Technische Hochschule Ingolstadt

Fakultät für Elektrotechnik und Informatik

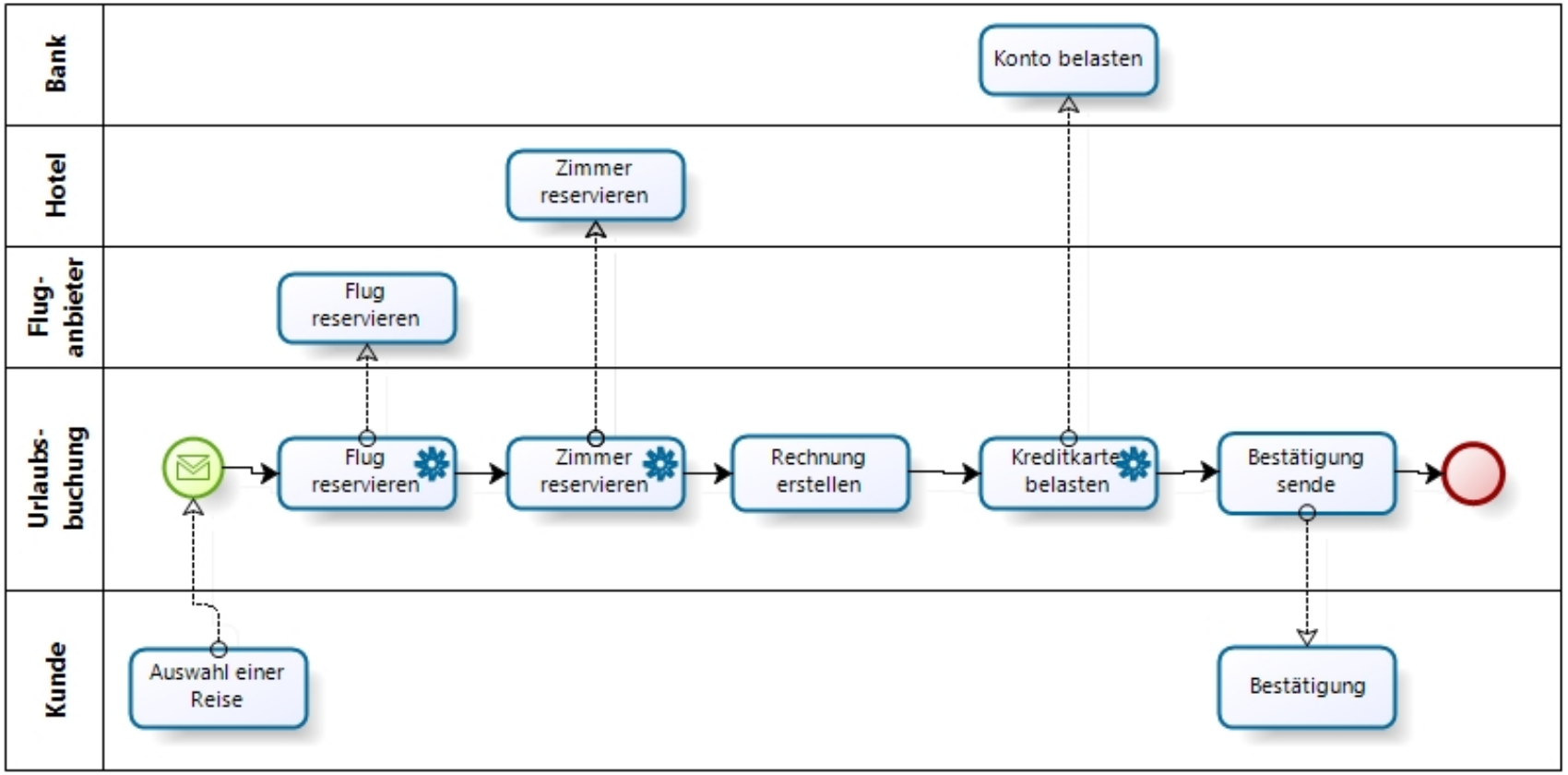
Zukunft in Bewegung

IT-Integrations- und Migrationstechnologien

Transaktionen

Prof. Dr. Bernd Hafenrichter 01.10.2014

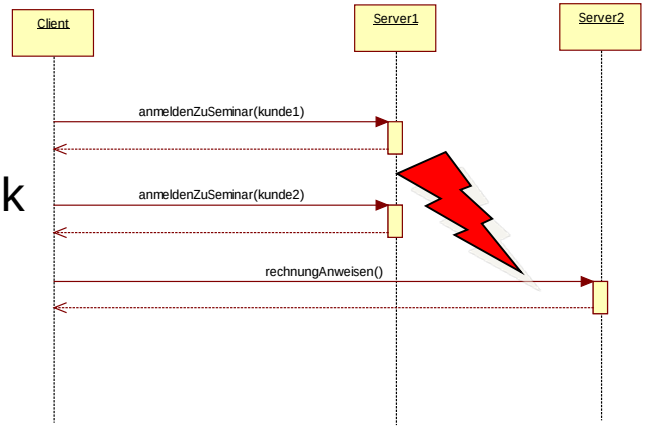
Motivation



- Was passiert wenn die Kreditkarte des Kunden nicht gedeckt ist?
- Was passiert wenn die Zimmerreservierung fehl schlägt?

Transaktionen in verteilten Systemen

