Diffusion

Notre C diffusion s'appuie sur notre classe **CBroadcastManager**.

Question 1

Cet algorithme est à implémenter sur les serveurs seulement. Il permet de faire passer les évenements qu'un serveur cherche à avertir les autres. Les clients reçoivent le dit évènement par l'intermédiaire de leur serveur si ils sont concernés.

Question 2

La ligne ** (p = q) ** signifie que ce serveur est à l'origine de ce message. Nous l'acceptons directement.

La ligne **v[q] = vp[q] + 1)** permet de vérifier que c'est bien le message suivant pour le noeud initiateur.

La ligne **v[k] <= vp[k]** permet de vérifier que nous n'avons pas besoin de délivrer des messages d'autres noeuds avant celui ci.

Question 3

Le vecteur d'orloge de chaque machine est contenu dans le CBroadcastManager. Il est transmis dans le champ **NeededData** des messages interserveurs.

Note : Le vecteur d'orloge est réinitialisé à chaque changement de topologie. La logique aurait été de les resynchroniser via un Echo, puis de les mettre à jour sur l'ensemble des noeuds à l'aide d'une diffusion fiable mais le temps me manque, et compte tenu de nos hypothèses de changement de topologie, ce scénario est largement acceptable.

Question 4

Nous avons deux autres structures de données :

- **messageBag** : collection de messages non encore C acceptés
- **cAcceptedMessages** : collection ordonée de messages c acceptés. L'idée est ici de respecter l'interface BroadcastManager, dont la méthode process input indique si au moins un message a été accepté. Il est de la responsabilité de l'utilisateur, en cas de message(s) C acceptés de les lire via la méthode **ArrayList getCAcceptedMessages()**

Les nouveaux messages sont passés via la méthode **processInput**.

Question 5

Les lignes 5 - 7 de l'algorithme correspondent à la méthode **public void launchBroadcast(InterServerMessage);**

Question 6

Les lignes 8 - 13 de l'algorithme correspondent à la méthode **public Boolean manageInput(InterServerMessage message)**

Question 7

Dans mon cas il n'y a malheureusement pas encore de méthodes implémenter pour celà. Devrais être implémenté prochainement.

Une double boucle, par exemple un huit, devrais aider à observer des non respects de l'ordre causal. Par exemple :