

Quelques éléments sur l'architecture système

inetdoc.net



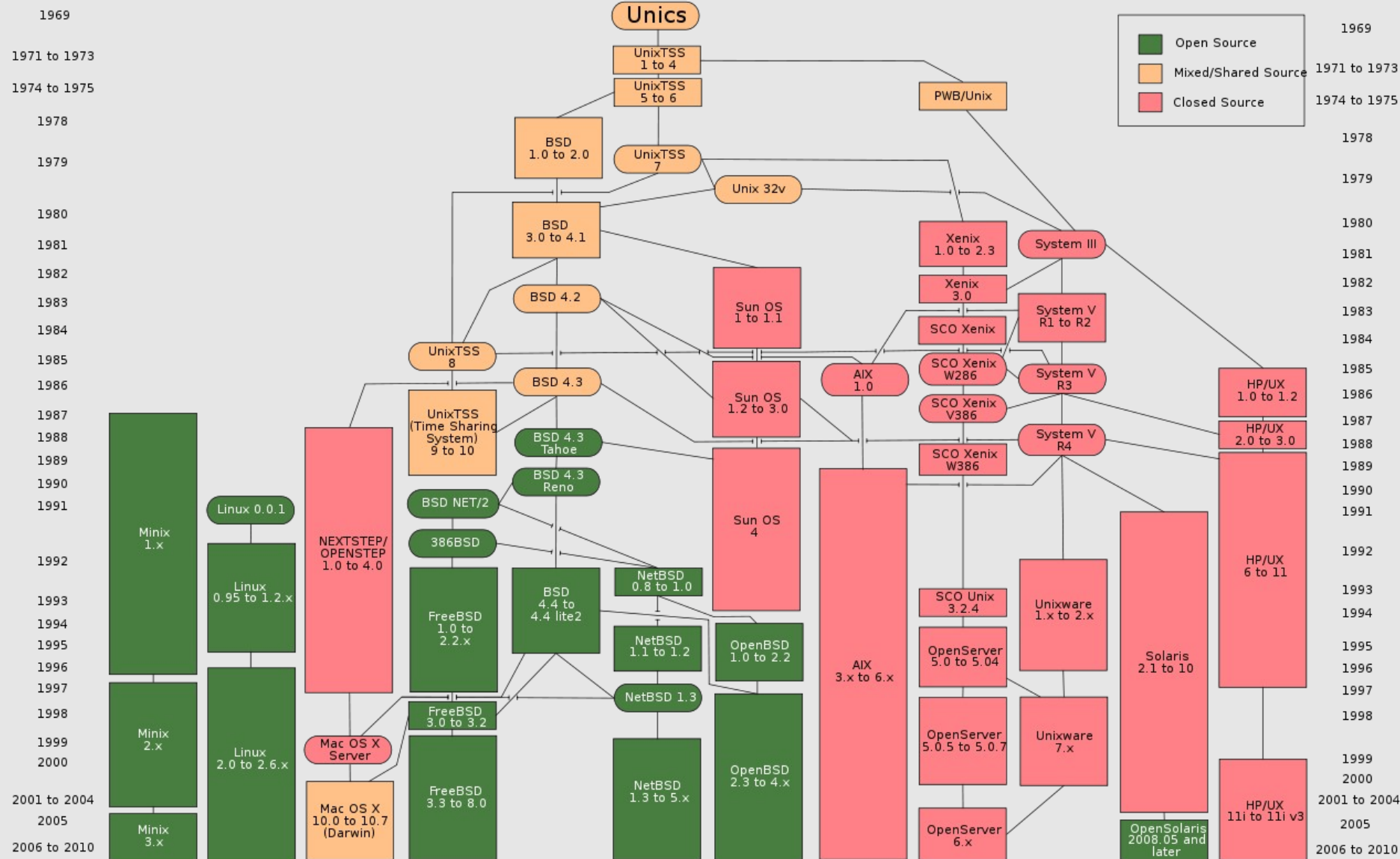
Philippe Latu / Université Toulouse 3 – Paul Sabatier
Document sous licence GNU FDL v1.3
<http://www.gnu.org/licenses/fdl.html>



Unix & GNU/Linux

- 5 fonctions de base des systèmes Unix
 - **Multi-tâches**
 - Temps processeur partagé entre plusieurs programmes
 - **Multi-utilisateurs**
 - Système partagé entre plusieurs utilisateurs
 - **Portabilité**
 - Outils système partagés entre ordinateurs différents
 - **Bibliothèques de développement standard**
 - Optimisation des développements en partageant le code source
 - **Applications communes**
 - Services système, services Internet, etc.

