Mobile nested transactions for nomadic teams

Luis A. Gama-Moreno\textsuperscript{a}, Matías Alvarado\textsuperscript{b,\textasteriskcentered}

\textsuperscript{a}CIC, Instituto Politécnico Nacional J. Batiz esq. O. de Mendizabal s/n. CP 07738, México DF, Mexico
\textsuperscript{b}PIMAyC, Instituto Mexicano del Petróleo, Eje Central Lazaro Cárdenas 152, San Bartolo Atehuacan CP 07730, México DF, Mexico

Abstract

This paper introduces the Mobile Nested Transactions (MNT) model for transaction processing coordination in groups of mobile devices. The model extends traditional approaches—centralized and distributed—for distributed transaction processing over mobile devices. Under the Nested Transactions approach, sub-transactions—logical work units—processing control is highly flexible and fault-tolerant. Hence, transaction processing with mobile groups lacking of access to a wired network or the Internet becomes possible. A MNT additional advantage is that sub-transactions could be distributed among mobile devices. This is, whenever some sub-transactions are successfully committed, they cannot be affected by failed or pending sub-transactions.

Keywords: Transactions; Mobile computing; Mobile transactions; Nested transactions

1. Introduction

This paper describes a transaction processing model based on the nested transaction scheme and oriented to wireless mobile devices (Doucet, Gancarski, Leon, & Rukoz, 2000; Gray & Reuter, 2003; Moss, 1985). Transaction processing has evolved from centralized (Gray & Reuter, 2003) and distributed systems (Marsh, 2000; Özsu & Valduriez, 1999; Ram, Do, & Drew, 1999), to transaction processing over mobile devices; the so called Mobile Transactions (Dunham & Kumar, 1999; Madria & Bhargava, 1998; Murthy, 2001; Serrano-Alvarado, Roncancio, & Adiba, 2001a,b). Currently, these mobile transactions access data managed by database management systems (DBMS) within a fixed wired network. That is, even when data is accessed by devices on-the-road, it is still processed by the originating device connected to a fixed network (Serrano-Alvarado et al., 2001a,b).

Data management in centralized transactions (CT) approach, is done by local DBMSs within a wired network (Gray & Reuter, 2003). On the other hand, in the Distributed Transactions processing (DT) scheme, a transaction can be split into sub-transactions (fragments). This depends on which network nodes contain the requested objects (Özsu & Valduriez, 1999). There comes the necessity for a coordinator located at the transaction’s home node; it is in charge of controlling the transaction fragments processing. The coordinator uses the Two-Phase Commit (2PC) protocol in order to confirm fragments execution, although the whole processing is carried out inside a wired network. However, it is possible that nodes are physically distant, ranging from a few feet to large geographical distances. The fact is that, they are connected to a network and their location is well-known. This applies both for the nodes where each fragment was processed and for the coordinator.

Mobile Transactions processing (MT) is different from DT processing in the sense that the coordinator which begins and ends a transaction, can change its physical location. It could be also possible that one mobile device begins a transaction and a different one finishes it at a different place (Kumar & Dunham, 1998). Moreover, transaction’s end may happen at a different place where it was started and/or with a different device—although the user remains the same (See Fig. 1). Nevertheless, the actual transaction processing is accomplished by local database management systems at the wired-network side.

Current research is focused on how to reproduce the traditional scheme for transaction processing—centralized or distributed (Gama & Alvarado, 2002)—by using groups of wireless connected devices. The point is that a group of mobile devices with enough internal computing capability,\textsuperscript{1}

\textsuperscript{1} A mobile device capable of transaction processing, must have enough computing processing capability. That is the case of portable computers, PDA’s and released mobile phones.
while being connected through wireless links, will perform tasks commonly assigned to database servers in wired networks (See Fig. 2). The goal to reach by introducing the Mobile Nested Transaction (MNT) model is:

Moving the work load currently performed at fixed network systems to mobile devices with enough computing capability.

This way is that transaction processing independence shall be achieved in zones with lack of access to a wired network or the Internet. There are circumstances where it is not possible to be supported by wired connections in order to process transactions; for instance, after a natural disaster occurs or at high-risk zones.

