LO41 - Architecture et utilisation des systèmes d'exploitation
Université de Technologie de Belfort-Montbéliard

Nascar, rapport de projet

Paul Locatelli et Pierre Rognon

Automne 2012

Sommaire

Introduction

L'unité de valeur LO41 a pour but de familiariser les étudiants aux différents mécanismes d'un système d'exploitation. De nombreuses notions ont donc été vues lors de ce semestre. La plupart ont été mises en pratique dans le langage C, qu'on peut qualifier d'universel dans le cadre d'un système d'exploitation tel que Solaris, système auquel nous nous sommes rapportés tout au long de cette U.V. C'est pour illustrer et mettre en pratique les différents mécanismes vus les uns avec les autres qu'un projet a été proposé. Ce projet a pour contexte les courses américaines bien connues de Nascar. Ces courses mettent en jeu des voitures durant des courses longues voire très longues. L'objectif de ce projet est donc de modéliser une course de Nascar en s'appuyant sur ce qui a été appris tout au long du semestre. Une simulation doit pouvoir être proposée en collant le plus possible à la réalité de ces courses.
Pour mener à bien ce projet, un cahier des charges a dû être mis en œuvre. Ce dernier a permis par la suite le développement d'un programme répondant aux besoins énoncés. C'est ces deux points qui seront abordés durant ce rapport suivi d'un bref bilan du déroulement de ce projet.

1 Cahier des charges

Le cahier des charges a inclut naturellement plusieurs parties au début du projet. En effet, une première partie doit décrire les besoins concernant les étapes d'avant développement, puis une seconde partie doit indiquer le cahier des charges fonctionnel de l'application, c'est-à-dire celui concernant le développement "pur". Nous aborderons donc tout d'abord les contraintes concernant l'étude du sujet et la conception relative à ce dernier puis nous décrirons le cahier des charges fonctionnel.

1.1 Étude du sujet

L'étude du sujet doit pour être menée à bien se diviser en étapes bien distinctes. Les besoins pour cette partie sont donc :

- d'effectuer des recherches sur le sujet qu'est la course de Nascar ;
- d'identifier les entités importantes concernant ces courses ;
- de bien comprendre les différentes règles de ce sport automobile.

1.2 Conception

La conception en elle-même demande plusieurs choses :

- tout d'abord, il doit être mis en place une arborescence bien précise du projet afin que celui-ci soit le plus clair possible ;
- ensuite, un choix doit être fait sur les différentes notions vu durant le semestre : toutes les notions ne sont pas forcément utilisées mais celles dont on a besoin doivent être reliées à une entité mise en avant dans l'étude du sujet ;
- enfin, les différentes interactions entre les entités doivent aussi être modélisées et être rapportées à une problématique bien précise du cours.

Suite à la mise en œuvre de cette étape de conception, un cahier des charges a été établi.

1.3 Application

Le cahier des charges fonctionnel de l'application énonce des besoins généraux et d'autres plus précis, relatifs à l'utilisation même du logiciel.

1.3.1 Besoins généraux

Ces besoins sont :
- l'application doit pouvoir simuler une course de Nascar ;
- cette simulation doit comprendre une séance de qualifications ainsi que la course proprement dite ;
- elle doit gérer les équipes et les voitures qui courent dans la course ;
- elle doit gérer des incidents pouvant être provoqués par l'utilisateur ;
- un système de stand doit aussi être géré.

1.3.2 Besoins spécifiques

Les besoins spécifiques se rapportent directement au fonctionnement de l'application dans son déroulement.

Début de la course Lors du lancement de l'application, on doit attendre le feu vert de l'utilisateur pour lancer la séance de qualifications.
Un second feu vert doit être donné lors du lancement réel de la course. De plus, l'utilisateur doit être informé du classement à l'issue de la séance de qualifications.

Durant la course L'application doit modéliser un affichage sommaire du circuit comprenant plusieurs informations :
- le placement des voitures sur le circuit : les voitures sont modélisées par leur numéro d'équipe et leur numéro de voiture dans l'équipe ;
- un affichage concernant les stands et la voiture éventuellement arrêtée à celui-ci ;
- si une voiture a subi un incident, elle doit être mise en évidence par sa couleur.

L'application doit aussi permettre par une combinaison de touches la mise en pause de la course en cours. L'appui sur ces touches doit impliquer l'apparition d'un menu et d'informations.
- Les informations sont le classement de la course, ainsi que le niveau de carburant et le nombre de tours effectués par chaque voiture ;
- le menu quant à lui doit permettre de : quitter l'application, revenir à la course, intégrer un incident léger ou intégrer un accident grave.

Une autre combinaison de touches, celle par défaut du système d'exploitation doit permettre de quitter l'application proprement. L'appui sur une combinaison de touche doit informer l'utilisateur de ce qu'elle entraîne :

- pour la mise en pause de la course,
- pour la fin de l'application, un message doit indiquer que l'on quitte celle-ci et dire si la fin s'est bien déroulée.

