## 06431 – Working Group Summary

# — Atomicity in Mobile Networks —

## Dagstuhl Seminar<br>Scalable Data Management in Evolving Networks

Joos-Hendrik Böse[1], Stefan Böttcher[2], Panos Kypros Chrysanthis[3], Alex Delis[4], Le Gruenwald[5], Anirban Mondal[6], Sebastian Obermeier[2], Aris Ouksel[7], George Samaras[8], Stratis D. Viglas[9]

[1] Freie Universitt Berlin, Germany
boese@mi.fu-berlin.de
[2] University of Paderborn, Germany
stb | so @upb.de
[3] University of Pittsburgh, USA
panos@cs.pitt.edu
[4] National and Capodistrian University of Athens, Greece
ad@di.uoa.gr
[5] US National Science Foundation - Arlington, USA
ggruenwald@ou.edu
[6] University of Tokyo, Japan
anirban@tkl.iis.u-tokyo.ac.jp
[7] University of Illinois at Chicago, USA
aris@uic.edu
[8] University of Cyprus, Cyprus
cssamara@cs.ucy.ac.cy
[9] University of Edinburgh, UK
sviglas@inf.ed.ac.uk

**Abstract.** We introduce different mobile network applications and show to which degree the concept of database transactions is required within the applications. We show properties of transaction processing and explain which properties are important for each of the mobile applications. Furthermore, we discuss open questions regarding transaction processing in mobile networks and identify open problems for further research.

**Keywords.** Data Management for Evolving Networks, Mobile Transactions, Atomicity

## 1 Introduction

Mobile devices are becoming more and more popular, and are going to be equipped with a processing capability that exceeds that of a workstation PC of the year 2000. Mobile networks are expected to fulfill the vision that each

Dagstuhl Seminar Proceedings 06431
Scalable Data Management in Evolving Networks
http://drops.dagstuhl.de/opus/volltexte/2007/952

and every device is connected to one another, which has an enormous impact on our everyday life. However, the use of mobile devices in business transactions will be limited as long as there is no adequate support for mobile transactions. In the following, we will point out different applications of mobile transactions (Section 2) and conduct investigations on the need for transaction processing support in the explained applications (Section 3). Furthermore, we outline open research topics and main challenges in Section 4 and give a short summary and conclusions in Section 5.

## 2 Example Applications for MANETS

There are several basic applications for MANETS, for which we will explain whether transactions are useful.

**Rescue applications** use mobile networks to communicate with different machines and human beings. A rescue application for firefighters can be used to develop rescue plans and form virtual teams of different fire brigades. Furthermore, the mobile networks can be used to locate fire trucks and give moving instructions to them. In such a scenario, there is the need for transaction support in the sense, that fire trucks and fire fighters are considered as resources that receive instructions. Since resources cannot perform contradictory instructions at the same time and most plans require that more than one unit processes the instructions, properties like atomicity and isolation must be supported.

**Mars rovers** explore the surface of the Mars by collecting measurements, taking rock samples and comparing them to previously identified materials. In order to analyze the surface of the planet, the Mars rovers must further combine their locally measured data with data that is already present in the network. Therefore, the network serves as a large database. While some types of Mars rovers move fast, some of them move at a moderate speed. To apply well tried and tested standard distributed database technology in such a scenario, we need a non-blocking atomic commit protocol that not only stabilizes the coordination process, but also reduces the blocking of participating databases, especially if the databases are suspected to frequently disconnect from the network, like moving Mars rovers do.

**Homecare** applications assist a nurse having mobile devices like PDAs to get information about the patient. Transaction support is required whenever patient data, medicine data, or subscriptions get updates. However, the amount of transactions is usually very low. Therefore, the concurrency control mechanisms used do not need to be efficient regarding the transaction throughput, but they must be efficient regarding the blocking of data.

**M-Commerce** scenarios like mobile auction applications assist sellers and buyers of a flea market in terms of searching and localizing desired items. Furthermore, contracts between several buyers can be signed to gain volume discounts. Transaction support is necessary in order to achieve consistency for contracts and buying/selling actions.

## 3 Properties of Transaction Processing in MANETS

Transaction processing in MANETs involves new challenges compared to fixed wired networks. We will explain different properties and show whether they are applicable to the mobile applications introduced in Section 2.

**Transaction Throughput**
Although a smaller amount of transactions must be processed in mobile networks, the throughput varies from application to application. Therefore, the protocols and concurrency control mechanisms used must take the transaction throughput into consideration.

**Minimized Blocking Time**
The time that the transaction processing nodes must block their resources must be minimized for some applications. The following dependency exists between the concurrency situation and the blocking time: The blocking of resources that are never needed by concurrent transactions has no harmful effect. Therefore, if there is no concurrency, the blocking time must not be minimized. [1] proved that there is no non-blocking atomic commit protocol in the sense that the transaction processing nodes must block the data after they have sent their commit vote on the transaction.

