CI 1 : Architecture matérielle et logicielle

Chapitre 2 – Systèmes d'exploitation et environnement de développement

Objectif

L'objectif est ici de se familiariser (Savoir Dec - C1) :
- à la manipulation d'un système d'exploitation (gestion des ressources, essentiellement : organisation des fichiers, arborescence, droits d'accès, de modification, entrées/sorties) ;
- à la manipulation d'un environnement de développement.

La principale capacité développée est :
- manipuler en mode « utilisateur » les principales fonctions d'un système d'exploitation et d'un environnement de développement.

1 Notion de système d'exploitation

1.1 Introduction

Le système d'exploitation (*Operating System* – *OS*) est un programme chargé en mémoire vive dès le démarrage de l'ordinateur et qui reste en mémoire jusqu'à son extinction.

Pour les **ordinateurs monoprocesseur** ce programme a pour but de donner l'illusion que l'ordinateur est multitâche lorsque plusieurs applications sont lancées en même temps. Pour cela, il stocke en mémoire les différentes

applications que l'on veut exécuter. Il lance l'exécution d'une première application. Dès qu'il se produit une entrée-sortie ou, à défaut, lorsqu'un certain temps est écoulé (de l'ordre de la centaine de millisecondes), le noyau du système d'exploitation reprend la main et lance l'exécution d'une autre application. En pratique, le temps d'exécution d'une tâche dépasse rarement la dizaine de millisecondes.

Pour les **ordinateurs multiprocesseurs** ce programme a pour but de gérer les accés de chacun des processeurs à la mémoire et aux périphériques, ce qui permet effectivement que l'ordinateur exécute plusieurs instructions à la fois. Les fabricants de processeurs proposent même maintenant des processeurs qui contiennent en fait plusieurs cœurs (dualcore, quadricore), c'est-à-dire que plusieurs processeurs (unité arithmétique et logique et unité de contrôle) ont été regroupés sur un seul processeur.

Par ailleurs l'OS permet de manière générale de gérer l'organisation des données sur le disque dur ainsi que leur droit d'accès. Il gère aussi les différentes ressources et sert de garde-fou en cas de tentative de mauvaise utilisation des ressources de l'ordinateur.

Pour finir, l'OS permet à un ou plusieurs utilisateurs de s'identifier leur permettant ainsi d'utiliser un seul ordinateur sans nécessairement partager les données et les programmes.

Il existe essentiellement deux grandes familles de systèmes d'exploitation.

Les systèmes d'exploitation de la famille Microsoft-Windows qui sont en situation de quasi-monopole sur les **ordinateurs personnels** (ils en équipent près de 90%)

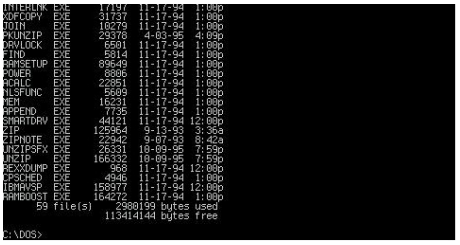
Les systèmes d'exploitation issus d'Unix (Mac OS X, iOS, GNU/Linux, Android, FreeBSD, NetBSD). Dans tous les domaines, sauf celui des ordinateurs personnels, les systèmes issus d'Unix sont majoritaires. Linux s'est imposé comme un système universel puisqu'il équipe aussi bien les téléphones portables (Android) que les boîtiers prêtés par les fournisseurs d'accès à Internet, les ordinateurs personnels (Ubuntu/GNU Linux), les serveurs web et les supercalculateurs (plus de 90% des calculateurs du TOP 500).

Il est possible d'installer plusieurs systèmes d'exploitation sur une même machine : double boot. Il est aussi possible d'émuler un système d'exploitation dans un autre système d'exploitation.

1.2 Petit historique

1969 Origines de UNIX.

1984 Sortie du premier OS de Microsoft : MS-DOS.



1984 Richard Stallman[1] crée le projet GNU dans le but de développer un OS basé sur Unix, mais libre.

1991 Linus Torvalds[2] entreprend de créer sur son temps libre son propre système d'exploitation. Ce système a pris le nom de Linux (contraction de Linus et Unix). Ce projet est complémentaire du projet GNU : tandis que Richard Stallman créait les programmes de base (programme de copie de fichier, de suppression de fichier, éditeur de texte), Linus s'est lancé dans la création du « cœur » d'un système d'exploitation : le noyau.

