

Verteilte Systeme – Übung

Synchronisation

Sommersemester 2021

Michael Eischer, Laura Lawniczak, Tobias Distler

Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)

www4.cs.fau.de



Lehrstuhl für Verteilte Systeme
und Betriebssysteme



FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

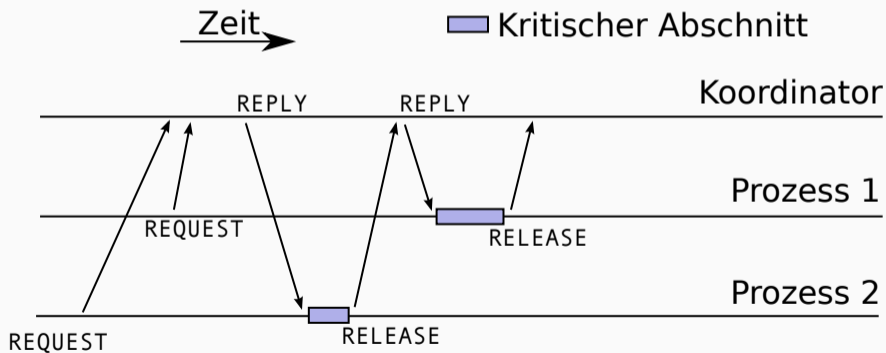
Synchronisation

Synchronisation

- Koordination von Zugriffen auf gemeinsame Betriebsmittel in verteilten Systemen notwendig
- Verschiedene Möglichkeiten:
 - Zentraler Koordinator
 - Koordination untereinander
- Exklusiver Zugriff äquivalent zur Bestimmung totaler Ordnung:
⇒ Einigung auf Reihenfolge der Zuteilung der Ressource

Zentraler Koordinator

- Zentraler Prozess ist zuständig für Koordination
- Anfragen werden geordnet und in Reihenfolge freigegeben
- Nachrichtenfolge: REQUEST, REPLY, RELEASE



- **Idee:** Ausnutzen der totalen Ordnung über logische Zeitstempel bezüglich Lock-Anfragen

- **Voraussetzungen:**
 - FIFO-Protokoll:
Nachrichten eines Absenders müssen in der Reihenfolge ankommen, in der sie abgeschickt wurden
 - Zuverlässiger Nachrichtenkanal
 - Toleriert ohne weitere Maßnahmen keine Ausfälle

- **Ablauf:**
 1. REQUEST via Broadcast an alle Prozesse versenden
 2. Warten bis eigene Anfrage vorne in der REQUEST-Warteschlange steht **und** kein anderer Prozess sich vor dem eigenen Eintrag einreihen kann
 3. Kritischen Abschnitt ausführen
 4. Broadcast der RELEASE-Nachricht zum Freigeben des Locks

- Warteschlangenverwaltung:
 - Einreihen von eingehenden REQUEST-Nachrichten (auch selbst gesendete)
 - Sortierung nach totaler Ordnung über Zeitstempel logischer Uhr
 - Entfernen des korrespondierenden Elements bei Empfang von RELEASE (auch selbst gesendete)
- Einreihen vor eigenem Eintrag nicht mehr möglich, wenn von allen Prozessen bereits Nachrichten mit größerem Zeitstempel als der des eigenen REQUESTS empfangen wurden
 - ⇒ Merken des jeweils zuletzt empfangenen Zeitstempels je Prozess
 - FIFO-Eigenschaft garantiert streng monotonen Anstieg
- Empfang einer REQUEST-Nachricht von anderem Prozess muss zudem mit ACK-Nachricht an Absender quittiert werden
 - Notwendig, um Fortschritt zu garantieren
 - Dient lediglich der Erhöhung und Übermittlung der logischen Uhr
 - Bestätigung durch Nachrichtenaustausch auf Anwendungsebene implizit möglich

Lock-Protokoll von Lamport (3)

■ Eigenschaften:

- RELEASE-Nachrichten sind total geordnet
- Erweiterungsmöglichkeiten bezüglich Fehlertoleranz, da REQUEST-Warteschlange implizit repliziert
- Geringe Latenzen bei häufig beanspruchten Locks
- Allerdings größeres Nachrichtenaufkommen als bei zentralem Koordinator

■ Beispiel:

