informatiCup 2006 · Aufgabe 1
Schnappschussalgorithmus

Einführung

Jeder Prozess in einem verteilten System besitzt seinen eigenen Zustand, im folgenden Prozesszustand genannt. Die Nachrichten, die gerade zwischen zwei Prozessen übertragen werden, bilden einen Kanalzustand der beiden Prozesse. Die Menge aller Prozesszustände und Kanalzustände bilden den globalen Systemzustand des verteilten Systems.

Die Kenntnis des globalen Zustands eines verteilten Systems ist aus vielen Gründen wichtig. Die Erfassung globaler Zustände wird z.B. in folgenden verteilten Anwendungen benötigt:

- Verteiltes garbage collection
- Verteilte Deadlock-Erkennung
- Verteilte Terminierungs-Erkennung
- Verteiltes Debugging
- Verteilter Speicherzustand

Da es in verteilten Systemen es weder einen (wirklich) gemeinsamen Speicher noch eine perfekt synchronisierte physikalische Uhr gibt, stellt die Erfassung eines globalen Systemzustandes (meist als Schnappschuss bezeichnet [KRS95]) eine große Herausforderung dar. Unternehmen, wie z.B. Banken, können ihre IT-Systeme nicht beliebig lange »einfrieren« (im Sinne von Anhalten), um einen globalen Systemzustand zu erfassen. Dies wäre eigentlich notwendig, um zu garantieren, dass keine Nachrichten mehr unterwegs sind.

Es gibt einige Algorithmen, die einen globalen Systemzustand bzw. einen Schnappschuss erfassen können. Jeder Algorithmus besitzt Vor- und Nachteile und auch unterschiedliche Voraussetzungen für den Einsatz, d.h. es gibt keinen best-performing Algorithmus, der für alle erdenklichen Szenarien gut einsetzbar wäre. Ein geeigneter Algorithmus muss daher anhand der konkret vorliegenden Anwendungsanforderungen ausgewählt werden. In dieser Aufgabenstellung ist einer der ersten bekannten Algorithmen zu implementieren. Er ist 1985 von Chandy und Lamport [CL85] veröffentlicht worden. Die wesentliche Idee dabei besteht darin, die für den globalen Systemzustand wichtigen Nachrichten durch FIFO-Kommunikationskanäle zu senden. Jeder beteiligte Prozess arbeitet während der gesamten Zeit »ganz normal« weiter!

Aufgabenstellung

Die aufgeführten Aufgaben stellen nicht die Implementierungsreihenfolge dar!

(a) Besorgen Sie sich die benötigten Quellen: Die angegebenen Quellen können im Internet als pdf-Dateien gefunden werden. In allgemeinen Büchern zu verteilten Systemen (z.B. Coulouris, G.; Dollimore, J.; Kindberg, T.: Verteilte Systeme, Pearson Studium) kann der Algorithmus auch gefunden werden.

b) Implementieren Sie die Kommunikationsplattform: Der zu implementierende Algorithmus benötigt eine Kommunikationsplattform, in der jeder beteiligte Prozess mit jedem anderen durch einen gerichteten FIFOKommunikationskanal verbunden ist. Alle den globalen Systemzustand betreffende Nachrichten müssen über diese Kanäle versendet werden! Um die Verteilung einer speziellen Nachricht (Marker genannt) zu vereinfachen, hat diese Kommunikationsplattform einen Broadcastmechanismus anzubieten, d.h. ein Prozess hat die Möglichkeit, per Broadcast an alle anderen Prozesse eine Nachricht zu versenden.

(c) Implementieren Sie die Speicherumgebung: Der zu implementierende Algorithmus benötigt für jeden Prozess einen Speicherbereich, in dem der lokale Zustand des Prozesses und alle während des Schnappschusses bei ihm über die FIFO-Kommunikationskanäle eintreffenden Nachrichten (die den Kanalzustand repräsentieren) sicher abgelegt werden können. Sicher heißt einmal, dass diese Daten nach der Speicherung nicht mehr verändert werden dürfen, und heißt auch, dass diese Daten nur von dem den Schnappschuss auslösenden Prozess gelesen und gelöscht werden dürfen.

(d) Implementieren Sie den Schnappschussalgorithmus: Jeder Prozess darf einen Schnappschuss auslösen. Achten Sie daher darauf, dass klar ist, wer welchen Schnappschuss ausgelöst hat! Am Ende eines ausgelösten Schnappschusses ist der globale Systemzustand von dem auslösenden Prozess lokal zu speichern. Bei allen anderen Prozessen sind die dort vorhandenenTeile dieses globalen Systemzustandes zu löschen.