Given the risk of frequent disconnections, as it happens very often in a mobile environment, transaction processing coordination could be endangered by events such as: unavailable connections, poor bandwidth for data transmission, a host failure, discharged batteries and limited storage. There are also external situations which complicate smooth communications: electromagnetic interference, moving into a weak or null reception area, mountains, and so on. Due to these reasons, a global transaction could stop or wait for the other nodes on the network to restart their work. Such mobility features characterizing mobile devices have introduced new challenges that concern:

- Concurrency control
- Distribution of transaction processing
- ACID properties preservation: atomicity, consistency, isolation and durability.
- Long-Lived transactions due to the time a host remains disconnected.

Using the MNT model, a transaction can be split into sub-transactions that define logical work units (Dogdu, 1998); thus creating a sub-transactions hierarchical tree. Each sub-transaction is processed by a host from the group of mobile devices. The advantages of this model as the solution for the problems above mentioned are:

1. Each sub-transaction can be split into sub-transactions as well
2. There is concurrent execution among sub-transactions
3. Work load is distributed among different devices
4. Global transaction validation is possible on behalf of some sub-transactions failure

Point (1) means that the global transaction and sub-transactions create a hierarchy of work units controlled by the global (top level) transaction. Point (2) means that each sub-transaction could be executed independently of the rest (the final validation depends on the global transaction). Point (3) shows the advantage of distribution among devices, so that processing is accomplished locally in relation to each device, thus reducing dependences among sub-transactions. Point (4) refers to the main advantage of NT model approach: the possibility for top-level transaction commitment despite of failed sub-transactions. The MNT model is illustrated in Example 1.

Considering the MNT features, work load could be also split into logical units where each piece of work is assigned to a wireless mobile device. Once a sub-transaction is successfully concluded, changes get reflected in the database despite of failures from other sub-transactions. This advantage represents additional value under difficult environmental circumstances, high-risk facilities such as oceanic oil platforms, ships at open sea and hardly accessible regions that complicate communications like those described in the Section 4 Example.

The rest of the paper is organized as follows. Section 2 describes the Nested Transaction model, features, and effects over ACID properties and concurrency control. Section 3 mentions different scenarios that are expected
during mobile devices operation. Section 4 shows an example on how this model applies to transaction processing by group of mobile devices in oceanic oil platforms. In addition, how a MNT Monitor is the first step in the development of software based on the model, is shown. Section 5 is a discussion about possible solutions to mobile computing challenges concerning: security, mobile devices location and concurrency control. Section 6 presents concluding remarks.

2. Nested transactions

In the NT model introduced by Moss (1985), a transaction can hold some sub-transactions, and each sub-transaction, on its own, can hold sub-transactions as well. The full transaction defines a hierarchical tree, where the top-level one is called root transaction. Nested transactions compose hierarchical pieces of work, accomplished by sub-transactions that can commit or abort independently of each other (Fig. 3). Transactions holding sub-transactions are known as parents and their sub-transactions are called children. Transactions no holding sub-transactions are named leaves; as a matter of fact, these latter last are flat transactions.2

A nested sub-transaction is either mandatory or optional. Whenever it is mandatory and aborts, its parent (immediate upper ancestor) must abort as well (Moss, 1985). On the other hand, if it is optional, the parent transaction is not required to abort as well. Having this behavior in a group of mobile devices, it is possible that a host H1 coordinating two hosts H2 and H3 can validate its work, even if H2 or H3 decide to abort (either one or both). The concluded tasks by each sub-transaction can be stored at the respective devices thus making data persistent (durability). Each sub-transaction inside a NT schema—including the root one—must behave this way.

