

Avoiding liveness hazards (10)

Simon Moissl, Christian Schürmann

27.06.2008

1 Introduction

2 Explicit hazards

- Lock-ordering deadlocks
- Dynamic lock order deadlocks
- Deadlocks between cooperating objects
- Resource deadlocks
- Starvation
- Poor responsiveness
- Livelock

3 Outlook & Questions

„Liveness Hazards“

Was versteht man unter „Liveness Hazard“

Ein gewisses Risiko für den Programmfluss (permanenter Status des „non-forward progress“)

...(einfachste Lösung: Vogel-Strauß-Algorithmus)

Was gibt es für „Liveness Hazards“

Man unterscheidet zwischen Deadlocks, Starvation, Livelock, ...

Was tun wir gegen „Liveness Hazards“

Analyse und fallbasierte Verklemmungsverhinderung (Prevention) bzw. Verklemmungsvermeidung (Avoidance)

„Liveness Hazards“

Was versteht man unter „Liveness Hazard“

Ein gewisses Risiko für den Programmfluss (permanenter Status des „non-forward progress“)

...(einfachste Lösung: Vogel-Strauß-Algorithmus)

Was gibt es für „Liveness Hazards“

Man unterscheidet zwischen Deadlocks, Starvation, Livelock, ...

Was tun wir gegen „Liveness Hazards“

Analyse und fallbasierte Verklemmungsverhinderung (Prevention) bzw. Verklemmungsvermeidung (Avoidance)

„Liveness Hazards“

Was versteht man unter „Liveness Hazard“

Ein gewisses Risiko für den Programmfluss (permanenter Status des „non-forward progress“)

...(einfachste Lösung: Vogel-Strauß-Algorithmus)

Was gibt es für „Liveness Hazards“

Man unterscheidet zwischen Deadlocks, Starvation, Livelock, ...

Was tun wir gegen „Liveness Hazards“

Analyse und fallbasierte Verklemmungsverhinderung (Prevention) bzw. Verklemmungsvermeidung (Avoidance)

Vorbeugung & Analyse

Vorbeugung

- Optimalerweise die Anzahl auf **ein** Lock beschränken
- Ansonsten: Lock-order Policy → Dokumentation!
- Vorbeugend: Open calls

Falls ein Deadlock auftritt kann ein Threaddump sehr hilfreich sein

Vorbeugung & Analyse

Vorbeugung

- Optimalerweise die Anzahl auf **ein** Lock beschränken
- Ansonsten: Lock-order Policy → Dokumentation!
- Vorbeugend: Open calls

Falls ein Deadlock auftritt kann ein Threaddump sehr hilfreich sein

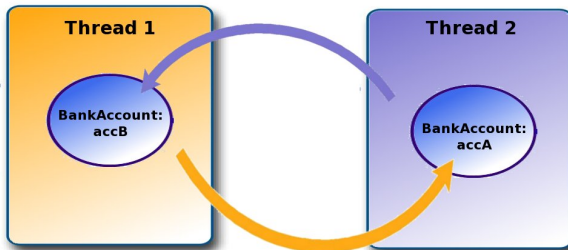
Lock-ordering deadlocks

Lock-ordering deadlock...

...entsteht durch unterschiedliche Reihenfolge beim Erwerb der Locks

Auflösung?

- Locks immer in gleicher Reihenfolge anfordern
- Lock-ordering policy



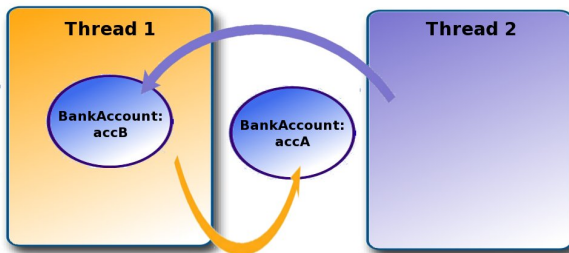
Dynamic lock order deadlocks

Dynamic lock order deadlocks...

- Objekte werden als Parameter übergeben
- Keine Kontrolle über Reihenfolge mehr

Auflösung?

Definition einer totalen Ordnung



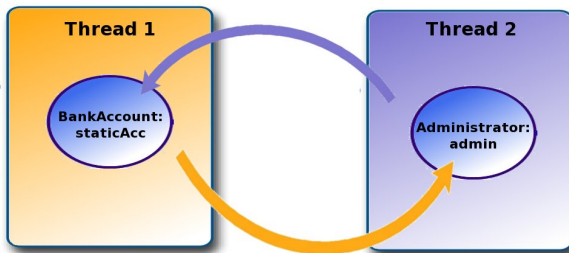
Cooperating objects deadlock

Cooperating objects deadlock...

