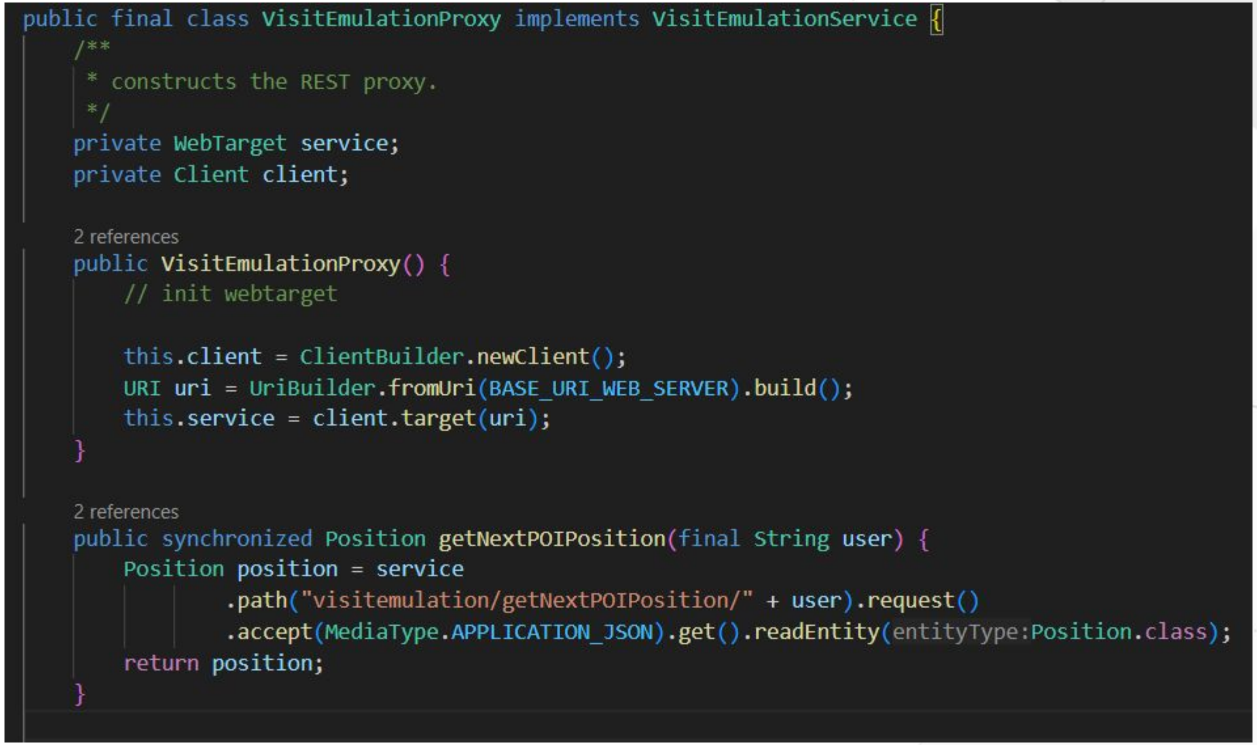
TourManagementAdmin



```java
public class VlibTourTourManagementAdminClient {

    private static VlibTourTourManagementService service;
    1 reference
    public VlibTourTourManagementAdminClient() throws Exception {
        Context myContext = new InitialContext();
        service = (VlibTourTourManagementService) myContext
                .lookup("vlibtour.vlibtour_tour_management_api.VlibTourTourManagementService");
    }

    Run | Debug | 0 references
    public static void main(final String[] args) throws Exception {
        new VlibTourTourManagementAdminClient();

        Tour tour = new Tour(ExampleOfAVisitWithTwoTourists.DALTON_TOUR_ID,
                "description of " + ExampleOfAVisitWithTwoTourists.DALTON_TOUR_ID);
        tour = service.createTour(tour);

        List<POI> poiList = ExampleOfAVisitWithTwoTourists.POI_POSITIONS_OF_DALTON_VISIT.stream()
                .map(position -> new POI(
                        position.getName(), position.getDescription(), position.getGpsPosition().getLatitude(),
                        position.getGpsPosition().getLongitude()))
                .collect(Collectors.toList());

        for (POI poi : poiList) {
            poi = service.createPoi(poi);
            service.addPOItoTour(tour.getId(), poi.getId());
```

