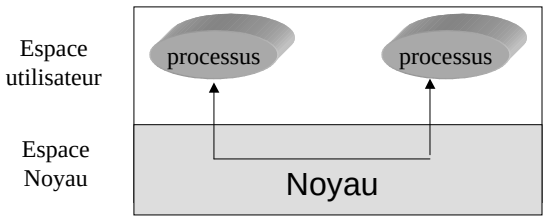
Les Sockets BSD

Hafid Bourzoufi

Université de Valenciennes
Institut des Sciences et Techniques de Valenciennes

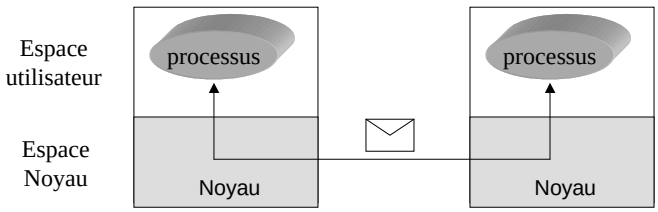
Les communications dans les systèmes centralisés (Rappel)



Les communications supposent
l 'existence d 'une mémoire partagée
Exemples : les pipes d 'unix, les IPCs système V

Les communications dans les systèmes répartis

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000



Les communications par envoi de message

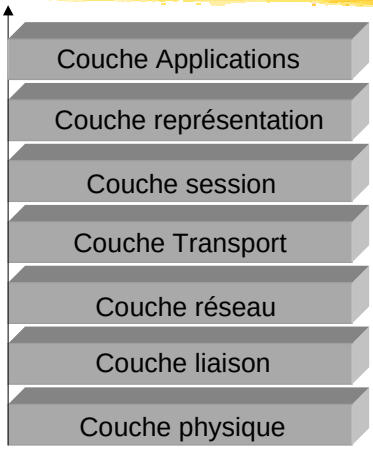
Protocole de communication

- La mise en œuvre des communications entre systèmes nécessite des protocoles de communication
- Protocole = un ensemble d 'accords sur
  - la représentation des bits
  - détection de la fin d 'un message
  - acheminement des message
  - représentation des nombres, des caractères
  - etc …

Il faut un accord à plusieurs niveaux, depuis les détails de bas niveau de la transmission des bits jusqu 'à ceux de plus haut niveau de la représentation des données

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000

La norme OSI *(Open System Interconnexion)*



Un ensemble de services standard : terminal virtuel, transfert de fichiers, ...

Codage des informations sous une forme standard

Gestion complète d 'une session de communications entre 2 utilisateurs.

Transport de l 'information entre deux processus (Multiplexage)

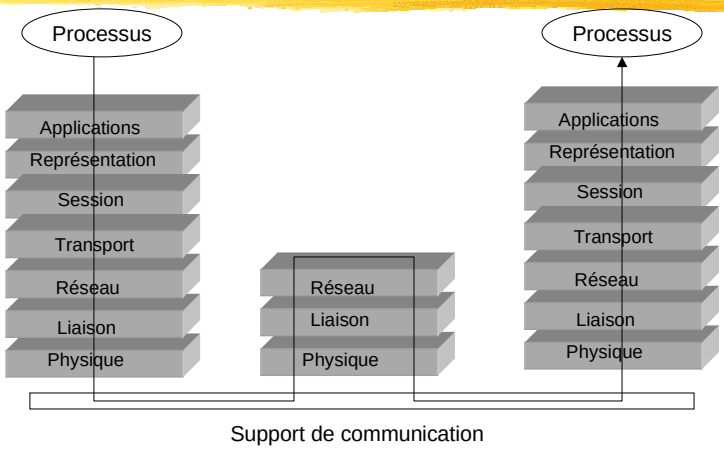
Transport de l 'information entre deux points du réseau (Routage)

Transport, sans erreur entre deux points, des blocs de données (trames)

