Implementace dávkových operací

Petr Steckovič

Dávkové zpracování dat

- Procesy běžící na pozadí
- Spouštěné
  - Časem
  - Stavem (např. dochází místo)
  - Ručně
- Obvykle se jedná o podpůrné funkce systému

exspirace, mazání, archivace, zálohování, synchronizace, párování, generování reportů, výpočty, grupování, rozesílání emailů, vyhledávání, stahování, aktualizace …

Specifika dávkových operací

- Zpracovávají velké množství dat
- Nezanedbatelná doba běhu
- Nezanedbatelná zátěž systému
- Krátkodobý výpadek obvykle nevadí
- Zanedbávají se při testování
- Zanedbává se logování / monitoring

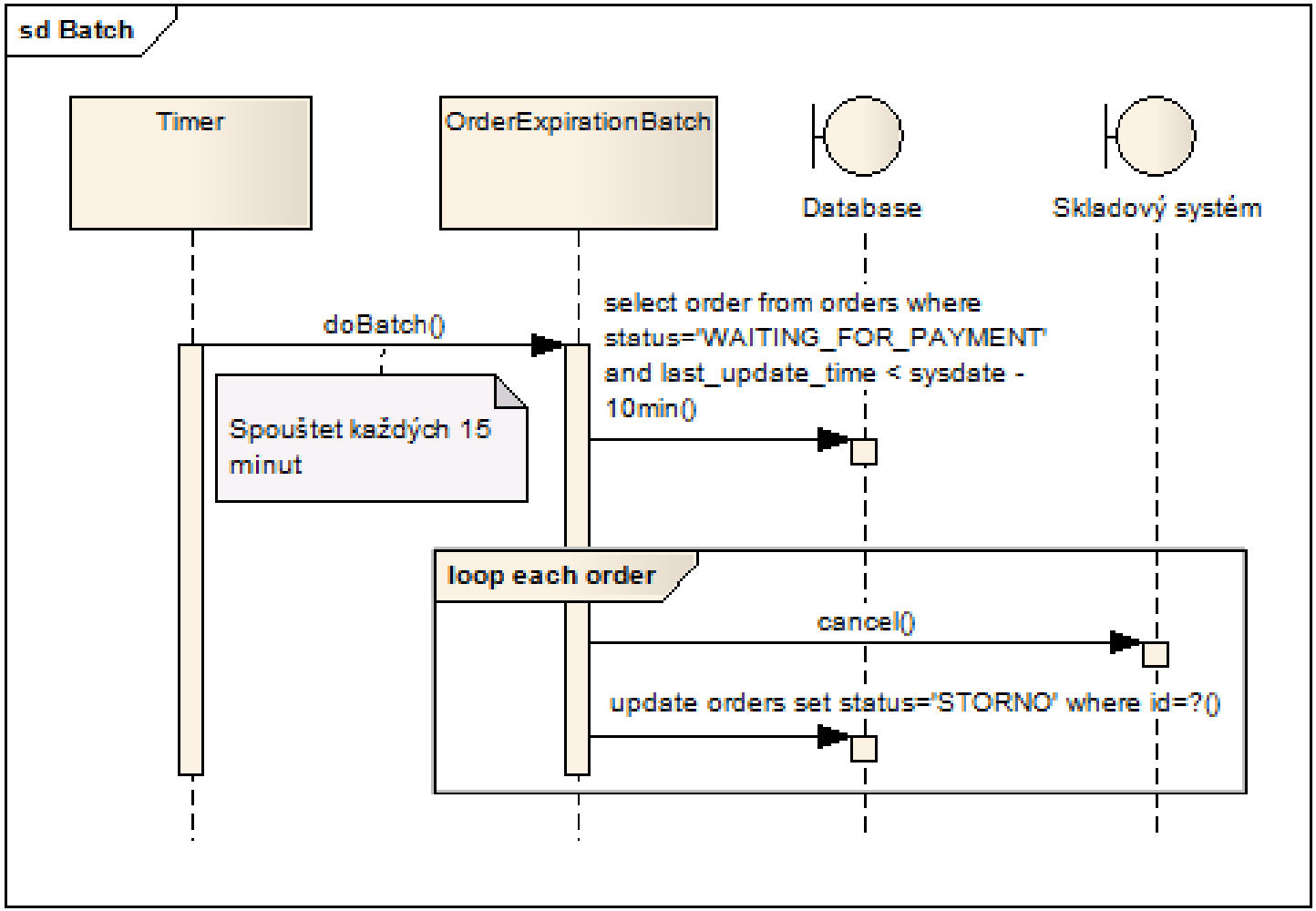
Fáze dávkové operace

- Získání dat pro zpracování
  - validace!!!!
- Vlastní zpracování
- Persistence zpracovaných dat

Úryvek ze specifikace

„... pokud zákazník nepotvrdí platbu do 10 minut, pošle se do skladového systému příkaz ke zrušení rezervace zboží a po té se objednávka automaticky stornuje …“

Zadání pro programátora



Anketa

Za jak dlouhou naprogramujete tuto funkcionalitu ve vašem frameworku?