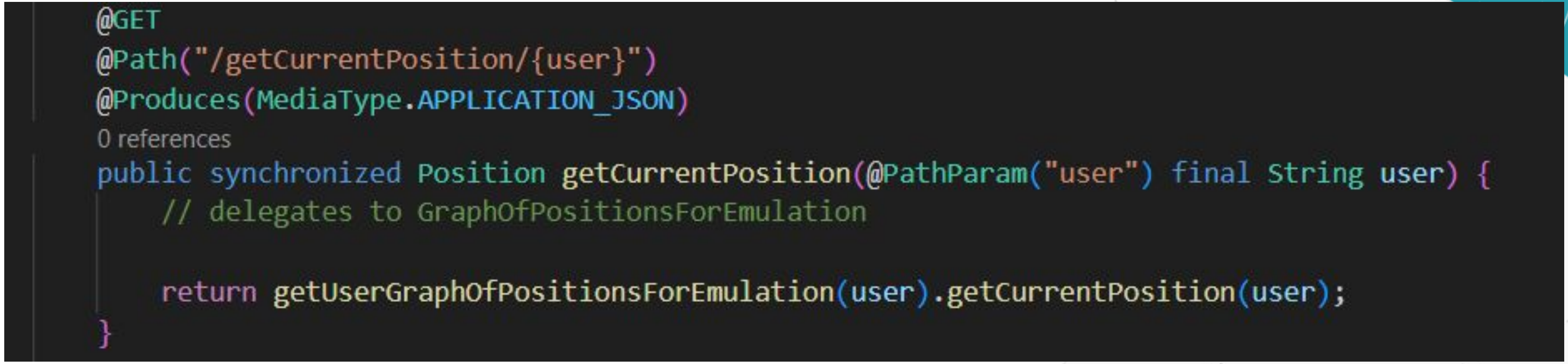
VisitEmulationServer

VisitEmulationServer : Proxy



```
public final class VisitEmulationProxy implements VisitEmulationService {
    /**
     * constructs the REST proxy.
     */
    private WebTarget service;
    private Client client;

    2 references
    public VisitEmulationProxy() {
        // init webtarget

        this.client = ClientBuilder.newClient();
        URI uri = UriBuilder.fromUri(BASE_URI_WEB_SERVER).build();
        this.service = client.target(uri);
    }

    2 references
    public synchronized Position getNextPOIPosition(final String user) {
        Position position = service
                .path("visitemulation/getNextPOIPosition/" + user).request()
                .accept(MediaType.APPLICATION_JSON).get().readEntity(entityType:Position.class);
        return position;
    }
```

