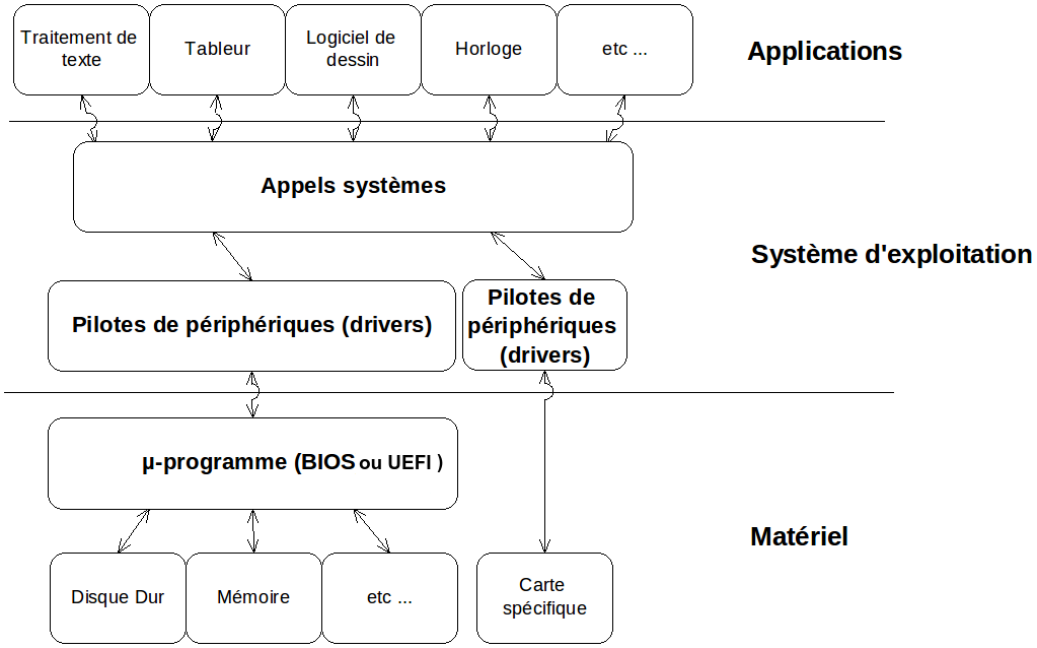
Fiche 2 : Introduction à la programmation système Linux

I - Introduction

Le **Système d'Exploitation** (**SE** ou **Operating System**) est la **couche logicielle** qui sert d'interface entre les applications et le matériel. C'est à dire qu'il prend appui à l'aide d'**appels système**, sur les circuits (la couche µ-programmée **BIOS** ou **UEFI**) pour mettre à la disposition des programmes d'application, les possibilités du matériel.

On peut modéliser le fonctionnement d'un ordinateur à l'aide d'un **modèle** à **3 couches** :



Remarque : L'***Unified Extensible Firmware Interface*** (**UEFI**), est un logiciel intermédiaire et est néanmoins un véritable petit système d'exploitation. Cette interface est le successeur du **BIOS**.

Remarque : La norme **POSIX** (**P**ortable **O**perating **S**ystem **I**nterface uni**X**) est un ensemble de standards de l'**IEEE** (**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers). **POSIX** définit notamment :

- Les commandes **shell** de base (bash, ls, man, ...) ;
- L'**API** (**A**pplication **P**rogramming **I**nterface) des **appels système** ;
- L'**API** des **threads**.

II - Arguments d'un programme

II.1. Arguments du main

Comme n'importe quelle fonction, la fonction principale **main()** d'un programme accepte des arguments. Ces derniers sont au nombre de **deux** : un **entier**, généralement appelé **argc** et un **pointeur** sur un **tableau de caractères**, généralement appelé **argv**.

- L'entier **argc** donne le nombre de paramètres + 1 ;
- Le second paramètre **argv** pointe sur ces derniers et contient comme éléments :
  - le premier élément **argv[0]** qui est une chaîne qui contient le nom du fichier exécutable du programme ;
  - les éléments suivants **argv[1]**, **argv[2]**, etc.. qui sont des chaînes de caractères qui contiennent les arguments passés en ligne de commande.

Soit le programme source suivant :

```
#include <stdio.h>
int main(int argc, char *argv[])
        {
        int i;
        for(i = 0; i < argc; i++) {
                printf("argument %d : %s\n", i, argv[i]);
                }
        if(strcmp(argv[1],"1")==0) printf("option: %s\n", argv[1]);
        return(0);
        }
```

Ce qui donne à l'exécution :

```
#./test 1 2
argument 0 : ./test
argument 1 : 1
argument 2 : 2
option: 1
```

II.2. La fonction system()

La fonction **system()** de la bibliothèque **stdlib.h** permet directement de lancer un programme dans un programme **C**.

```
#include <stdlib.h>
system(char *commande);
```

La commande **system()** n'accepte qu'une seule chaîne de caractères en argument et ne permet donc pas d'introduire plusieurs arguments dynamiques comme le fait la fonction **printf()**. Si tel est le cas, il faut alors coupler la fonction **system()** avec la fonction **sprintf()**.

```
char commande[BUFSIZ];
sprintf(commande, "%s %s", "chaine1",  "chaine2");
system(commande);
```

Soit le programme source suivant :

```
#include <stdio.h>
#include <stdlib.h> // pour utiliser system()
int main(void)
        {
        printf("Appel de la fonction system() \n");
        system ("pwd");
        return(0);
        }
```

Ce qui donne à l'exécution :

```
# ./test
Appel de la fonction system()
/home/etudiant
```