TD 6 - Les Sockets

Claude Duvallet

Université du Havre
UFR Sciences et Techniques
25 rue Philippe Lebon - BP 540
76058 LE HAVRE CEDEX
Claude.Duvallet@gmail.com

Introduction (1)

- Mécanisme de communication entre des processus appartement à des systèmes distants (machines).
- Une socket est à la fois :
  - une bibliothèque d'interface réseau,
  - et l'extrémité d'un canal de communication par lequel un processus peut émettre ou recevoir des données.
- Une socket est désigné par un descripteur (comme pour les fichiers).
- La bibliothèque socket masque l'interface et les mécanismes de transport des données.
- Tout service offert par un processus est identifié par un numéro de port. Certains services standards utilisent des ports connus et identiques sur toutes les machines (cf. fichier `/etc/services`).

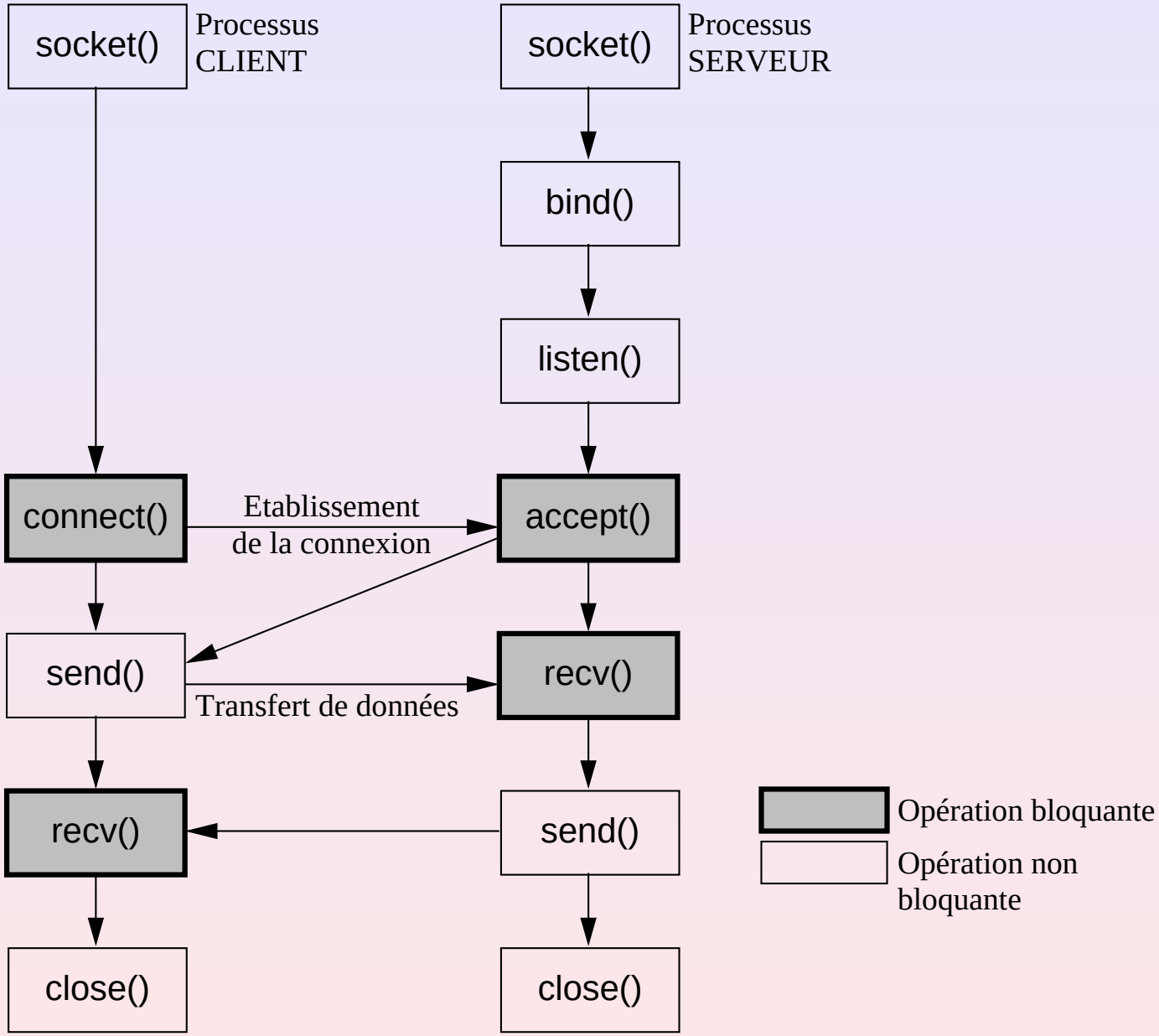
Introduction (2)