Transport de l 'information comme une suite de bits

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000

Communication sous OSI



Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000

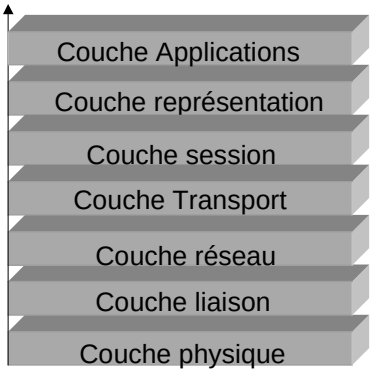
Les protocoles TCP/IP : Généralités

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000

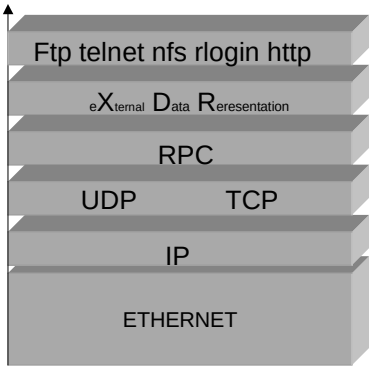
- Mis au point par l 'agence DARPA
- TCP/IP est un ensemble de protocole permettant l 'interconnexion de différents types de machines et de réseaux
- TCP/IP regroupe aussi UDP *(User Data Protocol)* : communications en mode non connecté
- Le réseau concerné à l 'origine était ARPANET (devenu INTERNET)
- Unix BSD 4.x premier a avoir intégré TCP/IP
- Contexte d 'utilisation
  - interconnexion des segments de réseaux
  - permettre aux machines non Unix d 'accéder aux services Unix

Le Modèle OSI vs TCP/IP

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000



Le modèle OSI



Monde Unix /Internet

Les sockets

- Les sockets = API *(Application Program Interface)*

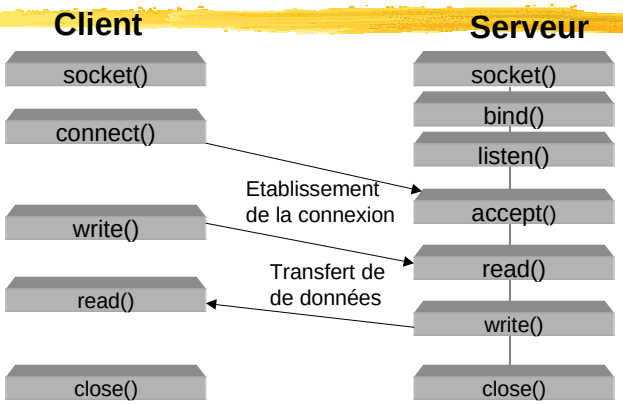
C 'est une interface entre les programmes d 'applications et les couches réseaux

- le terme Socket désigne aussi un canal de communication par lequel un processus peut envoyer ou recevoir des données
- Caractéristiques d 'une socket :
  - **domaine** : Unix (AF_INET) ou Internet(AF_INET)
  - **type** : datagram (SOCK_DGRAM) ou connecté(SOCK_STREAM)
  - **protocole** : UDP pour une socket de type SOCK_DGRAM
    TCP pour une socket de type SOCK_DGRAM
    il existe d 'autres protocoles
- dans le domaine AF_UNIX, il n 'y a pas de protocole

Utilisation des sockets en mode connecté

- Négociation de l 'établissement d 'un circuit virtuel
- **Côté Client (demandeur de la connexion)**
  - crée une socket
  - se connecte au serveur
  - lit et écrit dans la socket
  - ferme la socket
- **Côté Serveur**
  - crée une socket
  - associe une adresse à la socket
  - se met à l 'écoute des connexions entrantes
  - accepte une connexion entrante
  - lit et écrit sur la socket
  - ferme la socket

Utilisation des sockets en mode connecté



Adressage des sockets

- Une socket n 'est accessible par l ' extérieur que si on lui associé une adresse explicitement par ***bind()***
- Dans le domaine Unix :
  - une adresse = une entrée dans le système de fichiers
- Dans le domaine internet
  - une adresse = (adresse de la machine , **numéro de port)**
- les numéros de ports < 1024 sont réservés
  - ***Exemple de numéros réservés (ypcat services)***

    ftp : 21/tcp

    telnet : 23/tcp

Les sockets : Pratique

- Format général des adresses de sockets est représenté par le type générique

```
struct sockaddr {
    short sa_family;  /* domaine AF_UNIX, AF_INET */
    char sa_data[14]; /* adresse */
}
```

- Type associé aux adresses dans le domaine Unix:

```
#include<sys.un.h>
struct sockaddr_un {
    short sun_family; /* domaine AF_UNIX, AF_UNSPEC */
    char sun_path[128];   /* chemin  */
```


