

A Framework for Two Phase Reconciliation in Mobile Databases

Md. Abdur Rahman
***Souhail Abdala &
Abdulmotaleb El Saddik***

August, 2005
University of Ottawa

Table of Contents

- Introduction
- Problem Statement
- Our Contribution
- Assumption for the Protocol
- Working Principle
- Two Phase Commit Protocol
- Two Phase Reconciliation Algorithms
- Conclusion

Introduction

- *Mobile Database* – portion/replica of a database remain in mobile devices like PDA, Laptop etc.
- *Mobile devices* – remain disconnected or weakly connected most of the time
- *Before disconnection* – can download a fragment/subset of database
- *During disconnected* – can perform local transactions/updates offline

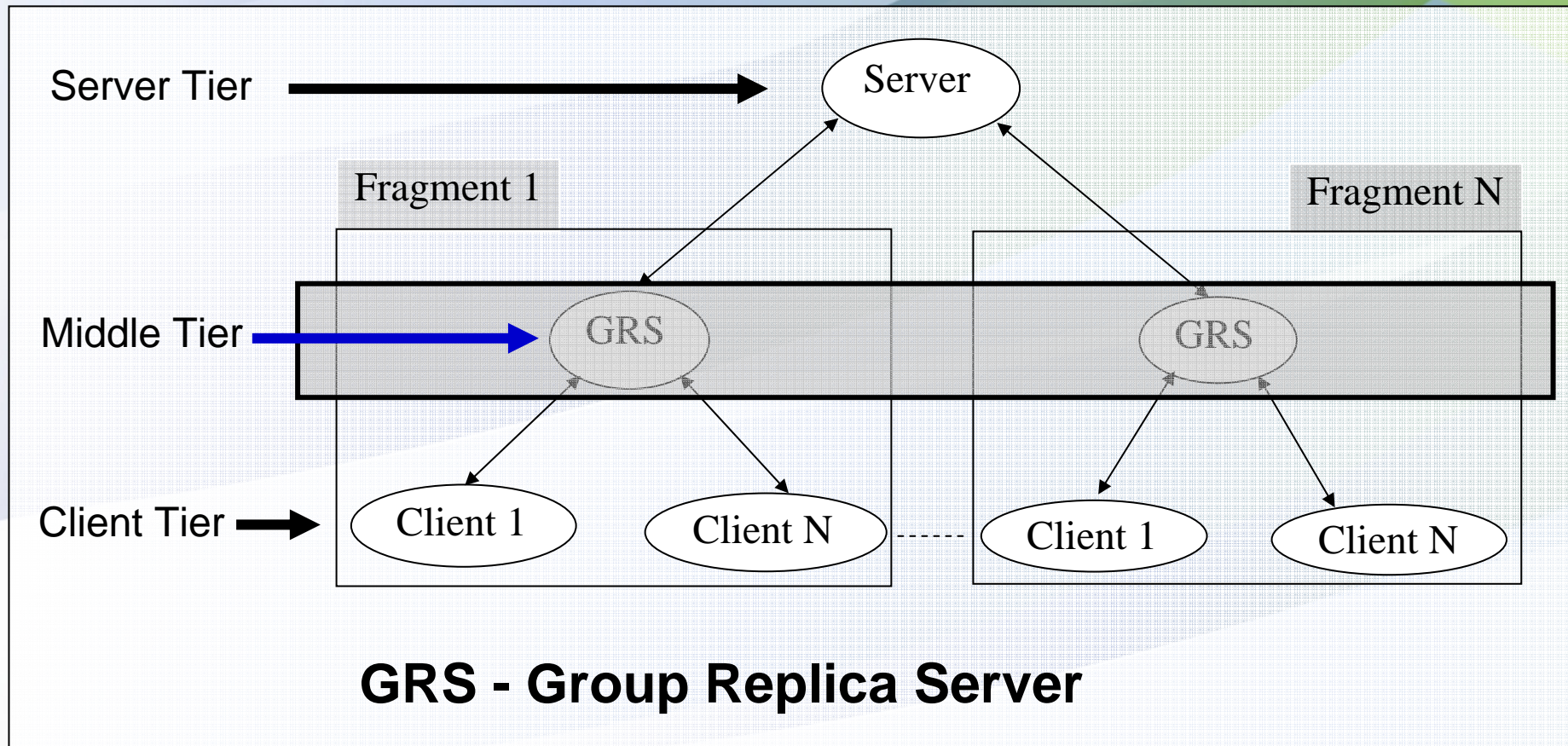
Introduction (Contd.)

- *At reconnection* – needs to globally commit offline transactions in the server DB.
- *At reconnection* – reconciliation issue
- *Reconciliation* – serializing potentially conflicting updates by disconnected clients at server DB

Problem Statement

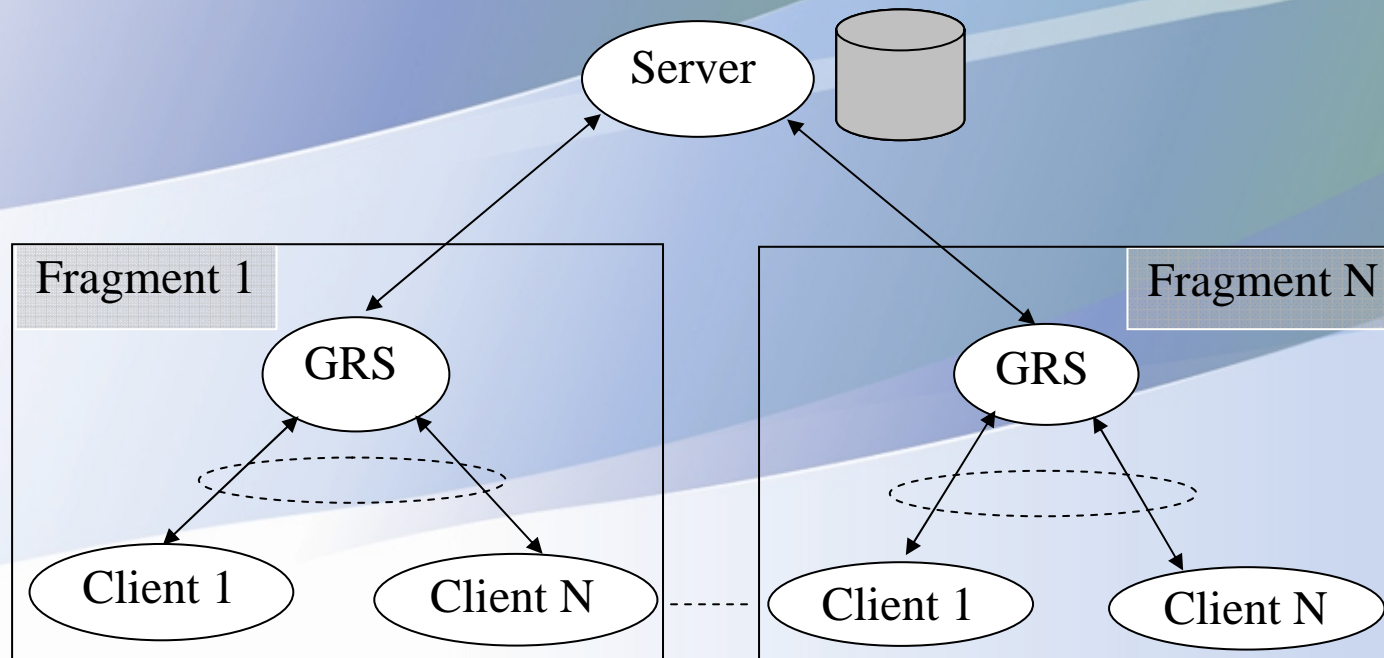
- *In most mobile DB architecture* – all mobile clients reconcile directly to the server DB resulting
 - DB resulting
 - ❖ Bottleneck and traffic congestion around server
 - ❖ Scalability and security issue
 - ❖ Requirement of highly complex and powerful server with all functionalities.

Our Contribution: Three tier Architecture



Assumption

- Fragments of server DB are replicated to GRS's such that most client queries can be reconciled in local GRS – single phase reconciliation
- Intelligent GRS – can download personalized data from Server.



