TD1: Introduction au multithreading

Marc Pérache

6 mars 2009

1 Introduction

1.1 Définitions

Définition 1: ***Processus*** **–** Un processus est une "coquille" dans laquelle le système exécute chaque programme (commande). Cette commande est un espace sûr ; en particulier chaque processus possède sa propre mémoire grâce au mécanisme de mémoire virtuelle. Il possède un numéro qui est unique sur le système, son pid, mais également un certain nombre de tables qui lui sont propres, comme la table des descripteurs ou celle de gestion des signaux. Chaque processus appatient à un utilisateur et un groupe et à les droits qui leur sont associés.

Définition 2: ***Espace d'adressage d'un processus*** **–** Tout processus UNIX a un espace d'adressage constitué de trois segments :
- code : les instructions qui forment le programme,
- pile : pile d'exécution,
- données : données statiques et dynamiques du programme.

Définition 3: ***Contexte d'un processus*** **–** Le contexte d'un processus comprend :
- le contenu de son espace d'adressage,
- les contenus des registres matériels,
- les structures de données du noyau qui ont un rapport avec le processus.

Définition 4: ***Thread*** **–** Un thread est une suite logique d'actions résultat de l'exécution d'un programme. Le contexte d'un thread comprend :
- une pile d'exécution,
- les contenus des registres matériels.

On peut donc dire qu'un processus contient un seul thread.

Définition 5: ***Processus multithread*** **–** Un processus multithread est un processus qui dispose de plusieur threads qui sont exécutés par la bibliothèque de threads.

2 Les bibliothèques de threads

2.1 Introduction

L'utilisation des bibliothèques de threads est l'approche classique pour concevoir une application mémoire partagée. Le niveau auquel est implémenté l'ordonnanceur constitue la caractéristique intrinsèque la plus importante des bibliothèques de threads. Les threads peuvent être gérés

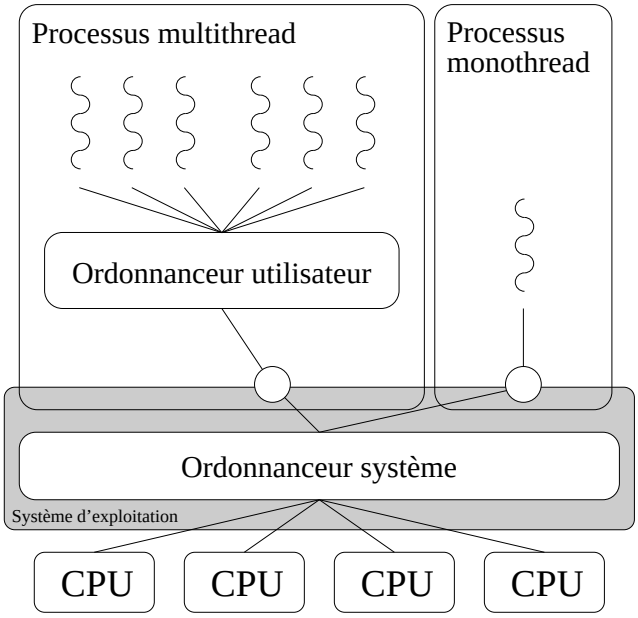
dans une bibliothèque entièrement en espace utilisateur, ou bien au sein même du noyau avec l'aide éventuelle d'une bibliothèque en espace utilisateur.

Il existe donc trois types de bibliothèques de threads :

– bibliothèque de niveau système,
– bibliothèque de niveau utilisateur,
– bibliothèque mixte.

