Univerzita Hradec Králové
Fakulta informatiky a managementu

Management procesu II

Mgr. Josef Horálek

Vlákna

- Vlákna (Threads)
  - proces je definován množinou zdrojů výpočetního systému, které používá a umístěním, kde je spuštěn;
  - vlákno (thread) nazýváme lehký proces
    - jde o základní jednotkou využití procesoru, která obsahuje:
      - ukazatel programu;
      - sadu registrů;
      - zásobník;
  - kódová, datová část procesu a přidělené zdroje OS jsou sdíleny všemi vlákny procesu;
  - tradiční nebo těžký proces je úloha s jediným vláknem;

Vlákna

- Úloha nic nedělá
  - pokud jí nepřísluší žádné vlákno a každé vlákno může příslušet právě jedné úloze;
- Vlákno snižuje režii operačního systému
  - při změně kontextu;
  - při zajišťování multitaskingu;
- Vlákna sdílí nejen společný paměťový prostor i další struktury
- Přepínání vláken nevyžaduje výpomoc OS
  - nevyvolá přerušení;
  - přepnutí mezi uživatelskými vlákny není závislé na OS a tedy velmi rychlé;
- Jedno vláknové jádro OS
  - každé volání jádra každým procesem zastaví celou jeho úlohu, dokud jádro neodpoví;

Vlákna

- Vlákna se v mnohém chovají jako procesy
  - mohou se nacházet v některém ze stavů:
    - připraveno;
    - blokováno;
    - spuštěno;
    - ukončeno;
  - stejně jako procesy vlákna sdílí CPU a pouze jedno z nich může být spuštěno;
  - během procesu se spouštějí jeho jednotlivá vlákna;
    - každé vlákno má svůj ukazatel programu a zásobník;
  - vlákna mohou vytvářet synovská vlákna
  - vlákna mohou být zablokována systémem;
    - jestliže je jedno vlákno zablokováno, jiné může být spuštěno;

Vlákna

- Na rozdíl od procesu jsou vlákna na sobě závislá
- Všechna vlákna mají přístup na libovolnou adresu úlohy
  - vlákno může číst do libovolného zásobníku jiného vlákna téže úlohy;
  - vlákno může zapisovat do libovolného zásobníku jiného vlákna téže úlohy;
  - neprovádí se žádná ochrana jednotlivých vláken;
- Procesy vznikají od různých uživatelů
  - mohou být nepřátelské navzájem
  - vlákna jsou naprogramována tak, aby navzájem kooperovala;

