JUST DO IT
HÖHERE ABTEILUNG FÜR INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Laborprotokoll

DEZSYS-L04
Verteilte Transaktionen

Systemtechnik Labor
5BHIT 2016/17, Gruppe B

Kocsis Patrick
Steinwender Jan-Philipp

Version 1.0

Note:

Betreuer: Th.Micheler

Begonnen am 24. März 2017

Beendet am 7. April 2017

Inhaltsverzeichnis

1. Einführung 3
1.1 Ziele 3
1.2 Voraussetzungen 3
1.3 Aufgabenstellung 3
2. Ergebnisse 5
2.1 Github 5
2.2 Konzept 5
Erfolgreiche Transaktion 5
Fehler in Prepare-Phase 5
Fehler in Transaktion 6
Entwicklungsvorbereitung 6
2.3 Transaktionsmanager 7
Ausführung 7
Funktionsweise 7
Klassendiagramm 8
2.4 Stationen 9
2.5 Client 10
2.6 Ablauf 11

1. Einführung

Die Übung soll die Grundlagen von verteilte Transaktionen mit Hilfe eines praktischen Beispiels in JAVA vertiefen.

1.1 Ziele

Implementieren Sie in JAVA einen Transaktionsmanager, der Befehle an mehrere Stationen weitergibt und diese koordiniert. Mit Hilfe des 2-Phase-Commit Protokolls sollen die Transaktionen und die Antwort der Stationen verwaltet werden. Der Befehl kann beliebig gewählt werden und soll eine Datenquelle (Datenbank oder Datei oder Message Queue) abfragen oder verändern.

Die Kommunikation zwischen Transaktionsmanagers und der Stationen soll mit Hilfe einer Übertragungsmethode (IPC, RPC, Java RMI, JMS, etc) aus dem letzten Schuljahr umgesetzt werden.

1.2 Voraussetzungen

- Grundlagen Transaktionen (allgemein, Datenbanksysteme)
- Anbindung Datenquelle in JAVA (JDBC, File, JMS)
- Kommunikation in JAVA (IPC, RPC, Java RMI, JMS)

