PolyMessages — Sprint 2

Eri Agnese, Marvin Bontemps, Lucas Nouguier

01 Mai 2022

Contents

1 Protocole de communication

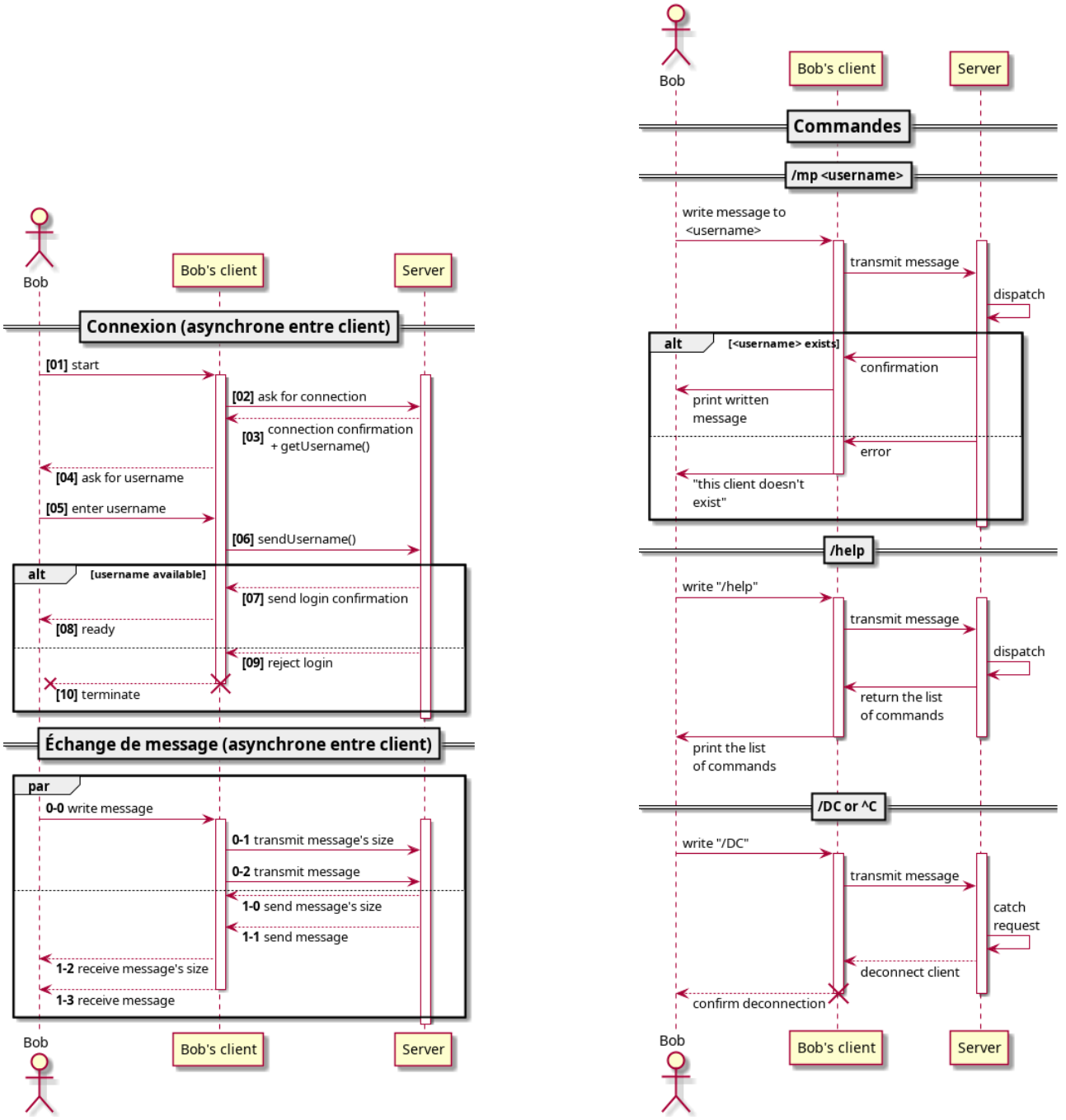

Figure 1: *Protocole de communication clients/serveur*

2 Architecture

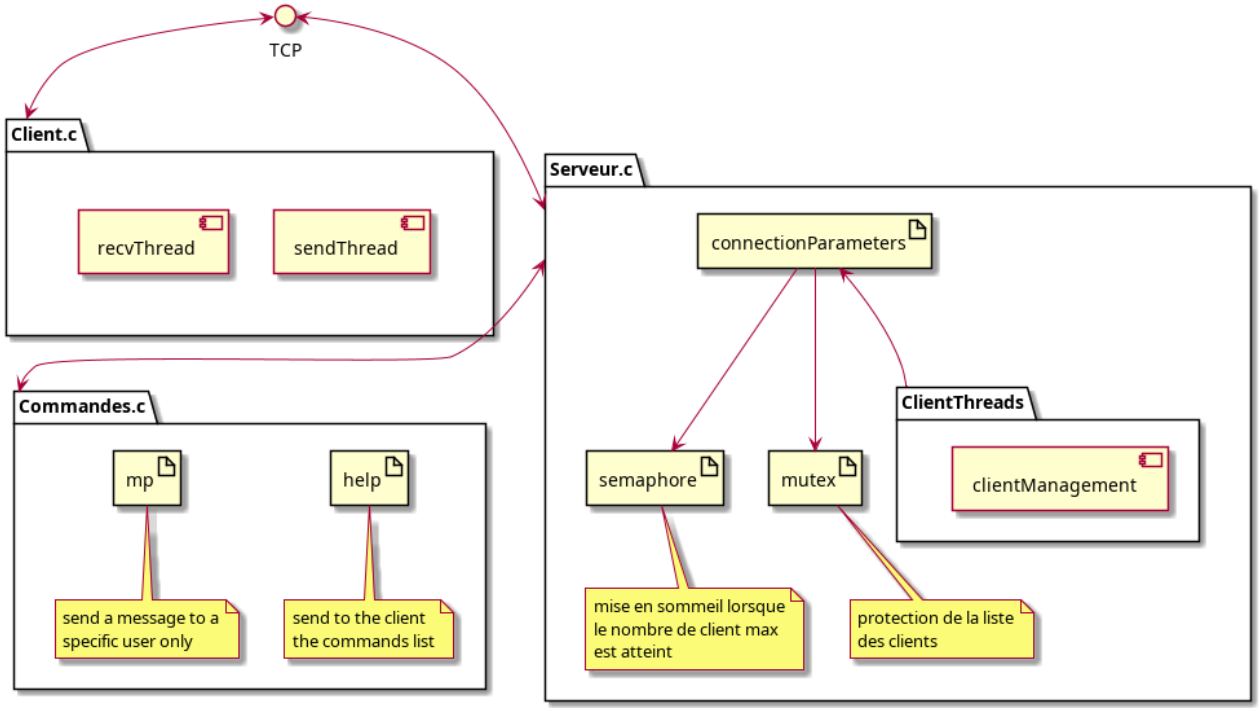

Figure 2: *Architecture de la messagerie*

3 Répartition du travail

La 2ème version de la messagerie consistait à avoir une gestion multiclient. Il fallait pouvoir gérer la connexion/deconnexion de chaque client indépendamment des autres clients. Comme il y avait plusieurs clients, on a dû créer des variables globales qu'il a fallu protéger via des *mutex* et *sémaphores*. Une fonctionnalité de message privée entre 2 utilisateurs a également été mise en place (voir Table. 1 & Fig. 3). Cette fois-ci, l'ensemble des fonctionnalités a été développé en asynchrone.

Tâches	Étudiants		
	Lucas	Éri	Marvin
Mutliclient	X		
MP			X
Help		X	X
Protection des données		X	

