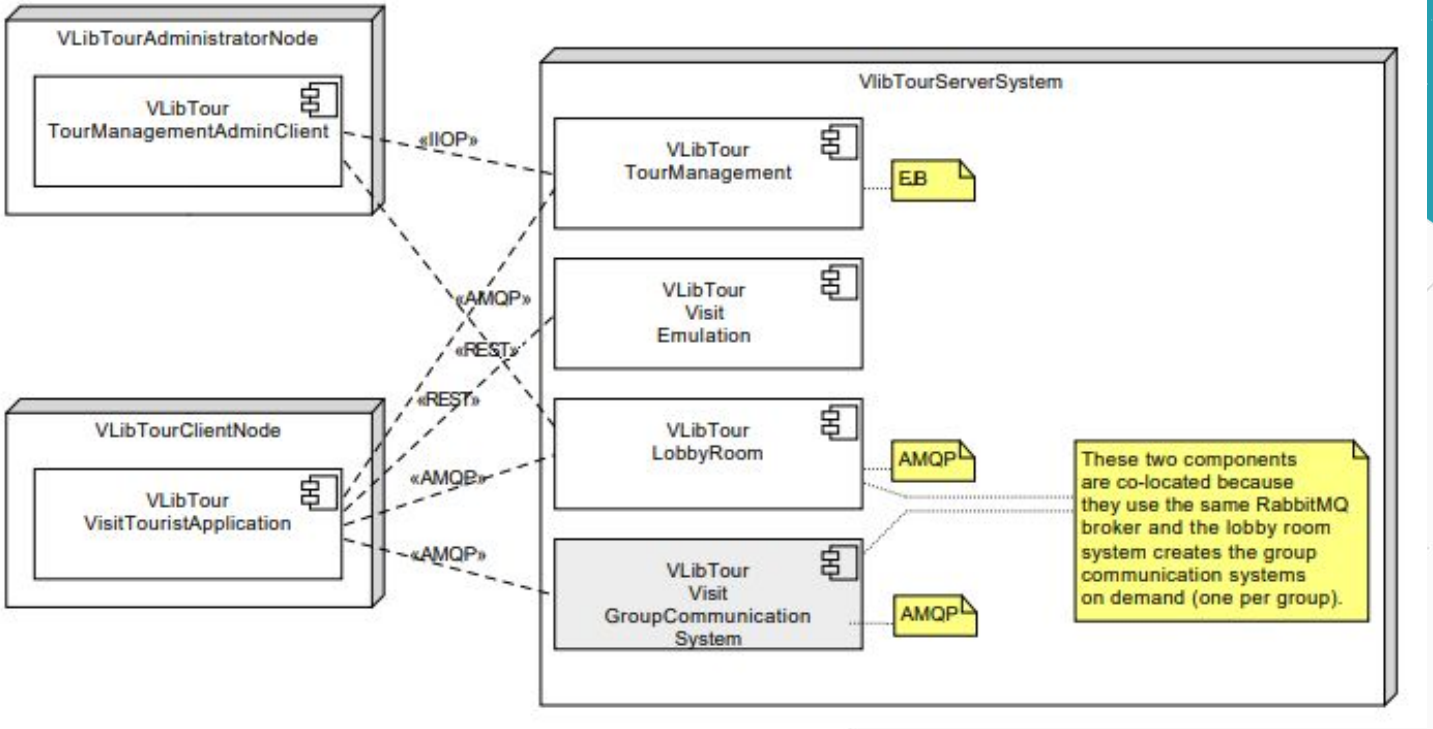
Samuel GUILLEMET, Nathan FERET
10 Novembre 2023

CSC 5002

Middleware and software architecture for distributed applications

Architecture de l'application



Architecture de l'application

Chaque touriste instancie chaque proxy dans son application

L'initiateur crée le groupe de communication et gère l'affichage des points pour chaque touriste

TourManagement

Responsable de la gestion de différents objets en base de données tels que les POIs ou les Tours.

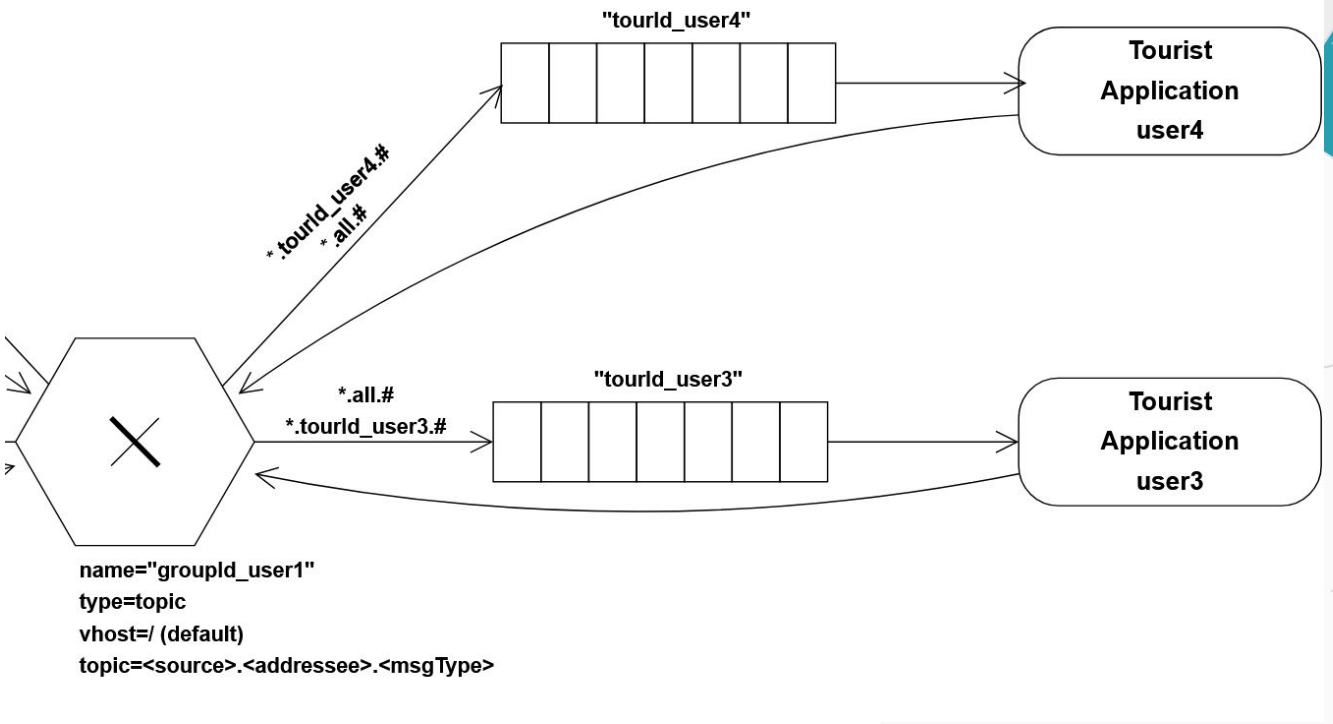
Un proxy appelle les méthodes de VlibTourTourManagementBean via un context lookup

VisitEmulation

Responsable du serveur REST

Envoie des informations telle que la position, le prochain POI …

Group Communication System



LobbyRoom

Le lobbyRoomProxy est uniquement utilisé pour la création du groupCommProxy.
Il est responsable de la connectionFactory

```
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);
}
```

Carte dynamique & attente des touristes

```
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
    }
}
```

Samuel GUILLEMET, Nathan FERET
10 Novembre 2023

CSC 5001: Middleware and software architecture for distributed applications

Merci