Example 1. Suppose there is one transaction Ti and it is split into sub-transactions Ti1, Ti2 and Ti3. Now, suppose that transaction Ti1 is split into T1i1 and T1i2, transaction Ti2 splits into T2i1, T2i2 and T2i3. Transaction Ti3 is not split (see Fig. 4). Ti commitment may happen even when one of its dependent transactions abort (Fig. 5). If transaction Ti2 decides to abort, then all the effects from its dependent transactions (T2i1, T2i2 and T2i3) shall be aborted as well. Alternatively, in the final execution, if Ti decides to commit, changes made by sub-transactions T1i1, T1i2 (controlled by Ti1) and Ti3 will be also committed (see Fig. 6). Transactions Ti1, Ti2, Ti3 and their descendants do only comply with atomicity, consistency and isolation properties. Durability property is only preserved by the global transaction as detailed in Section 2.1.

2 Under a NT schema, flat transactions are the only ones granted to make changes over database objects.

2.1. ACID properties

NT behavior with respect to ACID properties is as follows: only the root can ensure the durability property (Doucet et al., 2000). Whenever a sub-transaction at any level from the NT tree abandons, every children sub-transaction belonging to the failed one must abandon as well. This is quite a reason for children sub-transactions to comply with atomicity, consistency and isolation but durability. Thus, once changes done by a transaction are confirmed, they can only be cancelled by the parent transaction—from which it depends. In addition, if the root transaction decides to abandon, every sub-transaction must be cancelled. In this way, only the root transaction complies with all the ACID properties.

2.2. Concurrency control

Concurrency control algorithms are used in order to guarantee serialization and as a consequence, to ensure transaction isolation (Gray & Reuter, 2003). Several algorithms and protocols have been proposed in the current

---

2.1. ACID properties

NT behavior with respect to ACID properties is as follows: only the root can ensure the durability property (Doucet et al., 2000). Whenever a sub-transaction at any level from the NT tree abandons, every children sub-transaction belonging to the failed one must abandon as well. This is quite a reason for children sub-transactions to comply with atomicity, consistency and isolation but durability. Thus, once changes done by a transaction are confirmed, they can only be cancelled by the parent transaction—from which it depends. In addition, if the root transaction decides to abandon, every sub-transaction must be cancelled. In this way, only the root transaction complies with all the ACID properties.

2.2. Concurrency control

Concurrency control algorithms are used in order to guarantee serialization and as a consequence, to ensure transaction isolation (Gray & Reuter, 2003). Several algorithms and protocols have been proposed in the current

---

2 Under a NT schema, flat transactions are the only ones granted to make changes over database objects.
literature, but the most popular are: two-phase lock (2PL) and time-stamping ordering. Such algorithms were originally developed with the flat transaction model in mind (Dogdu, 1998). For the nested transactions model, Moss proposed the 2PL (Moss, 1985).

Under the 2PL protocol, the first phase comprises request locks to the required objects before their use. In the second phase, a transaction confirms its result (commit) by writing all the modified data to persistent storage and releases all the acquired locks. There are two types of locks in this protocol:

- **Shared**: if a transaction \( T \) gets a shared lock over an object \( Q \), then \( T \) can read but not write over \( Q \).
- **Exclusive**: if a transaction \( T \) gets an exclusive lock over an object \( Q \) then \( T \) can both read and write over \( Q \).

A transaction can only have access to a given object if and only if there is an adequate lock over such object. Each transaction must get a lock over an object \( Q \) depending on the type of operation to be executed. Given a set of locking modes, a compatibility function between them can be defined as follows:

- Let \( A \) and \( B \) represent two arbitrary locking modes. Suppose that a transaction \( T_i \) asks for a lock over an object \( Q \) under the \( A \) mode.
- \( \exists T_j (i \neq j) \) which currently holds a \( B \) mode lock over object \( Q \).
- If \( T_i \) is able to get the requested lock over \( Q \), then it can be said that \( A \) locking mode is compatible with \( B \) locking mode.

Such compatibility function can be represented by a matrix \( M \) where \( M(i,j) = True \), if \( i \) locking mode is compatible with \( j \) locking mode. Table 1 shows the compatibility matrix \( M \) between different locking modes.

### 2.3. Objects locking in NT

Moss (1985) proposed a 2PL protocol for operations control that may fall in conflict. The protocol is as follows:

- **Leafs** can get a lock over an object only if the object is not currently locked by another transaction or, it can get the lock if every transaction which has already locked the object is a predecessor, else, it must wait until the object is released.
- Once a transaction gets a lock, it can only be released until the transaction is concluded, either with commit or abort.
- When a sub-transaction abandons, all of its locks are released. Then, its parent will inherit them.
- After a sub-transaction confirms, all of its locks get to be managed by its parent.
- When a root transaction commits, every inherited lock from children transactions will be released.