Remarque

Un programme libre est un programme dont on peut avoir le code source, c'est-à-dire la « recette de fabrication ». Au contraire, Windows est un OS propriétaire dont le code source est conservé par Microsoft. On ne peut donc pas le modifier ou regarder comment il fonctionne à l'intérieur.

Un programme libre est donc la plupart du temps un programme gratuit. Mais c'est aussi un programme qu'on a le droit de copier, modifier, redistribuer. On dit aussi souvent que le programme est « Open Source » car son code source est ouvert ; tout le monde peut le voir.

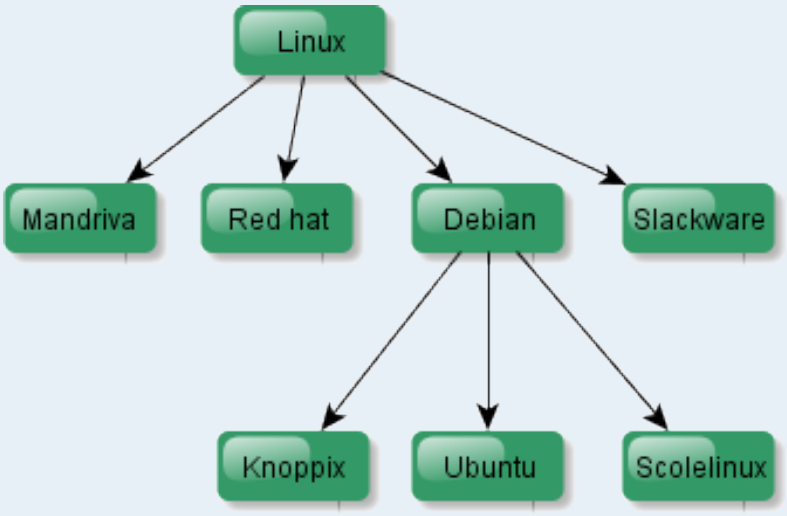
Il existe quelques légères différences entre un programme « Open Source » et un programme « libre », mais nous n'entrerons pas dans les détails ici.

1. Chercheur en intelligence artificielle au MIT.
2. Étudiant de l'Université de Helsinki (Finlande).

Remarque

Les distribution GNU/Linux

Une distribution est un ensemble de logiciels articulés autour d'un noyau. GNU/Linux comprend un grand nombre de distributions. Elles diffèrent par la méthode d'installation, la gestion des programmes, la méthode pour installer des programmes. Attention, le terme distribution est différent du terme logiciel. De manière générale, un logiciel libre est installable sur n'importe quelle distribution. Dans le cas le plus défavorable, il sera nécessaire de **compiler** le logiciel.



Différentes distributions GNU/Linux

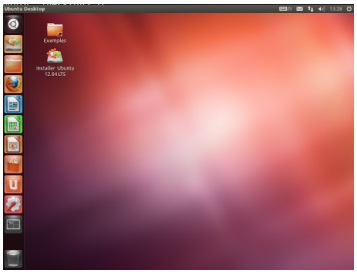
1.3 À quoi ressemble Linux ?

Une des réticences à l'utilisation d'une distribution GNU/Linux est le mode console.

