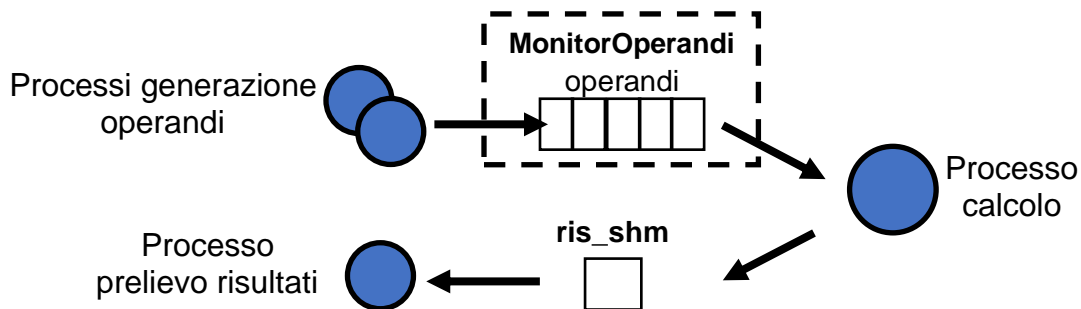


Testo della prova

Si realizzi in linguaggio C un'applicazione **multiprocesso** (vedi figura) per parallelizzare dei calcoli statistici, basata su costrutto **monitor signal and wait** e **semafori** e sullo **schema produttore-consumatore**.



In particolare:

- I processi *generazione operandi* (2 in totale), invocano ognuno 6 volte il metodo `inserisci_operando(...)` passando dei valori casuali tra 1 e 100. Il valore andrà inserito in un pool di buffer di **dimensione pari a 5**. Se tale pool risulta essere già pieno, il processo chiamante viene posto in attesa.
- Il processo di *calcolo* dovrà **prelevare 3 operandi** dal pool di buffer invocando il metodo `preleva_operandi(...)` (4 chiamate, per un totale di 12 prelievi). Tale metodo ritorna **tre valori da estrarre** (un array di 3 elementi) dal pool di buffer `operandi[]`. Se tale pool non contiene almeno 3 elementi, il processo chiamante viene posto in attesa. La sincronizzazione per l'accesso al pool di buffer deve essere disciplinato attraverso una **coda circolare**.

Dopo aver prelevato i 3 operandi, il processo di *calcolo* deve **effettuare il calcolo** della media, minimo, e massimo dei valori prelevati, attendere 2 secondi, ed inserire il risultato in una memoria condivisa (`ris_shm`) attraverso il metodo `inserisci_risultato(...)` (4 chiamate, per un totale di 4 produzioni). L'area di memoria condivisa deve essere disciplinata attraverso l'uso di semafori.

- Il processo di *prelievo del risultato* dovrà prelevare il risultato delle statistiche dalla memoria condivisa `ris_shm`, chiamando il metodo `preleva_risultato(...)` (4 volte in totale). Per ogni prelievo il processo dovrà stampare le statistiche a video.