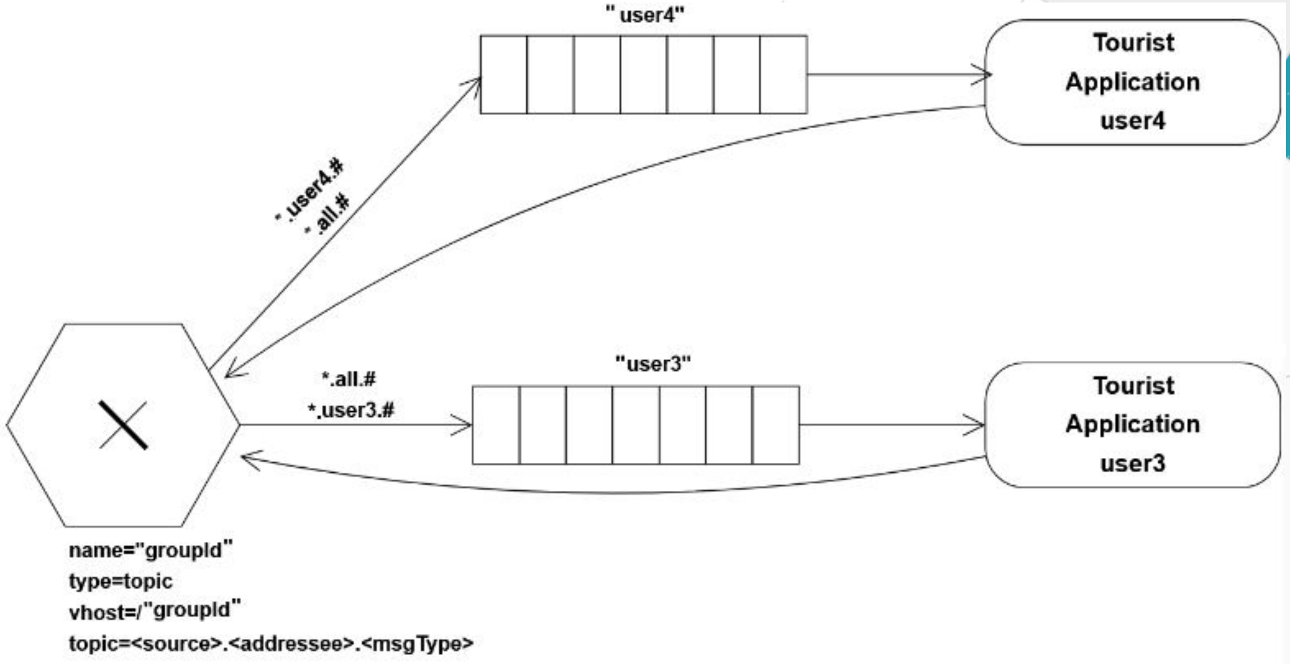
VisitEmulationServer



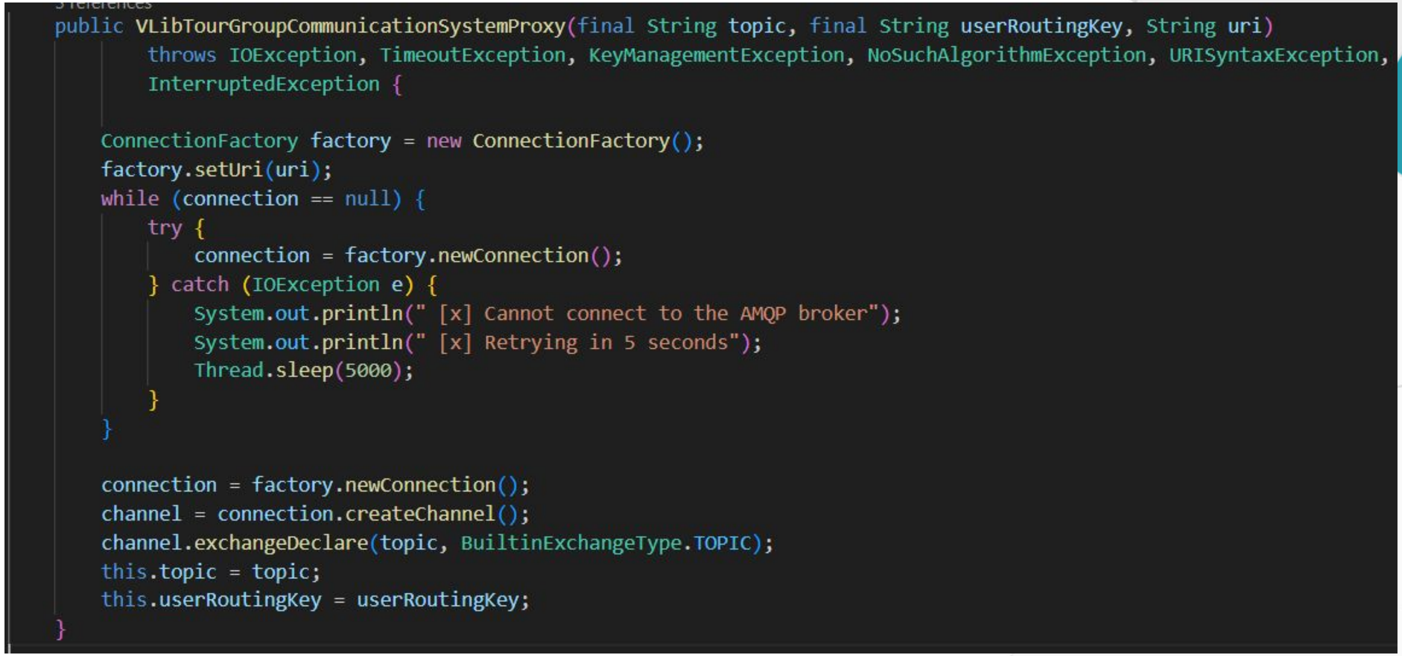
```
@GET
@Path("/getCurrentPosition/{user}")
@Produces(MediaType.APPLICATION_JSON)
0 references
public synchronized Position getCurrentPosition(@PathParam("user") final String user) {
    // delegates to GraphOfPositionsForEmulation

    return getUserGraphOfPositionsForEmulation(user).getCurrentPosition(user);
}
```