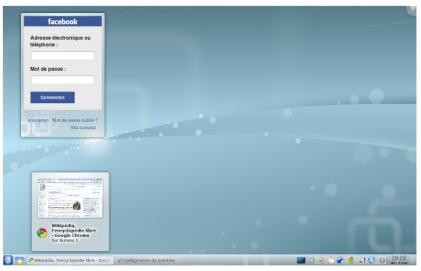
Ce mode de fonctionnement peut paraître effrayant et peu accessible pour un utilisateur novice. Cependant, le mode console permet de faire une grande partie des opérations faisable avec la souris ce qui, avec l'expérience, améliore l'efficacité de l'utilisateur.

```
ubuntu@ubuntu:~/Desktop$ ls
Examples  ubiquity-kdeui.desktop
ubuntu@ubuntu:~/Desktop$ cd Examples
ubuntu@ubuntu:~/Desktop/Examples$ ls
book                    logo-Kubuntu.png          oo-maxwell.odt
book-toc.html           logo-Ubuntu.png           oo-payment-schedule.ods
Experience ubuntu.ogg   oo-about-these-files.odt  oo-presenting-kubuntu.odp
fables_01_01_aesop.spx  oo-about-ubuntu-ru.rtf    oo-presenting-ubuntu.odp
gimp-ubuntu-splash.xcf  oo-access.odt             oo-trig.xls
kubuntu-leaflet.png     oo-cd-cover.odg           oo-welcome.odt
logo-Edubuntu.png       oo-derivatives.doc        ubuntu Sax.ogg
ubuntu@ubuntu:~/Desktop/Examples$ pwd
/home/ubuntu/Desktop/Examples
ubuntu@ubuntu:~/Desktop/Examples$ w
 22:44:02 up 15 min,  7 users,  load average: 0,07, 0,29, 0,26
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
ubuntu   tty1     -                 22:30    0.00s  2.93s  0.02s w
ubuntu   tty2     -                 22:30   15:25m  0.17s  0.14s -bash
ubuntu   tty3     -                 22:30   15:25m  0.15s  0.12s -bash
ubuntu   tty4     -                 22:30   15:25m  0.17s  0.14s -bash
ubuntu   tty5     -                 22:30   15:25m  0.15s  0.13s -bash
ubuntu   tty6     -                 22:30   15:25m  0.17s  0.15s -bash
ubuntu   :0       -                 22:30   ?xdm?  50.06s  0.15s /bin/sh /usr/bi
ubuntu@ubuntu:~/Desktop/Examples$ _
```

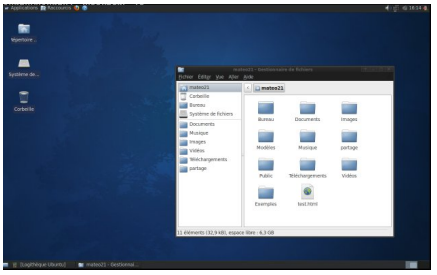
Le mode graphique semble beaucoup plus accueillant pour quelqu'un venant de Windows. Il y a plusieurs modes graphiques, tous les modes graphiques sont basés sur un programme appelé X. On parle aussi de serveur d'affichage.



Ubuntu



Kubuntu, basé sur KDE



Xubuntu, basé sur XFCE

1.4 Identification des utilisateurs

Les systèmes d'exploitation Unix comme Microsoft sont multi-utilisateur : chaque utilisateur dispose d'un identifiant auprès du système (et en général, d'un mot de passe correspondant).

Fenêtre d'identification de Windows

Fenêtre d'identification de Linux

1.5 La structure des dossiers et fichiers

Le nombre de fichiers étant généralement élevé, ils sont organisés en une structure arborescente de répertoires. Du point de vue de l'utilisateur, un répertoire est un ensemble de fichiers et de sous répertoires, désignés par des noms.

Du point de vue conceptuel, un fichier est une séquence finie d'octets, sans signification a priori : c'est le programme qui lira cette séquence d'octets qui décidera de la façon de la comprendre. Il peut contenir des données de tout type : texte, son, vidéo et même des programmes directement exécutables par le processeur (programmes dits « binaires » ou « en langage machine »).

Les fichiers présents sur le disque dur sont répertoriés dans une ou plusieurs tables, stockées à un endroit du disque conventionnellement fixé, qui donnent des informations relatives à ces fichiers, appelées métadonnées, notamment :

- date de création, de dernière modification, de dernière lecture ;
- taille du fichier ;
- emplacement des données du fichier sur le disque ;
- suivant les systèmes de fichiers employés, un numéro identifiant le fichier, appelé inode sous Unix.

La table des métadonnées est aussi appelée table des inodes. Du point de vue du système d'exploitation, un répertoire est juste un fichier particulier, qui contient une liste de couples (n, i) où n est un nom de fichier et i l'inode du fichier. Une information supplémentaire est stockée dans la table des inodes pour indiquer, pour chaque fichier, s'il s'agit d'un fichier de données ordinaire ou d'un répertoire.

