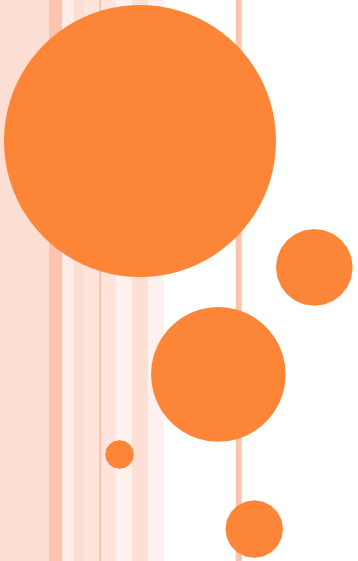
Communication Interprocessus (IPC)



Introduction

La communication interprocessus consiste à transférer des données entre les processus.

Exemple: un navigateur Internet peut demander une page à un serveur, qui envoie alors les données HTML.

La communication interprocessus peut se faire de différentes manières :

1. les files de messages (MSQ)
2. les segments de mémoire partagée
3. les sémaphores
4. les signaux
5. les tubes
6. Socket : communication (bidirectionnel) entre processus à travers un réseau

Définition et propriétés des MSQ

C'est une implantation UNIX du concept de boîte aux lettres, qui permet la communication indirecte entre des processus.

Les messages étant typés, chaque processus peut choisir les messages qu'il veut lire (extraire de la file).

```
struct msgbuf {
        long mtype;    /* type du message  >  0*/
        char mtext[100];   /* texte du message */
       };
```

Les informations sont stockées dans les files de messages en **DATAGRAM** (un message à la fois). Les files sont de type **FIFO** (**F**irst **I**n, **F**irst **O**ut).

➢Un processus peut émettre des messages vers plusieurs processus, par l'intermédiare d'une même file de message (**multiplexage**).

.

➢ L e mécanisme des files de message est externe au système de gestion de fichiers du système UNIX. Le système gère une table spécifique pour cet outil, qui est identifié par une clé. Cette clé est une valeur numérique et tout processus ayant connaissance de la clef peut utiliser la file de messages pour envoyer ou recevoir des messages.

Les primitives associées aux MSQ

1. Création et accès à une MSQ

La primitive msgget :

int msgid = msgget (int cle, int option)

permet de créer une nouvelle file de messages ou de récupérer une file de messages existante afin de l'utiliser çà à dire fournit l'identification de la MSQ de clé donnée.

- ✓ Le premier paramètre de la primitive, clé, permet de spécifier l'identifiant de la MSQ.
- ✓ Le second paramètre,option est une combinaison des constantes IPC_CREAT, IPC_EXCL et des droits d'accès associés à la file.
- ✓ La primitive en cas de succès retourne un identifiant interne au programme msgid.
- ✓ La récupération d'une file existante s'obtient en mettant l' option à 0.

Les primitives associées aux MSQ

- CREATION D'UNE FILE de clé 17 en lecture écriture pour tous

```
#define CLE 17
int msgid;
msgid = msgget (CLE, IPC_CREAT | IPC_EXCL | 0666);
```

- RECUPERATION D'UNE FILE de clé 17

```
#define CLE 17
int msgid;
msgid = msgget (CLE,0);
```

.

Les primitives associées aux MSQ

2. Envoi d'un message dans la file de messages

La primitive msgsnd :

int msgsnd (int msgid, struct message *buf, int lg, int option)

envoi du message **buf** de taille l**g** en octets dans la MSQ d'identifiant **msgid**.
Les options permettent de rendre l'envoi non bloquant si la file est pleine.

Les primitives associées aux MSQ

3. Réception d'un message depuis la file de messages

La primitive msgrv :

int msgrcv (int msgid, struct message *buf, int lg, long val, int option)

permet de recevoir un message depuis une file de messages. C'est à ce niveau qu'intervient le multiplexage dont nous avons parlé précédemment, c'est-à-dire que le récepteur peut désigner parmi les messages présents dans la file, un message qu'il souhaite plus spécifiquement recevoir.

- ✓ Le paramètre **val** spécifie le type de message à extraire.
- ✓ Les options permettent de rendre la réception non bloquante si la file ne contient pas de message avec le type attendu.

Les primitives associées aux MSQ

- Le multiplexage utilise le paramètre type des messages.
- Les interprétations possibles du paramètre **val** spécifié dans la primitive msgrcv sont les suivantes :
  - Si val > 0 alors le message le plus vieux dont le type est égal à **val** est extrait.
  - Si val = 0 alors le message le plus vieux, quel que soit son type est extrait.
  - Si val < 0, alors le message le plus vieux de type le plus petit inférieur ou égal à **|val|** est extrait.

Les primitives associées aux MSQ

4. Destruction d'une file de messages

La primitive msgctl :

int msgctl (int msgid, int op, struct msgid_ds *buf)

permet l'accès et la modification des informations contenues dans la table des files de messages gérée par le système.