Les sockets : Pratique

- Type associé aux adresses dans le domaine Internet

```
#include <netinet/in.h>
struct in_addr {
    union {
      struct {u_char s_b1, s_b2,s_b3, s_b4;} S_un_b;
      struct { u_short s_w1, s_w2;} S_un_w;
      u_long S_addr;
    } S_un;
#define s_addr S_un.S_addr;
#include<netinet/in.h>
struct sockaddr_un {
    short sin_family; /* domaine AF_INET */
    u_short sin_port];   /*  port de la socket  */
    struct in_addr sin_addr; /* N° IP de la machine format réseau */
    char sin_zero[8];
};
```

Fabriquer des adresses internet

- Fabrication d 'adresse Internet dépend de son contexte d 'utilisation
  - associer l 'adresse fabriquée à une socket locale pour qu 'elle soit accessible de l extérieur
  - fabriquer l 'adresse d 'une socket éloigné
- Fabrication d 'une adresse destiné à être attachée localement (par *bind*)

```
void myselfadress( struct sockaddr_in *s_loc) {
    s_loc->sin_family = AF_INET;
    s_loc->sin_port = 0; /* le port sera alloué dynamiquement */
    s_loc->sin_addr.s_addr = INADDR_ANY; /* adresse jocker */
}
```


Fabrication de l'adresse de sockets

- **Socket éloigné**

```
void Raddress (struct sockaddr_in *s_r, const char *rhost, int port)
{
    struct hostent *h;
    s_r->sin_family = AF_INET;
    s_r->sin_port = port;
    if((h=gethostbyname(rhost))==0) {
                fprint(stderr,  "%s : machine inconnue ",rhost);
    }
    bcopy((char *)h->h_addr,(char *)s_r->sin_addr, h->h_length);
}
```

Primitives générales sur sockets

- **Création**

  #include <sys/types.h>
  #include <sys/socket.h>
  int s = socket(int **domaine**, int **type** , int **protocole**);
  - crée une socket et renvoie son nom interne
  - le nom interne est un descripteur de fichier

- **Associer une adresse à une socket**

  bind(int **s**, struct sockaddr ***loc_saddr**,int **loc_saddrlen**)
  - associe l 'adresse locale *loc_saddr à la socket s
  - loc_saddrlen est la taille de *loc_saddr

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000

Primitives spécifique au mode connecté

- **Demande de connexion**

  #include <sys/types.h>
  #include <sys/socket.h>
  connect(int **s**, struct sockaddr ***s_r**, int **s_rlen**);
  - obligatoire en mode connecté (SOCK_STREAM)
  - primitive bloquante

- **Ecoute**

  listen(int **s**, int **dc**);

  - dc indique le nombre maximun de demandes de connexion mises en attente

- **Acceptation de connexion**

  int accept(int **s**, struct sockaddr ***Ent_saddr**,int ***Ent_saddrlen**)

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000

Exemple:

- **Code Client**

```
#include <sys/types.h>
#include <sys/socket.h>
#include<string.h>
main(int argc, char *argv[])
{ int sd;
struct sockaddr_un adresseS;
sd=socket(AF_UNIX, SOCK_STREAM,0);
adresseS.sun_family=AF_UNIX;
strcpy(argv[1], adresseS.sun_path, sizeof(argv[1]));
if(connect(sd, (sockaddr *)&adresseS, sizeof(adresseS)) <0) exit();
write(sd,    "Hello Server", 12);
}
```

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000

- **Code Serveur**

```
#include <sys/types.h>
#include <sys/socket.h>
#include<string.h>
main(int argc, char *argv[])
{ int sd, ns, nb,char buf[256];
   struct sockaddr_un adresseS;
   sd=socket(AF_UNIX, SOCK_STREAM,0);
   adresseS.sun_family=AF_UNIX;
   strcpy(argv[1], adresseS.sun_path, sizeof(argv[1]));
   bind(sd, (sockaddr *)&adresseS, sizeof(adresseS));
   listen(sd,1);
   ns=accept(sd,(sockaddr *)&adresseClient, &fromlen);
   nb=read(ns, buff, sizeof(buf));
   write(1,buf,nb);
}
```

Hafid Bourzoufi, Université de Valenciennes - ISTV, 1998-2000