2.2 Bibliothèque de niveau utilisateur ex : GnuPth

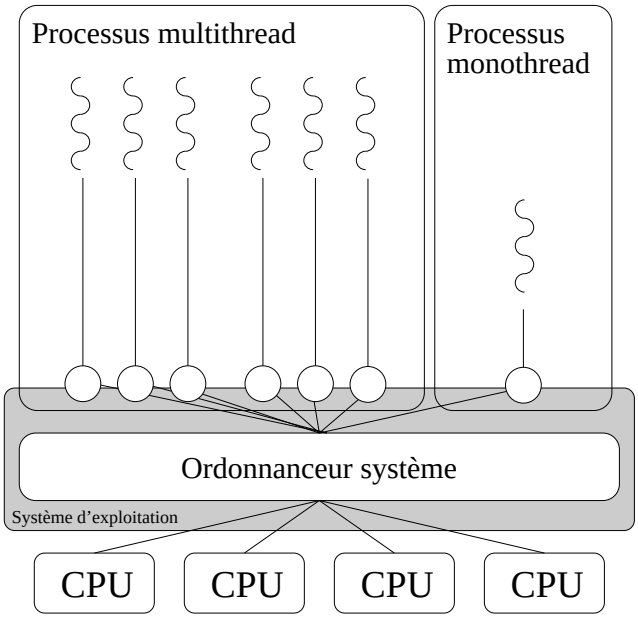
Les bibliothèques de threads de niveau utilisateur telle que GnuPth sont les premières apparues sur les systèmes d'exploitation, sans doute parce qu'il est plus facile de travailler en espace utilisateur que de modifier les systèmes d'exploitation. Il est facile de les adapter à un besoin particulier et, n'ayant pas à dialoguer avec le système d'exploitation, elles sont très efficaces. Le schéma suivant illustre ce type de bibliothèques :



Question 2.1: D'après vous quels sont les caratéristiques de ce type de bibliothèque ?

2.3 Bibliothèque de niveau système ex : Linux PThread

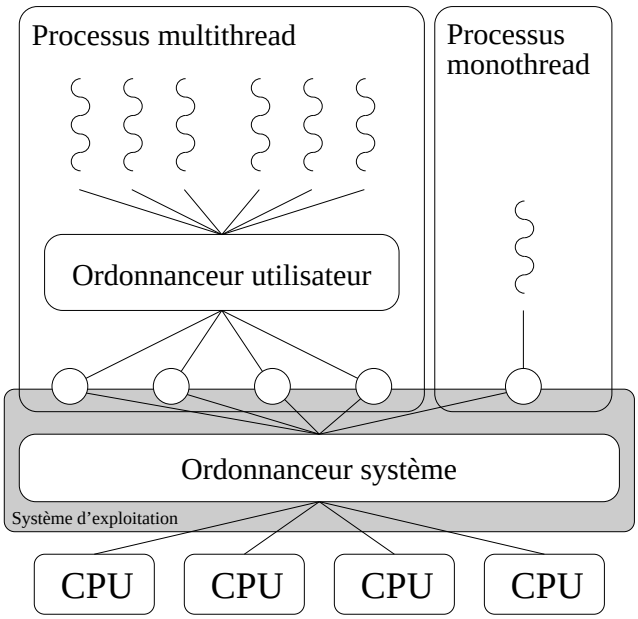
À l'opposé des bibliothèques de niveau utilisateur, les bibliothèques de niveau noyau sont généralement dédiées à un système particulier. Ces bibliothèques utilisent l'ordonnanceur du système pour gérer leurs processus légers ; une étroite intégration entre le système et la bibliothèque s'avère donc nécessaire. Les bibliothèques fournies avec les systèmes d'exploitation récents sont généralement des bibliothèques de niveau noyau (plus rarement à deux niveaux). Le schéma suivant illustre ce type de bibliothèques :



Question 2.2: D'après vous quels sont les caratéristiques de ce type de bibliothèque ?

2.4 Bibliothèque mixte ex : MThread

Les bibliothèque mixte ont l'avantage de garder un ordonnanceur en espace utilisateur et donc d'être efficaces et flexibles. Toutefois, la nécessité de faire coopérer deux ordonnanceurs rend l'écriture de telles bibliothèques plus difficile. Le schéma suivant illustre ce type de bibliothèques :



Question 2.3: D'après vous quels sont les caratéristiques de ce type de bibliothèque ?

