KIV / ZOS

Dodatky

Seznam dodatků

- I. Alokace paměti pro procesy
- II. Principy vstupně výstupního hw

Imperativní programovací jazyky

- statické proměnné
  - vyhrazena při spuštění programu
- lokální proměnné procedur a fcí
  - alokace na zásobníku
- dynamická alokace paměti
  - v oblasti hromada (heap)
  - pomocí služeb OS

Proces

- mapování po spuštění procesu
  - kód, statické proměnné, zásobník
  - není mapována veškerá adresovatelná paměť
  - 32bit systémy 2-3GB virt.pam. (zbytek OS)
- žádost procesu o paměť
  - knihovní funkce (alokátor paměti)
- zvětšení hromady
  - požádá OS

32bit OS

32bit OS, 2^32 = 4GB

2+2 Win: proces+OS

3+1 Win: prepinac /3GB

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(2)\WINNT
[operating systems]
multi(0)disk(0)rdisk(0)partition(2)\WINNT="????"
  /3GB
```

PAE (Physical Address Extensions)

podpora až 64GB fyzické paměti pro aplikace na IA-32 platformách – přepínač /PAE

Win2000, 32bit Win XP .. 4GB
Win Server 2003(8) .. 64GB

64bit Windows nepodporují PAE

expanze z 32bitů na 36bitů

2^4 = 16, 16x více, tj. 4x16 = 64GB

page directory, page tables na 8B
24 bitů místo 20 bitů

Explicitní správa paměti (C, C++, Pascal ..)

- *malloc()* a *free()* v C, *new delete* v C++
- alokátor paměti
  - spravuje hromadu
    - současná nestačí – požádá OS o další úsek virt.pam.
  - alokace např. metodou first fit
- problémy
  - nezapomenout uvolnit paměť, když není potřeba
  - roztroušení malloc, free v kódu

Snaha vyřešit správu paměti jiným způsobem,
automatickou správou paměti

Čítání referencí

- např. Perl, Python
- každá datová struktura
  - položka **počet referencí**
  - při snížení referencí – test == 0 uvolnění
  - počty referencí udržovány automaticky
- nevýhody
  - čas při vytvoření / zrušení odkazu
  - cyklická datová struktura – neuvolnění paměti

Garbage collection (GC)

- automatická detekce a uvolnění neodkazované paměti
- např. samostatné vlákno
  - spuštěno při *dostupná paměť* < *limit*
- součást virtuálních strojů (JVM, CLR)
  - potřebuje rozumět obsahu datových struktur

GC

- výhody
  - usnadnění pro programátora, redukce chyb
- nevýhody
  - plně v režii virtuálního stroje
  - aktivace GC v nevhodných okamžicích
    - explicitní spuštění GC
  - nepoužívaná paměť může zůstat alokována, pokud nenastavíme odkaz na null

GC: Mark-and-sweep

- první průchod
  - vyhledává dostupnou paměť
