AG Heterogene Informationssysteme
Prof . Dr.-Ing. Stefan Deßloch
Fachbereich Informatik
Technische Universität Kaiserslautern

Middleware for Heterogeneous and Distributed Information Systems – Exercise Sheet 1

Wednesday, October 29, 2008 – 10:00 to 11:30 – Room 48-379

ACID Properties

Transactions are considered as collections of actions with so called ACID properties. The term ACID is an acronym and stands for atomicity, consistency, isolation, and durability. Briefly describe each of these properties!

Distributed Transactions

The two-phase commit protocol (2PC) allows a set of autonomous agents to eventually all commit or all abort. Thus, the 2PC guarantees atomicity of distributed transactions.

Consider two autonomous agents and a coordinator involved in a distributed transaction using the 2PC. Further consider that the coordinator is asked to commit the distributed transaction. What messages are exchanged between the coordinator and the agents and what log entries are written in the following cases?

1. Both agents commit successfully.
2. One of the agents encounters an internal error within the prepare phase that prevents the transaction from completing successfully.
3. One of the agents crashes within the prepare phase.
4. One of the agents crashes within the commit/abort phase after both agents voted commit.

Distributed Information Systems

Information systems are most often composed of distributed systems such as database systems, application servers, or legacy applications, for instance. Middleware is used to hide the distribution from the application programmer and provide a single system image. Different units of distribution are feasible. Four alternatives have been presented in class, namely transaction routing, programmed distribution, distributed database operations, and DBMS-controlled distribution.

```
CREATE TABLE lecture (
      lectureId INTEGER NOT NULL PRIMARY KEY,
      title VARCHAR(255) NOT NULL,
      lecturer VARCHAR(255) NOT NULL,
      timeslot INTEGER NOT NULL,
      description VARCHAR(255)
);

CREATE TABLE reservation (
      roomId CHAR(6) NOT NULL,
      timeslot INTEGER NOT NULL,
      lectureId INTEGER NOT NULL,
      PRIMARY KEY(roomId, timeslot)
);
```

Figure 1: Sample database schemas

As an example, consider an information system that allows for planning and scheduling of university lectures. The system is based on two (legacy) systems namely the *lecture catalogue system* and the *room reservation system*. Scheduling a lecture involves both, updating the lecture catalogue and reserving a lecture room. Note that certain lectures must not overlap, i.e. scheduling a lecture may fail due to time conflicts. Similarly, reserving a lecture room fails if the room is unavailable.

1. Take transactions as the unit of distribution (alternative 1). Assume that remote procedures are available to reserve lecture rooms and catalog lectures. Specify a program for scheduling a lecture! Explain why distribution alternative 1 is unsuited here!
2. Take application programs/components as the unit of distribution (alternative 2). Assume that transactional remote procedures are available to reserve lecture rooms and catalog lectures. Specify a program for scheduling a lecture!
3. Take database operations as the unit of distribution (alternative 3). Think of the lecture catalogue and the room reservation system as being relational database tables as shown in Figure 1. Specify a program for scheduling a lecture!
4. Assume that distribution is controlled by a federated DBMS (alternative 4). Again, think of the lecture catalogue and the room reservation system as being relational database tables. Specify a program for scheduling a lecture!
5. Consider that you are given the name of a lecturer and you are interested in finding all rooms occupied by lectures taught by this lecturer. Specify suitable programs for this task for the distribution alternatives 3 and 4.