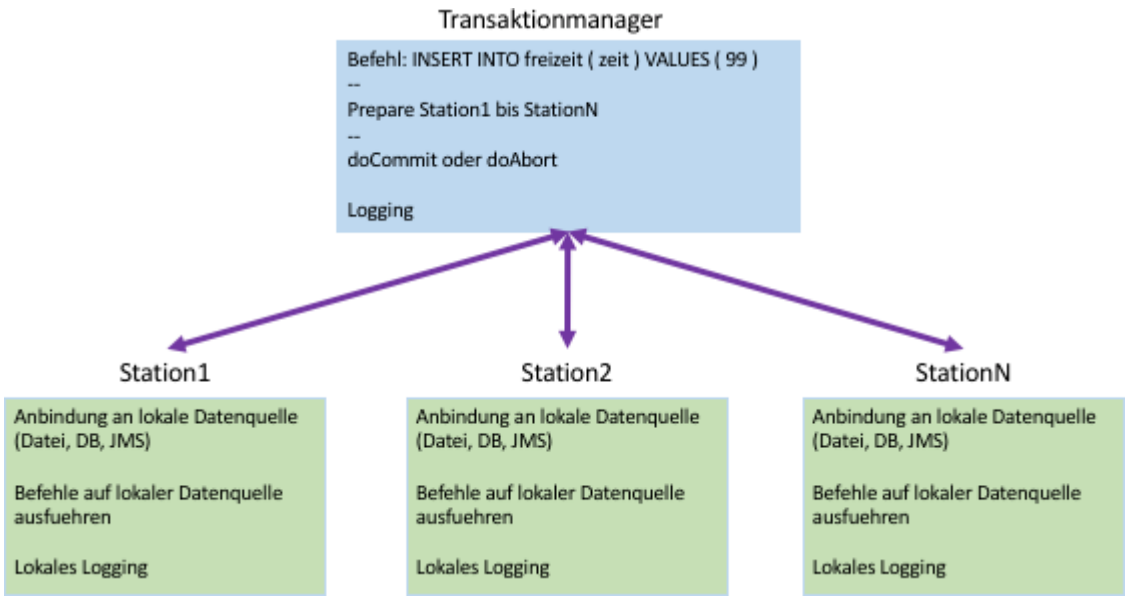
1.3 Aufgabenstellung

Der Transaktionsmanager läuft auf einer eigenen Instanz (bzw. eigenem Port) und stellt die Schnittstelle zwischen den Stationen und dem Benutzer dar. Der Benutzer gibt über die Konsole oder ein User Interface einen Befehl ein, der danach an alle Stationen verteilt wird. Da das 2-Phase-Commit Protokoll als Transaktionsprotokoll zugrunde liegt, soll der Transaktionsmanager jeweils nach einem Befehl,

- das Resultat nach der PREPARE-Phase (Bsp. 3xYES 0xNO 0xTIMEOUT) ausgeben
- welche Aktion der Transaktionsmanager danach durchführt (doCommit, doAbort)
- das Resultat der COMMIT-Phase (Bsp. 3xACK 0xNCK 0xTIMEOUT)
- danach kann ein neuer Befehl eingegeben werden

Logging:
Um im Einzelfall die Transaktionen und Resultat nachverfolgen zu koennen, sollen alle Befehle und deren Resultate mit Timestamp geloggt werden. Beim Transaktionsmanager soll dokumentiert werden, welcher Befehl zu welcher Station und zu welchem Zeitpunkt abgesendet wurde, ebenso beim Erhalt der Antwort. Ebenso sollen, bei den Stationen eingehenden Befehle und deren Resultate bei Ausführung an der lokalen Datenquelle mitdokumentiert werden.

Die folgende Grafik soll den Vorgang beim 2-Phase-Commit Protokoll verdeutlichen:



2. Ergebnisse

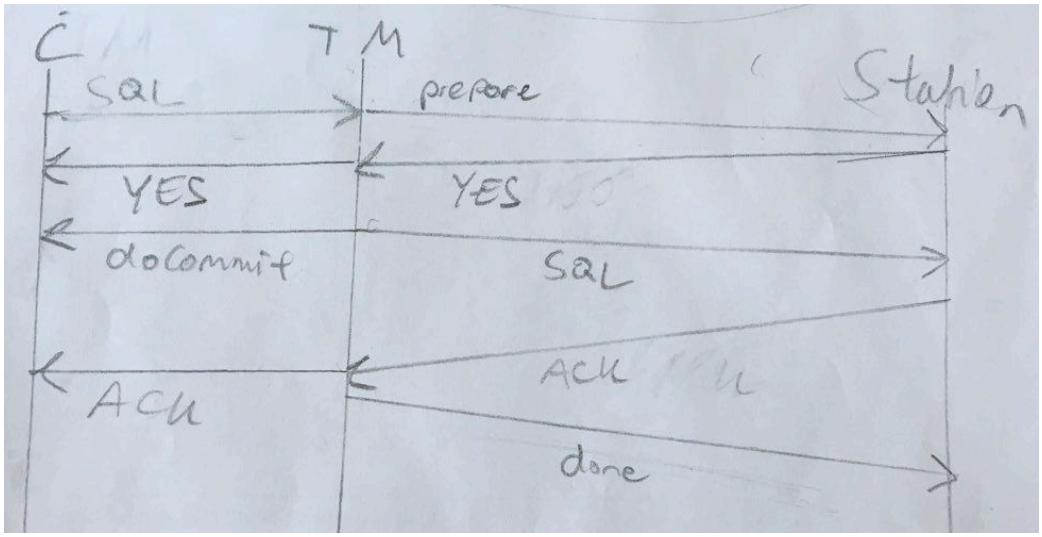
2.1 Github

https://github.com/jsteinwender-tgm/DEZSYS-L04-Verteilte-Transaktionen.git

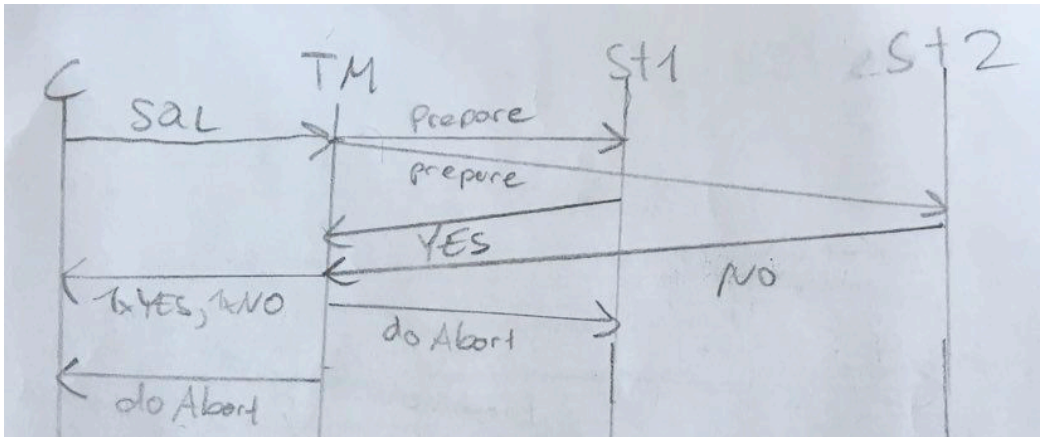
2.2 Konzept

Bei den nachfolgenden Skizzen entspricht die y-Achse der Zeit.

