Klasické problémy meziprocesové komunikace – producent-konzument aj.

Z FAV wiki

Meziprocesova komunikace: // tady by to chtělo rozvést a doplnit...

- Predavani zprav
- Primitiva send, receive
- Mailbox, port
- RPC
- Ekvivalence semaforu, zprav, ...
- Bariera, problem vecericich filozofu

Problem sdilene pameti

- umisteni objektu ve sdilene pameti
- Bezpecnost - globalni data pristupna kteremukoliv procesu bez ohledu na semafor
- Procesy bezi na ruznych strojich, komunikuji spolu po siti Reseni - predavani zprav

Obsah

Predavani zprav

- send, receive
- Zavedeme 2 primitiva
  - send (adresat, zprava) - odeslani zpravy
  - receive (odesilatel, zprava) - prijem zpravy
  - Send - Zprava (lib. datovy objekt) bude zaslana adresatovi
  - Receive - Prijem zpravy od urceneho odesilatele Prijata zprava se ulozi do promenne "zprava"

Synchronizace - blokující (synchronní) (recieve) / neblokující (asynchronní) (send)

- blokující send
  - čeká na převzetí správy příjemcem
- neblokující send
  - vrací se ihned po odeslání zprávy
  - většina systémů
- blokující receive
  - není-li ve frontě žádná zpráva, zablokuje se
  - většina systémů
- neblokující receive
  - není-li zpráva, vrací chybu

problémy, které nejsou u semaforů ani monitorů, zvláště při komunikaci po síti

- ztráta zprávy (potvrzení o přijetí (acknowledgement))
- ztráta potvrzení (číslování zpráv, duplicitní zprávy se ignorují)
- problém autentizace (zprávy je možné šifrovat)

lokální komunikace

- rendezvous - eliminuje frontu zpráv, send zavolán dříve než receive – odesílatel zablokován, vyvolán send i receive – zprávu zkopírovat z odesílatele přímo do příjemce
- využití mechanismu virtuální paměti paměť obsahující zprávu je přemapována z procesu odesílatele do procesu příjemce

typické aplikace typu klient - server

- WWW klient x WWW server (Apache, IIS)
- účetní aplikace x databázový systém (Oracle, MySQL)

Volání vzdálených procedur (RPC - Remote Procedure Call)

1. Klient zavolá spojku klienta, reprezentující vzdálenou proceduru
2. Spojková procedura argumenty zabalí do zprávy, pošle ji serveru
3. Spojka serveru zprávu přijme, vezme argumenty a zavolá proceduru
4. Procedura se vrátí, návrat. hodnotu pošle spojka serveru zpět klientovi
5. Spojka klienta přijme zprávu obsahující návrat. hodnotu a předá ji volajícímu

dnes nejpoužívanější jazyková konstrukce pro implementaci distribuovaných systémů a programů bez explicitního předávání zpráv

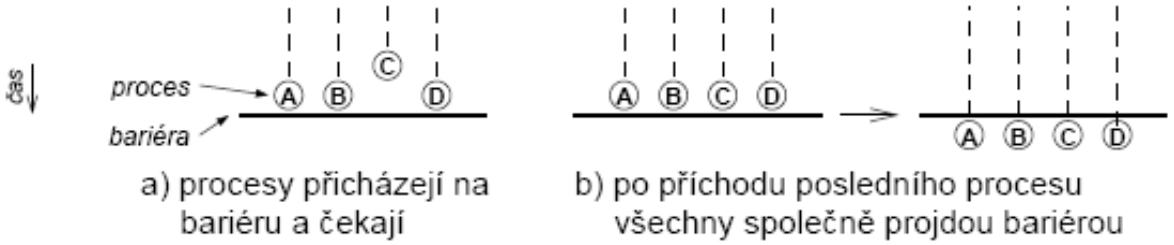
Ekvivalenty uvedených primitiv

Lze implementovat semafory pomocí zpráv a zprávy pomocí semaforů

Lze ukázat, že je možné implementovat

- Semafory pomocí monitoru
- Monitory pomocí semaforů

Bariéry



Producent-konzument

- Dva procesy sdílejí společnou paměť (buffer) pevné velikosti N položek
- Jeden proces je producent - generuje nové položky a ukládá je do vyrovnávací paměti
- Paralelně běží proces konzument, který data vyjímá a spotřebovává
- Procesy mohou běžet různými rychlostmi => musí být zabezpečeno, aby nedošlo k přetečení/podtečení:
  - Konzument musí být schopen čekat na producenta, nejsou-li data
  - Producent musí být schopen čekat na konzumenta, je-li buffer plný

Klasicky se zde sdílená paměť řeší pomocí pole, kde se z jedné strany položky přidávají a z druhé odebírají. Index kam zapisovat/odebírat počítáme modulo velikost pole. Kolizím zabráníme pomocí dvou semaforů, kde jeden obsahuje informaci o počtu plných prvků pole (f) a druhý o počtu prázdných prvků (e).

- přidání prvku: P(e), V(f)
- odebrání prvku: P(f), V(e)

Problém večeřících filosofů

- 5 filosofů sedí kolem kulatého stolu
- Každý filosof má před sebou talíř se špagetami
- Mezi každými dvěma talíři je vidlička
- Špagety jsou tak klouzavé, že filosof potřebuje 2 vidličky, aby mohl jíst
- Když filosof dostane hlad, pokusí se vzít 2 vidličky; pokud uspěje, nějakou dobu jí, pak položí vidličky a pokračuje v přemýšlení
- Problémy: uvíznutí(deadlock) (všichni filozofové zvednou levou vidličku, žádný z nich už nemůže pokračovat), vyhladovění (pokud by filozofové vzali najednou levou vidličku, budou běžet cyklicky - vidí, že pravá není volná, položí…)
- Řešení - Dijkstra (pomocí semaforů)

Problém čtenářů a písařů

- Představme si velkou databázi, množina procesů se pokouší souběžně číst a zapisovat
- Více požadavků čtení - akceptovatelné
- Při zápisu nesmí nikdo jiný přistupovat, ani žádní čtenáři
- První čtenář, který dostane přístup do databáze, provede P(w), další pouze zvětšují čítač
- Po skončení čtenáři zmenšují čítač, poslední provede V(w)
- Semafor w zabrání vstupu písaře, pokud jsou čtenáři
- Semafor w také zabrání vstupu čtenářům, je-li písař
- toto je s předností čtenářů

- nebo to jde udělat tak, že když chce vstoupit písař, tak už tam nepustí další čtenáře (s předností písařů)

Problémy

- Uvíznutí (Deadlock) - cyklické čekání dvou či více procesů na událost, kterou může vyvolat pouze některý z nich, nikdy k tomu však nedojde
- Vyhladovění (starvation) - proces se nedostane k požadovaným zdrojům

Kategorie: Fav-kiv-bzinf