

# Prova pratica di Calcolatori Elettronici (nucleo v6.\*)

*C.d.L. in Ingegneria Informatica, Ordinamento DM 270*

13 gennaio 2015

1. Introduciamo nel nucleo una versione semplificata del meccanismo dei socket per la comunicazione inter-processo, relativamente alla sola fase di connessione.

Un socket è un oggetto identificato tramite un numero naturale. È possibile creare una connessione tra due socket tramite le operazioni di `accept(natl id)` e `connect(natl src, natl dest)`. Un processo che esegue `accept()` su un socket  $s_1$  si pone in attesa di richieste di connessione verso  $s_1$ ; un processo può inviare una richiesta di connessione invocando `connect()` su un altro socket  $s_2$ , specificando  $s_1$  come destinazione. La `accept()` completa la connessione creando un nuovo socket,  $s_3$ , e ne restituisce l'identificatore al chiamante. A questo punto i socket  $s_2$  e  $s_3$  sono connessi e possono essere usati per scambiare dati tra i processi (cosa che non realizziamo), mentre il socket  $s_1$  è di nuovo disponibile per altri usi.

Su uno stesso socket ci può essere più di un processo in attesa di connessioni (più processi possono invocare `accept()` su uno stesso socket): in quel caso, le eventuali richieste verranno servite in base alla priorità dei processi in attesa. Per poter invocare `accept()`, il socket non deve però essere già connesso o essere sorgente di una richiesta di connessione in corso. Per poter invocare `connect()`, né il socket sorgente, né il socket di destinazione devono essere già connessi o sorgenti di altre richieste di connessione in corso; il socket destinazione deve essere in fase di accettazione, mentre il socket sorgente no.

Per realizzare questo meccanismo definiamo le seguenti strutture dati:

```
enum sock_state {
    SOCK_AVAIL,
    SOCK_ACCEPTING,
    SOCK_CONNECTING,
    SOCK_CONNECTED
};
struct des_sock {
    sock_state state;
    des_proc *connecting;
    des_proc *accepting;
};
```

I possibili stati di un socket sono i seguenti:

- **SOCK\_AVAIL**: il socket non è al momento utilizzato;
- **SOCK\_ACCEPTING**: c'è almeno un processo che sta accettando connessioni su questo socket;
- **SOCK\_CONNECTING**: un processo sta tentando di connettere questo socket (come sorgente) ad un altro;
- **SOCK\_CONNECTED**: il socket è connesso.

La lista **connecting** contiene i processi che stanno tentando di creare una connessione che ha questo socket come destinazione; la lista **accepting** contiene i processi che stanno accettando connessioni su questo socket.

Aggiungiamo inoltre le seguenti primitive (abortiscono il processo in caso di errore):

- **natl socket()** (già realizzata): crea un socket e ne restituisce l'identificatore (0xFFFFFFFF se non è stato possibile crearlo);
- **natl accept(natl id)** (da realizzare): pone il processo in attesa di connessioni sul socket **id**; restituisce 0xFFFFFFFF se il socket non è nello stato giusto; è un errore se il socket non esiste.
- **bool connect(natl src, natl dest)** (da realizzare): tenta di connettere il socket **src** con il socket **dest**; restituisce **false** se uno dei due socket non è nello stato giusto; è un errore se uno dei due socket non esiste.

Tenere conto di eventuali preemption. **Attenzione:** può essere necessario aggiungere informazioni ai descrittori di processo.