Transaction Management in Federated Databases

Paul Grefen
Center for Telematics and Information Technology
University of Twente
Transactions

Transactions form the basis for reliable operations on a database.

A transaction is a conceptual unit of work that should be handled as 'one operation'.

In centralized database systems, transaction support is well supported. In coupled database systems, things may be a bit more problematic .........
ACID properties

in traditional database environments we have ACID properties ensuring transaction semantics:

Atomicity: no partial effects of transaction execution

Correctness: no incorrect states or transitions w.r.t. constraints

Isolation: no incorrect concurrent execution of transactions

Durability: no loss of committed database updates
Coupled database systems

Parallel database system (PDBS):
  goal : more performance
  distrib. : same site
  coupling : tight, no autonomy

Distributed database system (DDBS):
  goal : local data access
  distrib. : multiple sites
  coupling : tight, little autonomy

Federated database system (FDBS):
  goal : data integration
  distrib. : varying
  coupling : loose, full autonomy
Example FDBS application

Hospital A

Physicians
Patients A

Hospital B

Patients B
in federated database systems, site autonomy key issue:

+ local arrangements for authorization and security
+ no local performance reduction because of remote waits
+ no local execution errors because of remote errors

but

- severe problems in guaranteeing ACID properties of multi-site (global) transactions
autonomy in FDBS leads to ACID problems for global transactions:
- Atomicity cannot be guaranteed
- Correctness may be hard to check
- Isolation cannot be guaranteed

possibly resulting in
- phantom records
- forgotten records
- lost updates
- incomplete transactions
- phantom global states
Problem: phantom record

\[ TA \]

\[
\text{SELECT * FROM PatientsB} \\
\text{INTO Temp} \\
\text{WHERE Physician = "Dr. No"}
\]

\[
\text{INSERT INTO PatientsB} \\
\text{VALUES} \\
("J.Bond", ... , "Dr. No")
\]

\[
\text{DELETE FROM PatientsB} \\
\text{WHERE SSN IN Temp.SSN}
\]

\[ TB \]
**Problem: forgotten record**

**TA**

MOVE FROM PatientsB TO Temp
WHERE SSN = 007

**TB**

SELECT * FROM PatientsA WHERE SSN = 007
UNION
SELECT * FROM PatientsB WHERE SSN = 007

INSERT INTO PatientsA VALUES IN Temp
Problem: lost update

TA

UPDATE Physician
SET Salary
TO Salary + 100

TB

SELECT * FROM Physicians
WHERE Name = "Dr. No"
LET NewSal = INPUT

UPDATE Physician
SET Salary TO NewSal
WHERE Name = "Dr. No"
Problem: incomplete transaction

TA
MOVE FROM PatientsB
TO Temp
WHERE SSN = 007

SELECT * FROM PatientsA
WHERE SSN = 007
UNION
SELECT * FROM PatientsB
WHERE SSN = 007

>> CRASH <<

INSERT INTO PatientsA
VALUES IN Temp

Transactions in Coupled Databases - Paul Grefen
Solutions to transactional problems

**Do Nothing**
- unreliable global applications

**Application Redesign**
- incomplete solution

**Master-Slave Transactions**
- violation of site autonomy
- performance bottleneck

**Global Transaction Monitor**
- violation of site autonomy
- performance bottleneck
- additional software costs

**Transactional Protocols**
- performance bottleneck
- additional complexity
Lost update revisited

\[ TA \]

```
SELECT * FROM Physicians
WHERE Name = "Dr. No"
LET Raise = INPUT

UPDATE Physician
SET Salary
TO Salary + 100
```

\[ TB \]

```
UPDATE Physician
SET Salary TO Salary + Raise
WHERE Name = "Dr. No"
```
Case: global integrity control

local integrity constraints define conditions over one database, e.g.:

the age of a patient is between 0 and 125

global integrity constraints define conditions over multiple databases in a federation, e.g.:

no patient can be registered at both Hospital A and Hospital B

checking global constraints requires global integrity control mechanisms, which suffer from the lack of global ACID transactions
Integrity checking process

**TA**

INSERT INTO PatientsA
VALUES (...., 007)

**TB**

MOVE FROM PatientsA
TO PatientsB
WHERE SSN = 007

**IC**

SELECT SSN
FROM PatientsA INTO T1

SELECT SSN
FROM PatientsB INTO T2
COMPARE T1 AND T2
INTEGRITY VIOLATION
Integrity checking aspects

Safety:
is every integrity violation signalled by integrity checking mechanism?
*note: safe checking may be too pessimistic!*

Accuracy:
is every integrity violation signal caused by an integrity violation?
*note: accurate checking may be too optimistic!*

Safe + Accurate = Ideal
Timestamped integrity checking protocol

Transactions in Coupled Databases - Paul Grefen
Selecting a solution

no ideal solution for transactional problems in FDBS, choice depends on various factors:

- required database consistency
  application dependent!

- permitted performance loss

- permitted autonomy loss

- permitted software complexity

- permitted software costs