- Wie können mehrere „Funktionsaufrufe“ in einem verteilten System mit einer Transaktions-Semantik versehen werden?
- Problem:
  - Wie kann über Service1 und Service2 eine übergreifende Transaktionssemantik realisiert werden ?
  - Beide Server z.B. auf unterschiedliche Datenbanken auf
  - Die Änderungen an den beiden Datenbanken sollte konsistent Rückgängig gemacht werden



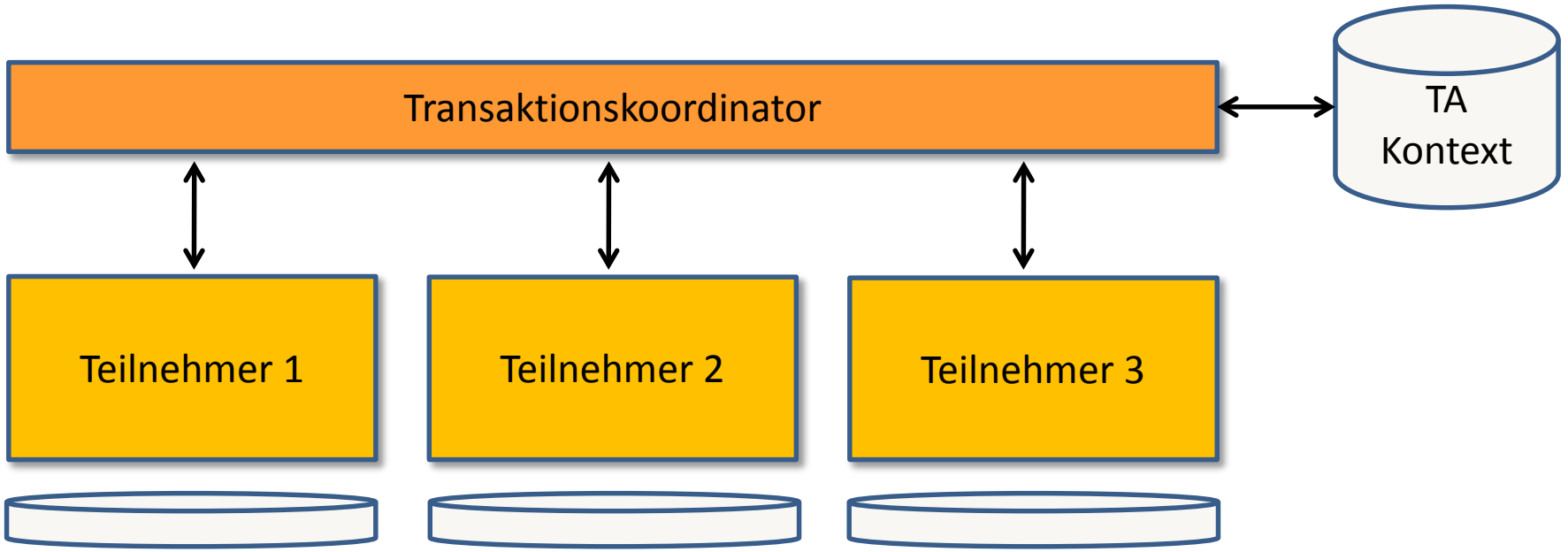
Buchungen rückgängig machen. Aber wie ??

Transaktionen in verteilten Systemen – Two-Phase-Commit

Grundidee

- Übertrage die Semantik von Datenbanktransaktionen auf die Arbeit in verteilten Systemen (ACID)
  - Atomar
  - Consistent
  - Isoliert
  - Dauerhaft
- Beliebige unabhängige Resourcen (Datenbanken, Filesystem, usw.) agieren als Gesamttransaktion

- Lösung: Das Two-Phase-Commit-Protokoll für verteilte Transaktionen



- Der Koordinator verwaltet den Zustand der Gesamttransaktion (Transaktionskontext)
- Die einzelnen Teilnehmer registrieren sich beim Koordinator und geben die Teilnahme an der Transaktion bekannt

Transaktionen in verteilten Systemen – Two-Phase-Commit

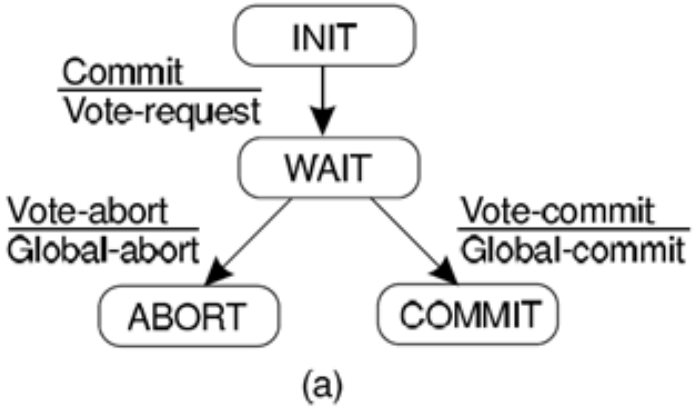
- 2PC – Zwei-Phasen-Commit (1978) – besteht aus zwei Phasen, mit jeweils zwei Schritten:

- Phase 1: Gesamtergebnis der Transaktion bestimmen
  - Koordinator sendet eine Vote-request an alle Teilnehmer
  - Teilnehmer antworten mit Vote-commit oder Vote-abort
    - Nach einem Vote-commit darf der Teilnehmer seine Entscheidung nicht mehr ändern.

- Phase 2: Transaktion abschließen
  - Koordinator sendet Global-commit wenn alle zugestimmt haben oder Global-abort wenn mind. einer abgelehnt hat
  - Die Teilnehmer reagieren entsprechend

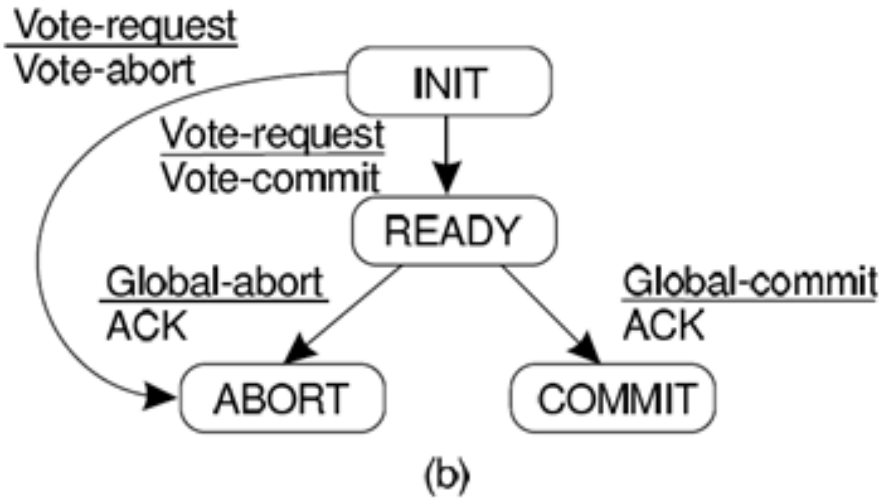
Transaktionen in verteilten Systemen – Two-Phase-Commit

Ablauf/Automat für den Koordinator im 2PC

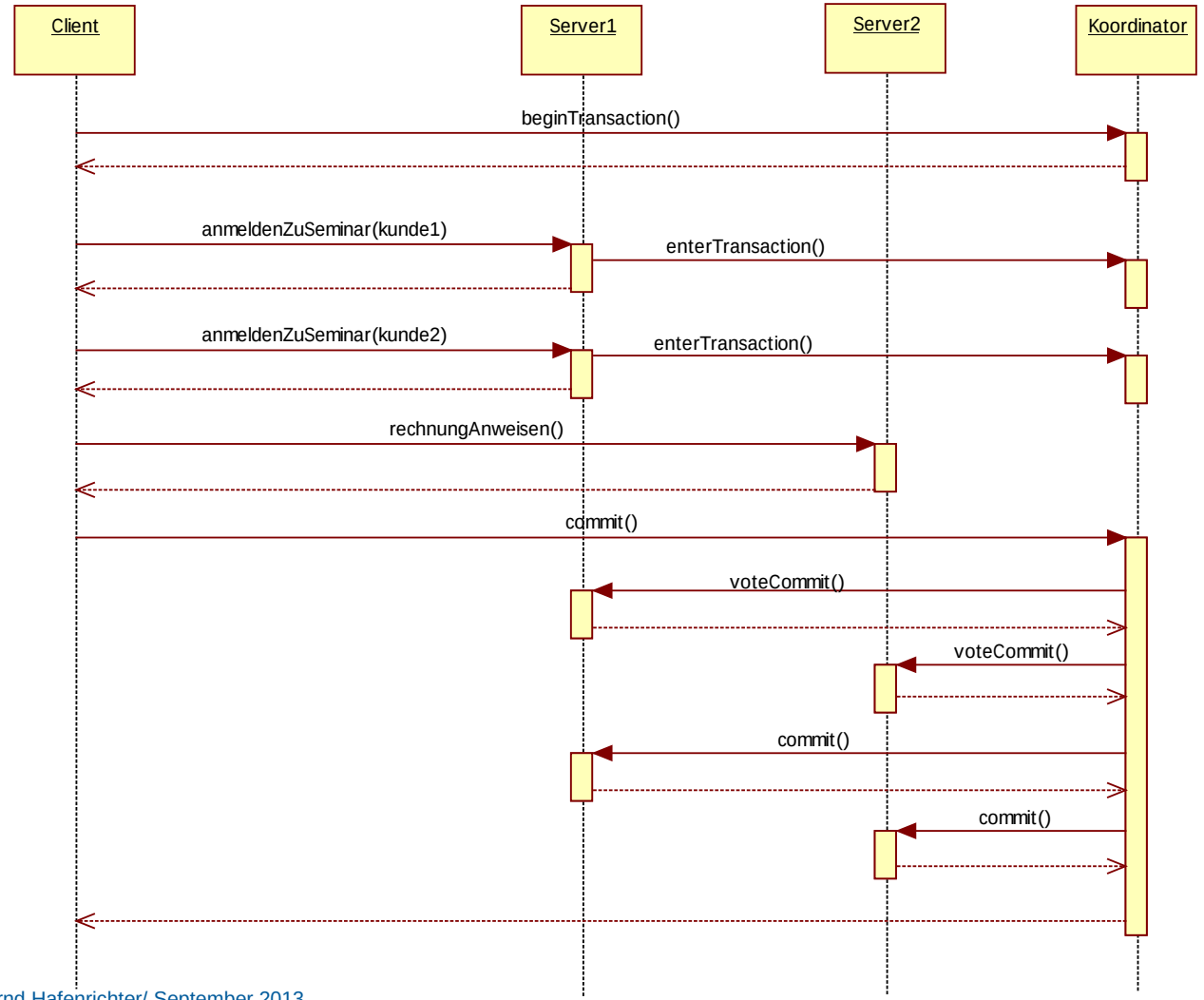


Transaktionen in verteilten Systemen – Two-Phase-Commit

Ablauf/Automat für einen Teilnehmer

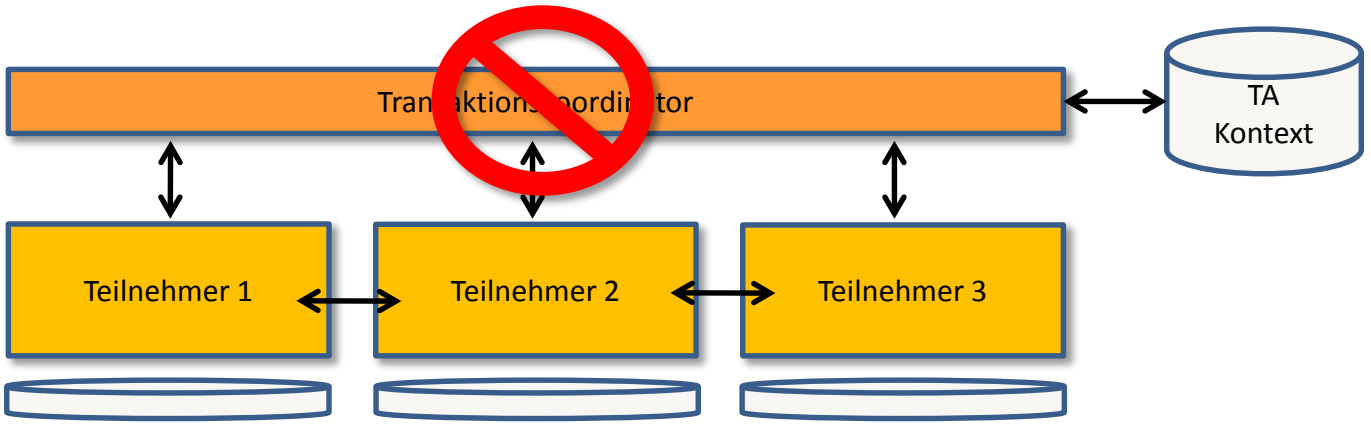


Transaktionen in verteilten Systemen – Two-Phase-Commit



Transaktionen in verteilten Systemen – Two-Phase-Commit

- Sowohl Koordinator als auch Teilnehmer haben blockierende Wartezustände
- Bei einem Absturz muss ein Timeout abgewartet werden, danach Abort
- Idee: Um das Verhalten des 2PC zu verbessen können sich Teilnehmer untereinander verständigen



Transaktionen in verteilten Systemen – Two-Phase-Commit

- READY-Zustand: Teilnehmer Q wendet sich an einen anderen Teilnehmer P um dessen Zustand zu erfragen. Abhängig davon kann er eine Entscheidung treffen.

Status von Q	Aktion von P
COMMIT	Übergang in den Status COMMIT
ABORT	Übergang in den Status ABORT
INIT	Übergang in den Status ABORT
READY	Kontaktierung eines anderen Teilnehmers

Aktionen eines Teilnehmers P, wenn er sich im Status READY befindet und sich an einen anderen Teilnehmer Q gewendet hat

Transaktionen in verteilten Systemen – Two-Phase-Commit

Problem beim Einsatz von Two-Phase-Commit

- Nicht alle beteiligten Systeme unterstützen dieses Protokoll
- Das TPC wird lokal durch die Verwendung von Sperren realisiert
- Dadurch werden evtl. unnötig lange parallel laufende Prozesse blockiert

Lang Laufende Transaktionen

Ausgangspunkt:

- Transaktionen innerhalb von verteilten Systemen können sehr lange Dauern
- Würde man ein 2PC-Protokoll verwenden würden Ressourcen unnötig blockiert.

- Könnte man das ACID-Prinzip aufweichen um ein besseres Verhalten zu erreichen
  - Isolation: Reduktion von Sperren. Zwischenergebnisse werden für andere Teilnehmer sichtbar. TA werden lokal commited
  - Atomarität: Teile der Gesamttransaktion werden commited. D.h. die Gesamttransaktion ist nicht atomar.

Lang Laufende Transaktionen – Compensation

Ausgangspunkt:

- Es stehen keine verteilten Transaktionen zur Verfügung (z.B. Kopplung von Legacy Systemen)

Grundidee:

- Jeder Funktionsaufruf wird in einer lokalen Transaktion ausgeführt
- Für jede aufgerufene Methode wird eine Compensation-Aktion aufgezeichnet
- Tritt ein Fehler auf, werden alle Compensations-Aktionen in umgekehrter Reihenfolge ausgeführt.
- Dadurch wird das Gesamtsystem wieder in einen konsistenten Systemzustand erreicht
- Um Stabilität über Abstürze hinaus zu gewährleisten sollten die Compensations-Aktionen persistent gespeichert werden

Lang Laufende Transaktionen – Compensation