Résumé du chapitre 1
Introduction aux systèmes d'exploitation

Jacques Mossière

21 septembre 2004

1 Fonctions d'un système d'exploitation

1.1 Définition

Tout système informatique a pour objet l'exécution d'applications. Comme la programmation de bas niveau est une activité complexe, on ne programme presque jamais un ordinateur directement au niveau du seul matériel. On programme un ordinateur doté d'un ensemble de programmes qui permettent d'améliorer l'interface offerte aux programmeurs et de simplifier la programmation d'applications. Ce sont ces programmes que l'on désigne sous le nom de système d'exploitation.

Exemples. Différentes versions du système Windows sont disponibles sur les PC ; le système Unix est disponible sur la plupart des ordinateurs actuels. Une version d'Unix, Linux, est disponible sous forme de logiciel libre. Le système des ordinateurs macintosh d'Apple, MACOS X, est construit à partir d'une autre version libre d'Unix, FreeBSD.

On peut traditionnellement répartir les services d'un système en deux classes : ceux qui étendent les fonctions offertes aux programmeurs, fournissant ainsi une **machine étendue ou virtuelle**, et ceux qui permettent la mise en commun des ressources d'une machine entre plusieurs applications, appelées aussi fonctions de **partage de ressources**.

1.2 Machine virtuelle

Le but de la machine virtuelle est de fournir une interface de programmation et d'exécution d'applications plus simple que celle fournie par le matériel. On abstrait en particulier les fonctions d'entrées-sorties élémentaires pour permettre la conservation d'informations sur disque, l'affichage d'informations sur écran et les lectures au clavier. La machine virtuelle permet aussi le déclenchement de l'exécution de nouvelles applications. L'interface étendue peut être utilisée à deux niveaux : par un utilisateur humain par le biais d'un langage de commande textuel ou graphique et par des programmes d'application par l'intermédiaire d'appels systèmes. Les langages de commande sont maintenant de véritables langages de programmation fournissant en général une notion de procédure avec des paramètres (« shellscript »).

1.3 Partage de ressources

L'utilisation d'un ordinateur, même simpliste, implique en général l'exécution de plusieurs programmes : par exemple, pendant que l'utilisateur consulte un serveur sur l'internet, une application de messagerie doit assurer le traitement de courriels. Pendant l'exécution d'un long programme de calcul, l'utilisateur peut souhaiter consulter sa boîte aux lettres, ou exécuter un jeu vidéo. Chacune de ces applications se traduit par l'exécution d'un programme, et le système d'exploitation doit répartir les moyens offerts par la machine (on dit plus savamment les ressources) entre les

différents programmes : coexistence en mémoire principale des différents programmes et données, affectation de l'unité centrale de façon équitable, etc. Cette répartition de ressources doit garantir l'isolation de ce qui doit l'être (le programme de calcul ne doit pas interférer avec le jeu vidéo) tout en permettant un partage de données entre applications (les résultats du programme de calcul peuvent être transmis à un programme de présentation des résultats sur écran).

2 Un peu d'histoire

Les premiers systèmes d'exploitation sont apparus dans les années 50. Ils avaient alors pour fonction de base d'assurer l'enchaînement automatique des différents programmes (traitement par lots ou « batch processing ») et fournissaient un ensemble de services facilitant la programmation, en particulier celle des entrées-sorties. L'évolution des systèmes a accompagné l'évolution des ordinateurs ; par exemple :

- l'augmentation des tailles de mémoire a permis de placer plusieurs programmes utilisateurs en mémoire (multiprogrammation), permettant de ne pas laisser l'unité centrale inactive pendant qu'un programme était en attente d'une entrée-sortie ;
- la généralisation des disques magnétiques a permis la mise en œuvre des systèmes de gestion des fichiers.

La structure des systèmes d'exploitation modernes est issue de recherches menées à la fin des années 60 et dans les années 70 (1967, premier symposium sur les principes des systèmes d'exploitation, SOSP). La réalisation phare est celle du système Multics au MIT qui a inspiré la réalisation d'Unix (article de Ritchie et Thompson, 1974). Depuis les années 80, les systèmes n'ont pas beaucoup évolué d'un point de vue conceptuel, mais ils ont dû prendre en compte les évolutions technologiques (taille des mémoires centrale et secondaire, vitesse des disques, etc.). L'introduction des ordinateurs individuels a permis le développement d'interfaces utilisateurs évoluées (écran graphique, clavier, souris) issues de recherche menées au centre de recherche Xerox de Palo Alto. Ces travaux ont été repris d'abord dans les Macintosh d'Apple, puis sur les IBM PC et sont depuis lors incontournables.

La généralisation de l'interconnexion des microordinateurs a conduit à des recherches sur les systèmes et les applications répartis, c'est-à-dire fonctionnant sur un ensemble d'ordinateurs reliés par des réseaux. Les recherches actuelles portent ainsi sur le développement des intergiciels (« middleware »), couches logicielles placées au dessus des systèmes d'exploitation traditionnels.

3 Différentes classes de systèmes

3.1 Ordinateur individuel

L'ordinateur individuel est présent partout dans notre société. Il doit permettre à de non spécialistes d'informatique d'exécuter des applications professionnelles, domestiques ou ludiques. Un accent tout particulier est mis (ou devrait être mis) :

- sur la qualité de l'interface utilisateur (présentation des informations sur l'écran, la plupart des commandes se faisant à la souris)
- sur la simplicité d'utilisation
- sur la fiabilité du système de conservation des informations.

Par rapport à nos deux volets précédents, c'est l'aspect machine virtuelle qui prédomine dans les systèmes pour ordinateurs individuels. Le partage de ressources reste cependant présent car l'exécution simultanée de plusieurs applications est possible.