### 2.4. 2PL protocol for NT

As it has already been mentioned, transaction processing with mobile groups can be done in poorly communicated zones, those where access to a fixed network or the Internet is difficult. TP control is done with NT, thus rising a model that will be called Mobile Nested Transactions. Under this scheme, a group of mobile devices connected with wireless network interfaces:

- Define a wireless network
- Name a coordinator (root node)

This root node controls transaction processing for its children nodes (sub-transactions). The global transaction
(root) must synchronize processed data against centralized databases in a wired-network or the Internet.

Given the mobility feature—a prime advantage from mobile computing devices are exposed to frequent disconnections. This way, transaction control becomes complex in order to comply with the ACID properties. Main challenge for the MNT model is to minimize disconnection consequences in mobile devices. The MNT model flexibility increase the probability to save more confirmed work units. Devices mobility is shown in the next scenarios:

(a) Unmoving (fixed) Hosts
(b) Moving children hosts
(c) Moving parent host
(d) All in moving.

Whenever the devices stay at the same place (a)—an exception—from the transaction beginning to the end, the scenario looks like a CT processing environment. However, quite often, one or many devices can change their location (b, c). As a result, the rest of the group members must get the new location in such a way that transaction processing remains transparent. If the coordinator device moves from one place to another (c), it must notify the new identity to the coordinated devices. Finally, there is the possibility for another scenario where all of the devices change their current location (d).

All those described situations point towards different mobility circumstances thus introducing more complexity in transaction processing. One problem that should be solved, concerns the increase of network traffic due to the numerous messages sent among devices. Besides, transactions are supposed to be long-lived (Ding, Meng, & Wang, 2001), whether because they are loaded with a large amount of tasks or due to long disconnection intervals. The consequences can go from locked objects over a large period of time, to the worst, deadlocks with other transactions.

In the mobile environment, each covered zone is a delimited transmission area, cell represented, where mobile devices get network access through wireless communication. Common use is found in cellular phones but now it is extended to any mobile device. Wireless devices going from cell to cell experiment handoffs, defined as a switching cell mechanism—sometimes all devices stay in the same cell. In the rest of the paper, zone and cell will be used as synonymous.

3.1. Scenario 1: unmoving (fixed) hosts

A transaction $T_i$ gets started at MH1 (Mobile Host) inside zone $Z_1$ (bounded by the signal range from a transmission antenna) and becomes the root transaction; MH1 is called Central Host or Global Coordinator. The transaction is then split in two children sub-transactions $T_{i1}$ and $T_{i2}$ that will be performed by mobiles hosts MH2 and MH3 under cells $Z_2$ and $Z_3$, respectively. During transaction processing, both MH2 and MH3 stay connected without experimenting any movement or change in their physical location, they remain the same until the root transaction concludes (Fig. 7).

3.2. Scenario 2: moving children hosts

The coordinator node MH1 remains still, but the other devices can change their location during transaction processing. Fig. 8 shows MH3 moving from cell $Z_3$ to cell $Z_4$, thus getting a new identity in the wireless network. MH1 must keep transparent control over the sub-transaction processed at MH3, even when MH3 has been displaced and got a new identity.

3.3. Scenario 3: moving parent host

In this situation, the coordinator (root) node changes its location within the wireless network. In Fig. 9 it can be seen that MH1 moves from cell $Z_1$ to cell $Z_3$, thus changing its identity. Change must be notified to all of the MHs currently processing sub-transactions and depending from $T_i$. This must be done in such a way that, children can remain the same after $T_i$ is displaced.