Unix & GNU/Linux



Source wikipedia
<http://en.wikipedia.org/wiki/Unix>

Unix & GNU/Linux

- **Unix est un système «accidentel»**
 - AT&T Bell labs – Ken Thompson – Dennis Ritchie
 - 1973 réécriture en Langage C
 - Diffusion sous licence AT&T incluant la totalité du **code source**
 - 1975 publication RFC681 **NETWORK UNIX**
 - 1978 première publication du livre **The C Programming Language**



Unix & GNU/Linux

- **Apparition de la représentation graphique**
 - 1986 **The Design of the UNIX Operating System**
 - Kernel
 - Shell représenté par ses commandes
 - Applications avec un secteur dédié au compilateur

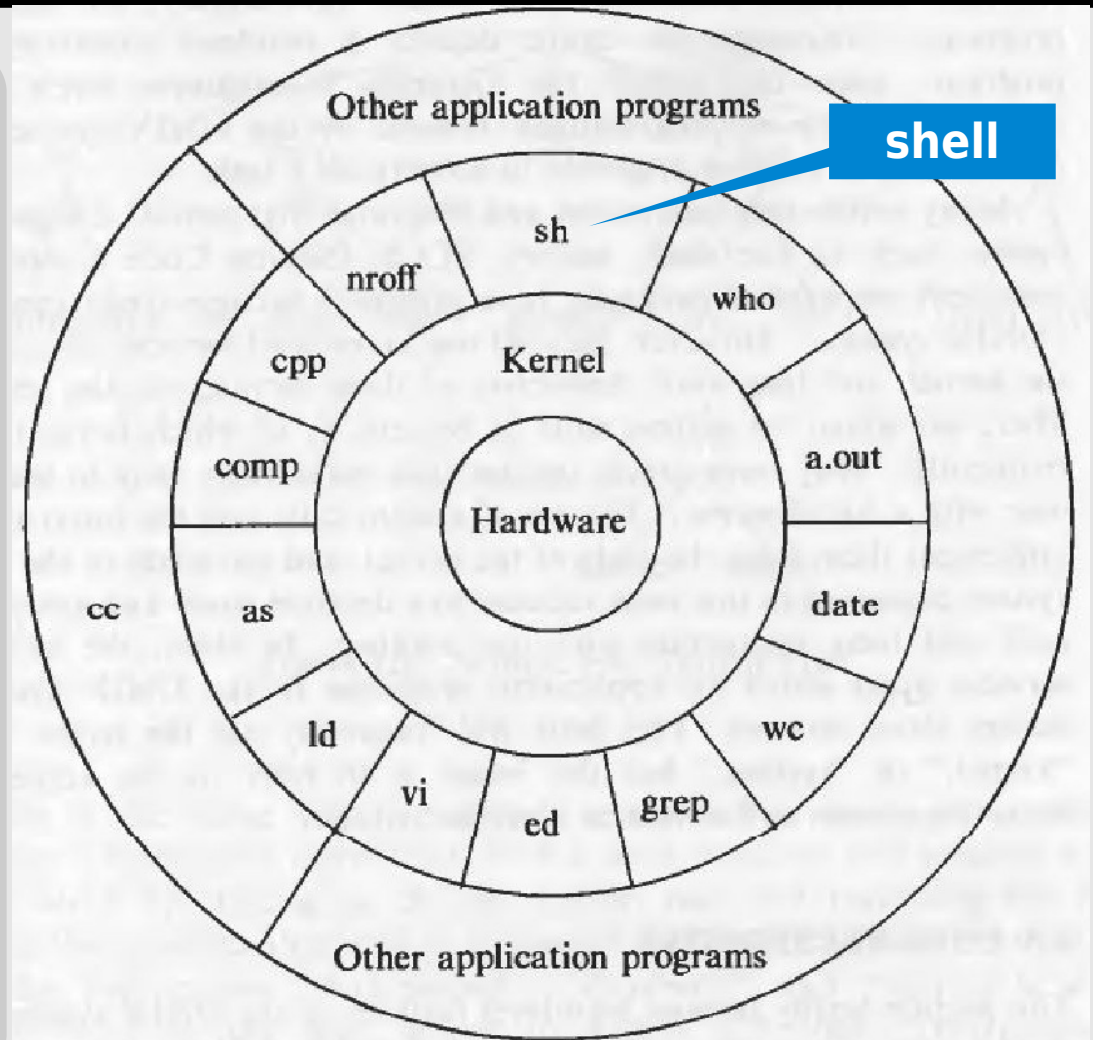
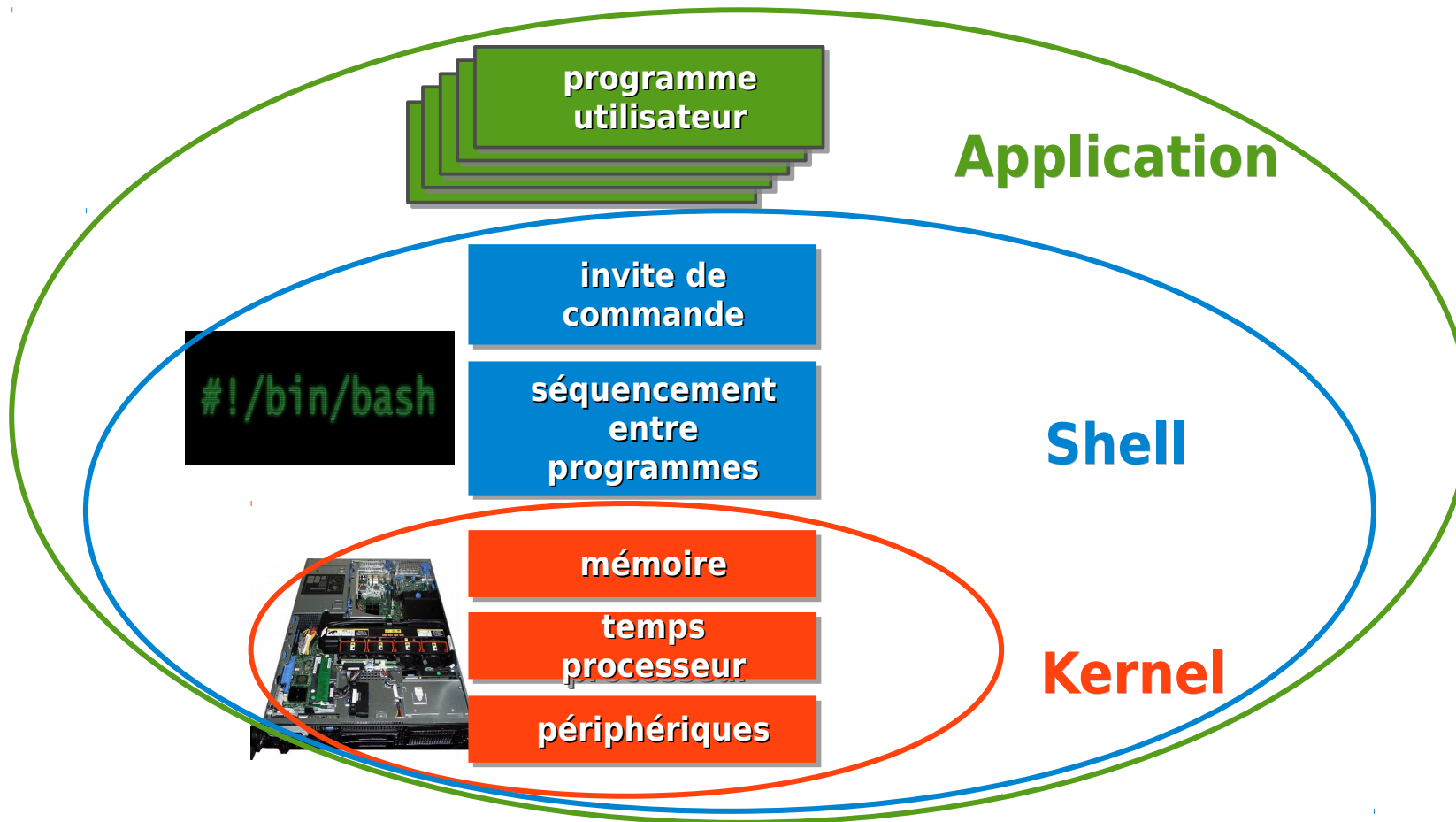