3.2 Serveurs en temps partagé

L'exemple type est celui des serveurs de traitement utilisés au département (ensibm, telesun). Ces serveurs doivent servir quelques dizaines de clients qui peuvent exécuter des tâches très variées

(édition de courriers, jeu vidéo en ligne, navigation internet, compilations, calcul, etc.).

Ce sont sur ces systèmes que la dualité machine virtuelle — partage de ressources est le plus difficile à gérer. Le système doit assurer une bonne isolation entre les utilisateurs et donner l'impression à chacun d'eux d'utiliser une machine individuelle, c'est-à-dire assurer des performances acceptables pour les travaux interactifs.

3.3 Systèmes à transactions

Nous regroupons sous cette classe des systèmes comme les systèmes de réservation de place ou les systèmes bancaires. Ces systèmes sont caractérisés par le grand nombre de clients potentiels, chacun de ces clients exécutant des opérations relativement simples. Outre la gestion des clients, la difficulté est ici de maintenir la cohérence des données et d'assurer la conservation des données à très long terme.

3.4 Commande de procédés industriels, systèmes embarqués

L'ordinateur est utilisé pour contrôler le fonctionnement d'une installation, que cette installation soit de grandes taille ou complexité (raffinerie, fusée) ou très primitive (machine à laver).

Il s'agit ici de prélever périodiquement, à l'aide de capteurs, des informations sur le système contrôlé, d'effectuer un traitement défini une fois pour toutes et de transmettre les résultats au système contrôlé à l'aide d'actionneurs. L'intervalle de temps maximum séparant la prise de données de la transmission du résultat est en général borné pour des raisons physiques (penser à l'exemple de la fusée, si la trajectoire n'est pas corrigée assez vite). La difficulté ici est de garantir le bon fonctionnement (penser à Ariane 5) et de dimensionner le système, matériel et logiciel, pour assurer le respect des contraintes le plus économiquement possible (penser à la machine à laver)

4 Principaux problèmes

4.1 Programmation sûre et efficace d'activités parallèles

Les dispositifs câblés de gestion du parallélisme se résument souvent à l'existence d'un mécanisme d'interruption. En conséquence, les systèmes d'exploitation peuvent avoir un comportement indéterministe car une interruption peut survenir de façon aléatoire pendant l'exécution d'un programme. Une fonction de base des systèmes d'exploitation est d'organiser les activités parallèles de façon à maîtriser cet indéterminisme.

Un système d'exploitation permet de créer des activités parallèles, ou processus, et d'assurer leur exécution. Il fournit également des outils permettant aux processus de coordonner leur exécution. Nous étudions en cours et TD les facilités introduites par Unix pour la gestion des processus. Nous introduisons et explicitons sur des exemples types deux outils de synchronisation, les moniteurs et les sémaphores. Enfin, nous examinons comment implanter la notion de processus sur les architectures de machines couramment disponibles.

4.2 Conservation à long terme d'informations

La conservation d'informations à long terme est un enjeu important. Depuis Multics et Unix, l'interface des systèmes de gestion des fichiers (SGF) est bien stabilisée. Le fichier, suite d'octets de longueur quelconque, est l'unité élémentaire de conservation d'informations. Chaque fichier est désigné par un nom symbolique contenu dans un catalogue. Enfin, les catalogues sont organisés en arbre pour structurer l'espace des noms.

Après quelques rappels sur l'interface des SGF, le cours se concentre sur leur implantation efficace. On aborde ensuite les questions de sauvegarde et de sécurité.

4.3 Gestion des ressources

Les principales ressources à gérer sont l'unité centrale, la mémoire principale et l'espace disque. L'allocation de l'unité centrale est abordée dans le chapitre sur les processus, celle de l'espace disque dans la partie consacrée aux fichiers. Nous développons donc les techniques modernes de gestion de mémoire principale qui assurent :

- l'isolation des espaces des différents processus par mémoire virtuelle,
- l'indépendance entre les tailles de mémoire disponibles pour les processus et celle de la mémoire centrale,
- le chargement en mémoire des programmes et des données au fur et à mesure des besoins.

5 Objectifs du cours

Ce cours s'adresse à des étudiants ayant des connaissances de base sur la structure des ordinateurs et de leur programmation. Les connaissances requises se trouvent dans les cours d'architecture, d'algorithmique et de logiciel de base de première année. Les TP et projets seront réalisés en C et en assembleur. Les compléments d'architecture nécessaires seront fournis au fur et à mesure des chapitres.

Destiné à de futurs ingénieurs, le premier objectif est de comprendre la structure et le fonctionnement d'un système d'exploitation, en vue de construire des applications qui en tirent parti efficacement.

Dans leur vie professionnelle, peu d'ingénieurs auront à participer à la conception et à la réalisation de systèmes complets. En revanche, beaucoup d'entre eux auront à faire des adaptations à des systèmes existants, conception d'interfaces matérielles et des pilotes associés, par exemple.

6 Organisation du cours

L'organisation du cours reprend les points abordés dans la description des principaux problèmes. On étudie tout d'abord la définition, la programmation et la synchronisation entre activités parallèles ou processus. Le chapitre suivant est consacré à la programmation des mécanismes de bas niveau que sont les interruptions et les entrées-sorties. On aborde ensuite les systèmes de gestion des fichiers. Le chapitre sur la gestion de mémoire principale est centré sur les notions de mémoire virtuelle paginée et de chargement des pages à la demande. On trouvera enfin une description simplifiée de l'architecture globale d'un système.