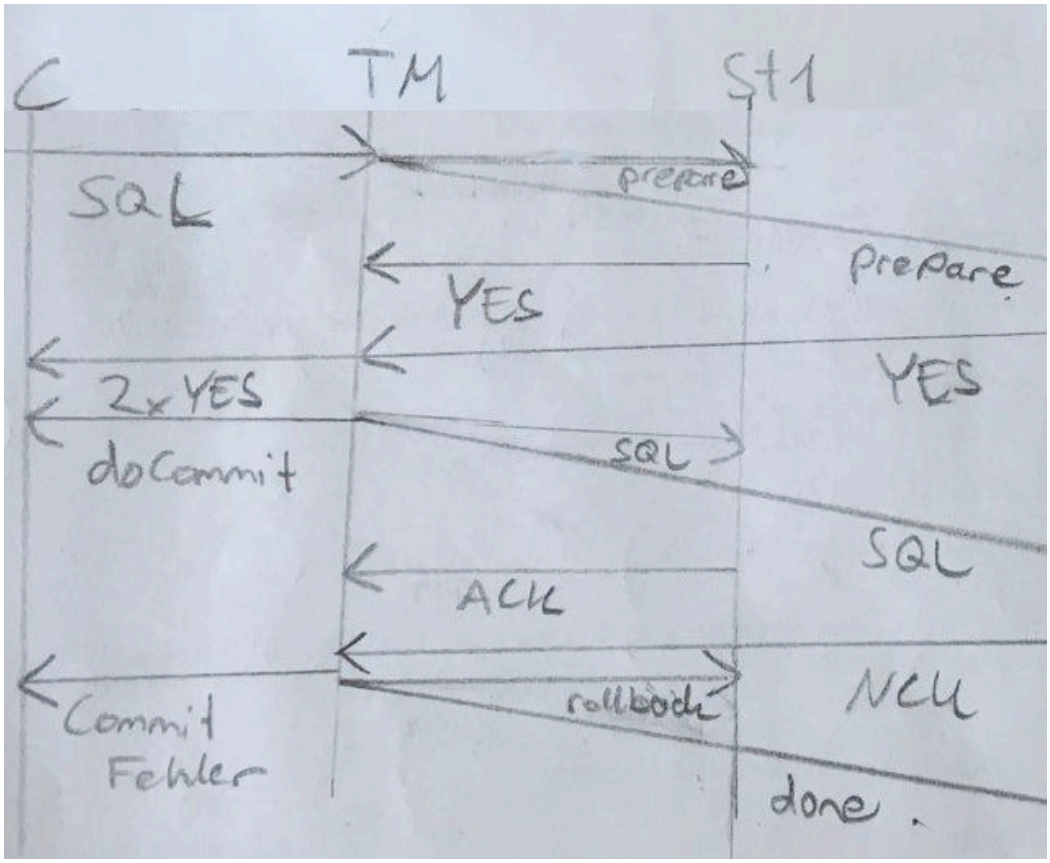
Erfolgreiche Transaktion



Fehler in Prepare-Phase



Fehler in Transaktion



Entwicklungsvorbereitung

Als Datenquelle wurden SQLite Datenbanken verwendet. Dafür benötigt man nur die JDBC .JAR. Für diese Übung wurde sqlite-jdbc-3.16.1.jar verwendet.

Die Kommunikation erfolgt via IPC. In der Übung wird der Port 4444 für die Verbindung zwischen dem Client und dem Transaktionsmanager. Die Verbindung von Transaktionsmanager zu den einzelnen Stationen benutzt den Port 6900.

2.3 Transaktionsmanager

Ausführung

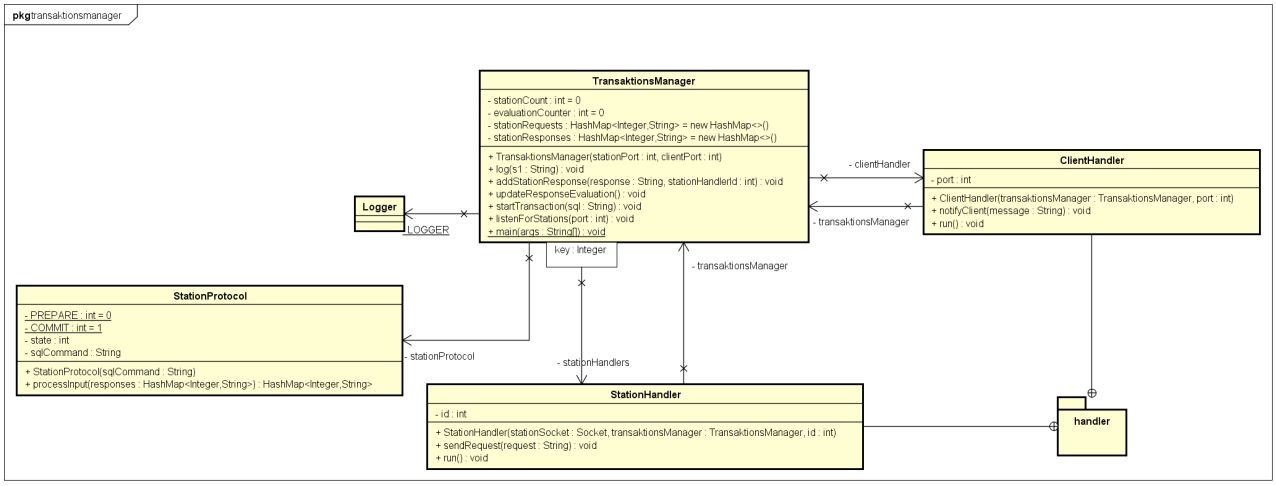
java TransactionManager <StationPortNumber> <ClientPortNumber>

