HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Prof. Yann Thoma
Gregory Medwed
Fiorenzo Gamba

Laboratoire de Programmation Concurrente
semestre printemps 2020

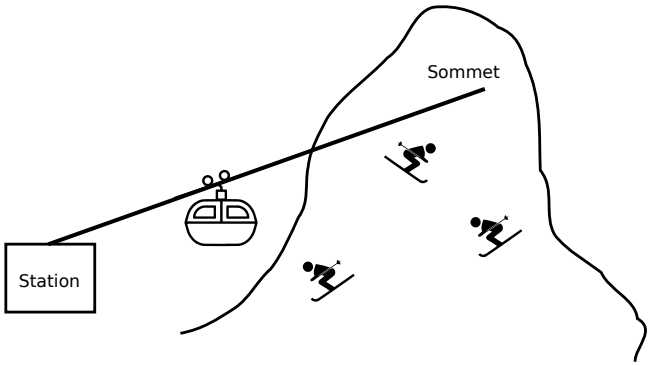
Synchronisation de threads avec semaphores

Temps à disposition : 6 périodes

1 Objectifs pédagogiques

— Mettre en évidence les problèmes liés à la synchronisation de tâches ;
— Modéliser un problème de synchronisation par sémaphores.

2 Cahier des charges



Dans ce laboratoire vous allez modéliser des skieurs et un télécabine. Le système est formé d'un ensemble de N threads skieurs et d'un thread télécabine. Le télécabine effectue des trajets entre la station et le sommet de la montagne. Les skieurs commençent en bas de la montagne est attendent que le télécabine arrive. Quand le télécabine arrive, uniquement les skieurs qui attendent déjà sur le télécabine peuvent monter. Les skieurs qui arrivent après l'arrivée du télécabine doivent attendre le prochain voyage. Le télécabine a une capacité de M places et s'il y a plus de M skieurs qui attendent le télécabine, certains devront attendre le prochain voyage. Si aucun skieur n'est présent à la station, le télécabine ne s'arrête pas et remonte. Tous les skieurs descendent du télécabine au sommet. Une fois sortis, les skieurs descendent en ski avant de revenir à la station pour attendre à nouveau le télécabine. Jamais les skieurs ne descendent avec le télécabine, il redescendra donc toujours vide et aucun skieur n'attendra non plus le télécabine au sommet.

3 Travail à faire

Complétez la fonction `run()` dans `cableCarBehavior.cpp` et `skierBehavior.cpp` afin de modéliser le comportement des skieurs et du télécabine. Plusieurs prototypes de fonctions sont fournis pour vous aider à modéliser le comportement et l'interaction entre les skieurs et le 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 le télécabine. La synchronisation entre les tâches doit être réalisée en utilisant uniquement des sémaphores (`PcoSemaphore`).

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 `goDownTheMountain()` permet de modéliser le fait que tous ne skient pas de la même manière. Les skieurs ont un comportement cyclique : ils prennent le 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 le télécabine les tâches skieurs doivent être inactives et attendre d'arriver à destination.

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

Note : Il faut faire attention que le télécabine ne parte pas avant que les skieurs ne soient montés et qu'il 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()` du télécabine depuis la fonction `main()`, il faut que les skieurs rentrent, c'est à dire qu'ils arrêtent d'attendre le 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()`. Le télécabine lui aussi doit terminer son service comme décrit ci-dessus et terminer en bas à la station avant qu'il ne quitte sa fonction `run()`.

4 Consignes

- Ne pas créer de nouveaux fichiers.
- Modifiez judicieusement les fichiers `pcocablecar.h` et `pcocableclar.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	25%
Exécution et fonctionnement	10%
Codage	15%
Documentation et analyse	25%
Commentaires au niveau du code	10%
Robustesse et traitement de la terminaison	15%