1.5.1 La racine

Dans un système de fichiers, il y a toujours ce qu'on appelle une racine, c'est-à-dire un gros dossier de base qui contient tous les autres dossiers et fichiers.

Sous Windows, il y a en fait plusieurs racines. *C :* est la racine de votre disque dur, *D :* est la racine de votre lecteur CD (par exemple).

Sous Linux, il n'y a qu'une et une seule racine : « / ». Il n'y a pas de lettre de lecteur car justement, Linux ne donne pas de nom aux lecteurs comme le fait Windows. Il dit juste « La base, c'est / ».

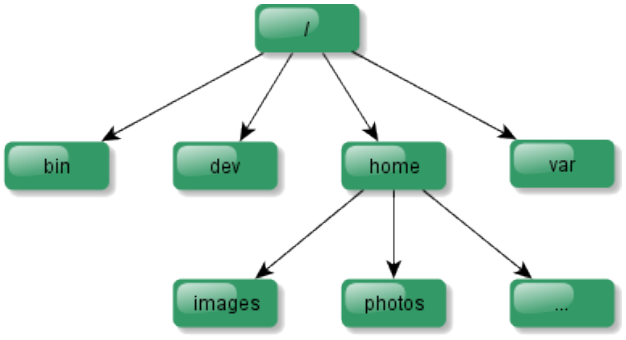
Il n'y a pas de dossier de plus haut niveau que /, c'est-à-dire qu'il n'existe pas de dossier qui contienne le dossier /. Quand on est à la racine, on ne peut pas remonter en arrière car… on est déjà tout au début.

1.5.2 Architecture des dossiers

Sous Windows, un dossier peut être représenté de la manière suivante : *C :\Program Files\Gimp 2*. On dit que Gimp 2 est un sous-dossier du dossier Program Files, lui-même situé à la racine.

Vous noterez que c'est l'antislash \(aussi appelé backslash) qui sert de séparateur aux noms de dossiers. Sous Linux, c'est au contraire le / (slash) qui sert de séparateur. Il n'y a pas de *C :* sous Linux, la racine (le début) s'appelant juste /.

Le dossier de notre superprogramme ressemblerait plutôt à quelque chose comme cela : */usr/bin/*. On dit que *bin* est un sous-dossier du dossier *usr*, lui-même situé à la racine.



Organisation des dossiers de Linux

Sous Windows, on a l'habitude de trouver souvent les mêmes dossiers à la racine : Documents and Settings, Program Files, Windows… Sous Linux, vous vous en doutez, les dossiers sont complètement différents.

Quelques dossiers courants sous linux :
- *boot* : fichiers permettant le démarrage de Linux ;
- *etc* : fichiers de configuration ;
- *home* : répertoires personnels des utilisateurs. On en a déjà parlé un peu avant : c'est dans ce dossier que vous placerez vos fichiers personnels, à la manière du dossier Mes documents de Windows.

1.5.3 Quelques commandes classiques pour se déplacer dans l'arborescence :

- *pwd* : afficher le dossier actuel

- *ls* : lister les fichiers et dossiers
- *cd* : changer de dossier
- *du* : taille occupée par les dossiers
- *cp* : copier un fichier
- *mv* : déplacer un fichier
- *rm* : supprimer des fichiers et dossiers
- *mkdir* : créer un dossier

Ces quelques fonctions peuvent s'avérer nécessaires car elles sont utilisables dans Python pour manipuler les fichiers.

Le symbole * est appelé joker, ou encore wildcard en anglais sous Linux.

Terminal

Le joker vous permet de copier par exemple tous les fichiers image .jpg dans un sous-dossier :

```
cp *.jpg mondossier/
```

Terminal

Vous pouvez aussi vous en servir pour copier tous les fichiers dont le nom commence par « so » :

```
cp so* mondossier/
```

1.6 Droits d'accès

Les droits d'accès font partie des propriétés d'un fichier. On peut recenser 3 principaux droits pour un fichier :
- le droit en lecture (*r – read*), qui permet à l'utilisateur de lire le contenu du fichier ;
- le droit en écriture (*w – write*), qui permet à l'utilisateur d'écrire dans le fichier ;
- le droit d'exécution (*x – execute*), qui permet à l'utilisateur d'exécuter le fichier.

