RE203 - Projet de Réseau

Rapport Intermédiaire

Maxime Bellier Jean-Michaël Celerier
Julien Chaumont Bazire Houssin Sylvain Vaglica

Groupe 3

23/04/2013

ENSEIRB
MATMECA
BORDEAUX

Introduction

L'objectif du projet est de réaliser un programme capable de simuler un réseau. Il se base sur deux composantes principales : les routeurs et le contrôleur. Le résultat final devra se présenter sous la forme de deux programmes distincts, car ceux-ci ne communiqueront que par le réseau, de sorte que leur code source ne doit pas influencer. Pour ce faire, il a été imposé de développer en utilisant deux langages distincts, le langage C, ainsi qu'un langage orienté objet. D'un commun accord au sein du groupe, c'est le langage C++ qui a été choisi pour coder les routeurs, tandis que le C servira à la réalisation du contrôleur.

Contrôleur et routeurs, ainsi que l'état d'avancement de leur implémentation, seront détaillés dans la suite de ce premier rapport.

1 Le contrôleur

1.1 Son rôle

Le contrôleur, comme son nom l'indique, a pour mission du superviser l'intégralité du réseau. C'est à lui de l'initialiser, à partir d'une topologie donnée par l'utilisateur, et de maintenir la communication avec les routeurs pour mettre à jour ses informations et les transmettre.

Il faut cependant noter que son rôle est purement théorique, puisque dans la plupart des réseaux réels, ce sont aux routeurs de communiquer entre eux pour mettre à jour leurs connaissances sur le réseau. Dans le cadre du projet, cette mise à jour des connaissances des routeurs se fait justement auprès du contrôleur qui, lui, doit avoir la connaissance de la totalité du réseau. Pour ce faire, les routeurs se connectant ou se déconnectant du réseau doivent se signaler auprès du contrôleur avant d'effectuer l'action.

1.2 Structure

Plusieurs modules distincts composent le contrôleur, chacun ayant ses propres besoins en matière de structures de données :

- une invite de commande pour communiquer avec l'utilisateur et afficher les résultats des différentes requêtes entre lui et les routeurs ;
- la gestion du réseau sous forme d'un graphe initialisé à partir d'un fichier au format *dot* [1] ;
- un moyen de communiquer via le réseau avec les routeurs.

La communication sur le réseau se fait à l'aide de la bibliothèque C `sys/socket.h`, complète et largement utilisée. La gestion des graphes, quant à elle, utilise la bibliothèque C `graphviz/graph.h` fournie avec l'ensemble d'outils *Graphviz*. Le principal avantage est qu'elle permet très facilement le passage d'un fichier au format *it* à une structure de graphe, et inversement.

Plus bas niveau, afin d'améliorer la complexité générale du programme, des tables de hachage sont utilisées. A cet effet, la bibliothèque C standard ne contenant pas de telle structure de données, l'implémentation utilisée se base sur une bibliothque C tierce publiée sous licence GPL [2].

1.3 Etat actuel de l'avancement

Actuellement, les différentes parties du contrôleur ont été implémentées de manière séparée :

- L'invite de commande est opérationnelle : reconnaissance des différentes commandes possibles à l'aide d'expressions régulières et échappements des caractères ;

1. http ://www.graphviz.org/content/dot-language
2. Le code source est consultable à cette adresse : `https://github.com/ankurs/Hash-Table/`

- La gestion des graphes est mise en place : création à partir d'un fichier *dot*, ajout/retrait/-modification d'arête ou de nœud, sauvegarde dans un fichier ;
- Une bibliothèque de gestion de sockets utilisant les fonctions de *sys/socket.h* est prête à l'utilisation. Elle permet la connexion, la déconnexion, l'envoi de messages, . . .Son implémentation est générique et permet une programmation évènementielle via des pointeurs de fonction correspondants aux actions à effectuer lors d'une connexion, de la réception d'un message, etc.

Il ne reste donc qu'à *lier* les différentes composantes entre elles pour obtenir un contrôleur respectant les attentes.

2 Les routeurs

2.1 Leur rôle

Les routeurs du projet simulent les routeurs d'un réseau réels. La principale différence est la connexion permanente qu'ils doivent maintenir avec le contrôleur. Leurs connaissances sur leur partie du réseau sont régulièrement actualisées par un appel de mise à jour au contrôleur. Parallèlement, ils sont connectés à leurs voisins et maintiennent une table de routage interne.

2.2 Structure

De même que le contrôleur, le programme du routeur se décompose en plusieurs parties distinctes :

- une invite de commande pour afficher le résultat des échanges avec le contrôleur et les autres routeurs, ainsi que pour communiquer avec l'utilisateur ;
- la gestion de leur table de routage ;
- un moyen de communiquer via le réseau avec le contrôleur et les routeurs voisins.

Pour des raisons d'optimisation en complexité, la table de routage est implémentée à l'aide de la classe `map` de la bibliothèque C++ standard.

