Programmation répartie

HEIG-VD / PRR / Labo 2

Temps à disposition : 10 périodes
Travail à réaliser par groupe de 2 étudiants.
La présence aux labos est obligatoire. En cas d'absences répétées, l'étudiant sera pénalisé.
En cas de copie manifeste entre les rendus de deux labos, tous les étudiants des deux groupes se verront affecter la note de 1.
Distribué le mercredi 6 novembre 2019 à 14h55.
A rendre le mercredi 4 décembre 2019 à 13h15. *Une démonstration pourra être demandée.*

Forme du rendu : impression des fichiers sources à remettre en séance, archive du projet par email à l'assistant et à l'enseignant. Readme indiquant comment l'utiliser et précisant ce qui a été réalisé, ce qui fonctionne et ce qui reste à faire.

Objectifs

- Comprendre le fonctionnement d'un algorithme d'exclusion mutuelle réparti.
- Utiliser l'algorithme de Carvalho et Roucairol pour maintenir la cohérence d'une variable répartie sur un ensemble de processus.
- Réaliser des communications TCP en langage Go

Critères d'évaluation

La note du labo sera constituée de 2 notes au demi-point prêt, chacune étant comprise entre 0 et 2,5 points : une pour le bon fonctionnement du programme et une pour la qualité du rendu. La note du labo (sur 6 points) sera obtenue par : 1 + la somme de ces 2 notes.

Le bon fonctionnement du programme est obtenu s'il couvre toutes les fonctionnalités de l'énoncé du labo, s'exécute de façon performante et ne présente pas de bugs.

La qualité du rendu sera évaluée en tenant compte des points suivants :

- Facilité et rapidité de la mise en œuvre du programme (activation, aide, paramétrage), en particulier si on utilise un seul PC pour le tester (options par défaut bien choisies).
- Facilité et rapidité de la vérification de son fonctionnement : traces éventuellement paramétrables, datées et signées, mais en tout cas claires et adéquates.
- Possibilités de paramétrage pour simuler des conditions réseau réelles (uniquement les délais de transmission car le réseau est supposé sans pannes) dans les limites de l'énoncé.
- Réalisation de tests automatisés avec simulation de processus ou d'une partie de l'application (mocking).
- Conception du code source (structure et décomposition). Possibilité de réutilisation d'une partie du code avec un autre énoncé (autre couche de transport réseau, …). Cependant, il ne doit pas faire plus que ce qui est demandé, ni être prévu pour des extensions hypothétiques.
- Qualité, simplicité, concision et lisibilité du code source. Conformité au format de code source et aux bonnes pratiques préconisées pour le langage.
- Documentation des entêtes de modules et de fonctions. Commentaires adéquats des instructions : un commentaire trivial ou inutile est aussi pénalisant que l'absence d'un commentaire nécessaire à la compréhension.
- Lisibilité du code imprimé : pagination adéquate (fonction sur une seule page ou bien découpée), entêtes et fin de pages, titres des modules et fonctions apparents, police adéquate et indentation bien choisie (pas de retour ligne automatique).

Énoncé du problème

Partager une donnée parmi un ensemble de processus est un problème qui peut se résoudre par le biais d'un algorithme d'exclusion mutuelle. Dans ce laboratoire, nous allons utiliser l'algorithme de Carvalho et Roucairol, une optimisation de l'algorithme de Ricart et Agrawala, comme algorithme d'exclusion mutuelle.

Chaque processus détient une variable entière qui doit être cohérente. Les tâches applicatives peuvent faire 2 opérations sur cette variable : consulter sa valeur, et modifier sa valeur. La consultation revient à obtenir la valeur la plus récente. Par contre, une modification se passe en 3 étapes :

1. obtenir l'exclusion mutuelle sur la variable ;
2. modifier la valeur en section critique, par exemple l'incrémenter ;
3. informer tous les autres processus de la nouvelle valeur ;
4. libérer la section critique.

Travail à faire

Chaque processus de l'environnement exécute une application qui intègre la partie applicative (gestion de la variable partagée et interface utilisateur) et la partie gestionnaire de l'exclusion mutuelle en mode réparti. Il sera possible :

1. d'afficher la valeur de la variable partagée obtenue (opération de consultation);
2. d'obtenir la permission de modifier la variable partagée, puis de réaliser cette modification par une saisie au clavier.

La première opération n'est jamais bloquante et pourra être effectuée par toutes les tâches applicatives qui ne sont, ni en attente d'une exclusion mutuelle, ni en cours de modifier la variable partagée. C'est uniquement la seconde opération qui nécessite l'obtention de l'exclusion mutuelle.

Afin de vérifier le fonctionnement de votre implémentation, vous devez afficher la valeur de la variable partagée en section critique avant sa modification.

Avant libération de la section critique, la nouvelle valeur de la variable sera transmise aux autres processus formant l'environnement.

Hypothèses

1. La configuration du réseau (nombre de processus, leurs numéros, adresses et numéros de port) est connue avant le démarrage du 1er processus et partagée par tous les processus. Chaque processus est démarré avec son numéro en paramètre. On suppose qu'on les démarrera tous, chacun avec un numéro différent en paramètre.
2. Ces processus et le réseau qui les interconnecte sont entièrement fiables.
3. La communication entre les différents processus se réalisera uniquement par TCP.
4. Il n'y a qu'une seule variable globale à partager sur l'ensemble des processus.
5. Tous les messages échangés devront être le plus petit que possible.
6. Vous devez attendre que l'ensemble des processus soient prêt avant d'accepter les demandes applicatives.