- Un processus peut s'attribuer un numéro de port à condition qu'il n'appartiennent pas à la plage 1-1024, réservée aux services systèmes et qu'il ne soit pas attribué à un autre service. Il est préférable de choisir un numéro supérieur à 5000.
- Une communication entre deux processus est repéré de manière unique par :
  - le `protocole transport` : soit UDP ou TCP.
  - l'`adresse1` : adresse internet de la machine 1.
  - le `numport1` : numéro de port du processus de la machine 1.
  - l'`adresse2` : adresse internet de la machine 2.
  - le `numport2` : numéro de port du processus de la machine 2.
- Cet ensemble est appelé **socket**.

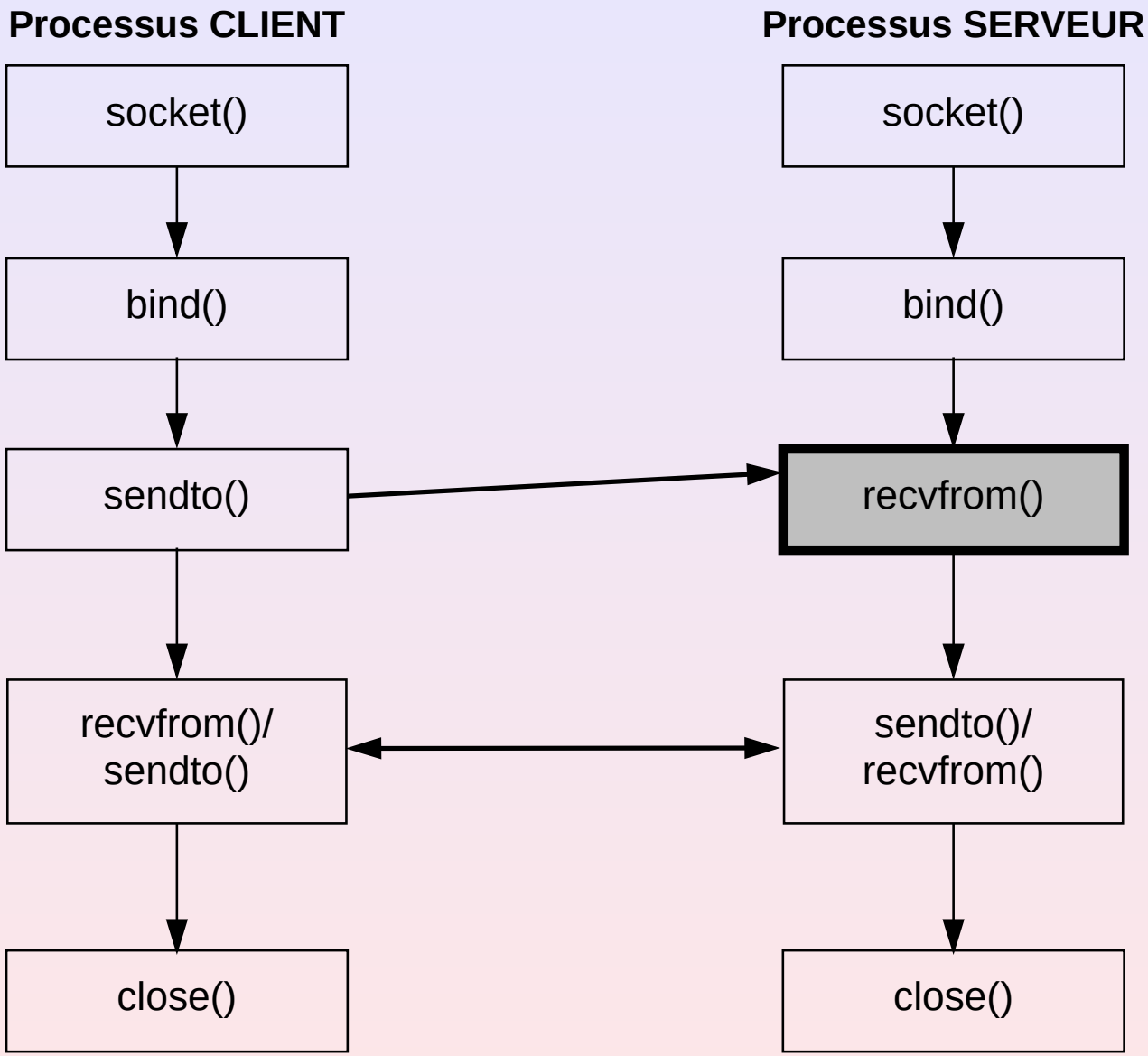
Mécanisme de mise en œuvre

- Modèle client/serveur.
- L'utilisation des sockets passent par la définition du domaine de travail mais aussi du type de communication.
- Les domaines de travail possibles sont :
  - `AF_UNIX` : en mode local sous Unix.
  - `AF_INET` : sur le réseau internet.
  - `AF_NS` : sur le réseau Xeros NS.
  - `AF_Decnet` : sur le réseau DecNet.
  - `AF_APPLETALK` : sur le réseau AppleTalk.
- Les types de communication possible :
  - `SOCK_DGRAM` : envoi de messages en mode datagramme.
  - `SOCK_STREAM` : envoi de messages en mode flot d'octets.
  - `SOCK_RAW` : envoi de messages en accédant à des protocoles de bas niveau (comme IP dans le domaine AF_INET).

Communication en mode TCP



Communication en mode UDP



