INF2610
TP 2 : Communication interprocessus et synchronisation

Polytechnique Montréal
Auteurs : Antoine Bourassa et Ann-Sophie St-Amand
Automne 2025

Pondération : 15%
Dates de remise :

Groupe 01L	10 novembre à 23:59
Groupe 02L	17 novembre à 23:59
Groupe 03L	11 novembre à 23:59

Présentation

Ce travail pratique a pour but de vous familiariser avec les mécanismes de communication interprocessus et de synchronisation dans les systèmes d'exploitation, en utilisant les appels système de la norme POSIX. Vous explorerez différentes primitives permettant à plusieurs processus ou threads d'échanger des informations et de coordonner leur exécution. La

communication interprocessus et la synchronisation jouent un rôle essentiel dans l'informatique moderne, où plusieurs tâches doivent s'exécuter en parallèle tout en évitant les conflits d'accès aux ressources. Par exemple, dans une application serveur, plusieurs processus ou threads traitent simultanément les requêtes des clients. Sans synchronisation, deux requêtes pourraient accéder à la même donnée en mémoire et créer des incohérences. De même, sans mécanismes de communication, il serait impossible de transmettre efficacement des informations entre ces tâches concurrentes.
Les systèmes d'exploitation fournissent donc des outils permettant de coordonner ces exécutions : les sémaphores assurent un accès exclusif aux ressources partagées, les mémoires partagées permettent un échange rapide de données, et les tubes de communication offrent une communication structurée entre processus. Dans ce travail pratique, vous appliquerez ces concepts dans un contexte de producteurs-consommateurs.

Prise en charge du laboratoire

Il y a deux options pour préparer l'environnement de travail pour le laboratoire :

1. Utiliser les ordinateurs de laboratoires de Polytechnique.
2. Télécharger et installer VMware.

VMware est un logiciel qui fait la gestion de machines virtuelles (création, copie, suppression, etc.). Il fonctionne sur Windows et sur Mac. Il sera utile tout au long du baccalauréat et pour la suite. Le point négatif est qu'il prend beaucoup de ressources sur un ordinateur. Ceci étant dit, voici les instructions pour télécharger VMware :
`https://www.polymtl.ca/gigl/guides-informatiques#vmware`

La prochaine étape consiste à créer la machine virtuelle en utilisant le fichier fourni sur Moodle dans la section TP1. Une fois la machine virtuelle créée, tu trouveras le dossier du travail pratique directement sur le bureau. Pour te connecter à la machine, utilise le compte suivant :
Nom d'utilisateur : user
Mot de passe : user

Si vous avez un processeur d'ordinateur ARM, votre meilleur option est de télécharger une VM par vous même à partir d'un fichier ISO. La VM utilisé pour les TPs est lUbuntu. La version desktop ne se télécharge pas directement en iso. Si vous voulez le même OS (lUbuntu) il vous suffie de télécharger la version server et ensuite télécharger la version desktop à partir du terminal. Vous pouvez très bien utiliser un autre OS. Tant que make et git sont installer sur la VM vous pouvez faire le TP.

Il est conseillé de lire l'énoncé en entier avant de commencer le TP. Il n'est pas demandé de traiter les erreurs éventuelles liées aux appels système. Par contre, si besoin est, à chaque fois que votre programme effectue un appel système (directement ou via une fonction de librairie), vous avez la possibilité d'afficher un message d'erreur explicite en cas d'échec de cet appel système. Pour ce faire, il vous suffit d'utiliser la fonction `perror` après l'appel système (ou l'appel de fonction de librairie). Consultez sa documentation !

Mise en situation

Un centre de contrôle de vidéosurveillance doit gérer plusieurs caméras disposées sur un site industriel et plusieurs écrans regroupés dans une salle de contrôle. Les caméras génèrent des flux vidéo qu'elles souhaitent transmettre au serveur central. Les écrans affichent en continu les vidéos reçues depuis le serveur. Finalement, le serveur agit comme un intermédiaire entre les caméras et les écrans : il reçoit les flux des caméras, les stocke dans un espace partagé et les distribue aux écrans.

1 Mise en place du système de vidéosurveillance (12 points)

Vous devez implémenter une simulation du système de contrôle de vidéosurveillance. D'abord, les caméras produisent en continu des vidéos puis les envoient au serveur. Utilisez la fonction createVideo() pour générer des vidéos. Par la suite, les caméras, lorsqu'elles sont disponibles, prennent une vidéo du serveur et la diffusent. Utilisez la fonction broadcastVideo() pour diffuser. Cette vidéo est alors supprimé de l'espace mémoire du serveur. Voici quelques contraintes additionnelles :

- Vous devez implémenter le serveur central avec un moniteur de Hoare.
- L'exclusion mutuelle doit être respectée lors de l'accès à la mémoire du serveur principal.
- Le total d'écrans et de caméras doit être supérieur à quatre.