Group Communication System

Group Communication System

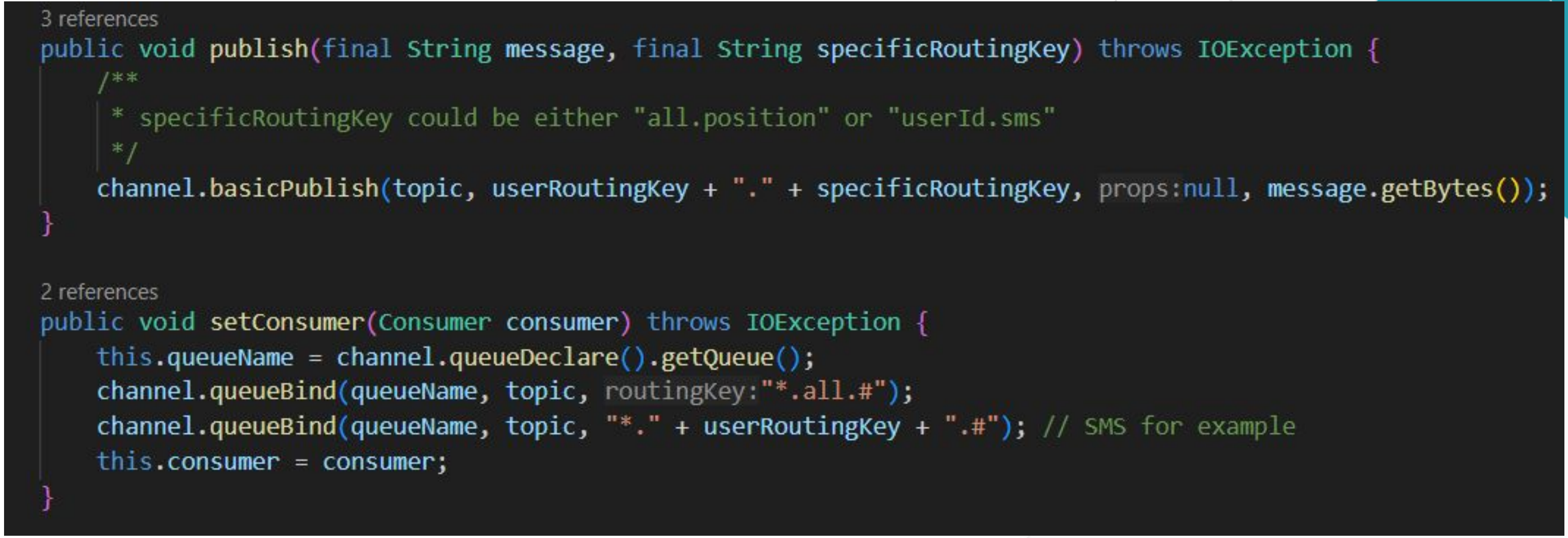


Group Communication System : Proxy



```
    public VLibTourGroupCommunicationSystemProxy(final String topic, final String userRoutingKey, String uri)
            throws IOException, TimeoutException, KeyManagementException, NoSuchAlgorithmException, URISyntaxException,
            InterruptedException {

        ConnectionFactory factory = new ConnectionFactory();
        factory.setUri(uri);
        while (connection == null) {
            try {
                connection = factory.newConnection();
            } catch (IOException e) {
                System.out.println(" [x] Cannot connect to the AMQP broker");
                System.out.println(" [x] Retrying in 5 seconds");
                Thread.sleep(5000);
            }
        }

        connection = factory.newConnection();
        channel = connection.createChannel();
        channel.exchangeDeclare(topic, BuiltinExchangeType.TOPIC);
        this.topic = topic;
        this.userRoutingKey = userRoutingKey;
    }
```

