Système d'exploitation et programmation système

Cours 5

EISTI

2008/2009

Département informatique

Communication Inter-processus

Files de messages

Files de messages

- C'est une implantation UNIX du concept de boîte aux lettres, qui permet la communication indirecte entre des processus.
- Fonctionnement FIFO
- Permet envoi et réception de messages typés
- Pas de lien de parenté nécessaire (récupération de l'identifiant de file grâce à la clé)
- Lecture destructrice
- Un processus peut émettre des messages vers plusieurs processus, par l'intermédiare d'une même file de message (**multiplexage**).
- Dans un message l'ensemble des informations doit être contigu en mémoire. Ceci interdit l'usage de pointeurs d'indirection.
- Les files d'attente de message fournissent des liaisons asynchrones normalisées. l'expéditeur et le récepteur ne soient pas contraints de s'attendre mutuellement.

Files de messages (suite)

Une file de message est définie par un entier positif unique (son *msqid*) et dispose d'une structure associée de type **struct** msquid_ds, définie dans <*sys/msg.h*> :

```
 struct msqid_ds
{
    struct ipc_perm msg_perm; /* gestion des droits d'acces  */
    ushort msg_qnum; /* nb messages dans la file */
    ushort msg_qbytes; /* octets maxi dans la file */
    ushort msg_lspid; /* PID dernier appel msgsnd */
    ushort msg_lrpid; /* PID dernier appel msgrcv */
    time_t msg_stime; /* heure dernier appel msgsnd */
    time_t msg_rtime; /* heure dernier appel msgrcv */
    time_t msg_ctime; /* heure dernière modification */
}
```

Messages

- Les messages étant typés, chaque processus peut choisir les messages qu'il veut lire (extraire de la file)
- Les messages sont toujours de type :
  - **struct** msgbuf {
    long mtype; //*type de message*
    char mtext[…]; /* *texte du message* ***/***
    int tab[4]; /* *texte du message (suite)* */
    … };
- Les informations sont stockées dans les files de messages en **DATAGRAM** (un message à la fois).
- Le type des messages est spécifié par un entier strictement positif
- La taille du message (msgsz) correspond à la taille du texte (sans l'entier (type de message)

Files de messages : utilisation

- Création de la file avec les droits permettant son utilisation (au moins lecture et écriture pour l'utilisateur)
- Récupération de l'identifiant de la file (de type entier)
- Envoi / réception de messages
- **msgget** : fournit l'identification d'une f.d.m. de clé donnée.
- **msgsnd** : utilisée par un processus pour envoyer un message sur la f.d.m. connue du processus.
- **msgrcv** : permet à un processus connaissant une f.d.m. donnée, d'y extraire des messages.
- **msgctl** : pour accéder et modifier les informations de la table des f.d.m.

Création d'une files de messages

#include <sys/types.h>

#include <sys/ipc.h>

#include <sys/msg.h>

int **msgget** (key_t *key*, int *msgflg*)

- Création / récupération de la file à partir d'une clé
- Renvoie l'identifiant de la file (interne au programme)
- *key* : n° de clé d'une fdm (existante ou non) ou **IPC_PRIVATE** (fdm non nommée)
- *msgflg* : Les droits et les paramètres de création
  - **IPC_CREAT**,
  - **IPC_EXCL** (création exclusive),
  - **0666** (droits d'accès)

Contrôle d’une file de message

- Permet d'intervenir sur la file (la plupart du temps pour la supprimer

  int msgctl (**int** *msqid*, **int** *cmd*, **struct** msqid_ds **buf*)

msqid : identificateur de la fdm visée

cmd : type d'opération à effectuer sur la fdm,

- IPC_RMID : Détruire la fdm
- IPC_STAT : Copier les informations depuis la structure représentant la file de messages dans la structure pointée par *buf*.
- IPC_SET : Ecrire la valeurs de certains champs de la structure **msqid_ds** pointée par *buf* dans la structure représentant la file de messages
- IPC_INFO,
- MSG_INFO,
- MSG_STAT

buf : pour récupérer la table associée à la fdm

msgctl renvoie 0 s'il réussit, ou -1 s'il échoue auquel cas **errno** contient le code d'erreur.

Emission de messages

```
# include <sys/types.h>
# include <sys/ipc.h>
# include <sys/msg.h>
```

int msgsnd (**int** *msqid*, **struct** msgbuf **msgp*, **int** *msgsz*, **int** *msgflg*);

msqid : identificateur de la fdm visée

msgp : le message à inclure dans la fdm

msgsz : taille de l'objet place dans la fdm

msgflg : option de fonctionnement du processus vis-à-vis de la fdm ;

- IPC_NOWAIT : ne bloque pas le processus en cas de file pleine
  c'est-à-dire, éviter l'attente active
- MSG_NOERROR : tronque le message si pas assez de place

Réception des messages

ssize_t msgrcv (**int** *msqid*, struct msgbuf **msgp*, size_t *msgsz*,
long *msgtyp*, **int** *msgflg*)

msqid : identificateur de la fdm visée

msgp : zone de récupération du message à lire dans la fdm

msgsz : taille maximale de la zone mémoire pointée par ***msgp***

msgtyp : type du message à lire (deux options sont possibles en réception);

- 0 signifie que l'on prend le 1er message de la fdm quelque soit son type (le plus ancien dans FIFO)
- ≠ 0 on reçoit le message le plus ancien correspondant au type donné

msgflg : option de fonctionnement du processus appelant

- **IPC_NOWAIT** évite l'attente passive
- Si aucun message n'est disponible, par défaut la lecture est bloquante

Exemple : headers

- **$ cat amphi.h**

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <stdio.h>

struct exemple {
        long id_msg;
        char nom[50];
        char tel[50];
};
const key_t KEY_MSQ = 1;
```

Exemple : création

```
#include "amphi.h"

int main (int argc, char ** argv)
{
    int msg_id;
    msg_id = msgget(KEY_MSQ, 0700 | IPC_CREAT);
    printf("ID msg : %d\n",msg_id);
    //boucle de reception de messages
    return 0;
}
```

Exemple : envoi

```
#include "amphi.h"
int main (int argc, char ** argv) {
    struct exemple m_msg;
    int msg_id;
    msg_id = msgget(KEY_MSQ, 0700 | IPC_CREAT);
    printf("ID msg : %d\n", msg_id);
    m_msg.msg_type = 1515;
    strcpy(m_msg.nom, "EISTI");
    strcpy(m_msg.tel, "0135251010");
    msgsnd(msg_id, &m_msg, sizeof(m_msg)-sizeof(long), 0);
}
```

Exemple : réception

```
#include "amphi.h"

int main (int argc, char ** argv) {
    struct exemple m_msg;
    int msg_id;
    msg_id = msgget(KEY_MSQ, 0700 | IPC_CREAT);
    printf("ID msg : %d\n", msg_id);
    msgrcv(msg_id, &m_msg, sizeof(m_msg)-sizeof(long), 1515, 0);
    printf("Recu : Nom:%s:\nTel:%s:\n", m_msg.nom, m_msg.tel);
}
```

FIN