3 Avis préliminaire

Question 3.1: D'après la description que nous venons de faire, déterninez les caratéristiques des différentes bibliothèques de threads. Pour cela, remplir le tableau suivant avec $+$ ou $-$.

	Caractéristiques			
Bibliothèque	Performances	Flexibilité	SMP	Appels systèmes bloquants
Utilisateur				
Système				
Mixte				

4 Évaluation des différents types de bibliothèques de thread

4.1 Mise en place des évaluation

1. Se connecter à `rabelais` login m12009 passwd XazGp34w{.
2. Copier le répertoire `thread_libs` dans votre "home".
3. Aller dans le repertoire `thread_libs/exemple`.
4. Utiliser la commande `compile` pour créer les binaires.
5. Tester les binaires `hello_world_*` en spécifiant en argument le nombre de threads.

Si tout s'est bien passé, vous devez avoir un environnement prêt pour les évaluations.

4.2 Analyse du code d'exemple

Question 4.1: Décrire le comportement ligne par ligne du code suivant (celui du ficher `hello_world_pthread.c`). Ne pas hésiter à utiliser la commande man pour obtenir de l'information

```
#include <stdio.h>
#include <assert.h>
#include <stdlib.h>
#include "pthread.h"

int NB_THREAD;

void* run(void* arg){
  long rank;
  rank = (long)arg;

  fprintf(stdout,"Hello, I'am %ld (%p)\n",rank,pthread_self());

  return arg;
}

int main(int argc, char** argv){
```

```
  pthread_t* pids;
  long i;

  NB_THREAD=atoi(argv[1]);
  pids = (pthread_t*)malloc(NB_THREAD*sizeof(pthread_t));

  for(i = 0; i < NB_THREAD; i++){
    pthread_create(&(pids[i]),NULL,run,(void*)i);
  }

  for(i = 0; i < NB_THREAD; i++){
    void* res;
    pthread_join(pids[i],&res);
    fprintf(stderr,"Thread %ld Joined\n",(long)res);
    assert(res == (void*)i);
  }


  return 0;
}
```

4.3 Évaluation des différentes bibliothèques

4.3.1 Évaluation des capacitées sur architectures SMP

Définition 6: ***SMP*** **–** Un système SMP est constitué de plusieurs processeurs identiques connecté à une *unique* mémoire physique.

Question 4.2: Décrire un moyen qui permet caractériser si une bibliothèque utilise correctement tous les processeurs à sa disposition.

Question 4.3: A partir des exemples `hello_world_*`, construire des exemples qui caractérisent l'utilisation des processeurs.

Question 4.4: Évaluer les caractéristiques SMP de la bibliothèque PThread.

Question 4.5: Évaluer les caractéristiques SMP de la bibliothèque GnuPth.

Question 4.6: Évaluer les caractéristiques SMP de la bibliothèque MThread.

4.3.2 Évaluation des capacités envers les appels bloquants

Question 4.7: Trouver un appel système bloquant permettant d'évaluer la capacité de gestion des appels bloquants des bibliothèques.

Question 4.8: A partir des exemples `hello_world_*`, construire des exemples qui caractérisent la gestion des appels bloquants.

Question 4.9: Évaluer les caractéristiques envers les appels bloquants de la bibliothèque PThread.

Question 4.10: Évaluer les caractéristiques envers les appels bloquants de la bibliothèque GnuPth.

Question 4.11: Évaluer les caractéristiques envers les appels bloquants de la bibliothèque MThread.

5 Récapitulatif

Question 5.1: D'après les évaluations que nous venons de faire, déterninez les caratéristiques des différentes bibliothèques de threads. Pour cela, remplir le tableau suivant avec $+$ ou $-$.

	Caractéristiques			
Bibliothèque	Performances	Flexibilité	SMP	Appels systèmes bloquants
Utilisateur				
Système				
Mixte				

6 Découverte du code de la bibliothèque MThread