HMIN105M : Principes de la programmation concurrente et répartie

Responsable : Hinde Bouziane (bouziane@lirmm.fr)
Intervenants : H. Bouziane et T. Lorieul

UM - LIRMM

1 Introduction

Organisation

- Cours 13,5h, TD 13,5h, TP 15h
- Les supports de cours/TD-TP sont disponibles sur Moodle
- Contenu global : principes et outils pour la programmation concurrente (multi-threads et multi-processus) et la programmation d'applications distribuées
- Contrôle de connaissances : écrit (coeff. 0.6) + TP (coeff. 0.4)
  - documents non autorisés à l'écrit
  - pas de session 2 pour le TP
  - la présence en TP est vivement encouragée
- Pré-requis : cours système et réseaux de la licence et langage C

Organisation des contrôles de TP (prévisions)

- 2 contrôles de TP
  1. TP noté sur le multithreading : le 22/10/2019
  2. Projet court sur la programmation d'une application client-serveur et IPC
     - en binôme
     - disponibilité du sujet autour du 05/11/2019
     - fin le 01/12/2019 pour une évaluation le 02/12/2019
- Evaluation par les pairs.

Questions

- Que signifie le terme concurrence ?
- A votre avis, pourquoi apprendre la programmation concurrente ?
- Donnez une définition d'une application distribuée.
- Quels sont les moyens de communication que vous connaissez dans le cadre des applications distribuées ?

Quelques définitions

- Concurrence : exécution en même temps (en parallèle) de plusieurs codes sur une même machine
  - plusieurs exécutions simultanées d'un même programme
  - exécution en parallèle de plusieurs programmes/processus
    - indépendants ou non (même application)
  - Exécution d'un programme engendrant plusieurs processus lourds (fork()) ou légers (threads)
- Programme : entité statique (fichier(s) avec un point d'entrée pour l'exécution (main))
- Processus : programme en cours d'exécution (entité dynamique)
- Remarque : une application ou un logiciel peut être composé de plusieurs programmes (exemple ?)

Contenu du cours

- Concurrence et communications intra-système
  1. Activités dans les processus (*processus légers*, ou *threads*), communication et synchronisation entre activités
  2. Outils de communication et de synchronisation inter-processus (IPC) : Files de messages, mémoire partagée et ensembles de sémaphores
- Programmation distribuée : communications distantes (Réseaux)
  1. Mises en oeuvre d'applications client-serveur avec gestion simultanée de plusieurs clients et utilisation des sockets
     - Remarque : mise en oeuvre différente de celle vue en L3.
  2. Mécanismes de communication de plus haut niveau : RPC (Remote Procedure Call)

A retenir

- Application des concepts et outils étudiés en langage C
  - ne signifie pas que ces concepts n'existent pas dans d'autres langages
  - faire la différence entre les propriétés d'un concept et leur implémentation
- Il est primordial de passer par une étape de réflexion et raisonnement avant l'étape d'implémentation
  - définir des algorithmes avant de coder
- les illustrations dans le cours sont des exemples d'utilisation des concepts présentés
  - savoir s'adapter à d'autres contextes

Bibliographie

J.M. Rifflet, J.B. Yunès *Unix, Programmation et communication*, Dunod, 2003

Joëlle Delacroix *Linux, Programmation système et réseau*, 4^{th} edition, Dunod, 2012

Andrew Tanenbaum, *Réseaux*, $5^{ème}$ édition, Pearson Éducation, 2011
Computer Networks, 5^{th} edition.