En fin de course, l'application doit afficher un classement.

2 Mise en œuvre

La mise en œuvre a été menée en deux parties principales. La première a consisté à étudier le sujet, la seconde à développer l'application en elle-même.

2.1 Étude du sujet

Lors de la rédaction du cahier des charges, l'étude du sujet devait permettre de faire diverses recherches sur les courses de Nascar et d'en ressortir les différentes entités importantes. Il en est donc ressorti plusieurs entités principales :

- le circuit, qui est le support même de l'application. Il permet d'y placer les voitures et de les faire évoluer tout au long de la course ;
- les équipes, qui sont constituées chacune de deux voitures ;
- les voitures, qui sont le support de différents paramètres de la voiture tels que le niveau d'essence ;
- les sections, qui sont en fait des parties de circuit. Un circuit comporte alors 100 sections et chaque voiture passe d'une section à l'autre ;
- les directeurs de course, qui s'occupe de la gestion des évènements ;
- les stands gèrent l'arrivée des voitures pour se ravitailler.

Les différentes situations problèmatiques qui ont été mises en avant sont :

- quand un voiture veut accéder à une section et que deux sont déjà présentes dans celle-ci ;
- quand deux voitures veulent aller dans le même temps dans une même section et qu'il y a déjà une voiture dans celle-ci ;
- quand une voiture veut accéder à un stand et qu'une autre est déjà dans celui-ci.

2.2 Développement de l'application

2.2.1 Architecture

Le développement a été largement inspirée de l'étude du sujet. Elle avait pour but de mettre en place une arborescence des fichiers dans le projet et d'expliciter les liens entre ces fichiers. Ainsi, la plupart des entités ont été portées sur un fichier. L'architecture est donc la suivante :

- circuit.c et circuit.h ;
- directeur.c et directeur.h ;
- equipe.c et equipe.h ;
- section.c et section.h ;
- stand.c et stand.h ;
- voiture.c et voiture.h ;

En plus de ces fichiers ont été mis en place :

- main.c et main.h : ce sont les fichiers principaux du jeu ;
- menu.c et menu.h : s'occupe de l'affichage d'un menu lorsque le jeu est mis en pause après l'envoi d'un signal par l'utilisateur ;
- semaphore.c et semaphore.h : permet d'utiliser les sémaphores ;
- system.c et system.h : s'occupe de plusieurs choses comme par exemple l'affichage ou le choix aléatoire d'une entité.

Un Makefile est aussi mis en place pour une compilation plus aisée lors du développement du projet.

2.2.2 Agencement des informations

Afin de gérer plus facilement les informations relatives aux différents objets manipulés dans l'application, nous avons décidé de mettre en place des structures. Les structures Voiture, Circuit, Stand, Equipe, Section, Directeur, Evenement ont donc été crées. La structure Evenement est contenue dans le fichier directeur.h.
Pour utiliser plus facilement les structures, une approche orientée objet a été effectuée. Des fonctions d'initialisation ont donc été mises en places pour créer les différentes structures.

2.2.3 Vie des entités

En parallèle des structures, il a été décidé d'utiliser des threads pour matérialiser la "vie des entités". Ce choix permet ensuite de modéliser les situations d'interblocage en utilisant des mutex.

2.2.4 Description des entités

Circuit Le circuit ne comporte pas de situation compliquée à gérer au niveau de l'implémentation. Il contient les informations sur le nombre de tours, la longueur du circuit et la vitesse maximum des voitures. Il permet seulement l'affichage du classement et du circuit.

Section La section est modélisée par une structure comprenant deux voitures. Ces deux voitures représentent les deux voies du circuit. Chaque section possède un verrou qui permet de bloquer l'arrivée d'une voiture si deux sont déjà présentes sur celle-ci.
Elle dispose de fonctions permettant d'entrer et de sortir une voiture. Ce sont ces fonctions qui gèrent le fonctionnement du mutex.

Stand Chaque stand est représenté par une structure et un thread.
La structure est liée aux voitures de l'équipe à laquelle le stand appartient. Elle possède aussi un paramètre permettant de savoir quelle voiture est attendue au stand et un autre permettant de savoir quelle voiture y est actuellement. Un mutex permet de se verrouiller durant la recherche de la prochaine voiture à attendre.
Le thread du stand effectue la recherche de la voiture à attendre. Une voiture est attribuée dès que l'une des deux voitures de l'équipe a atteint la moitié de sa réserve de carburant.

Voiture Les voitures sont représentées tout comme les stands par une structure et un thread.
La structure contient des informations sur l'essence, la position de la voiture sur le circuit, la vitesse à laquelle elle se déplace, l'équipe et son numéro dans celle-ci et son statut (arrêt, en course, stand, arrivée, panne d'essence, incident mineur, accident grave).
Le thread quant à lui initialise les paramètres de la voiture, puis tant que la voiture peut rouler va tenter de la faire de section en section tout au long du circuit.