3.4. Scenario 4: all in moving

In this case, every host will perform displacement, if a child host changes its location, this must be notified to the coordinator; for instance, MH3 moves from cell $Z_4$ to cell $Z_1$ (like in scenario 2). On the order hand, if MH1 (coordinator) moves from $Z_1$ to $Z_3$, then it must notify the new location to all of the dependent hosts. Fig. 10 shows the final state for this scenario. Every device moves from its original location to a different one.
4. Case of study

MNT based applications over wireless networks are a suitable solution for zones with difficult or lack of access to wired networks. This access might be restricted by geographical circumstances (as it happens with oil extraction facilities) or by hazardous environmental conditions (as it is the case of disaster zones). Example in 4.1 illustrates the MNT model performing transactions over portable computers, PDAs and cell phones located at oil platforms, ships and helicopters (Fig. 11). Both the ship and the helicopter are in constant movement with the risk of frequent disconnection and wireless cell interchange.

4.1. Mobile group in oil platforms

Suppose that it is required to process the temperature from a set of extraction wells in ocean platforms and to store it in a database. Statistical information will be later retrieved from such measurements. The application starts transaction $T_i$ from ship $S_1$ and registers the beginning of the root transaction (See Fig. 12); $T_i$ then gets split into subtransactions $T_{i1}$ and $T_{i2}$ that will be located on the helicopters $H_1$ and $H_2$, respectively. Next, $T_{i1}$ and $T_{i2}$ get split into subtransactions $T_{i11}$, $T_{i12}$; $T_{i21}$, $T_{i22}$ for each one of these (the only ones granted for objects manipulation) are located at different oil platforms. If the device at $H_1$ fails because of a natural event that compromises communication with the rest of the group, and considering $H_2$ is at an undisturbed zone, transaction may continue here. In this way, with the MNT model, harmful effects over the global transaction derived from bad climatic conditions or similar, are reduced. This allows confirmation of those accomplished work pieces under consistent state, despite of faulty transactions from same group.

Note that transaction processing is manipulated only by those devices at OP1, OP2, OP3 and OP4. Transactions processed at $H_1$ and $H_2$—with greater failure probability—at
any time, can confirm or cancel the accomplished work, but such changes will only become durable if and only if the root transaction \( T_i \) commits. If \( T_i \) aborts, operations from \( T_{i11}, T_{i12}, T_{i21}, \) and \( T_{i22} \) will be cancelled as well. Sub-transactions that have already finished can commit their results even when some other show failures.

4.2. MNT monitor

Transaction processing (TP) monitors are responsible of coordinating distributed transaction processing among different servers. Upon availability, they balance workload among servers and guarantee both database consistency and a high level of fault tolerance. Additionally, TP monitors ensure right fulfillment of transaction processing, or else, they undertake the necessary steps to overcome errors—2PC is used for this purpose.

In order to coordinate transaction processing under the MNT model, a monitor will be designed to control MNT transactions in mobile devices. The next services are considered:

1. Incoming requests.
2. Nested transactions workflow control.
3. Data synchronization against DB on the fixed side.
4. Data processing during disconnections.

Considering the frequent disconnections in mobile environments, it is the intention to take advantage of the time a device remains disconnected by processing the assigned sub-transaction. At reconnection, the node will inform its ancestor the current state of the transaction; this will allow the node to know what to do depending on the results from other nodes and the global transaction. By the time of reconnection, these can be the expected events:

Let NT be a sub-transaction at any tree level. Let \( T \) be NT’s ancestor denoted by the Predecessor(\( x \)) function. Let Commit(\( x \)) and Abort(\( x \)) be validation and abortion functions for a transaction \( x \), respectively. The Reconnection(\( x \)) function denotes transaction \( x \) reactivation into the system after having gone through a disconnection. In a formal way we have:

\[
\text{Reconnection} (\text{NT}) \land \text{Commit-Predecessor}(\text{NT}) \rightarrow \text{Commit} (\text{NT}).
\]

\[
\text{Reconnection} (\text{NT}) \land \text{Abort-Predecessor}(\text{NT}) \rightarrow \text{Abort} (\text{NT}).
\]

Under these premises, a transaction that has been disconnected for a while, will be able to incorporate or discard completed tasks, and this depends on the current state of its parent coordinator or the global transaction state.