Figure 1.1. Architecture of UNIX Systems

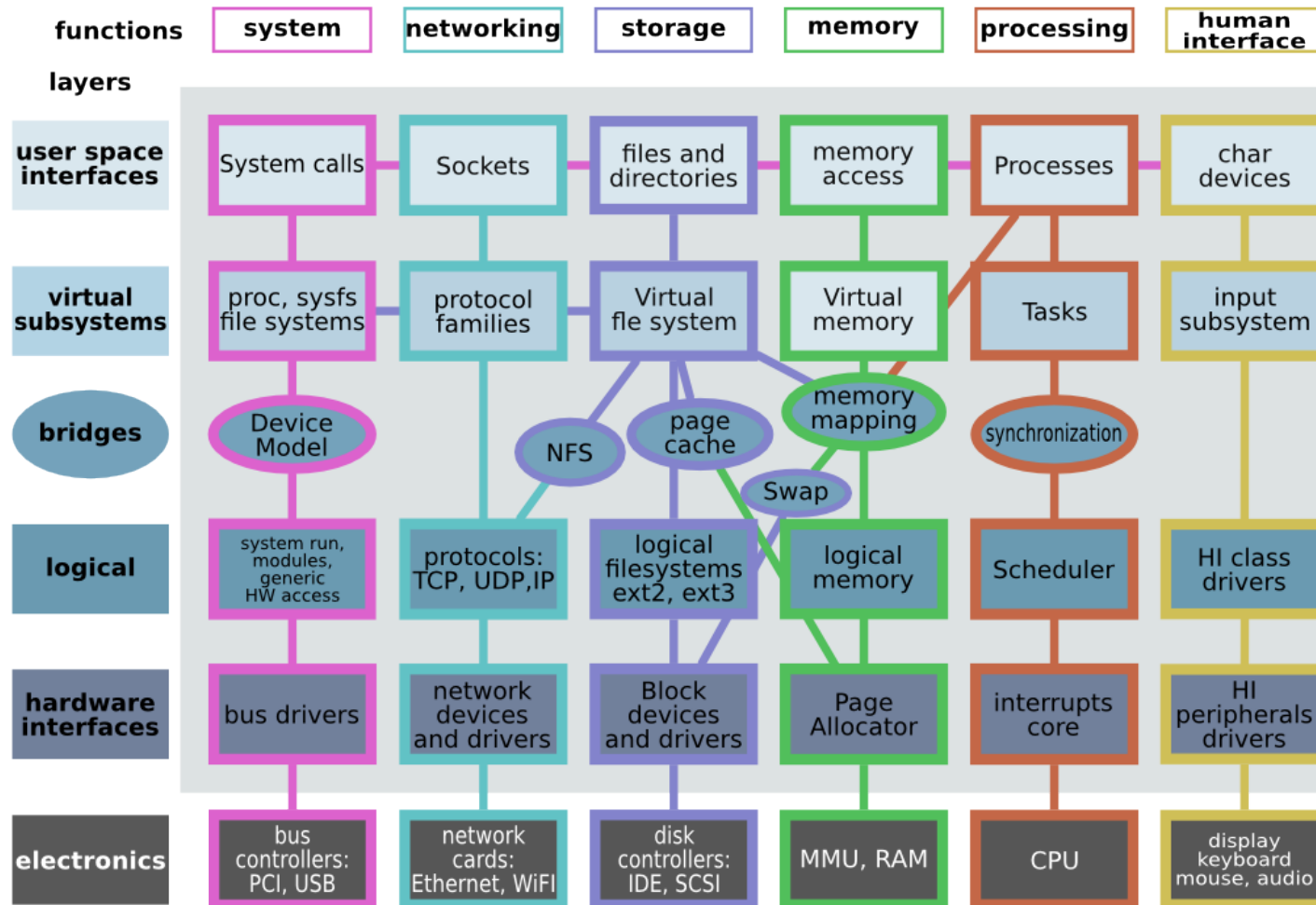
Unix & GNU/Linux

- Plusieurs décennies passent → représentation graphique identique



Kernel ou noyau

Linux kernel diagram



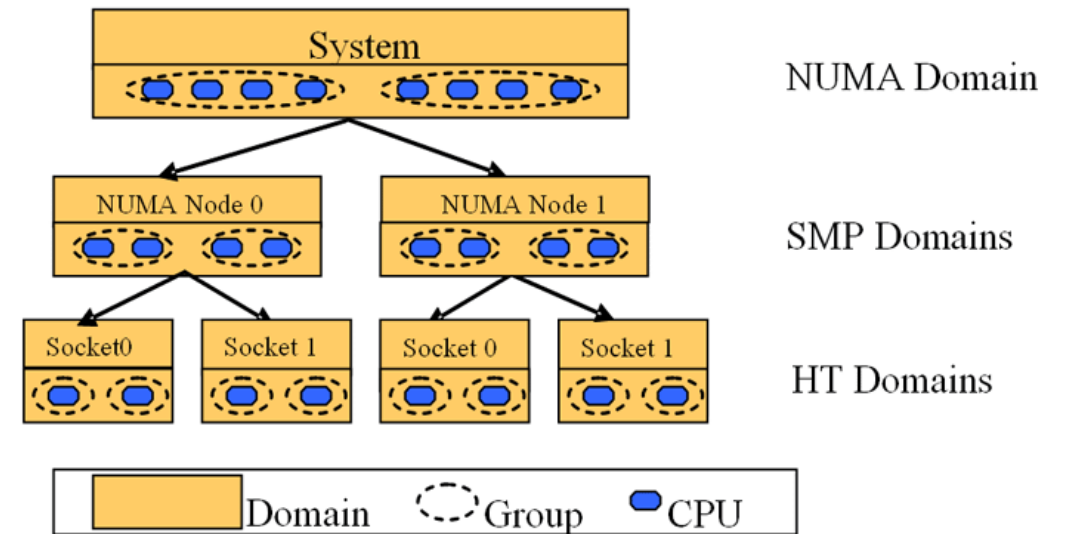
© 2007-2009 Constantine Shulyupin <http://www.MakeLinux.net/kernel/diagram>

Noyau Linux

- Fonctions simplifiées des niveaux kernel/Shell/Application
 - **Kernel → 3 fonctions de gestion**
 - Mémoire
 - Arbitrage des accès aux ressources entre programmes/processus
 - Virtual Memory (VM) & Memory Management Unit (MMU)
 - Distinction entre les espaces mémoire noyau et utilisateur
 - Périphériques
 - Arbitrage des accès aux entrées/sorties avec synchronisation
 - Process → Block I/O → I/O Scheduler → device driver
 - Processeur
 - Répartition du temps processeur entre programmes/processus
 - Scheduler

Noyau Linux

- Ordonnanceur ou Scheduler
 - **3 domaines ou types de tâches**
 - **Domaine temps réel**
 - Contraintes de temps d'exécution élevées
 - Fréquence d'exécution garantie
 - **Domaine entrées/sorties**
 - Attente de disponibilité des périphériques
 - **Domaine CPU**
 - Temps consacré aux calculs
 - **Tranche de temps CPU - time slice**
 - Durée d'exécution d'un processus sur un cœur
 - **Préemption**
 - Interruption d'un processus par un second de priorité plus élevée