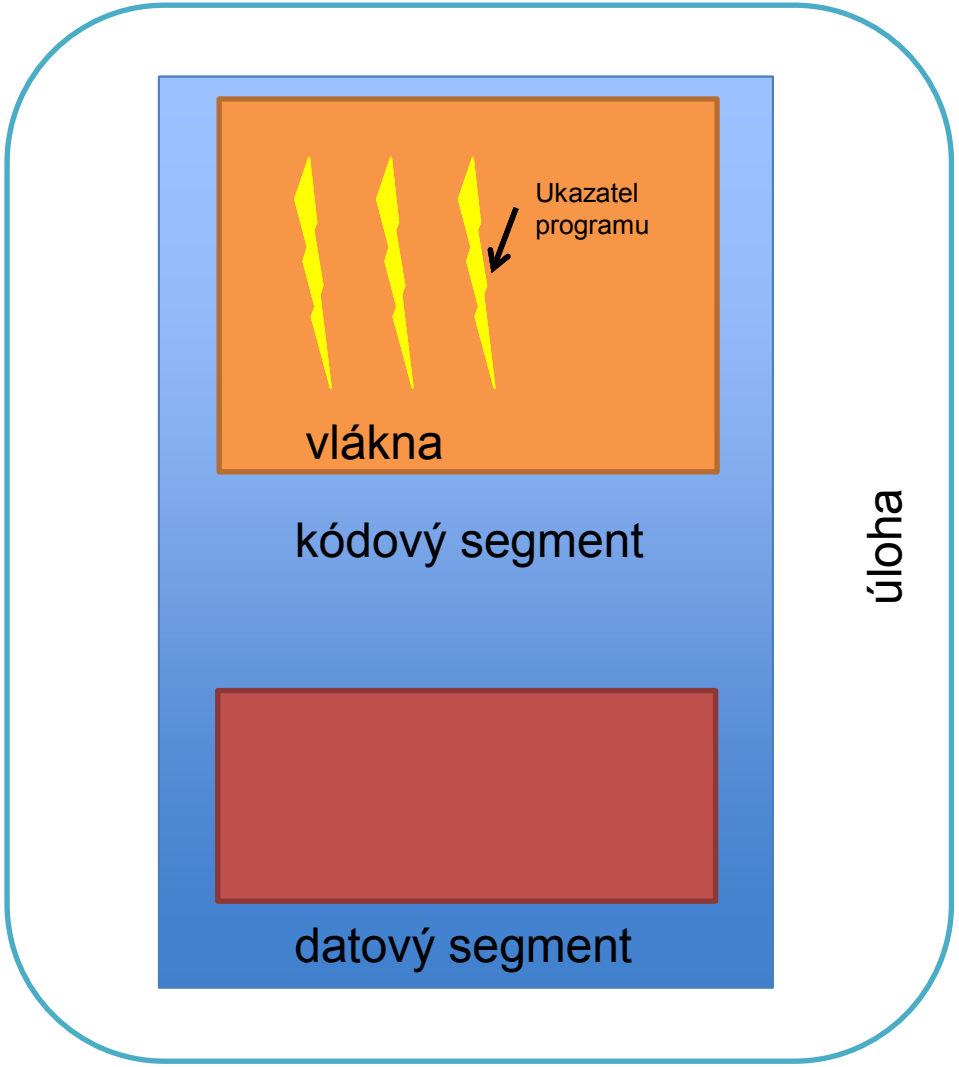
Vlákna

- Vlákna mohu být podporována jádrem
  - pak jsou volání jádra vykonávána souběžně s procesy;
  - tato volání mohou být také podporována jádrem přes množinu volání knihoven na uživatelské úrovni;
  - vlákna na uživatelské úrovni nepoužívají jádro
    - proto jsou při přepínání mezi sebou rychlejší než vlákna podporovaná jádrem;
  - jakékoli volání jádra způsobí zastavení celého procesu;
    - protože jádro plánuje jen procesy, o existenci vláken nemá potuchy a proces, který je čekající nemá naději na přidělení CPU;

Vlákna

- Shrnutí:
  - vlákna jádra mají pouze malé datové struktury a zásobník
  - přepínání mezi nimi nevyžaduje přistup do paměti a je poměrně rychlé;
  - uživatelské vlákno obsahuje pouze zásobník a ukazatel programu;
  - nepotřebuje žádné zdroje jádra;
  - jádro se nepodílí na plánování uživatelských vláken;
    - přepínání mezi uživatelskými vlákny je rychlé;

Vlákna


Ukazatel programu
vlákna
kódový segment
datový segment
úloha

FIM UHK

Komunikace procesu

- IPC (interprocess-communication)
  - umožňuje procesům komunikovat a synchronizovat své akce;
- Základní struktura:
  - systém zpráv musí umožňovat minimálně dvě operace:
    - send(zpráva);
    - receive(zpráva);
  - zprávy doručované systémem mohou být proměnné nebo pevné délky;

FIM UHK

Komunikace procesu

- Procesy P a Q chtějí komunikovat
  - musí být navzájem schopny přijímat a vysílat své zprávy;
  - musí mezi nimi existovat komunikační spojení;
    - spojení může být navázáno i mezi více než dvěma procesy;
    - spojení je nesměrové;
- Možné logické implementace spojení:
  - přímá nebo nepřímá komunikace;
  - symetrická nebo asymetrická komunikace;
  - automatické nebo explicitní bufferování;
  - posílání kopie nebo posílání odkazu;
  - zprávy pevné nebo proměnné délky;

Komunikace procesu
Přímá komunikace

- V tomto typu komunikace musí každý proces znát jméno příjemce
  - v tomto schématu jsou definovány funkce:
  - send(P, zpráva);
    - pošli zprávu procesu P;
  - receive(Q, zpráva);
    - přijmi zprávu od procesu Q;

- Komunikační spojení má následující vlastnosti:
  - spojení je vytvořeno mezi právě dvěma procesy;
  - spojení je automaticky povoleno mezi každými dvěma procesy;
  - pro komunikaci musí procesy pouze navzájem vědět svou identifikaci;
    - mezi každými dvěma procesy může existovat maximálně jedno spojení;
  - spojení může být nesměrové, ale většinou je dvousměrové;