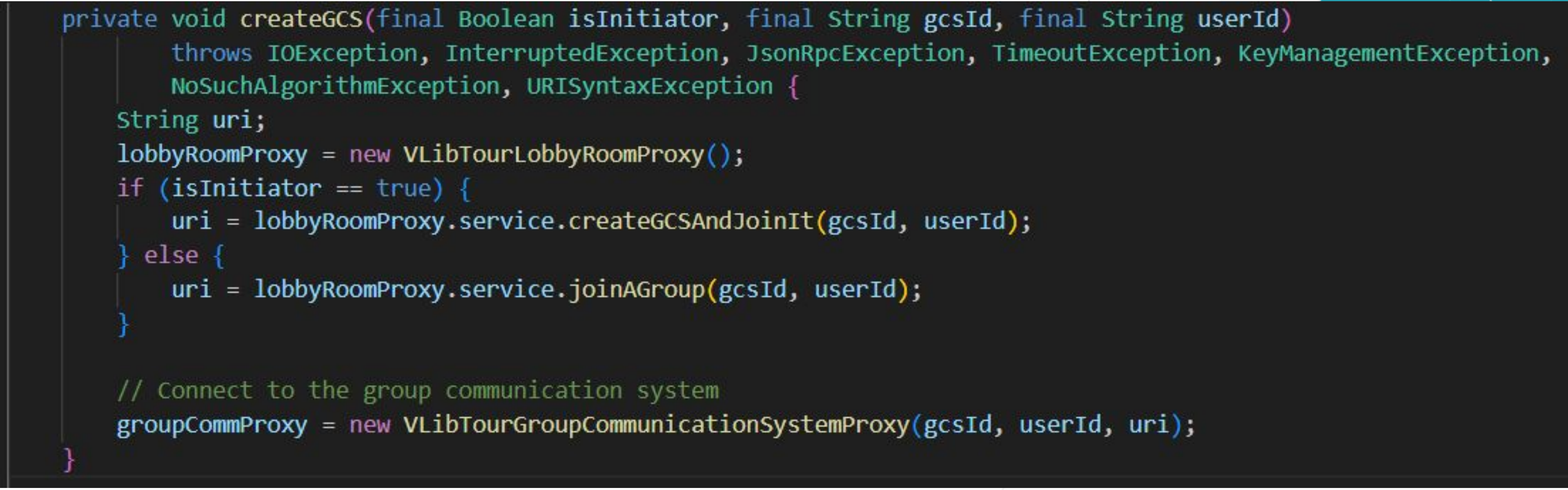
Group Communication System : méthodes



```
3 references
public void publish(final String message, final String specificRoutingKey) throws IOException {
    /**
     * specificRoutingKey could be either "all.position" or "userId.sms"
     */
    channel.basicPublish(topic, userRoutingKey + "." + specificRoutingKey, props:null, message.getBytes());
}

2 references
public void setConsumer(Consumer consumer) throws IOException {
    this.queueName = channel.queueDeclare().getQueue();
    channel.queueBind(queueName, topic, routingKey:"*.all.#");
    channel.queueBind(queueName, topic, "*." + userRoutingKey + ".#"); // SMS for example
    this.consumer = consumer;
}
```