Equipe Toutes les équipes ont une structure. Cette structure est composée de deux voitures et d'un numéro qui qualifie l'équipe. L'équipe ne fait pas d'autres traitements. Elle permet simplement de regrouper deux voitures.

Directeur et évènements Les évènements peuvent être de deux types : incident mineur ou accident grave. Ils sont déclenchés par l'utilisateur depuis le menu de pause. Chaque évènement est représenté par une structure et un thread. La structure contient uniquement le thread et le statut de l'évènement (fini ou non).

Le thread est utilisé uniquement lors d'un incident mineur. Il choisit aléatoirement une voiture qui va subir l'incident et l'arrête (passe en statut incident mineur). Il ralentit ensuite les autres voitures à la manière d'une voiture de sécurité. Puis il rétablit la situation normale après un temps tiré aléatoirement.

Le directeur est composé d'un thread et d'une structure contenant la liste de tous les évènements en cours. Durant sa vie, le directeur va vérifier si les évènements se terminent ou non. Si c'est le cas pour un, il le détruit de façon propre.

2.2.5 Déroulement du programme

Le déroulement du programme a lieu dans le fichier main.c.

- tout d'abord, les signaux sont redéfinis pour l'application. On change le traitant des signaux SIGTERM et SIGSTOP (correspondant respectivement aux raccourcis ctrl+c et ctrl+z).
  - Pour SIGTERM, on ordonne l'arrêt propre immédiat de la course. Pour cela, on arrête les threads et on supprime tout ce qui a été crée dynamiquement ;
  - pour SIGSTOP, on met en pause toutes les voitures sur le circuit à l'aide d'un verrou et on lance un menu pour l'utilisateur. Depuis celui-ci, il peut soit arrêter la course, soit la reprendre ou encore envoyer un incident ou accident ;
- il crée les équipes et les voitures pour tout le long de l'application ;
- il crée le circuit et les stands pour la phase de qualification ;
- il crée les threads correspondant aux stands et aux voitures pour cette phase ;
- il exécute ensuite les threads des voitures puis ceux des stands et enfin un autre pour l'affichage (crée en même temps) ;
- le programme se met alors en attente de l'arrivée de toutes les voitures (et donc la fin des threads des voitures) ;
- il termine le thread d'affichage et ceux des stands ;
- il détruit toutes les données temporaires liées au qualifications sauf le classement qui est récupéré pour la grille de départ ;
- le même procédé est mis en place pour la course réelle.

2.2.6 Situations d'interblocage

Les situations d'interblocage concernaient les stands et les sections.

- pour les sections, si une voiture veut accéder à une section déjà remplie, elle libère le verrou associé à la section qui permettait d'accéder aux variables de celle-ci puis se met en attente. Lorsqu'elle termine son attente elle tente à nouveau d'accéder à la section ;

– quand deux voitures veulent accéder en même temps à une section où il reste une seule place, le principe des mutex est utilisé mettant en place une règle de type FIFO. Le premier ayant été exécuté par le sytème accède à la section ; l'autre se met en attente ;
– pour les stands, c'est celui-ci même qui choisit quelle voiture accédera à celui-ci. Pour faire ce choix, il regarde le niveau d'essence des deux voitures de l'équipe et prend celle dont le niveau est le plus critique. Pour qu'une voiture soit choisie, il faut que son niveau d'essence soit en dessous de la moitié du plein. Si aucune voiture n'a besoin d'aller au stand, celui-ci attend.

3 Problèmes rencontrés

Quelques problèmes sont apparus lors du développement de cette application.

Tout d'abord, de nombreuses modifications ont du être faites au début du projet. Les mécanismes utilisés au début se sont en effet révélés impossibles à mettre en place et des modifications dans la conception ont du être faites.
Les processus ont été par exemple utilisés au départ mais devant la difficulté de mettre en place la mémoire partagée selon notre conception était peu aisé.

L'application ayant été implémentée sous des systèmes Linux, nous avons du porter le projet sous Solaris comme il était demandé. Cette opération n'a posé qu'un problème majeur avec l'utilisation des structures. Cependant, cela n'a pas été un problème important lors de projet.

Quelques problèmes au niveau de l'affichage ont été rencontrés, le rafraichissement ne se faisant pas toujours correctement. Cependant, l'affichage n'étant pas le but premier du projet, nous avons préféré nous concentrer sur l'application en elle-même. Par chance, il semblerait que ces problèmes n'aient pas lieu sous Solaris.

Conclusion

Ce projet a permis d'exploiter de nombreuses notions vues lors de ce semestre en LO41. Il nous semble que la plupart des objectifs ont été accomplis même si toutes les fonctions demandées n'ont pas pu être mises en œuvre, faute de temps. L'application est cependant pleinement jouable puisque la course peut se dérouler en entier sans problème.
De plus, le fait d'avoir été dans des groupes de deux nous a permis, ayant des emplois du temps proches, de travailler en duo et de mettre en commun nos idées. Cela nous a aussi aidé à être plus efficaces notamment lors de l'étude du sujet qui demandait une compréhension aisée de tous les membres.

Table des matières