Notes de Cours INF 8725

Olivier Sirois

2018-10-10

Contents

Introduction

0.1 Outline

On retrouve les systèmes temps réels un peu partout. toutes les applications critiques contiennent une forme de systèmes/aspects temps réelles.

on retrouve les systèmes temps réelles dans:

- factory - usine
- drone
- AI
- medicale
- automobile
- jeux vidéos

Les systèmes ou les décisions doivent être fait dans un délai temporel sont techniquement considérés temps réelles. Cela s'applique à tous les systèmes rattachés aussi.

Les systèmes temps réeles sont généralement des systèmes qui interagisse avec des environnements externes.

Par nature, ils sont réactif. Mais normalement, on ne cherche pas seulement la bonne réponse, mais on veut l'avoir dans un temps données (spécifications). Ils contiennent aussi beaucoup de capteurs étant donnée qu'ils interagissent avec un environnement externes.

Definition : Deadline = temps ou il faut avoir une réponse (contraintes)

on peut différencier les systèmes temps réelles en deux différentes catégories :

- Hard real time systems : ils doivent réponde sans faute a leur contraintes

- Soft real time systems : la réponse n'est pas aussi critique mais elle est quand même voulus

Definitions : Deterministe = les contraintes déterministe sont exprimés en terms de valeur fixe, i.e. aucune moyenne et autre valeur statistique

Normalement, lorsqu'on ne répond pas a une contraite déterministe, sa équivaut à un systems failure.

exemple : la porte de fermeture d'un train doit toujours marché..

Définitions : Concurrences = trois types:

- True concurrency : plusieurs processeurs en parallels
- pseudo concurrency : Model qui représente logiquement des activités en parallele, sauf qu'il s'execute généralement sur un processeur mono-coeur.
- physical concurrency : lorsqu'on parle d'une environnement qui évolue/la physique.

On peut aussi différencier l'envioronnement externe, dans le sens qu'on exploite normalement le patrons d'arrivés des événements. Comme par exemple un événemet périodique ou apériodique. Les systèmes temps réelles ont généralement aussi un partage de resources assez robustes.
Normalement, l'utilisation des techniques de programmation temps réelles augmente considérable la complexité de la programmation.

Définitions : Correct = le systèmes donne un résultats qui est vraie, (qu'il calcule bien les chose)

Définiciton : Robuste = le systèmes réagit bien à des conditions qui ne sont pas planifier, mêmes en présence de défaillance systèmes non-prévues.

Reliabitility = une mesure de comment souvent le systèmes va défaillé

Fault tolerance = la reconnaissance et la gestion de fautes dans le systèmes en les évitants avec certaines techniques

Criticalités = mesure d'importance d'une défaillance -¿ crash ¿ division par 0

d'autres aspect assz important est la prédictabilité d'un systèmes temps réelles. l'interface human-machines doit être designer de sorte qu'il prévient l'erreur humaine et la confusion.

Évidemment, le design d'un systèmes temps réelles se réalise de la même manière qu'on logiciel traditionelle (waterfall, spiral). On commence avec un énoncés des différents requis, un design préliminaire, on fais ensuite un design détaillé +

réalisations + teest pour le logiciel et matériel avec finalement une intégration et une validation.

On travaille généralement avec plusieurs abstractions. On ne peut pas controler tout les détails de l'environnement en plus que les systèmes d'aujourd'hui sont massif.

INSÉRÉ CLASSIQUE MOORE'S LAW..

Cela fait en sorte qu'on ne peut pas suivre l'amélioration des chips. C'est pour sa qu'on se sert maintenant de languages haut niveau + conception automatique qui nous aide a suivre l'évolution technologique.

Les méthodologies aujourd'hui ont déjà beaucoup d'abstraction pris en compte. C'est méthodologies sont essentiels pour pouvoir réduire le gap de productivités.

Chapter 1

Cours 1 - Analyse de systèmes temps réelles.

1.0.1 Example - Guichet automatique de banque

Normalement, les systeme automatiser de banque offrent les services suivant:

- Distribution d'argent à tous les porteur de cartes.
- Consultation de solde pour les clients
- toutes les transactions sont sécurisé (QoS?)
- possibilité de rechargé le distributeur

les acteurs possible:

- la banque
- le technicien pour la recharge
- le client
- la device

INSÉRÉ PHOTO SYSTEM (ACTEUR)

On peut voit aussi la multiplicité, sa veut dire qu'il peut seulement avoir entre 0 et 1 acteurs qui travaille sur le système. (le système n'est pas multi-user).

On peut aussi ajouter des tableaux pour des cas plus spécifique (un porteur + un client).

Pour ce système, on a certain use-case. Un use-case sers a mettre en évidence une des fonctionnalités/utilisation du système.

On peut énumérer plusieurs cas possible. Le scope des use-case peuvent être variable étant donné les spécifications/requis.

On associe normalement chaque use-case avec un acteur. On peut ensuite les trier/..

INSÉRÉ SLIDE 4 - 6

Normalement, les systèmes sont définis par des contraintes très spécifique pour répondre aux besoin. Les contraintes peuvent avoir des natures complètement différentes. Un exemple de différentes contraintes sont figurés dans le tableau ci-dessous.

INSÉRÉ TABLEAU CONTRAINTES

Le principe est d'arriver avec un document qui illustre le mieux possible les contraintes sur le systèmes pour qu'il puisse répondre au requis.

INSÉRÉ EXEMPLE DE CAS D'UTILISATION

Dans le cas du GAB, on fait l'hypothèse que tous les conditions de fonctionnement pour retirer de l'argent sont respecté.

nota:

- E = erreur
- A = choix (affichage)

INSÉRÉ IMAGE DU GAB