Funktionsweise

1. Methode listenForStations der Klasse Transaktionsmanager wird gestartet, hört auf einkommende Stationsverbindungen
2. ClientHandler wird als Thread gestartet, handhabt die Benutzerkommunikation
3. Client verbindet sich
4. N Stations verbinden sich
5. Client gibt einen SQL Befehl ein
6. startTransaction in TreansaktionManager wird vom ClientHandler aufgerufen, somit startet die prepare Phase
7. Empfangene Nachricht von Station wird mittels addStationResponse dem Transaktionsmanager hinzugefügt
8. updateResponseEvaluation wird von jedem StationHandler aufgerufen, sobald alle Antworten erhalten wurden. Diese werden mittels der processInput Methode des StationProtocol Interpretiert und über die sendRequest Methode des StationHandlers an die Stations weitergegeben. Im Weiteren wird der Client über die notifyClient Methode des ClientHandlers vom Fortschritt der Transaktion benachrichtigt.

Weitere Informationen können den Code Kommentaren entnommen werden.

Klassendiagramm



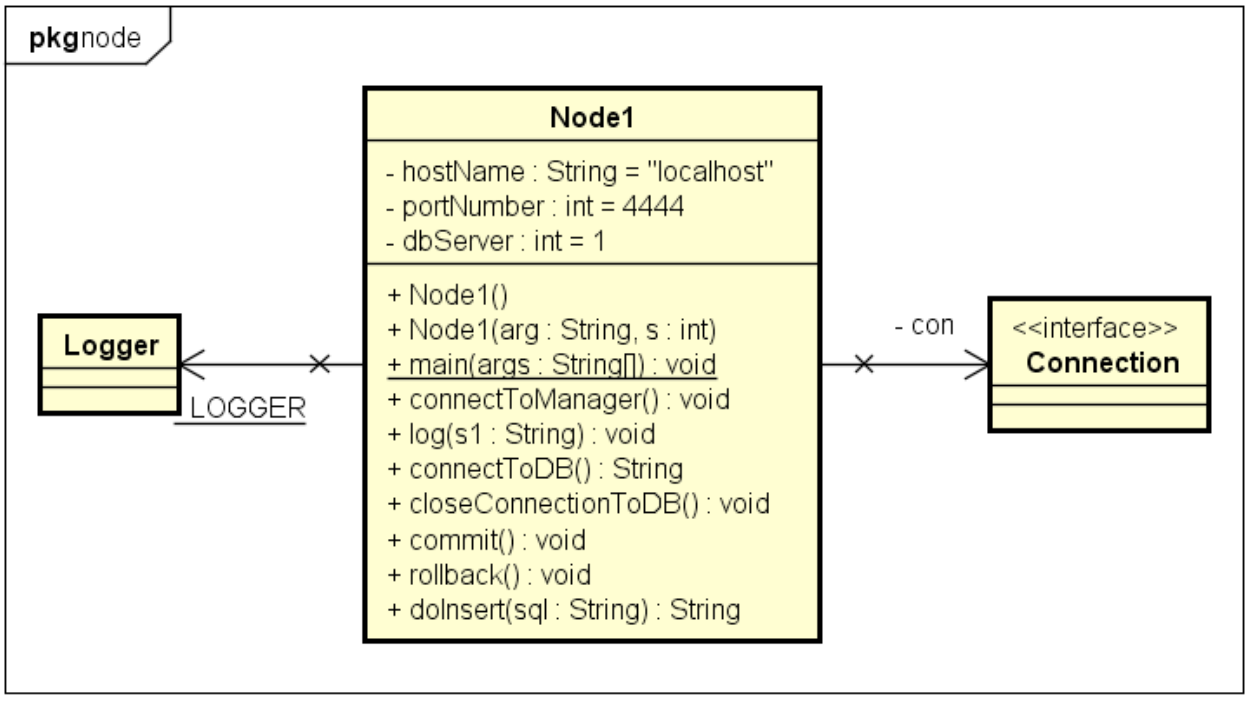
2.4 Stationen

Wenn die Station ohne Parameter gestartet wird, wird als Hostname localhost und der Port 4444 verwendet. Will ein anderer Server und / oder Port verwendet werden, muss die Station mit dem Hostname als ersten und dem Port als zweiten Parameter gestartet werden.