  - začne od globální proměnné, lok. proměnné procedur a funkcí (kotevní objekty)
  - odkazy – postupuje dále na odkazovaná data
  - označení – bit „objekt je dostupný“
- druhý průchod
  - neoznačená alokovaná paměť je uvolněna
- nárazové citelné zpomalení systému

GC: Baker Collector

- inkrementální verze
- paměť na 2 semiprostory
  - jeden aktivní, v něm vytvářeny nové objekty
- po určitém čase
  - jako aktivní označen druhý semiprostor
    - projde původní "mark-and-sweep", místo označení prostor evakuuje do nového semiprostoru
    - na místě původního objektu – náhrobní kámen s novou adresou

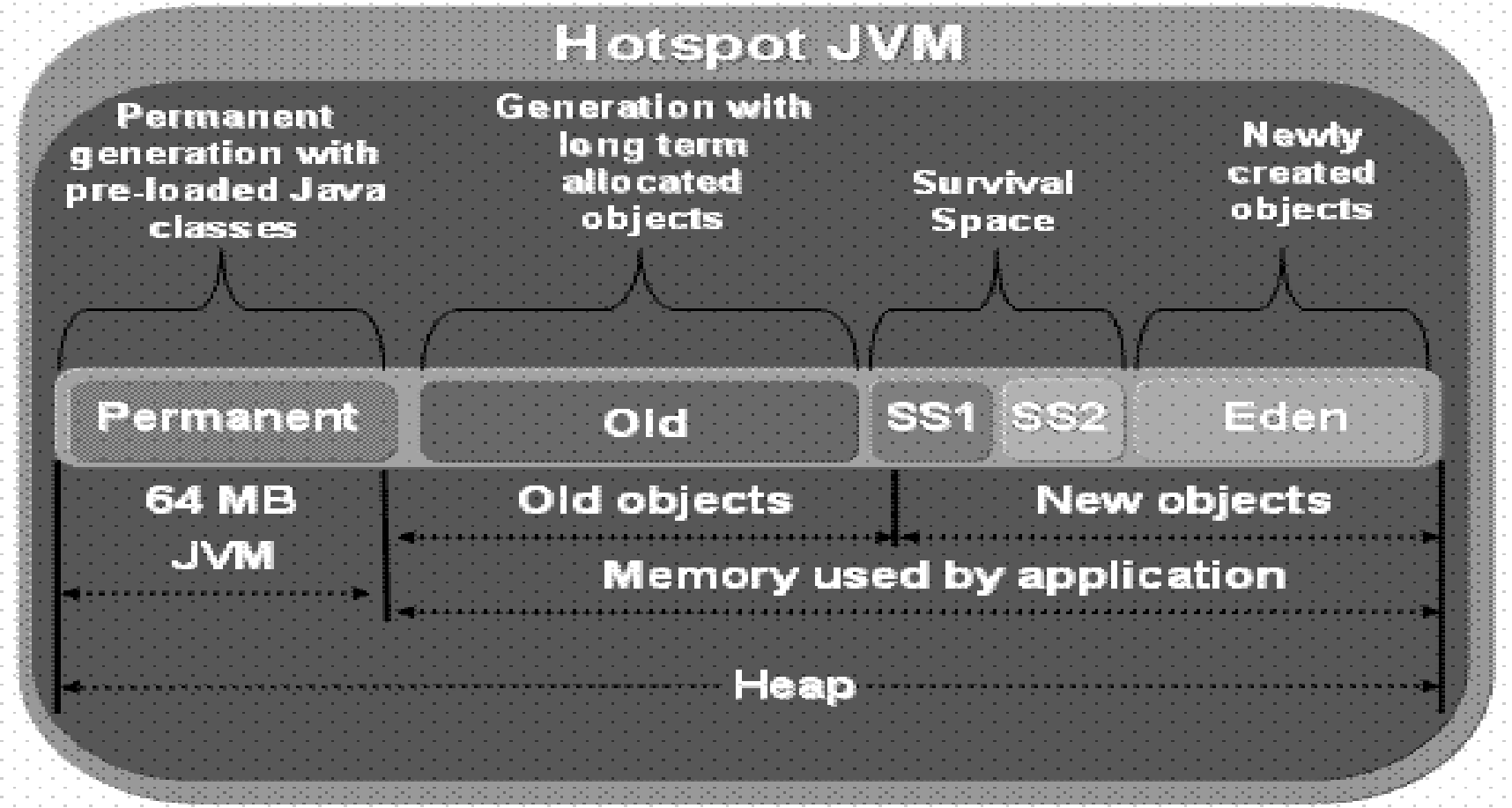
GC: Baker Collector cont.

- původní semiprostor
  - po průchodu jen smetí, lze zrušit
- nové alokace před úplnou evakuací starého semiprostoru
  - hned přesunout všechny odkazy z nového objektu
- výsledkem – čistý semiprostor
- inkrementální algoritmus
  - řeší nárazovou aktivaci mark-and-sweep

Další GC

- nevýhoda Baker C.
  - neustále přesouvá objekty – nákladné
- generační GC
  - CLR (.NET) – třígenerační GC
  - mnoho objektů krátká doba života, některé ale NE
  - přesun objektu
    - zvýšeno číslo generace objektu
    - objekt přežije N generací – do privilegované oblasti, kde není tak často zpracováván GC

Hotspot JVM


Hotspot JVM
Permanent generation with pre-loaded Java classes
Generation with long term allocated objects
Survival Space
Newly created objects
Permanent
Old
SS1
SS2
Eden
64 MB JVM
Old objects
New objects
Memory used by application
Heap

Odkazy

- http://programujte.com/index.php?akce=clanek&cl=2006060902-architektura-microsoft-net-framework-%96-3-dil

- http://www.skilldrive.com/book/DOTNETinSamples.htm

II. Principy vstupně výstupního hw (pAio.pdf)

1. CPU řídí přímo periferii
2. CPU – řadič – periferie
   aktivní čekání CPU na dokončení operace
3. řadič umí vyvolat přerušení
4. řadič umí DMA
5. I/O modul
6. I/O modul s vlastní pamětí

Magnetické disky

- geometrie
  - několik ploten – každá dva povrchy
  - stopa (track)
  - stopa rozdělena do sektorů (sectors)
    - nejmenší velikost dat číst či zapisovat
  - všechny stopy pod sebou – cylindr
    - lze přistupovat bez přesunu hlav

Disky

- disková adresa (povrch, stopa, sektor)
- virtuální geometrie
  - aby nebyla nižší hustota na vnějších stopách
- logická adresa
  - sektory číslovány od 0 bez ohledu na geometrii disku

- logická adresa = číslo sektoru