Table 1: *Tâche effecutée par étudiant*

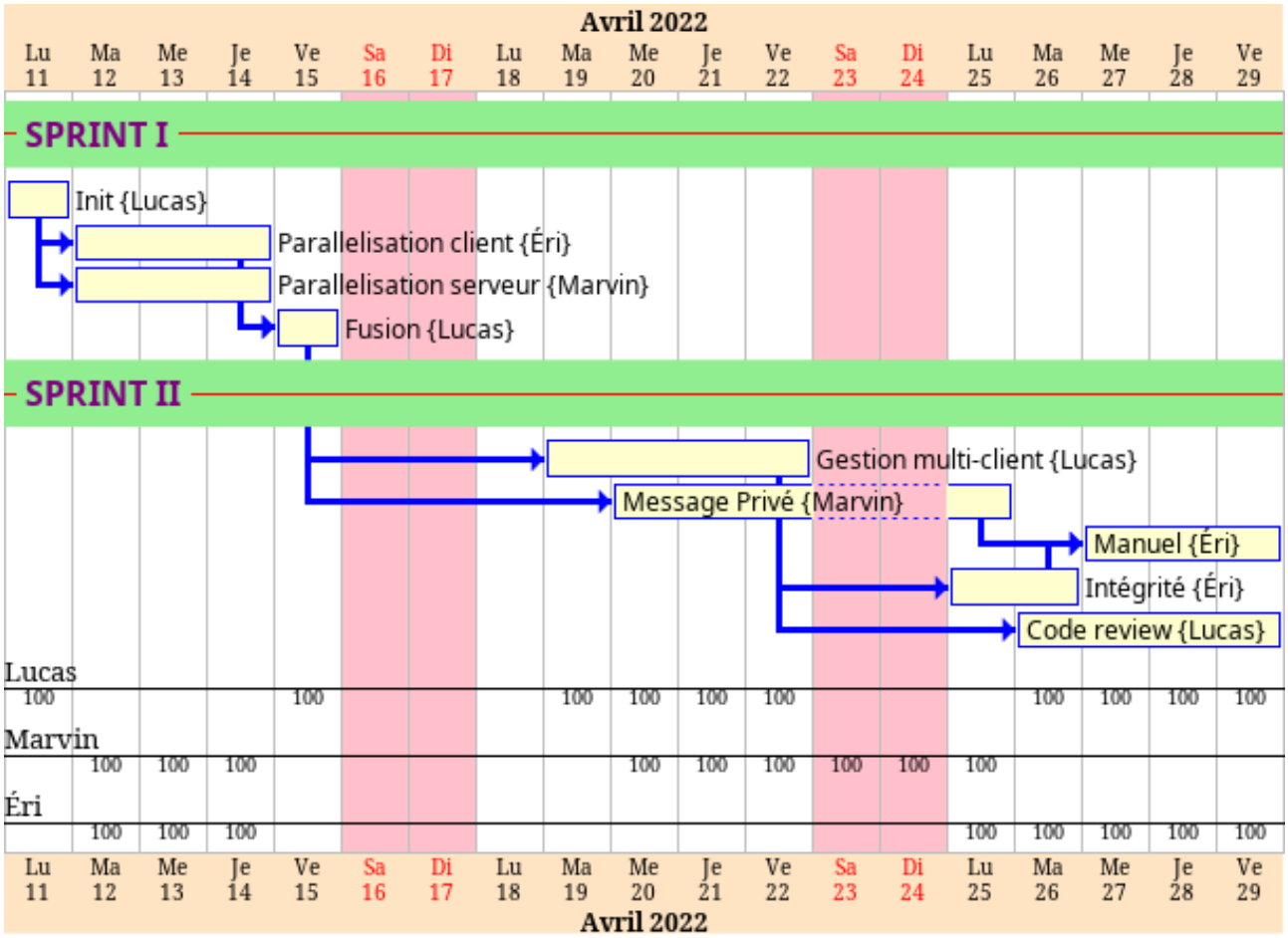

Figure 3: *Diagramme de Gantt sur la réalisation du projet*

4 Exécution du code

Pour lancer la messagerie, il faut commencer par compiler et lancer le serveur

```
[lucas@xps-lucas ~]$ gcc -o server serveur.c commandes.c
[lucas@xps-lucas ~]$ ./server <PORT>
```

On peut alors lancer les clients

```
[lucas@xps-lucas ~]$ gcc -o client client.c
[lucas@xps-lucas ~]$ ./client <IP> <PORT>
```

5 Difficultés

- Changement des structures de données pour les rendre plus facile d'utilisation & de compréhension
- Fin des threads résolus via un *exit(0)* lorsqu'un *receive()* recevait une chaîne particulière