Noyau Linux

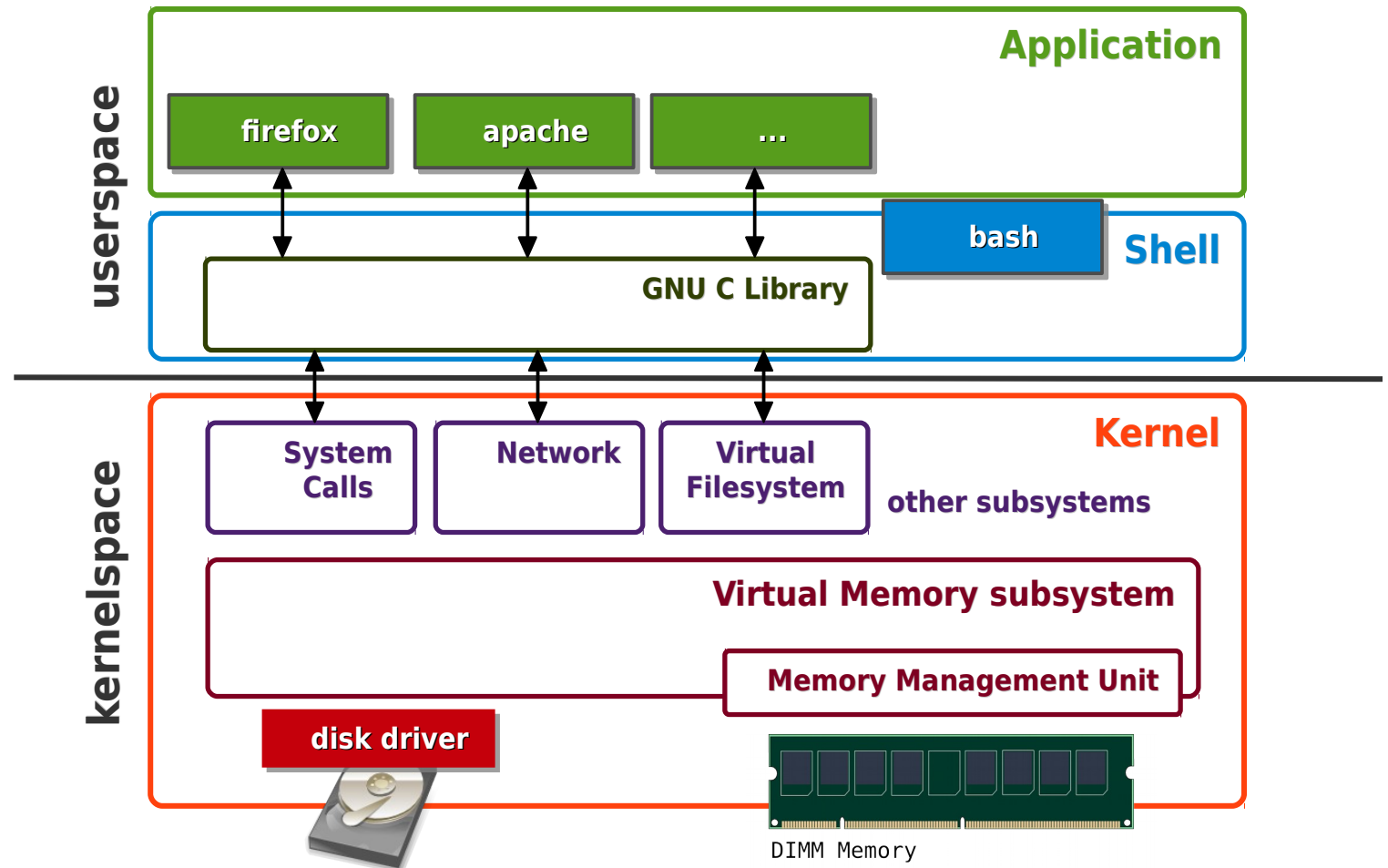
- Mémoire physique & mémoire virtuelle

- Userpace

- Espace mémoire propre pour chaque processus

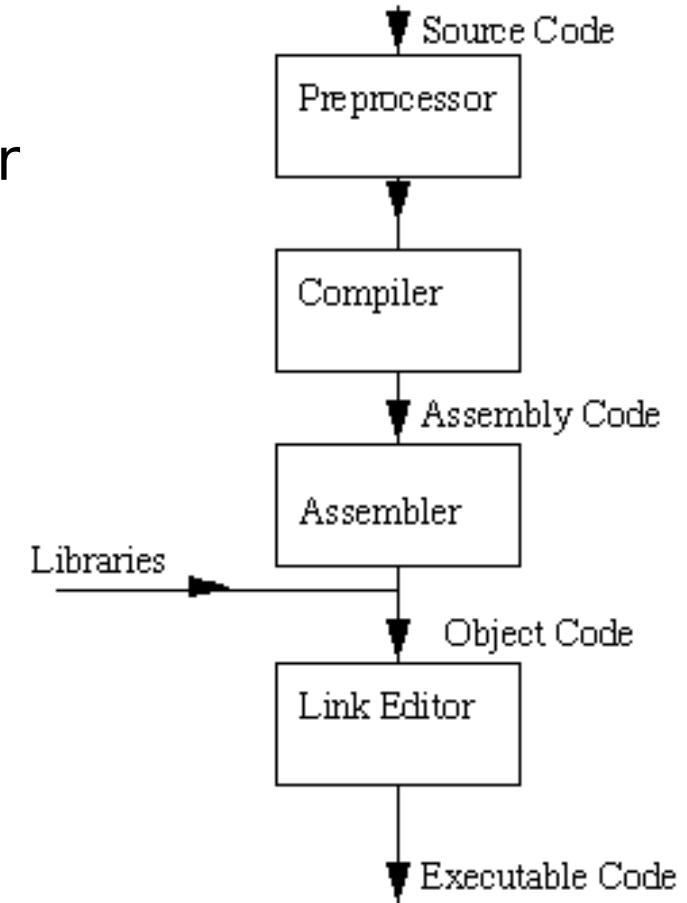
- Kernespace

- Espace réservé aux fonctions du noyau
 - Espace accessible aux processus via appel système (syscall)



Code source → code exécutable

- **Tout programme est écrit dans un langage**
 - Noyau Linux → écrit en Langage C ← *code source*
 - Code source → interprétation impossible par le processeur
- **Compilation**
 - Transformation code source → *code exécutable*
 - Code exécutable → interprétation uniquement par le processeur
- **Logiciel propriétaire**
 - Droit d'utilisation limité d'un code exécutable
- **Logiciel libre**
 - Accès au code source
 - Droit d'utilisation, d'échange, de modification et de redistribution



Programme showip.c

```
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <netdb.h>
#include <arpa/inet.h>
```

Catalogues des
bibliothèques standard

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

```
    struct addrinfo hints, *res, *p;
    int status;
    char ipstr[INET6_ADDRSTRLEN];
```

Enregistrement 'addrinfo'
défini dans la bibliothèque

```
    if (argc != 2) {
        fprintf(stderr, "usage: showip hostname\n");
        return 1;
    }
```

fprintf → appel système
vers la sortie 'stderr'

```
    memset(&hints, 0, sizeof hints);
    hints.ai_family = AF_UNSPEC; // IPv4 or IPv6
    hints.ai_socktype = SOCK_STREAM;
```