(e) Implementieren Sie einen Monitor, der einen Überblick über die durchgeführten wie auch laufenden Schnappschüsse liefert.

(f) Implementieren Sie einen Koordinator, der für den Aufbau, den Lauf und den Abbau des Systems verantwortlich ist. Der Koordinator kennt alle beteiligten Prozesse und muss in der Lage sein, die jeweils benötigten Umgebungen zu instanziieren.

(g) Implementieren Sie eine Anwendung: Implementieren Sie ein verteiltes System, das den Schnappschussalgorithmus verwendet.

Hinweise

In der Aufgabenstellung sind bestimmte Punkte noch nicht festgelegt worden, die für die Lösung festgelegt werden müssen. Die aufgeführten Prozesse sind entsprechend der gewählten Lösung zu erweitern.

- Als Programmiersprache empfiehlt sich Java. Es kann aber auch eine andere Sprache gewählt werden. DieWahl ist zu begründen. Vor- und Nachteile (jeweils mindestens zwei) sind zu nennen.
- Zur Kommunikation kann ein Protokoll festgelegt werden. Die Wahl ist zu begründen. Vor- und Nachteile (jeweils mindestens zwei) sind zu nennen.
- Die Fehlersemantik ist nicht festgelegt worden: Es wird empfohlen, eine minimale Fehlersemantik zu realisieren. Nachrichtenverluste oder Prozessausfälle sollen nicht zum Absturz des Systems führen (undefinierter Zustand). Das System ist ggf. »sauber« mit einer entsprechenden Fehlermeldung zu beenden (definierter Zustand).
- Die Anwendung ist nicht festgelegt worden: Sofern Sie keine weitere Idee haben ist ein Bankenszenario mit verschiedenen Banken, die jeweils eigene Konten besitzen, zu implementieren. Die Buchungen (Einzahlungen, Überweisungen, Abbuchungen) können als fester (Test-)Ablauf realisiert werden. Wichtig ist, dass Sie die globalen Systemzustände auf ihre Korrektheit prüfen können!

Literatur

[CL85]
K. Mani Chandy and Leslie Lamport. Distributed snapshots: Determining global states of distributed systems. ACM Trans. Comput. Syst., 3(1):63–75, 1985.

[KRS95]
Ajay D. Kshemkalyani, Michel Raynal, and Mukesh Singhal. An introduction to snapshot algorithms in distributed computing. Distributed Systems Engineering, 2(4):224–233, 1995.

Kontakt und Einreichung

Gesellschaft für Informatik e.V. (GI) · Ludger Porada · Ahrstraße 45 · 53175 Bonn
E-Mail: ludger.porada@gi-ev.de · http://www.informaticup.de

Termine

- 20. Dezember 2006: Einsendeschluss der Beiträge
- 28. Februar 2007: Benachrichtigung der Teilnehmerinnen/Teilnehmer
- 30. – 31. März 2007: Endrunde
- 31. März 2007: Siegerehrung

FAQ zur Aufgabenstellung und Einreichung

Wer darf teilnehmen?

Teilnehmen dürfen Studierende bis zum 5. Semester (Bachelor- oder Diplomstudium). Unser Wettbewerb richtet sich ausschließlich an Gruppen von mindestens zwei bis maximal vier Personen. Wenn Sie interessiert sind, suchen Sie Mitstreiterinnen und Mitstreiter. Die Teilnahme am Wettbewerb ist kostenlos.

Wieviele Aufgaben müssen wir lösen?

Für die Teilnahme am InformatiCup muss nur eine Aufgabe gelöst werden.

Was müssen wir als Lösung einreichen?

Eine gelöste Aufgabe umfasst:

- ein lauffähiges Programm
- eine Installationsanleitung
- ggf. eine Bedienungsanleitung
- eine schriftliche Ausarbeitung wie in den Aufgaben gefordert

Wie können wir eine Lösung einreichen?

In jeder Form, per Brief, auf Papier, mit CD, per E-Mail.

Welche Programmiersprachen dürfen wir verwenden?

Es darf jede Programmiersprache verwendet werden, die für die Lösung einer Aufgabe geeignet erscheint.

Welche Bibliotheken dürfen wir verwenden?

Es darf jede kostenlos verfügbare Bibliothek verwendet werden. Die Bibliothek muss nicht als Quelltext verfügbar sein.

Wie werden die eingereichten Lösungen bewertet?

Bei der Bewertung der eingereichten Lösungen werden sowohl der theoretische Lösungsansatz (z.B. Komplexität verwendeter Algorithmen) als auch die praktische Umsetzung (z.B. Bedienbarkeit und Zuverlässigkeit der Software) bewertet.