5. Discussion

MNT model offers solutions to those problems mentioned through this paper. Current research involves development of environments considering: high security levels, performance and availability. Besides, there are
additional efforts focused on prediction algorithms, concurrency control, independence levels, fault tolerance and bandwidth restrictions. These issues are studied with read-only transactions over a mobile environment. This is an undergoing research aimed to extend the MNT model.

5.1. Prediction algorithms

Walther and Fischer (2002) introduced a service featuring a prediction algorithm for saving bandwidth. It is aimed to location of networking users. This algorithm makes feasible the implementation of a devices location service when they change from one cell to another or when they get disconnected; the device’s location becomes predictable as well. In turn, this facilitates recognition of the current state for those transactions executed at the disconnected device, thus allowing process confirmation or abortion.

5.2. Security

Mobile transactions are endangered by security issues; the fact is that any user having a wireless device (e.g. a portable computer with a wireless network interface card) can access the wireless workgroup, thus compromising integrity and consistency database by interfering with transactions processing. Some protocols related to wireless network security and communications are: Wired Equivalent Privacy (WEP), WEP2 and Extensible Authentication Protocol (EAP). However, security issues are still under continuous improvements. Luo, Kong, Zerfos, Lu, and Zhang (2002) introduced an approach named self-securing on which multiple nodes collaborate for providing authentication services to other nodes within the wireless network. Up to foreseeing is possible, transactions secure processing under the MNT model is guaranteed in such a way that, only those authenticated devices have authorized access to the wireless network.

5.3. NT concurrency control

Pavlova and Nekrestyanov (1997) introduced a hybrid concurrency control for solving deadlocks in nested transactions. Lee and Lam (1999) introduced a new alternative for optimistic concurrency control under broadcast environments, there, read-only transactions can be processed locally at mobile clients. Only those that perform updates are sent to the servers for final validation. Chan, Lee, and Lam (2003) defines a requirement for keeping consistency and a high concurrency level in read-only transactions groups, and it is called Group Strong Consistency. Ding et al. (2001) explained there is a combination between optimistic concurrency control with two-phase commit (2PC) aimed to solve common problems in mobile transactions such as: mobility, disconnections and long-lived cases.

5.4. Monitors

Commercial monitors currently found are: BEA Systems’ TUXEDO (Bernstein & Newcomer, 1997; Gray
& Reuter, 2003), IBM Corp.’s Encina Transaction Server (Bernstein & Newcomer, 1997), Microsoft Corporation Transaction Server (Limprecht, 1997), among others. Provided services by those monitors assume persistent and high connections among participant hosts, monitors and servers. However, these issues are not compatible at all with mobile computing environments. Here (frequent) device disconnections during a transaction, force users to go disconnected for unknown time intervals and then reconnect. Walborn and Chrysanthis (1999) introduced PRO-MOTION as a disconnected transaction processing system; there, consistent information units are processed locally at the device. The Off-Line Transaction Processing Monitor allows users to take fragments of the Database and to process them while being disconnected (Nance 2003)); later updates are applied over central Databases. If there is any anomaly, the OFTP monitor can solve such conflict.

6. Conclusions

The MNT model allows work splitting into logical units that are executed by sub-transactions. Prime advantages are:

- Sub-transactions distribution among devices.
- Consistent work pieces validation.
- Successful sub-transactions confirmation despite of faulty sub-transactions from the same group.
- Isolation guarantee at the sub-transaction level which in turn allows control over conflicting operations at each level in the transaction tree.

Applications performance is improved under this model as independent sub-transactions (tree branches) that reduce the possibility of inconsistency between operations. Also, long-lived transactions effects—at bad environmental conditions—are less conflictive as each transactional tree branch is not supposed to wait for the results from other branches. Open challenges for improvement concern: security, concurrency control, transactions isolation and read-only transactions, among some of them. A MNT monitor for transactions processing control of nomadic teams has been designed and the implementation proofs are ongoing.

Acknowledgements

First author is supported with PhD grants of the National Science and Technology Council, Mexican Institute of Petroleum and Technical Institute of Zacatepec, Mor., all from México. Reported research is funded by the D.00006 Intelligent Distributed Computing IMP project.

References