2.3 Etat actuel de l'avancement

L'avancement du programme du routeur est semblable à celui du contrôleur : les différents modules évoqués dans la partie précédente sont opérationnels mais ne sont pas encore connectés entre eux ; à l'exception des algorithmes de maintenance de la table de routage qui n'ont pas encore été écrits. Il ne reste donc plus qu'à effectuer cette liaison pour arriver à un routeur respectant les règles attendues.

Sauf retard, il est également prévu de consacrer une séance à la réalisation de tests de recette.

2.4 Routeur ↔ Routeur

Après avoir reçu sa liste de voisins par le contrôleur, un routeur qui vient de rejoindre le réseau doit tout d'abord se signaler auprès de l'ensemble de ses voisins et établit avec eux une connexion permanente. De façon régulière, il doit échanger avec ses voisins ses vecteurs distances, et reçoit également les vecteurs distances de ces derniers. Ces nouvelles données lui permettent de mettre à jour sa table de routage si besoin est.

Un routeur doit également communiquer avec un voisin lorsqu'il doit transmettre un paquet en passant par ce voisin. Pour savoir à quel voisin transmettre le paquet, il lui suffit de regarder le destinataire du message et sa table de routage, et de le transmettre au bon routeur.

Quel que soit le type d'échange, le récepteur envoie systématiquement un message d'aquittement au routeur émetteur.

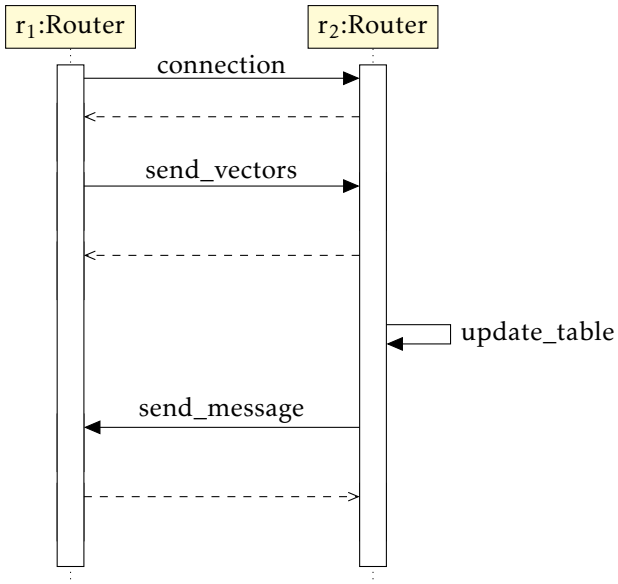

Figure 1 – Diagramme de séquence des communications entre deux routeurs

3 Nos objectifs pour la suite du projet

Le déroulement prévisionnel du projet est indiqué sur le diagramme de Gantt donné en figure 2.

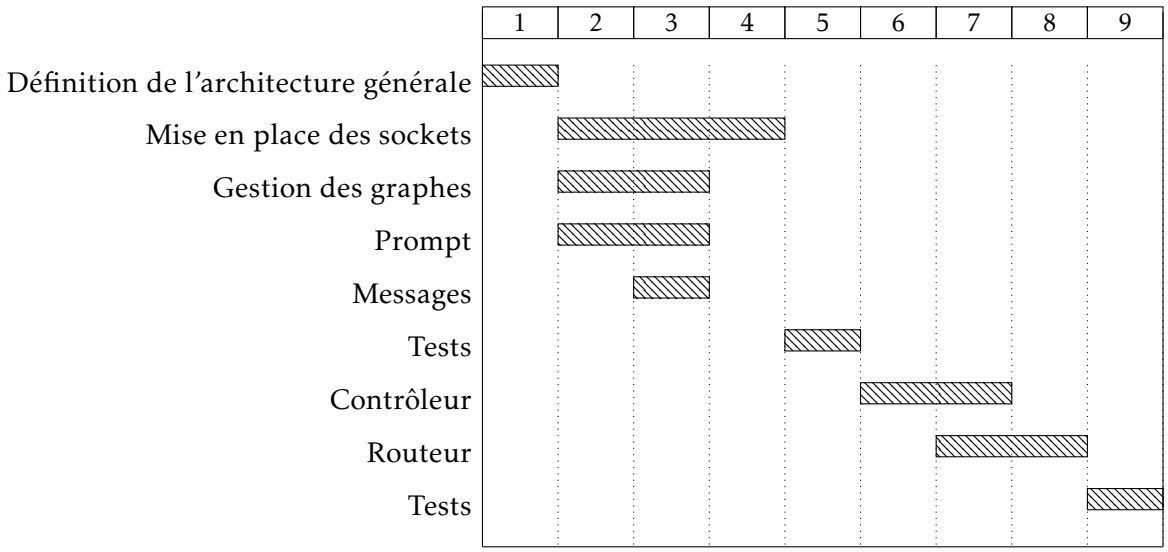

Figure 2 – Diagramme de Gantt pour le projet