Par ailleurs pour un même ordinateur, il peut exister plusieurs utilisateurs. Ces utilisateurs peuvent de plus appartenir à un groupe.

Ainsi il est possible de distribuer des droits à un utilisateur, à un groupe d'utilisateurs ou à tous les utilisateurs.

Attention

Toutes les données que ce soit dans votre espace personnel ou sur les serveurs du lycée ne sont stockées qu'a titre provisoire. Il vous appartient de faire des sauvegardes régulières.

Attention

Tenter d'accéder aux données d'autrui est considéré comme une atteinte à la vie privée.

2 Python

Python est un langage de programmation, dont la première version est sortie en 1991. Créé par Guido van Rossum, il a voyagé du Macintosh de son créateur, qui travaillait à cette époque au Centrum voor Wiskunde en Informatica aux Pays-Bas, jusqu'à se voir associer une organisation à but non lucratif particulièrement dévouée, la Python Software Foundation, créée en 2001. Ce langage a été baptisé ainsi en hommage à la troupe de comiques les « Monty Python ».

2.1 À quoi peut servir Python ?

Python est un langage puissant, relativement facile à apprendre et riche en possibilités. Il est possible d'étendre ses fonctionnalités. On peut aussi appeler des bibliothèques qui aident le développeur à travailler.

Python permet :
- de réaliser de petits programmes très simples, appelés scripts, chargés d'une mission très précise sur votre ordinateur ;
- des programmes complets, comme des jeux, des suites bureautiques, des logiciels multimédias, des clients de messagerie…
- des projets très complexes, comme des progiciels (ensemble de plusieurs logiciels pouvant fonctionner ensemble, principalement utilisés dans le monde professionnel).

Voici quelques-unes des fonctionnalités offertes par Python et ses bibliothèques :
- créer des interfaces graphiques ;
- faire circuler des informations au travers d'un réseau ;
- dialoguer d'une façon avancée avec votre système d'exploitation ;
- …

2.2 Un langage de programmation interprété

Python est un langage de programmation interprété, c'est-à-dire comme on l'a spécifié au que les instructions que vous lui envoyez sont « transcrites » en langage machine au fur et à mesure de leur lecture. D'autres langages (comme le C / C++) sont appelés « langages compilés » car, avant de pouvoir les exécuter, un logiciel spécialisé se charge de transformer le code du programme en langage machine. On appelle cette étape la « compilation ». À chaque modification du code, il faut rappeler une étape de compilation.

Les avantages d'un langage interprété sont la simplicité (on ne passe pas par une étape de compilation avant d'exécuter son programme) et la portabilité (un langage tel que Python est censé fonctionner aussi bien sous Windows que sous Linux ou Mac OS, et on ne devrait avoir à effectuer aucun changement dans le code pour le passer d'un système à l'autre). Cela ne veut pas dire que les langages compilés ne sont pas portables, loin de là ! Mais on doit utiliser des compilateurs différents et, d'un système à l'autre, certaines instructions ne sont pas compatibles, voire se comportent différemment.

En contrepartie, un langage compilé se révélera bien plus rapide qu'un langage interprété (la traduction à la volée de votre programme ralentit l'exécution), bien que cette différence tende à se faire de moins en moins sentir au fil des

améliorations. De plus, il faudra installer Python sur le système d'exploitation que vous utilisez pour que l'ordinateur puisse comprendre votre code.

2.3 Différentes versions de Python

Lors de la création de la Python Software Foundation, en 2001, et durant les années qui ont suivi, le langage Python est passé par une suite de versions que l'on a englobées dans l'appellation Python 2.x (2.3, 2.5, 2.6...). Depuis le 13 février 2009, la version 3.0.1 est disponible. Cette version casse la compatibilité ascendante qui prévalait lors des dernières versions.

Références

[1] Wack, Conchon, Courant, deFalco, Dowek, Filliatre, Gonnord, Informatique pour tous en classes préparatoires aux grandes écoles, éditions Eyrolles.

[2] Apprenez à programmer en Python http://www.siteduzero.com/.