memset → appel système
vers la gestion mémoire

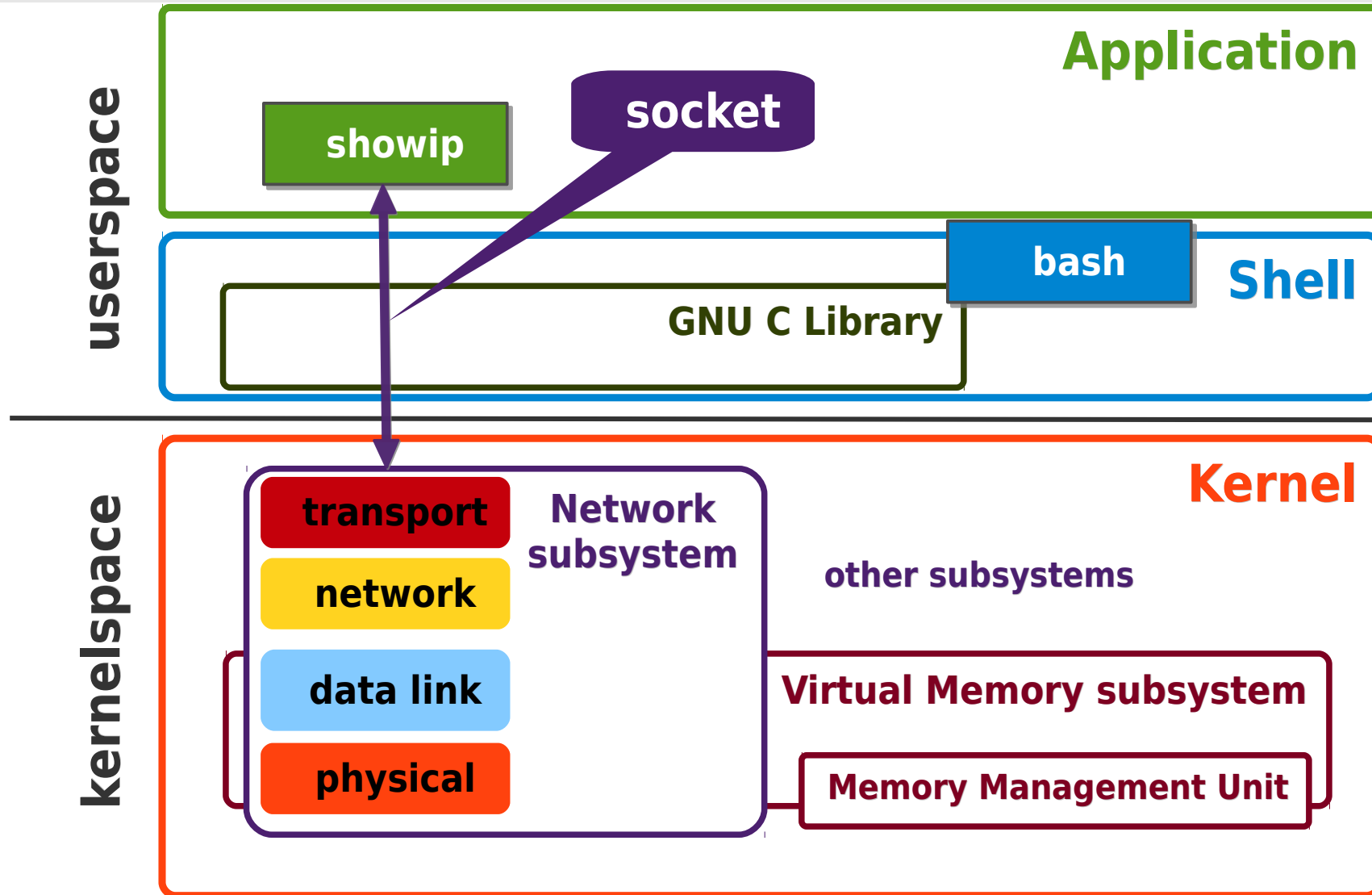
```
    if ((status = getaddrinfo(argv[1], NULL, &hints, &res)) != 0) {
        fprintf(stderr, "getaddrinfo: %s\n", gai_strerror(status));
        return 2;
    }
```

getaddrinfo → appel
système vers le sous-
système réseau

```
$ ./showip www.nic.fr
IP addresses for www.nic.fr:

IPv6: 2001:67c:2218:2::4:20
IPv4: 192.134.4.20
```

Programme showip.c



Quid de la représentation des données ?

- Dans l'espace utilisateur
 - **Faible** stress
 - Sous-système de gestion mémoire
 - Ramasse miettes
 - 'sizeof' → allocation indépendante de la cible
 - Arithmétique sur les pointeurs «transparente»
 - Alignement géré par le MMU
- Dans l'espace noyau
 - **Gros** stress
 - Pilotes de périphériques → pas de ramasse miettes
 - Accès aux registres de configuration → MMU
 - Fonctions logiques, décalages et masquages
 - Pointeur générique 'void'