Spouštění časem

- Implementace časovačem (Quartz, EJB Timer, JDK Timer)
- JavaEE5 – velmi špatná podpora
- JavaEE6
  - Rozšíření o cyklické timery
  - Daný čas / časový interval
  - @Schedules, @Schedule
  - Možnosti vytvořit programově

Problémy

- Spouštění do sebe - @Singleton
- Změna konfigurace za běhu
  - @Schedule ☹
- Pád časovače runtime výjimkou
- JavaEE5 časovače – přeplánování
- Redelivery / Persistence
- Spuštění ihned po startu serveru
- Lokální / globální časovač přes cluster

Ruční spouštění

- Vhodné vždy i když není v zadání
- Dlouhotrvající operace = timeouty rozhraní
- Vhodná implementace
  - start operace
  - sledování běhu (alespoň běží-neběží)
- Pozor na vícenásobné spuštění
  - interakce s timerem
  - dvojité spuštění přes rozhraní

Načítání dat

- Vždy limitovat počet objektů, které se načítají
- Pozor na výpočty „vzniká 10 záznamů za minutu => nepotřebuji limit“
- Problém načítacího okna
  - validační chyba
  - umět automaticky vyřadit záznam ze zpracování
  - umožnit návrat záznamu do zpracování

Transakční schéma 1

```
BEGIN TRANS
WHILE (moreDataExists) {
data<D> = load(1);
 x = process (data);
 store (x);
}
END TRANS
```

Transakční schéma 2

```
WHILE (moreDataExists) {
BEGIN TRANS
 data<D> = load(1);
 x = process (data);
 store (x);
END TRANS
}
```

Transakční schéma 3

```
WHILE (moreDataExists) {
BEGIN TRANS
 data<D> = load(max_items);
END TRANS
for (D d : data) {
 BEGIN TRANS
 x = process (d);
 store (x);
 END TRANS  // commit interval ? > 1
}}
```

Chybové stavy

- Permanentní - chyba nastává vždy znovu
- Dočasné - po zopakování nenastane
- Transakční / netransakční zdroje
- Řešení
  - Restart při dočasné chybě
  - Čítače špatných pokusů (po n neúspěšných pokusech je to permanentní chyba)
  - Při permanentní chybě vyřadit (záznam) z dalšího zpracování
  - Netransakční zdroje jako poslední

Paralelní dávkové operace

- Některé operace se musí provést právě jednou
- Distribuce práce
- Mutex
- Partisioning
- Problém řídících příkazů v clusteru
  - Abort – musí zrušit všechny instance DO apod.

Mutex

- Mutex = vzájemné vyloučení
- Jednoduché
- Obvyklá implementace = DB zámky na úrovni typu dávkové operace
- Nevyužívá výkon všech strojů v clusteru
- V případě, že délka operace je srovnatelná s intervalem spuštění = jede pouze jeden
- Sklony k divergenci při přetížení

Partišnování

- Rozdělit data do disjunktních částí (např. where podmínkou)
- Není potřeba zámkování ☺
- Problém statistické distribuce
- Nevázat natvrdo se strojem !!! (např. nodeId)
- Nedodržuje pořadí mezi partition

Zámkování jednotek práce

- Implementace rozděl a panuj
  - Zámkování na úrovni záznamu
  - Dynamické vytvoření partition (registrace partition)
  - Problém statistické distribuce obvykle nevýznamný
- Lze dosáhnout velké úrovně paralelizmu
  - paralelizmus na úrovni vláken
  - paralelizmus na úrovni strojů

Implementace zámků

- Paměťové
  - rychlé, nejsou vhodné do clusteru
  - zámkování vláken na jednom stroji
- Databázové
  - vhodné do clusteru,pomalejší
  - !!! persistentní !!!!
  - Expirace (expirace >> maximální doba běhu)
  - Vlastní transakce, pesimistický přístup

Monitoring

- Vhodná technologie – JMX / Web
- Čas posledního spuštění
- Čas příštího spuštění
- Status poslední operace (success / failed)
- Doba trvání výkonné operace
- Metriky zpracovaných dat
- Metody pro ruční řízení

Logování

- Log by měl obsahovat
  - Začátek dávky (spuštěno ručně / timerem)
  - Pro každý zpracovaný záznam primární klíč
  - Délka zpracování pro každý záznam
  - Při výjimka musí jednoznačně identifikovat záznam, který se nepovedl spolu s důvodem
  - Konec dávky (statistiku)
    - Celkový čas běhu
    - Počet success / failed / (popřípadě počet vyřazených)

Vyhodnocení odhadů

Komu se jeho původní odhad nezměnil?

„... pokud zákazník nepotvrdí platbu do 10 minut, pošle se do skladového systému příkaz ke zrušení rezervace zboží a po té se objednávka automaticky stornuje …“

Implicitní požadavky

aneb co ve specifikaci nebylo:

- Vždy mít možnost operaci vyřadit
- Vždy mít možnost operaci monitorovat
- Vždy mít možnost operaci přeplánovat
- Vždy mít možnost operaci ovládat

Q&A