**Concurrency**
The amount of parallelly processed transactions effects the the choice for optimistic or pessimistic concurrency control mechanisms. Furthermore, the amount of conflicting transactions that must be processed concurrently influences the blocking time.

**Atomicity**
Whenever a transaction consists of more than one sub-transaction, atomic commit protocols can be used to ensure an atomic transaction execution.

Table 1 shows the importance of different transaction properties for the applications in our example. As we can see, atomicity is an important property whose support is essential for a lot of applications. In the following, we will investigate the challenges of guaranteeing atomicity in mobile networks.

## 4 Challenges and Research Topics in the Field of Guaranteeing Atomicity

We have identified the following challenges and open questions for atomic transaction processing in mobile networks:

**Guaranteeing global data integrity**
One of the main challenges which motivates the use of atomic commit protocols is to guarantee global integrity constraints. If transactions are not

| Application | Transaction Throughput | Need for Reduced Blocking Time | Concurrency | Atomicity |
|---|---|---|---|---|
| **Rescue App.** | high | critical | only one command at a time | required |
| **Mars Rovers** | medium | medium | high | depending on desired consistency |
| **Homecare** | low | low | low | required |
| **M-Commerce** | depending on market situation | medium | medium | required for distributed transactions |

**Table 1.** Transaction properties for different applications

processed in an atomic fashion, the result can harm integrity constraints. How can we ensure that defined global integrity constraints hold even if we have completely independent database systems?

**Global Commit ⟶ Individual Commit**

How can we break down the global commit decision to a set of individual commit decisions, in order to decentralize the commit decision making process?

**Localize Commit Decisions**

How can participants efficiently localize commit decisions in the network, even if participants fail? To which extend should the commit decision be disseminated in order to be reliably stored?

**Disconnections and Network Partitioning**

In mobile networks, disconnections may occur at any time and are no failure at all. Furthermore, network partitioning can divide the network into several partitions that cannot communicate with each other. How can we design a protocol that does not suffer from network partitioning as much as current protocols do?

**One Sided Disconnections**

In comparison to disconnections and network partitioning, in scenarios with one sided disconnections, some participants may only receive messages but are not able to send messages to other participants, e.g. due to a low battery, limited transmitting power, or geographical issues. Can we define a protocol that takes advantage of the one sided disconnection?

**Reduce Risk of Infinite Blocking**

Although the possibility of having an infinite blocking is proven to be unavoidable [1], we may not only reduce the chance that this infinite blocking occurs by reducing the time frame in which participants are blocked (cf. e.g. [2]), we may also want to reduce the number of blocked participants using multiple coordinators (cf. e.g. [3]). Can we find criteria that allow us to predict whether some participants are more susceptible to infinite blocking than others?

**Sacrifice Consistency for Energy**

Can we sacrifice consistency constraints in order to save energy? How can we detect inconsistent states and how can we repair inconsistent participants?

**Redo / Retry in Case of Failures**

When a failure occurs during transaction processing, how can we reuse old results to a maximum extent instead of repeating the whole transaction? Can we redo only parts of the transaction without losing serializability?

**Heuristics**

Can we identify situations in which relaxing atomicity or consistency is not harmful? Think, for example, of a rescue operation that has blocked resources but does not reach a commit decision. In such a case, relaxing atomicity would mean that other transactions can be processed first even if the blocked transaction cannot be committed anymore. Can we use a heuristic approach to ensure consistency and compensate later on by spawning new transactions that restore consistency? Can we, as another possibility, compensate to an equivalent prior state instead of compensating the transaction? To what extent do we have to allow inconsistency between certain points in time in order to speed up transaction execution?

## 5 Summary and Conclusions

We have introduced different application scenarios based on mobile networks, which all have in common that they profit from transaction support. However, as we pointed out, each scenario has individual properties that must be supported. Furthermore, we have shown open research questions and challenges for transaction processing in mobile networks. In summary, research on transaction execution in mobile networks is still a challenging and interesting topic.

## References

1. Bernstein, P.A., Hadzilacos, V., Goodman, N.: Concurrency Control and Recovery in Database Systems. Addison-Wesley (1987)
2. Böttcher, S., Gruenwald, L., Obermeier, S.: Reducing sub-transaction aborts and blocking time within atomic commit protocols. In: BNCOD. (2006) 59–72
3. Böse, J.H., Böttcher, S., Gruenwald, L., Obermeier, S., Schweppe, H., Steenweg, T.: An integrated commit protocol for mobile network databases. In: 9th International Database Engineering & Application Symposium IDEAS, Montreal, Canada (2005)