Rapport de projet LO41

Nicolas Ballet

5 janvier 2017

Table des matières

1 Sujet

Mise en situation

Nous devions mettre en place un programme simulant une chaine d'usinage. Un sujet nous a été proposé, mais nous étions libre de faire nos propres choix concernant le fonctionnement de la chaine. Une seule contrainte était à respecter : pas d'attente active.

J'ai donc choisi le processus suivant :

Un produit à usiner est d'un certain type est necessite une table d'usinage du même type afin d'être usinée. Un produit est envoyé par le superviseur au robot d'alimentation, qui le dépose sur le convoyeur, le produit est déposé sur la première table disponible etant du même type que le produit. Le produit met un certain temps à être usiné, ce temps dépend du type du produit. Une fois le produit usiné, il est renvoyé sur le convoyeur, qui le renvoi au robot de retrait.

2 Conception

Architecture et structures de données

Afin de stucturer mon programme j'ai voulu reproduire un fonctionnement orienté objet via des structures, des constructeurs, destructeurs et des accesseurs.

J'ai créé plusieurs 'Classes' :

- Une usine(Factory), possede un superviseur, et ne communique qu'avec lui.
- Un superviseur(Supervisor), connait l'ensemble du système, le robot d'alimentation, le convoyeur, les tables, le robot de retrait, mais aussi les produits.
- Un produit, ne connait rien à part son propre état. Les classes suivantes connaissent toute le superviseur auquel elle sont affectées.
- Un robot d'alimentation(Supplier), connait le convoyeur auquel il doit envoyer les produits.
- Un convoyeur(Conveyor), connait les tables avec qui il echange les produits, et le robot de retrait, afin de renvoyer les produits. Il connait le produit qu'il est en train de transporter.
- Une table(Table), connait le convoyeur auxquel il doit renvoyer le produit dès lors qu'il est usiné.
- Un robot de retrait(Retrait), ne connait rien d'autre que le supervisor.

Les robots de retrait, d'alimentation, le convoyeur, les tables et le superviseur exécute tous leur propre thread.

La synchronisation des threads s'effectue à l'aide de moniteurs.

3 Développement

J'ai donc commencé le développement par mettre en place l'architecture orientée objet, ce qui m'a pris un certain temps afin de touver le compromis idéal. Puis j'ai commencé à développer un système simple afin de passer des objets provenant du robot d'alimention directement au rebot de retrait.

A ce stade, le robot d'alimention possédait une liste de produits à traiter. Cela m'obligeait à réallouer la liste si je voulais ajouter un produit pendant l'execution du programme. J'ai ensuite implémenté les tables, ce qui n'était plus tres compliqué étant donné que la base était stable.

J'ai aussi développer un afficheur qui s'occupe de l'affichage des informations à l'écran.

C'est à partir d'ici que cela ce complique. Lors de la conception, je n'avais pas remarqué le problème d'interblocage qui pouvait se présenter lorsque les tables d'usinages sont toutes occupées et que le robot d'alimentation envoie un produit sur le convoyeur. Je m'étais trop concentré sur les ressources partagées. J'ai donc changé le système d'approvisionnement en cours de route. Le robot d'alimentation ne possède maintenant qu'un seul produit, celui à distribuer dans l'instant. Et c'est au superviseur de s'occuper de l'envoi des produits dans la chaine. Ce qui lui permet de vérifier les disponibilités des machines. Je n'ai malheureusement pas eu le temps de reglé le problème d'interblocage. J'ai pourtant une solution : à chaque fin d'usinage, il faudrait en notifier le superviseur, et lorsque le produit sort de la chaine, si il reste un produit encore non introduit dans la chaine du même type que celui qui est sortit, on l'envoie.

4 Conclusion

J'ai voulu développer trop de fonctionnalités et au final la base du projet n'est pas entièrement stable à cause du problème d'interblocage et la simulation des défaillances n'est pas faite.

Pour finir, j'ai trouvé ce projet intéressant. J'ai aimé le concevoir et le développer. Malgré le fait que j'aurais aimé faire plus, et avoir un système plus modulaire et paramètrable.