Adressage dans le domaine AF_INET

```
#include<netinet/in.h>

struct in_addr{
  u_long s_addr; /*adresse internet d'une machine*/
}

struct sockaddr_in {        /* in comme Internet */
  short sin_family;         /* vaut AF_INET */
  u_short sin_port;         /* numero du port en ordre reseau */
  struct in_addr sin_addr; /* adresse internet de la machine en ordre reseau */
  char sin_zero[8];         /* champ inutilise */
}
```

Principaux fichiers d'en-tête nécessaires :

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
```

Création/Suppression d'une d'une socket

Création d'une socket :

```
int socket (
         int domaine,  /* AF_UNIX ou AF_INET */
         int type,     /* SOCK_DGRAM ou SOCK_STREAM */
         int protocole /* 0 pour le protocole par defaut */
                       /* IPPROTO_UDP pour le protocole UDP */
                       /* IPPROTO_TCP pour le protocole TCP */
);
```

On ne peut utiliser que le protocole UDP en mode datagramme (`SOCK_DGRAM`) et TCP en mode flot d'octets `SOCK_STREAM`.

Suppression d'une socket :

```
int close (int socket); /* socket = descripteur de la socket */
```

```
int shutdown (
  int socket,   /* descripteur de la socket */
  int controle  /* 0, 1 ou 2 */
);
```

- 0 : pour ne plus recevoir de données sur la socket ;
- 1 : pour ne plus envoyer de données sur la socket ;
- 2 : pour ne plus ni envoyer ni recevoir de données sur la socket.