Zyklischer Ringschluss über
mehrere Klassen

Auflösung?

Open calling



Open calls

Definition

Open call: Aufruf einer (Alien-) Methode ohne ein Lock zu halten

Open calls...

- ... erleichtern Analyse und Debugging
- ... können durch Kopieren realisiert werden (lesend)
- ... führen zu Problemen bei schreibendem Zugriff

Open calls

Definition

Open call: Aufruf einer (Alien-) Methode ohne ein Lock zu halten

Open calls...

- ... erleichtern Analyse und Debugging
- ... können durch Kopieren realisiert werden (lesend)
- ... führen zu Problemen bei schreibendem Zugriff

Resource deadlocks

Resource deadlocks...

Thread wartet auf bestimmte Ressourcen

Beispiel?

Semaphoren, Ergebnisse anderer Tasks...



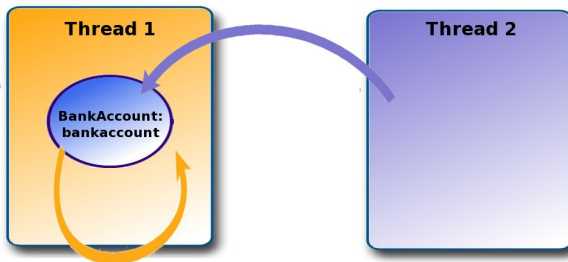
Starvation

Starvation...

- ... ist ein Deadlock-Derivat
- Thread bekommt ständig keinen Zugang zu benötigter Ressource

Auflösung?

- Signalling
- korrekte Priorisierung



Poor responsiveness

Poor responsiveness...

Anstatt notwendige Ressourcen überhaupt nicht zu erhalten (Starvation) werden sie nur sehr verzögert übergeben

Auflösung?

Priorisierung ein sinnvoller Ansatz

Allerdings:

Bei rechenintensiven Threads müssen andere Threads entsprechend lange warten

Poor responsiveness

Poor responsiveness...

Anstatt notwendige Ressourcen überhaupt nicht zu erhalten (Starvation) werden sie nur sehr verzögert übergeben

Auflösung?

Priorisierung ein sinnvoller Ansatz

Allerdings:

Bei rechenintensiven Threads müssen andere Threads entsprechend lange warten

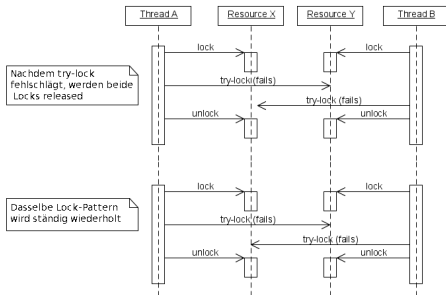
Livelock

Livelock...

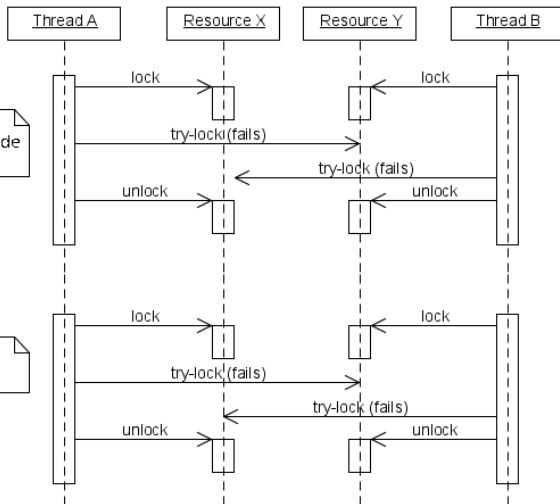
- ... ist ein Deadlock-Derivat
- Threads verändern ständig den Zustand

Auflösung?

Unregelmäßigkeit (Zufälligkeit)
einbauen



Livelock



Nachdem try-lock fehlschlägt, werden beide Locks released

Dasselbe Lock-Pattern wird ständig wiederholt

Was nehmen wir mit nach Hause?

Jetzt solltet Ihr folgendes wissen/können/verinnerlicht haben

- Es gibt verschiedene Arten von Deadlocks (ordering, starvation, ...)
- Eine Thread-policy hilft dabei, Deadlocks zu vermeiden (Doku nicht vergessen!)
- Open calls helfen ebenfalls, Deadlocks zu vermeiden
- Kommt es trotzdem zum Deadlock, kann ein Threaddump nützlich sein (muss aber nicht)
- Zur Analyse von Threaddumps gibt es sehr nützlich Werkzeuge (lockness eclipse plugin)

Was nehmen wir mit nach Hause?