Komunikace procesu
Přímá komunikace

- Komunikace:
  - oba procesy jsou spuštěny současně a konzument zpracovává položku zatímco producent již vytváří jinou;
  - v okamžiku kdy ji vytvoří pošle ji pomocí funkce send příjemci;
  - ten ji přijme prostřednictvím funkce receive;
  - není- li vyprodukována žádná zpráva, je producent ve stavu čekající;
  - oba musejí znát navzájem své identifikace, aby mohli komunikovat;
  - nebo lze použít asymetrické adresování;
    - pouze producent zná adresu příjemce;
    - příjemce adresu producenta znát nemusí;
  - nevýhodou obou těchto schémat komunikace (symetrické i asymetrické) je limitovaná modularita konečné definice procesu;

- V této komunikaci si producent a příjemce vyměňují zprávy pomocí schránky (mailbox)
  - někdy označované port;
  - schránku můžeme chápat jako objekt;
    - do objektu mohou procesy vkládat zprávy;
    - z objektu mohou být zprávy procesy vyzvedávány;
  - každá schránka má jedinečnou identifikaci;
    - proces může komunikovat s jinými procesy prostřednictvím množství různých schránek;
    - dva procesy mohou komunikovat pouze sdílí-li společný mailbox;
  - funkce send a receive mají následující hlavičku:
    - send(A, zpráva);
      - pošli zprávu do schránky A;
    - receive(A, zpráva);
      - přijmi zprávu ze schránky A;

- V tomto případě má komunikační spojení následující charakteristiky:
  - spojení je navázáno mezi dvěma procesy pouze mají-li sdílenou schránku;
  - spojení může být navázáno mezi více než dvěma procesy;
  - mezi každým párem komunikujících procesů může být navázáno více spojení;
    - každé spojení vyžaduje jednu schránku;
  - spojení může být nesměrové nebo dvou směrové;

- Uvažujme, že procesy P1, P2 a P3 sdílí společnou schránku A
  - P1 pošle zprávu do schránky A;
    - zatímco procesy P2 a P3 mají spuštěnou funkci receive ze schránky A;
  - který proces přijme zprávu od P1?
    - otázku můžeme řešit více způsoby:
      - povolit spojení mezi více než dvěma procesy;
      - povolit nejvýše jedno spuštění operace receive v čase;
      - povolit systému, aby vybíral, který proces zprávu obdrží;

- Vlastníkem schránky může být buď OS nebo proces
  - každá schránka má jedinečného vlastníka;
  - jak se stanovuje vlastník a uživatel schránky?
    - zavést typ proměnné schránka;
    - schránky ve vlastnictví operačního systému;
    - proces, který schránku vytvořil je implicitně jejím vlastníkem;
      - jedině on může do schránky zasílat zprávy;
    - procesy mohou sdílet schránku prostřednictvím techniky tvorby procesu;

Komunikace procesu
Buffery

- Kapacita spojení je dána množstvím zpráv, které mohou čekat v záloze, než si je příjemce odebere
  - spojení je možno chápat jako frontu zpráv;
  - pro implementaci fronty jsou možné tři základní způsoby:
    - fronta s nulovou kapacitou;
    - fronta s omezenou kapacitou;
    - fronta s neomezenou kapacitou;
- Existují další výjimečné formy komunikace procesu:
  - proces, který produkuje zprávy nikdy nečeká;
  - proces, který vyslal zprávu čeká dokud nedostane odpověď;

- Proces byl ukončen:
  - odesilatel nebo příjemce může být ukončen dříve, než byla zpracována zpráva;
    - mohly vzniknout zprávy, které nikdy nebudou přijaty;
    - zablokované procesy čekající na zprávu, která nikdy nebude odeslána;
    - dvě možné situace:
      - Proces P čeká na zprávu od procesu Q, který byl zrušen;
      - Proces P odeslal zprávu procesu Q, který byl zrušen;

- Ztracená zpráva
  - zpráva odeslaná procesem P procesu Q se ztratila kdesi v síti;
    - chybou hardwaru nebo komunikačního spojení;
  - základní metody ošetření této události:
    - OS je schopen detekovat tuto událost a znovu poslat zprávu;
    - odesílající proces je schopen detekovat tuto událost a znovu odeslat zprávu, je-li to zapotřebí;
    - OS je schopen detekovat tuto událost a upozorní odesílající proces, že zpráva byla ztracena;
      - odesílající proces potom může udělat co uzná za vhodné;

- Porušená zpráva
  - zpráva byla doručena příjemci
    - cestou došlo k jejímu poškození;
    - důsledky jsou totožné se ztracením zprávy a většinou OS znovu posílá originál porušené zprávy;
    - mechanismus detekce poškození spočívá v kontrolním součtu zprávy (parita nebo CRC);

Univerzita Hradec Králové
Fakulta informatiky a managementu

Děkuji za pozornost…