Die einzelnen Stationen beinhalten folgende Funktionen:

- Starten des Loggers,
- Socket-Aufbau zum TM,
- Verbindungsaufbau zu der Datenbank (Prepare-Phase),
- Einen Datensatz in die DB eintragen,
- Ein Commit oder Rollback durchführen (Commit-Phase) und
- Die folgende Verbindung zu der DB beenden.

Klassendiagramm:



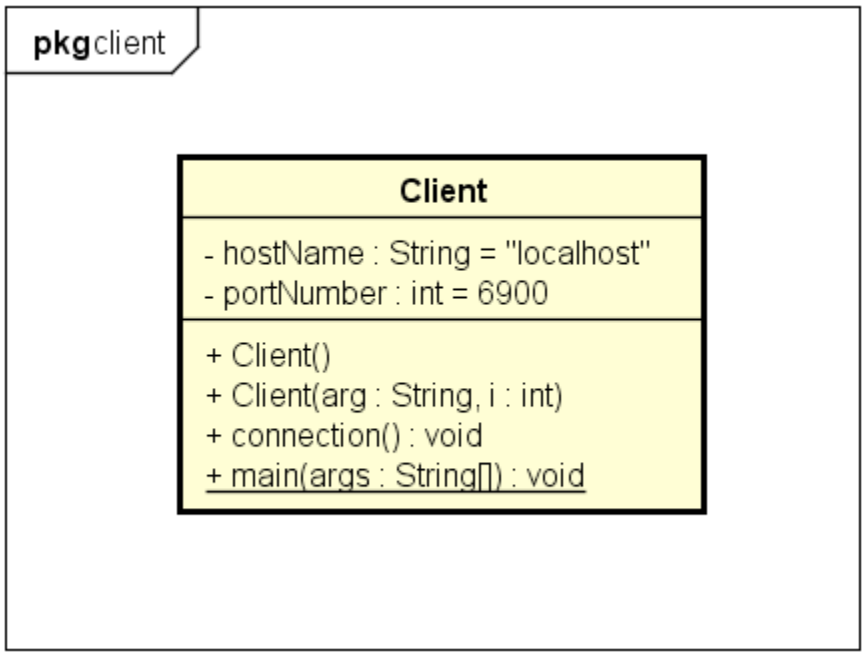
2.5 Client

Der Client ist eine simple Applikation, die Usereingaben an den Server schickt und die Antwort des Transaktionsmanagers ausgibt.

Er kann ebenfalls ohne Parameter gestartet werden, dann werden die Standardwerte eingesetzt, Hostname ist localhost und Port 6900.

Will ein anderer Server genutzt werden, muss das Programm mit dem Hostname und dem Port als Parameter ausgeführt werden.

Klassendiagramm:



2.6 Ablauf

1) Zuerst muss der Transaktionsmanager gestartet werden.
2) Nun sollten die Stationen hochgefahren werden.
3) Jetzt kann sich der User aka Client verbinden.
4) Der Client sendet einen SQL Befehl an den Server (TM).
5) Transaktionsmanager leitet die Prepare-Phase ein.
6) Die Stationen verbinden sich mit der Datenbank.
7) Eine Meldung über den Aufbau wird den Transaktionsmanager geschickt.
8) Bei Erfolg, Transaktionsmanager leitet den Befehl an alle Stationen weiter.
9) Die Operationen werden an den Stationen ausgeführt.
10) Eine Rückmeldung der Operation wird an den Transaktionsmanager zurückgesendet.
11) Bei Erfolg, schickt der Transaktionsmanager eine Erfolgsmeldung an den Client und eine Nachricht an die Stationen, dass sie die Verbindung zu der Datenbank beenden können.