Plus précisément, utilisée comme option la valeur **IPC_RMID**, elle permet la destruction de la file de message dont l'identifiant est passé en paramètre:

int msgctl (msgid, IPC_RMID, NULL)

Un exemple de programmation avec les MSQ

Processus1: crée la file et envoi un message dans cette file

/* fichier env.c */

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define CLE 20
struct  msgbuf  {
                    long mtype;
                    char mtext[100];
                   };
 struct msgbuf msgp;
 char *msg="ceci est un message";
```

Un exemple de programmation avec les MSQ

```
main()

{  int e,msgid;                                          /* identificateur msg */

   msgid = msgget(CLE, IPC_CREAT | 0777);                /* creation msg */

   msgp.mtype=12;                                        /* le type */

   strcpy(msgp.mtext,msg);                               /* le message */

   e=msgsnd(msgid, &msgp, strlen(msg), 0) ;              /* envoi message */

   if  (e==-1) printf("erreur envoie \n");
                else printf("msg  bien envoyé\n");
 }
```

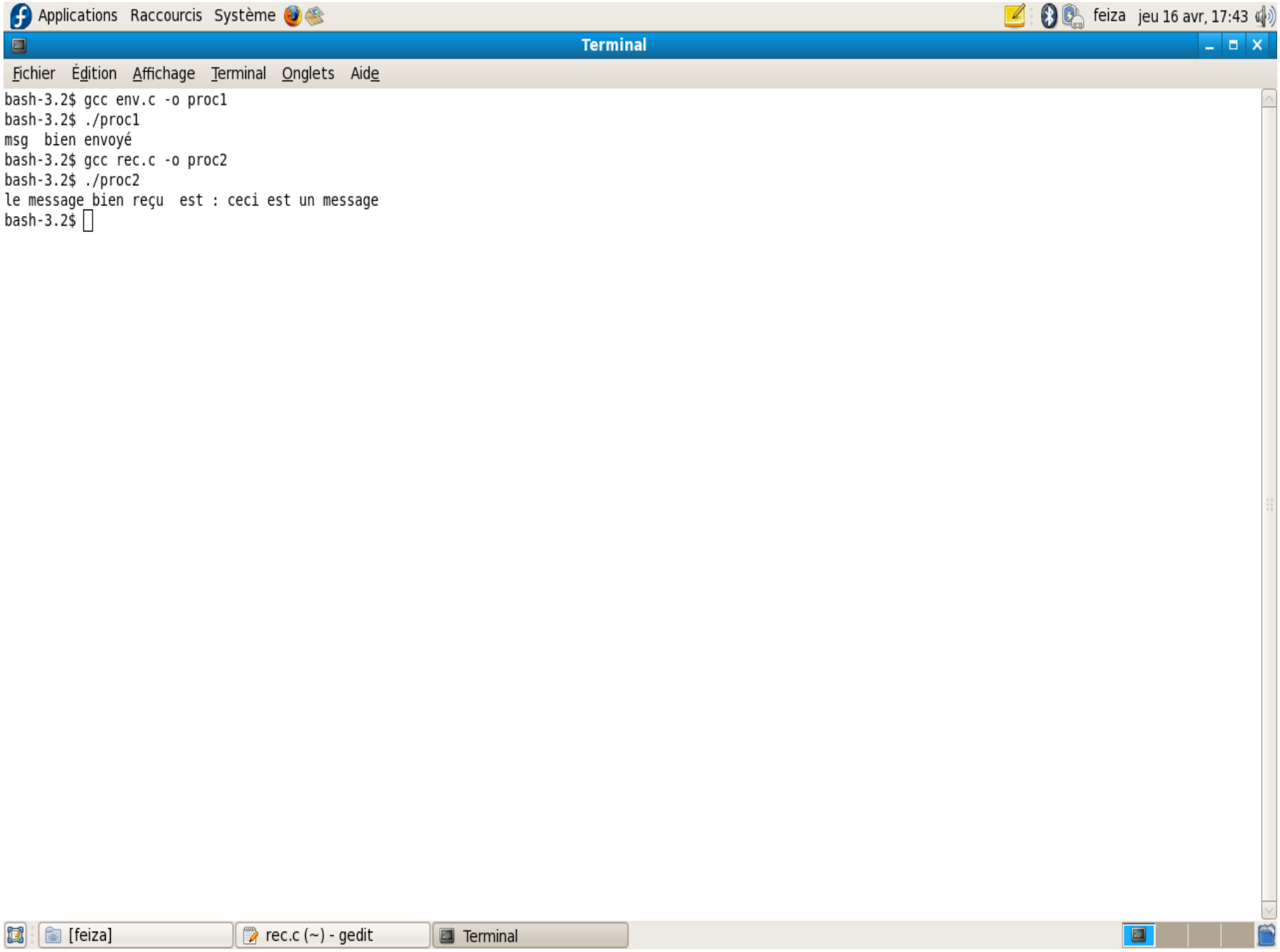
Un exemple de programmation avec les MSQ

Processus2: reçoit un message depuis la file 20 puis la détruit */
/* fichier rec.c */

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define CLE 20
struct  msgbuf  {
                  long mtype;
                  char mtext[100];
                 };
struct msgbuf msgp;
```

Un exemple de programmation avec les MSQ

```
main()
{   int r,msgid;

    msgid = msgget(CLE, 0);                                          /* récup msgid */

    r=msgrcv(msgid,&msgp,50,12,0);                                   /* lecture type 12*/

    if (r==-1) printf("erreur reception du msg\n");
                else printf("le message bien reçu  est : %s \n", msgp.mtext);

     msgctl(msgid,IPC_RMID, NULL);                                   /* destruction */

 }
```


Applications Raccourcis Système
feiza jeu 16 avr, 17:43
Terminal
Fichier Édition Affichage Terminal Onglets Aide
bash-3.2$ gcc env.c -o proc1
bash-3.2$ ./proc1
msg bien envoyé
bash-3.2$ gcc rec.c -o proc2
bash-3.2$ ./proc2
le message bien reçu est : ceci est un message
bash-3.2$
[feiza]
rec.c (~) - gedit
Terminal