

# Großvolumige Batch-Verarbeitungen auf komplexen OO-Modellen: Performance ohne redundante Entwicklung

SEACON, 22./23.6.2009, Hamburg  
Ralf Degner, Techniker Krankenkasse



  
Techniker Krankenkasse  
Gesund in die Zukunft.

## Überblick

- Das Problem
- Die Lösung – Parallelität
- Parallelität sicherstellen
- Erfahrungen

# Das Problem

## Klassische Massenverarbeitung

- Nah an den Daten
- Datenbankzugriffe hoch optimiert

## Geschäftsobjekte - Abstraktion von den Daten

- Optimierungsmöglichkeiten eingeschränkt
- Deutliche Einbußen bei der Performance (typisch über eine Größenordnung)

## Beides Gleichzeitig: redundante Entwicklung

- Hoher Aufwand
- Redundanzen sind riskant
- Komplexe OO-Modelle schwer „datennah“ abbildbar

Beispiel: Änderungen am „Partner“ löst Folgen in diversen Systemen aus

3

© 2009 Techniker Krankenkasse

# Die Lösung - Parallelität

- 1) Fachlich einfache Selektion potentiell zu bearbeitender Datensätze  
z.B. alle Verträge, für die eine Abrechnung notwendig sein könnten
- 2) Für jeden Datensatz einen „Auftrag“ erzeugen  
z.B. Auftrag in ein Message-System einstellen
- 3) Parallele Abarbeitung der Aufträge mit den Geschäftsobjekten

- Zur Minimierung von redundanter Entwicklung nur einfache Selektion der Datensätze, unnötige gewählte Sätze sortiert BO-Logik aus.
- Selektion und Erzeugung der Aufträge gut optimierbar  
=> kurze Laufzeit
- Gesamtlaufzeit im Wesentlichen durch Abarbeitung der Aufträge durch Geschäftsobjekte bestimmt.

4

© 2009 Techniker Krankenkasse

# Parallelität ermöglichen

Effiziente Parallelität kann leicht „kaputt gehen“

- Kurze DB-Transaktionen sind hilfreich
- Row Level Locking vs. Page Level Locking
- Bottlenecks vermeiden (Klassiker: zentrale Schlüsselvergabe)
- Application-Server-Architektur darf sich nicht selbst behindern

Fachliche Behinderungen: „Gleichzeitige“ Verarbeitung von Datensätzen, die selbe Datenressourcen benötigen

- Oft hilft „Mischen“
- Fachlich sortiert abarbeiten: z.B. 26 Threads, wovon jeder nur einen Anfangsbuchstaben abarbeitet => innerhalb eines Anfangsbuchstaben keine Parallelität (Flexibler: Hash-Code auf Sortierbegriffen)

# Erfahrungen

- TKeasy skaliert bis zu einer Parallelität von ca. 40 gut
  - Limitierend: DB-Engine
  - 2 Application-Server-Instanzen ausreichend (Limit nur durch Speicherverbrauch/GC)
- Kaum redundante Entwicklung außerhalb der BOs
- Gesamtlaufzeit gut
- Verbrauch an Rechenleistung hoch
- In Nebenzeiten (TK: 16 bis 7 Uhr) häufig Rechenzeit verfügbar
- Sicherstellung der Parallelität ist komplexes Thema  
=> Test unter Realistischen Bedingungen

Vielen herzlichen Dank für  
Ihre Aufmerksamkeit!

Ralf.Degner@tk-online.de



**TK**  
**Techniker Krankenkasse**  
Gesund in die Zukunft.