HEIG-VD HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Chargés de cours : Florian Vaussard, Rick Wertenbroek

Laboratoire de Programmation Concurrente

semestre automne 2022

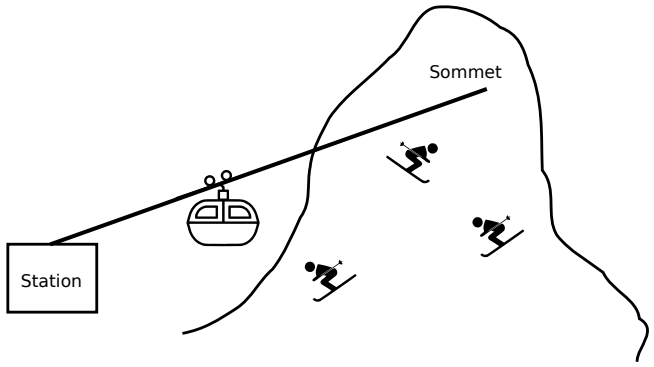
Synchronisation de threads avec des moniteurs de type Mesa

Temps à disposition : 6 périodes (trois séances de laboratoire)

1 Objectifs pédagogiques

— Mettre en évidence les problèmes liés à la synchronisation de tâches ;
— Résoudre un problème de synchronisation avec des moniteurs de type Mesa [1].

2 Cahier des charges



Dans ce laboratoire vous allez modéliser des skieurs et une télécabine. Le système est formé d'un ensemble de N threads skieurs et d'un thread télécabine. La télécabine effectue des trajets entre la station et le sommet de la montagne. Lorsqu'elle se trouve à la station, elle va récupérer des skieurs, et lorsqu'elle arrive au sommet de la montagne, elle va les décharger pour qu'ils puissent partir skier.

Les skieurs commencent en bas de la montagne et attendent que la télécabine arrive. Quand la télécabine arrive, uniquement les skieurs qui attendent déjà sur la télécabine peuvent monter. Les skieurs qui arrivent après l'arrivée de la télécabine doivent attendre le prochain voyage. La télécabine a une capacité de M places et s'il y a plus de M skieurs qui attendent la télécabine, certains devront attendre le prochain voyage. Si aucun skieur n'est présent à la station, la télécabine ne s'arrête pas et remonte. Les skieurs qui peuvent monter dans la télécabine doivent attendre dans celle-ci jusqu'à ce qu'elle arrive au sommet de la montagne. Le respect est une denrée rare dans cette station, de fait, il n'est pas rare de voir certains skieurs passer devant les autres dans la file d'attente du télécabine. Un avantage certains pour l'écriture du code, qui n'implique donc pas de gérer l'attente avec une FIFO pour entrer dans le télécabine !

Tous les skieurs descendent de la télécabine au sommet. Une fois sortis, les skieurs descendent en ski avant de revenir à la station pour attendre à nouveau la télécabine. Jamais les skieurs ne descendent par la télécabine, elle redescendra donc toujours vide et aucun skieur n'attendra non plus la télécabine au sommet.

1. Mesa est un langage de programmation innovant développé à la fin des années 70 avec de nombreuses nouveautés pour l'époque, par exemple ; les exceptions, la synchronisation des threads, ainsi que la compilation incrémentale. Il est considéré comme l'un des ancêtres de Java. `https://en.wikipedia.org/wiki/Mesa_(programming_language)`

3 Travail à faire

Complétez la fonction `run()` dans `cableCarBehavior.cpp` et `skierBehavior.cpp` afin de modéliser le comportement des skieurs et de la télécabine. Plusieurs prototypes de fonctions sont fournis pour vous aider à modéliser le comportement ainsi que les interactions entre les skieurs et la télécabine. Lisez la description de ces fonctions dans les fichiers `cablecarinterface.h` et `cablecarskierinterface.h` avant de commencer pour vous en faire une idée.

Modifiez la classe `PcoCableCar` dans `pcocablecar.h/.cpp` afin de gérer la synchronisation entre les skieurs et la télécabine. La synchronisation entre les tâches doit être réalisée en utilisant des moniteurs de type Mesa.

La tâche `run()` lancée dans un thread représente le comportement des acteurs du système, skieurs et télécabine. Pour les skieurs, ils ont tous un comportement identique mais un délai aléatoire dans la fonction `slideSlope()` permet de modéliser le fait que tous ne skient pas de la même manière. Les skieurs ont un comportement cyclique : ils prennent la télécabine pour arriver au sommet puis descendent de la montagne à ski, ceci jusqu'à ce que le service soit terminé (fermeture en fin de journée). Durant le trajet dans la télécabine les tâches skieurs doivent être inactives et attendre d'arriver à destination.

Pour l'unique télécabine, elle ne fait que des aller-retours entre la station et le sommet de la montagne. elle charge les skieurs en attente à la station et les décharge au sommet. Une fois le service terminé, la télécabine finit sont trajet actuel. Si elle monte, elle arrive au sommet, décharge les skieurs et redescend à la station pour s'arrêter. Si elle est déjà en train de descendre, elle finit son trajet et s'arrête.

Note : Il faut faire attention que la télécabine ne parte pas avant que les skieurs ne soient montés et qu'elle ne redescende pas avant que les skieurs à l'intérieur ne soient descendus.

Lorsque la fin du service est annoncé, signalé par l'appel à la fonction `endService()` de la télécabine depuis la fonction `Station::launchStation()`, il faut que les skieurs rentrent, c'est à dire qu'ils arrêtent d'attendre la télécabine si ils étaient en attente, ou qu'ils descendent de la montagne (si encore dans le télécabine ou au sommet). Le fait de rentrer et symbolisé par le retour de la fonction `run()`. La télécabine elle aussi doit terminer son service comme décrit ci-dessus et terminer en bas à la station avant qu'elle ne quitte sa fonction `run()`.

4 Consignes

— Ne pas créer de nouveaux fichiers.

— Modifiez judicieusement les fichiers `pcocablecar.h` et `pcocablecar.cpp` afin de gérer la synchronisation des tâches.

— Remplissez la fonction `run()` dans `cablecarbehavior.cpp` et `skierbehavior.cpp`.

— Remplissez l'en-tête de ces fichiers avec vos noms.

— Vous pouvez changer le nombre de skieurs N et la capacité du télécabine M dans `main.cpp` avec les constantes `NB_SKIERS` et `CABLE_CAR_CAPACITY`.

— Il n'est pas nécessaire de modifier les autres fichiers.

— La description de l'implémentation, la manière dont vous avez vérifié son fonctionnement et toute autre information pertinente doivent figurer dans le fichier `rapport.pdf`.

— Inspirez-vous du barème de correction pour connaître là où il faut mettre votre effort.

— Vous devez travailler en équipe de deux personnes.

— Rendez votre code selon la procédure décrite dans les consignes de laboratoire.

5 Barème de correction

Conception	40%
Exécution et fonctionnement	15%
Documentation et analyse	25%
Commentaires au niveau du code	10%
Robustesse et traitement de la terminaison	10%