Il est possible que le serveur central tombe en panne. Pour éviter une coupure totale du service, un autre serveur de secours est hébergé et prêt à être sollicité. Si une panne survient, le système au complet est mis en arrêt, puis les données du serveur sont envoyées au serveur secondaire. Voici d'autres contraintes :

- Le serveur secondaire peut être implémenté de la manière de votre choix.
- Les deux serveurs sont sur des processus différents.
- Une fois que le serveur secondaire a terminé de recopier les données du serveur primaire, il lui envoie un message de confirmation pour dire qu'il a terminé.
- Une fois la confirmation reçue, il est possible de procéder à un "reset" total du système. Le serveur principal peut se redémarer en recevant une entrée utilisateur (clavier) : lorsque ce signal est envoyé, le serveur secondaire renvoit les données au serveur principal, puis celui-ci reprend son fonctionnement normal.

Aux fins de cet exercice, la panne devra être simulée à l'aide d'un signal. Dans votre programme, vous devez lancer un signal après un certain nombre de secondes aléatoire. Les deux premières lignes du main sont à cet effet. Le traitement du signal devra être codé par vous-même.
Puisqu'il y a plusieurs solutions à ce TP, vous devez commenter votre code pour faciliter notre lecture. C'est aussi une bonne manière pour vous de valider votre compréhension.

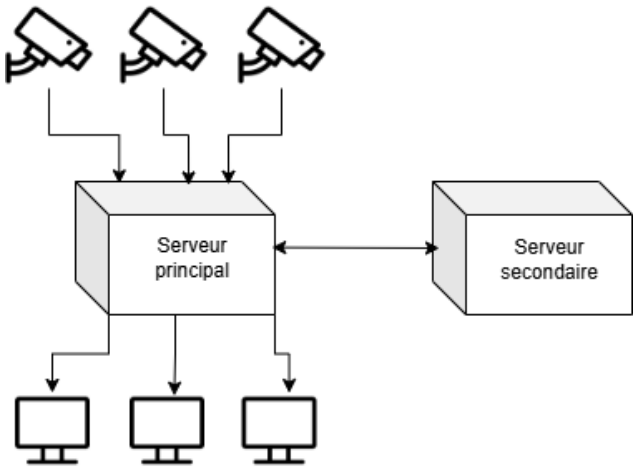

FIGURE 1 – Interactions du système

Question 1

Quel problème de synchronisation avez-vous rencontré dans cet exercice ? Expliquez le mécanisme mis en place pour le régler (0.5 point).

Question 2

Expliquez, dans votre implémentation, comment vous avez fait pour respecter les quatre conditions de Coffman (0.5 point).

Question 3

Quelle est la différence entre un moniteur de Hoare et un sémaphore ? Dans ce contexte, pourquoi un moniteur est-il est bon choix pour gérer la mémoire du serveur central (0.5 point) ?

Question 4

Comment garantir que les données copiées du serveur principal vers le serveur secondaire ne soient pas perdues ou corrompues lors du transfert (0.5 point) ?

2 Intégrité des données lors d'une panne (6 points)

Si une panne survient alors qu'une vidéo est en cours de traitement (copie ou diffusion), il est possible de perdre ces données. Il pourrait aussi y avoir une incohérence des données si une vidéo est partiellement écrite en mémoire.

Dans les vrais systèmes, on essaie d'éviter les pertes de données en gardant toujours une copie de secours. Par exemple, on peut enregistrer les opérations importantes dans un journal (log) pour pouvoir reprendre là où on s'était arrêté si une panne arrive. Aussi, les échanges de données passent souvent par des protocoles fiables qui vérifient que l'information est bien arrivée avant de l'effacer du côté de l'expéditeur. Grâce à ces précautions, même si un problème survient, le système peut redémarrer sans perdre ce qui avait déjà été fait.

On vous demande d'améliorer votre système pour empêcher la perte de données lors de la diffusion par les caméras. Si une panne survient lors de la lecture d'une vidéo par une caméra, le contenu ne doit pas être perdu, c'est-à-dire qu'on doit être en mesure de continuer ou de reprendre le visionnement lorsque la panne est résolue.

Barème

Important : Un programme qui ne compile pas entraine automatiquement une perte de la moitié des points de code attribués à cette section.

Votre rapport est aussi à remettre sur Git.

Section 1	Code	10
	Questions	2
		/12
Section 2		/6
Qualité du code		/2
Total		/20