Context Aware Mobile Transactions

Patricia Serrano-Alvarado*, Claudia Roncancio, Michel Adiba, Cyril Labbé
LSR-IMAG Laboratory, Grenoble France
Firstname.Lastname@imag.fr http://www-lsr.imag.fr/storm

Applications in mobile environments are confronted to limitations imposed by wireless networks and mobile hosts – low/variable bandwidth, frequent disconnections, high communication prices, limited battery autonomy, etc. These limitations lead to several potential failures that affect data management (e.g., queries, replication, transactions) [1]. In the mobile context, in some cases these failures must be handled as "normal" or "weak performances" and not as misfunctions. In traditional environments, application designers do not care about dynamic variability of host/network characteristics. Nevertheless, in the mobile context, it is crucial to overcome infrastructure variability to better fit application/user requirements. To overcome context variability, systems need to be *context aware*.

This work focuses on mobile transaction management. There exist several proposals for mobile transactions (see a survey in [2]). They offer solutions in several aspects but few of them take into account the importance of mobile environment variability and context awareness.

We propose a general solution allowing to execute transactions over one or several autonomous databases located on one or several mobile hosts – laptops, PDA, cell phones – or fixed hosts on the wired network. Our proposal includes an Adaptable Mobile Transaction (AMT) model and the TransMobi middleware that support it. The AMT model allows to define mobile transactions – T_{AMT} – with several *execution alternatives* each of them associated to a particular mobile *environment descriptor*. Execution alternatives may be semantically equivalent. The successful execution of one of them represents a correct execution of the corresponding T_{AMT}. When a T_{AMT} is launched, the appropriate execution alternative is initiated by the system depending on the current state of the mobile context. Our analytical study shows how the AMT model increases transaction commit probability according to expected costs. Indeed, selecting dynamically appropriate execution alternatives improves the application's quality of service (commit rate, execution costs, response times, application availability, etc).

* Supported by a CONACyT scholarship of the Mexican Government

Execution alternatives (EA_i) contain a set of *component transactions* with execution dependencies (precedence, parallelism). EA_i and T_{AMT} are coordination units; data access is done only by component transactions. *Compensating transactions* may be associated to component transactions. Component transactions must be ACID whereas EA_is are *semantically atomic* and T_{AMT}s are *semi-atomic*.

As a middleware between application code and existing DBMSs, TransMobi coordinates the execution of T_{AMT}s and provides context awareness (e.g., bandwidth rate, communication price, MH disconnections, MH available energy, etc.). It uses suitable protocols to guarantee AMT properties. We consider that ACID properties of component transactions are ensured by the underlying DBMS. To ensure *semantic atomicity*, we propose the CO2PC (Combination of an Optimistic approach and 2PC) commit protocol. It allows compensable component transactions to commit unilaterally whereas non-compensable ones synchronize their commit with the corresponding EA_i commit. Aborting an EA_i implies to abort non-compensable transactions and to compensate compensable ones. The CO2PC protocol requires standard interfaces. It minimizes wireless communication, allows MH disconnections and reduces resource blocking. Compensating transactions play a crucial role in this respect. To ensure a correct global execution order of T_{AMT}s, we adapt the *optimistic ticket method* (proposed by Georgakopoulos et al). It leads to global serializability without compromising performances and allowing to run component transactions on top of autonomous heterogeneous DBMS.

TransMobi has a client/agent/server architecture. Main features has been implemented on a prototype in Personal Java on Ipaq H3850 (Compaq) and a WLAN 802.11b. For more details on our work see [3].

References

[1] E. Pitoura and G. Samaras. *Data Management for Mobile Computing*. Kluwer Academic, 1998.
[2] P. Serrano-Alvarado, C. Roncancio, and M. Adiba. A Survey of Mobile Transactions. *DAPD Journal*, 2004. To appear.
[3] P. Serrano-Alvarado, C. Roncancio, M. Adiba, and C. Labbé. Adaptable Mobile Transactions and Environnement Awareness. In *BDA*, 2003.