M1 Info – IDS - Chat avec RabbitMQ

Julien ALAIMO - Olivier HUREAU

22/03/2020

1 Introduction

Nous avons repris la même interface de l'application crée avec le tp rmiserver en l'implémentant avec RabbitMQ. On garde alors plusieurs principe d'utilisation pour notre chat :

- Une configuration de la connextion (ip, nom d'utilisateur)
- Plusieurs salles de chat

Il faut dans un premier temps lancer rabbitMQ. la solution la plus simple est de lancer RabbitMQ sous docker :

```
docker run -it --rm --name rabbitmq -p 5672:5672 -p 15672:15672 rabbitmq:3-management
```

Un .jar est fournis pour lancer l'application.

2 RabbitMQ : Topics et exchange

Lorsqu'un client se connecte à une salle de chat, il bind la queue avec une routing key correspondant au nom de la salle de chat. Soit le routing key :

```
chat.<nom de la salle de chat>
```

Alors, quand l'on envoie un message, on publie sur l'exchange principale "CHAT" avec la routing key correspondan à la salle de chat.

Il suffiras alors d'unbind la queue lors de la déconnexion de la salle.

Voici un schéma synthétisant notre implémentation.

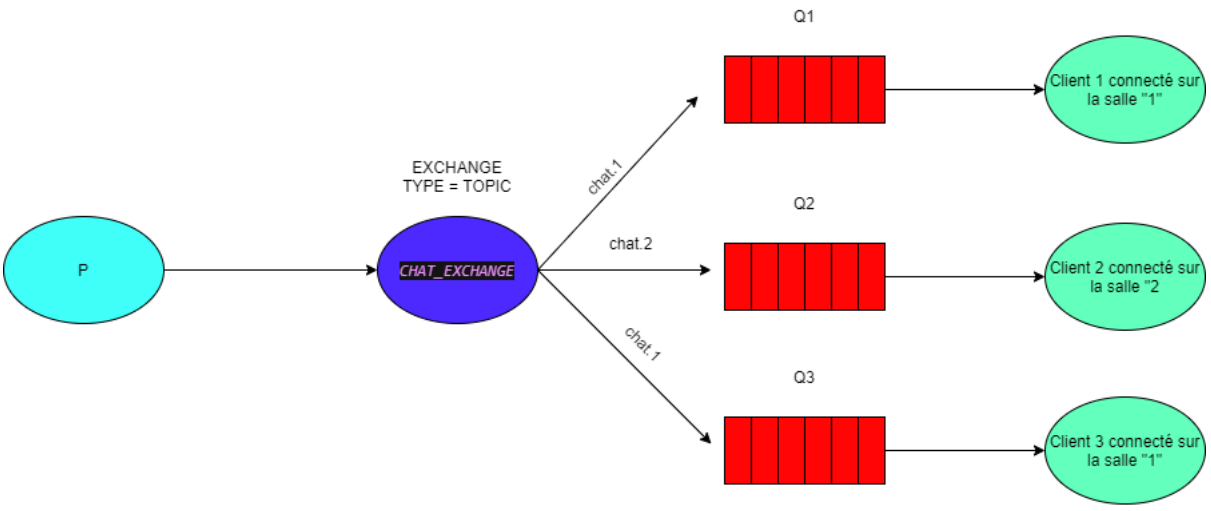

Figure 1: Implémenation du chat avec RabbitMQ

3 Architecture serveur

3.1 Décentralisé

RabbitMQ porpose un service de messenging décentralisé. De ce fait, nous avons choisi une architecture en concordance. Il n'y a pas de serveur (a part le borker de rabbitmq). Chaque client se connecte aux broker est peuvent echanger des messages dans differentes salles de chat.

3.2 Possiblité d'implémentation en centralisé

Nous avons voulu implémenter des fonctionalités basiques de chat tel que le log-in ou encore l'historisation. Malheureusement, ces fonctionnalités demandent des obtions que rabbitMQ n'offre pas.

Dans l'idée il aurait été possible d'implémenter une persistance et un system de log-in avec une application qui écoute en permanence sur les différents topics et qui enregistre les diferents messages:

A chaque connection d'un nouveau client, celui-ci envoi une demande à l'application sur un topic spécifique pour que celle-ci lui renvoie l'historique ainsi que la confirmation que l'utilisateur est connu par le système.

4 Conclusion

En se qui concerne la partie envoi et recpetion de message pur, effectivement, cela est beaucoup plus simple à mettre en oeuvre qu'avec RMI, que se soit en therme de compléxité ou encore en nombre de lignes de code. Par ailleur, il est plus difficile d'implémenter les focntionnalité basique d'un chat qui demande d'avoir un systeme centralisé.

D'aprés ce TP, nous pensons que rabbitMQ est destiné plus à des applications tel que la communication entre objets IOT.