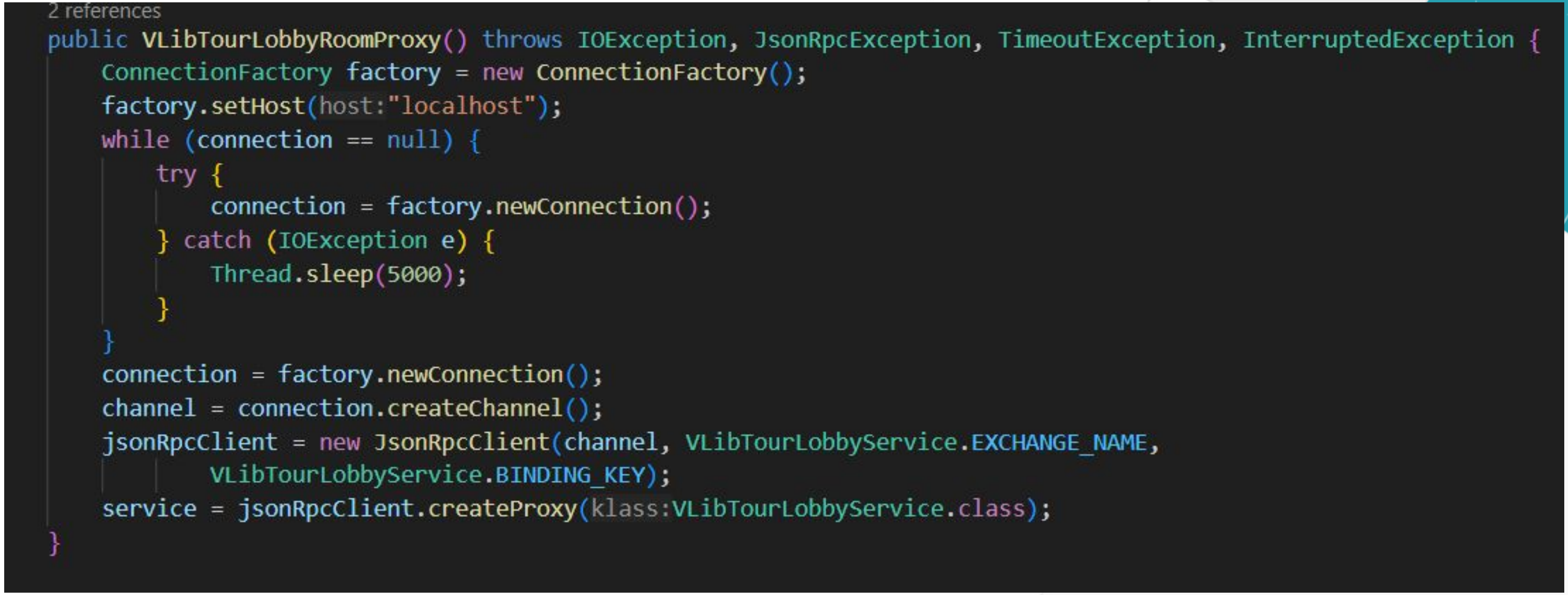
LobbyRoom

LobbyRoom



```
private void createGCS(final Boolean isInitiator, final String gcsId, final String userId)
        throws IOException, InterruptedException, JsonRpcException, TimeoutException, KeyManagementException,
        NoSuchAlgorithmException, URISyntaxException {
    String uri;
    lobbyRoomProxy = new VLibTourLobbyRoomProxy();
    if (isInitiator == true) {
        uri = lobbyRoomProxy.service.createGCSAndJoinIt(gcsId, userId);
    } else {
        uri = lobbyRoomProxy.service.joinAGroup(gcsId, userId);
    }

    // Connect to the group communication system
    groupCommProxy = new VLibTourGroupCommunicationSystemProxy(gcsId, userId, uri);
}
```

