TP 3 (à rendre)

1 Introduction

Une grande agglomération portuaire souhaite rénover son port de commerce. À cette fin, elle conduit des études pour vérifier si les infrastructures prévues sont dimensionnées correctement pour accueillir tous les trafic de marchandises présents et à venir. C'est la raison pour laquelle elle vous charge de concevoir un programme de simulation.

Le port à simuler est constitué de n quais pouvant accueillir chacun un navire porte-conteneurs. Chaque quai dispose d'une grue pour décharger les conteneurs sur les camions.

Lorsqu'un navire est en approche du port, il demande à la capitainerie le numéro du quai où s'amarrer. Si aucun quai n'est disponible, il doit patienter au large jusqu'à ce qu'un autre navire quitte le port. Lorsqu'il est à quai, les camions arrivent pour récupérer les conteneurs débarqués du navire par la grue et les emmener. On supposera dans cette simulation qu'on ne fait que décharger les navires et que ceux-ci repartent à vide. Lorsque tous les conteneurs sont déchargés, le navire peut quitter le port.

On souhaite modéliser ce fonctionnement sous forme de processus distincts communiquant via un ensemble d'objets (mémoire partagée et sémaphores) IPC System V.

2 Travail à réaliser

On demande de réaliser 6 programmes (mono-processus, mono-threadés).

— `pcap` n
Le programme `pcap` est central car il simule la capitainerie du port. Il est lancé au démarrage de la simulation avec en paramètre le nombre de quais ($n > 0$) dans le port. Il crée les objets IPC System V nécessaires au fonctionnement de la simulation et reçoit les requêtes des navires et des camions. Il ne s'arrête que lorsque le programme `pstop` le signale.

— `pstop`
Le programme `pstop` demande au programme `pcap` de terminer la simulation : `pcap` doit alors attendre que tous les navires aient terminé de décharger et aient quitté le port. Les navires en attente pour rentrer dans le port doivent être informés de la fermeture : ils doivent dès lors s'en aller pour chercher un autre port. Lorsque tous les navires sont sortis du port, le programme `pcap` peut alors supprimer les objets IPC System V qu'il a créés. Le programme `pstop` se contente d'envoyer une requête à la capitainerie, il n'a pas besoin d'attendre la terminaison du système.

— `pnav` v c t_a t_d
Le programme `pnav` simule un navire. Il admet 4 arguments : le nom v du navire, le nombre c de conteneurs à décharger (on suppose que tous les conteneurs sont identiques) et deux temps t_a et t_d. Ce programme démarre la simulation en supposant que le navire est en dehors du port : il doit demander à la capitainerie l'autorisation de rentrer dans le port et le numéro du quai où accoster. L'accostage prend un certain temps (fonction du navire et de la dextérité de son capitaine) que l'on fixera à t_a. Lorsque le navire est enfin à quai, il attend que les différents camions lui demandent les conteneurs (un seul à la fois puisqu'il n'y a qu'une seule grue). Chaque conteneur prend un temps t_d à décharger du navire. Lorsque les c conteneurs sont débarqués, le navire peut quitter le port et reprendre sa route (i.e. se terminer).

— `pfcam` v c t_c
Le programme `pfcam` simule une flotte de camions. Il prend en argument le nom v du navire à décharger, le nombre c de camions à simuler et un temps t_c. Le programme interroge la capitainerie pour connaître le quai où est amarré le navire. Si celui-ci n'est pas à quai, la capitainerie signale le problème et la flotte repart (i.e. le programme se termine avec un code non nul). Lorsqu'un camion arrive sur le quai approprié, il demande un conteneur au navire, le charge sur le camion en un temps t_c et quitte le port. Lorsque les c camions de la flotte ont chargé leur conteneur, le programme se termine.

— `pdump`
Le programme `pdump` affiche différentes informations sur l'état actuel du système : entre autres, le

nombre de quais, et pour chaque quai, le nom du navire qui y est amarré, nombre de conteneurs, etc. Ces informations sont extraites à un instant donné, il n'est pas requis qu'elles soient cohérentes (par exemple si un navire est en train de quitter le port ou si un conteneur est en cours de chargement sur un camion).

— `pclean`
Enfin, le programme `pclean` supprime les objets IPC System V s'ils existent. Il s'agit d'une aide au cas où `pcap` ne se terminerait pas proprement.

On prendra quelques hypothèses simplificatrices :

— Seul le premier caractère du nom d'un navire est significatif, autrement dit les navires « roi des mers » et « reine des neiges » correspondent tous deux au navire de nom « r ».
— On supposera que les noms des navires sont tous différents, c'est-à-dire qu'on ignorera le cas où deux navires portent le même nom.
— Seul un navire peut être en phase d'accostage dans le port à un instant donné.
— Les temps sont tous indiqués en millisecondes, et peuvent être nuls pour accélérer la simulation.
— On ignorera le cas où il y a plus de camions que de conteneurs à décharger d'un navire.

3 Rapport

Vous rédigerez un rapport (au format PDF obligatoirement) comprenant la description des synchronisations entre les différents acteurs, ainsi que la justification des informations placées en mémoire partagée.

4 Implémentation

Le code de retour de vos programmes devra indiquer le succès ou l'échec de l'action associée. En cas d'erreur, vous adopterez la stratégie simple de terminer l'exécution du programme concerné avec un message explicite et un code de retour adéquat.

On vous demande de mettre en place un système d'affichage de l'état de vos programmes sous forme de messages de débogage contrôlés par la variable d'environnement `DEBUG_PORT`. Si celle-ci existe, elle doit contenir un entier. Si elle n'existe pas, ou si la valeur est 0, vos programmes doivent être muets (sauf pour les erreurs, bien évidemment). Si la valeur égale 1, les programmes doivent afficher un court message lors des étapes importantes. Avec des valeurs supérieures, vos programmes doivent afficher des informations de débogage plus complètes, notamment le détail des synchronisations. N'oubliez pas d'utiliser `fflush` pour avoir ces messages dès leur affichage lorsque la sortie est redirigée dans un fichier.

Votre implémentation ne doit pas comporter d'attente active. Vous devez utiliser la mémoire partagée et les sémaphores IPC System V comme uniques outils de synchronisation entre les différents programmes. L'utilisation d'autres dispositifs de communication entre processus, comme par exemple les signaux, est exclue.

Un ensemble de jeux de tests est à votre disposition. Vous ne devez en aucun cas le modifier, mais vous pouvez le compléter avec de nouveaux scripts. La machine `turing` est la machine de référence : votre programme doit y compiler avec les options `-Wall -Wextra -Werror -g` à l'exclusion de toute autre option, et les jeux de tests seront exécutés sur cette même machine.

5 Modalités de remise

Vous déposerez votre TP, avec votre rapport, sous forme d'une archive (au format « tar.gz », sur Moodle, dans l'espace de devoirs prévu à cet effet. Vous supprimerez les fichiers binaires et autres fichiers de log.

Le travail à réaliser est individuel. On rappelle que la copie ou le plagiat sont sévèrement sanctionnés.