Jetzt solltet Ihr folgendes wissen/können/verinnerlicht haben

- Es gibt verschiedene Arten von Deadlocks (ordering, starvation, ...)
- Eine Thread-policy hilft dabei, Deadlocks zu vermeiden (Doku nicht vergessen!)
- Open calls helfen ebenfalls, Deadlocks zu vermeiden
- Kommt es trotzdem zum Deadlock, kann ein Threaddump nützlich sein (muss aber nicht)
- Zur Analyse von Threaddumps gibt es sehr nützlich Werkzeuge (lockness eclipse plugin)

Was nehmen wir mit nach Hause?

Jetzt solltet Ihr folgendes wissen/können/verinnerlicht haben

- Es gibt verschiedene Arten von Deadlocks (ordering, starvation, ...)
- Eine Thread-policy hilft dabei, Deadlocks zu vermeiden (Doku nicht vergessen!)
- Open calls helfen ebenfalls, Deadlocks zu vermeiden
- Kommt es trotzdem zum Deadlock, kann ein Threaddump nützlich sein (muss aber nicht)
- Zur Analyse von Threaddumps gibt es sehr nützlich Werkzeuge (lockness eclipse plugin)

Was nehmen wir mit nach Hause?

Jetzt solltet Ihr folgendes wissen/können/verinnerlicht haben

- Es gibt verschiedene Arten von Deadlocks (ordering, starvation, ...)
- Eine Thread-policy hilft dabei, Deadlocks zu vermeiden (Doku nicht vergessen!)
- Open calls helfen ebenfalls, Deadlocks zu vermeiden
- Kommt es trotzdem zum Deadlock, kann ein Threaddump nützlich sein (muss aber nicht)
- Zur Analyse von Threaddumps gibt es sehr nützlich Werkzeuge (lockness eclipse plugin)

Was nehmen wir mit nach Hause?

Jetzt solltet Ihr folgendes wissen/können/verinnerlicht haben

- Es gibt verschiedene Arten von Deadlocks (ordering, starvation, ...)
- Eine Thread-policy hilft dabei, Deadlocks zu vermeiden (Doku nicht vergessen!)
- Open calls helfen ebenfalls, Deadlocks zu vermeiden
- Kommt es trotzdem zum Deadlock, kann ein Threaddump nützlich sein (muss aber nicht)
- Zur Analyse von Threaddumps gibt es sehr nützlich Werkzeuge (lockness eclipse plugin)

Frage 1

Frage:

Gibt es weitere Möglichkeiten, Open calls bei schreibendem Zugriff auf Objekte zu realisieren

Antwort:

Ja, z.B. nur einem Thread Zugriff gewähren (oben erwähnt), Atomarität sicherstellen, `tieLock` verwenden

Frage 1

Frage:

Gibt es weitere Möglichkeiten, Open calls bei schreibendem Zugriff auf Objekte zu realisieren

Antwort:

Ja, z.B. nur einem Thread Zugriff gewähren (oben erwähnt), Atomarität sicherstellen, tieLock verwenden

Frage 2

Frage:

Gibt es noch andere Möglichkeiten um Deadlocks allgemein zu verhindern (sowohl Resource-Deadlocks als auch „normale“ Deadlocks)?

Frage:

Warum habt Ihr die Timed lock attempts unterschlagen?

Antwort:

In Kapitel 10 wird nur die Existenz erwähnt, das tryLock Feature wird erst in Kapitel 13 eingeführt (explicit Locking)

Frage 2

Frage:

Gibt es noch andere Möglichkeiten um Deadlocks allgemein zu verhindern (sowohl Resource-Deadlocks als auch „normale“ Deadlocks)?

Frage:

Warum habt Ihr die Timed lock attempts unterschlagen?

Antwort:

In Kapitel 10 wird nur die Existenz erwähnt, das tryLock Feature wird erst in Kapitel 13 eingeführt (explicit Locking)

Frage 2

Frage:

Gibt es noch andere Möglichkeiten um Deadlocks allgemein zu verhindern (sowohl Resource-Deadlocks als auch „normale“ Deadlocks)?

Frage:

Warum habt Ihr die Timed lock attempts unterschlagen?

Antwort:

In Kapitel 10 wird nur die Existenz erwähnt, das tryLock Feature wird erst in Kapitel 13 eingeführt (explicit Locking)