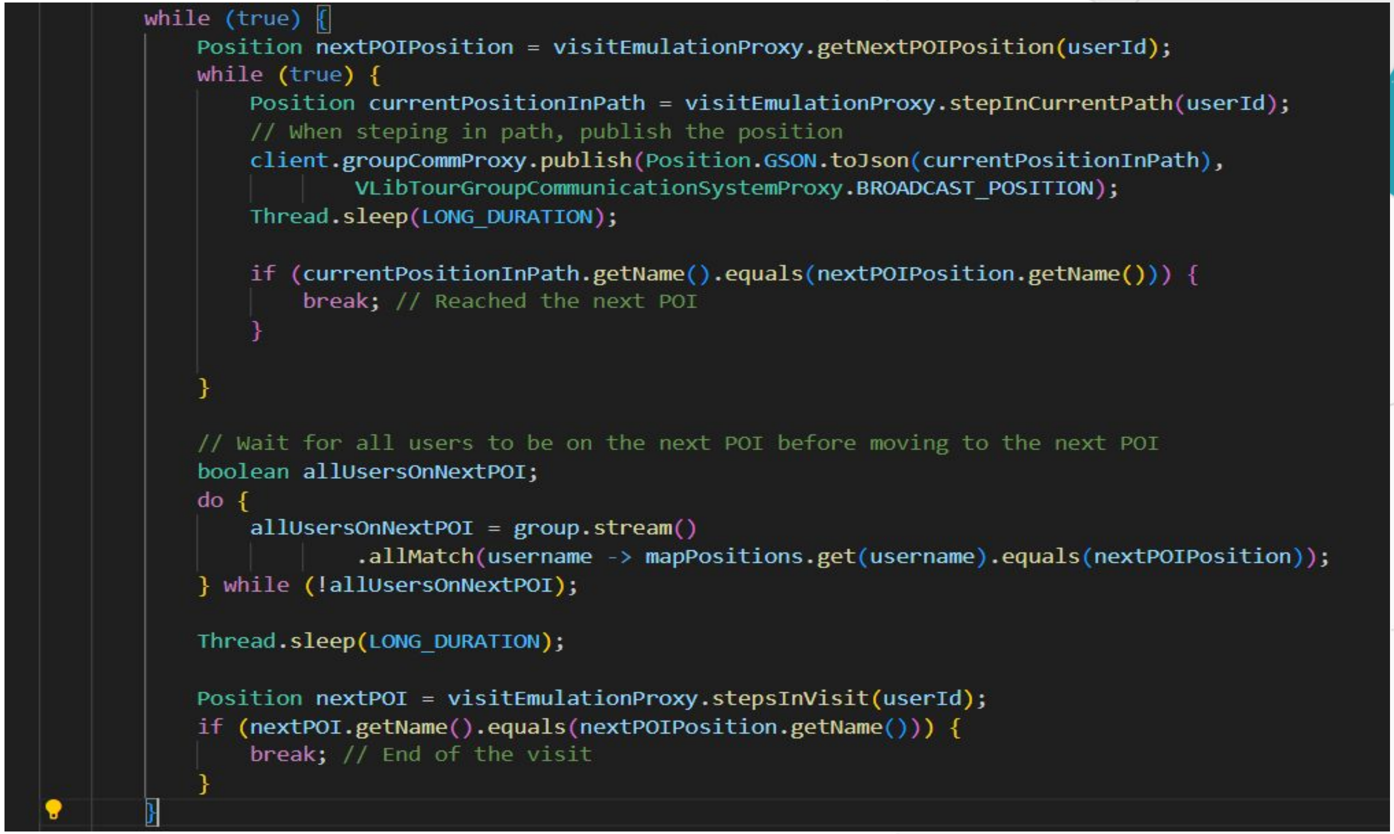
LobbyRoom : Proxy



```
2 references
public VLibTourLobbyRoomProxy() throws IOException, JsonRpcException, TimeoutException, InterruptedException {
    ConnectionFactory factory = new ConnectionFactory();
    factory.setHost(host:"localhost");
    while (connection == null) {
        try {
            connection = factory.newConnection();
        } catch (IOException e) {
            Thread.sleep(5000);
        }
    }
    connection = factory.newConnection();
    channel = connection.createChannel();
    jsonRpcClient = new JsonRpcClient(channel, VLibTourLobbyService.EXCHANGE_NAME,
            VLibTourLobbyService.BINDING_KEY);
    service = jsonRpcClient.createProxy(klass:VLibTourLobbyService.class);
}
```

Carte dynamique & attente des touristes

Carte dynamique & attente des touristes



```java
while (true) {
    Position nextPOIPosition = visitEmulationProxy.getNextPOIPosition(userId);
    while (true) {
        Position currentPositionInPath = visitEmulationProxy.stepInCurrentPath(userId);
        // When steping in path, publish the position
        client.groupCommProxy.publish(Position.GSON.toJson(currentPositionInPath),
                VLibTourGroupCommunicationSystemProxy.BROADCAST_POSITION);
        Thread.sleep(LONG_DURATION);

        if (currentPositionInPath.getName().equals(nextPOIPosition.getName())) {
            break; // Reached the next POI
        }

    }

    // Wait for all users to be on the next POI before moving to the next POI
    boolean allUsersOnNextPOI;
    do {
        allUsersOnNextPOI = group.stream()
                .allMatch(username -> mapPositions.get(username).equals(nextPOIPosition));
    } while (!allUsersOnNextPOI);

    Thread.sleep(LONG_DURATION);

    Position nextPOI = visitEmulationProxy.stepsInVisit(userId);
    if (nextPOI.getName().equals(nextPOIPosition.getName())) {
        break; // End of the visit
    }
}
```

Démonstration

Résumé des fonctionnalités

- ManagementBean, ManagementAdmin
- EmulationServer (REST),
- CommunicationSystem (RabbitMQ)
- LobbyRoom
- Affichage dynamique de la carte

Samuel GUILLEMET, Nathan FERET
10 Novembre 2023

CSC 5001: Middleware and software architecture for distributed applications

Merci