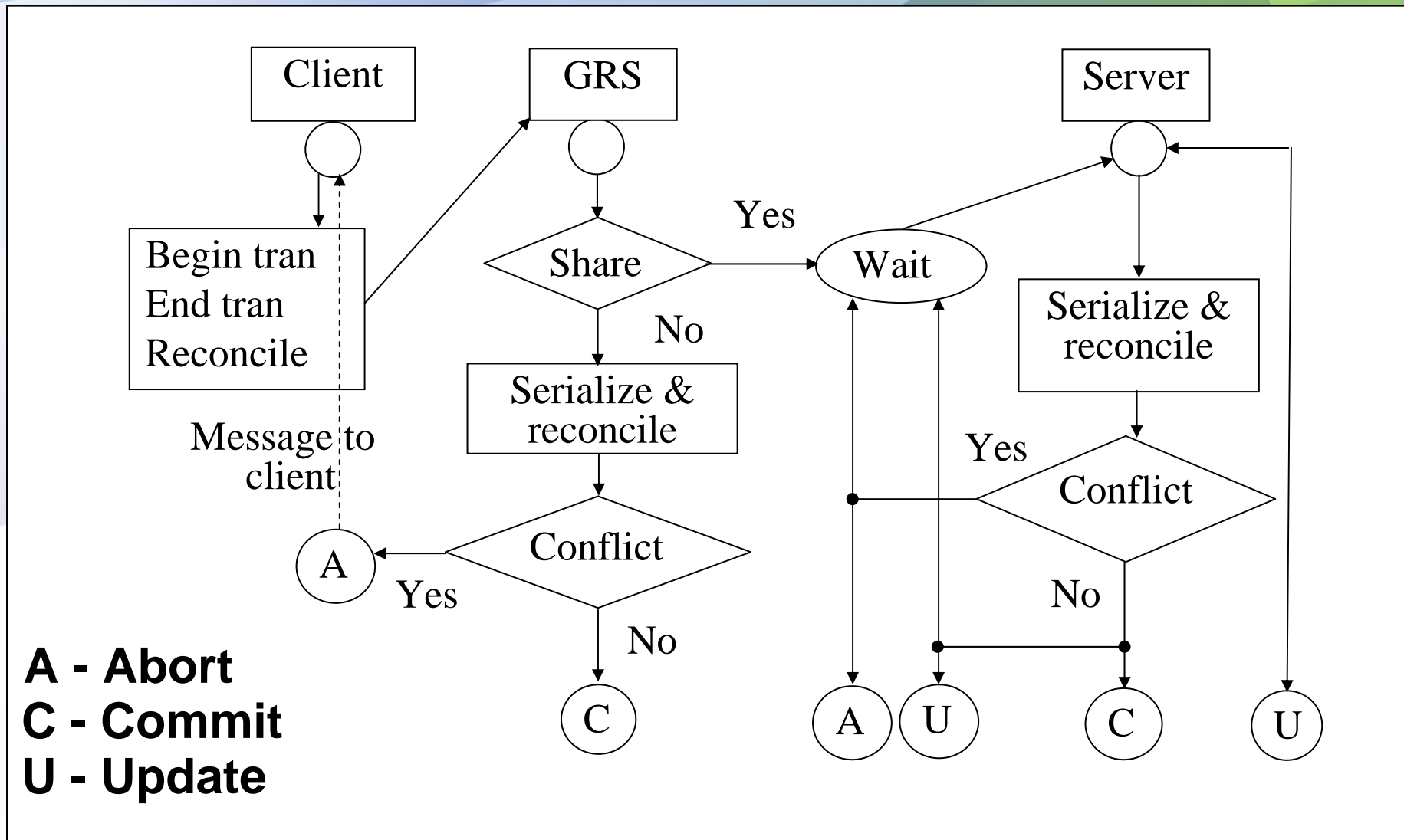
Working Principle

- At reconnection a mobile user tries to commit the offline transaction to its connected GRS
- Data accessed in the submitted queries might be shared or local.
- Local – GRS runs reconciliation algorithm [1], if successful, commits in the replica DB and updates the server DB asynchronously. Only one reconciliation process runs.

Working Principle (Contd.)

- Shared – Other GRS's also hold some chunk of the same DB object (e.g. table)
 - Needs integrity checking in the server at the cost of 2nd phase reconciliation.
 - First phase reconciliation [1] in the GRS for data integrity checking in the local GRS
 - if successful, server runs 2nd phase reconciliation [1] to check data integrity with other GRS's.
 - If successful, server commits the transactions and notifies the GRS, which in turn notifies the user.

Two Phase Commit Protocol



Two Phase Reconciliation Algorithm

Algorithm:
First Phase
Reconciliation
in GRS

```
If message from client="reconcile" Then  
  If accessed data="non-shared" Then  
    Call Reconciliation algorithm  
    If Reconciliation successful Then  
      Update main server asynchronously  
    Else  
      Send Abort notification to client  
    End If  
  Else  
    Send Reconcile message to main Server  
    Wait (message from server)  
    If Reconciliation is successful Then  
      Update GRS Database state  
      Send Successful message to client  
    Else  
      Send Abort message to client  
    End If  
  End If  
End If
```

Two Phase Reconciliation Algorithm (Contd.)

```
If message from GRS = "successful reconciliation" Then
    Update server database state
    Send Successful message to GRS
Else
    Send Abort notification to GRS
End If
End If
```

***Algorithm: GRS Updates the
main server database***

Two Phase Reconciliation Algorithm (Contd.)

```
If data = "shared" Then
  If message from GRS ="reconcile" Then
    Send the data item to GRS's holding the data items
    If that/those GRS can reconcile Then
      Update server database state
      Send Successful message to originator GRS
    Else
      Send Abort notification to originator GRS
    End If
  End If
End If
End If
```

Algorithm:

2nd Phase Reconciliation in Main Server DB

Conclusion

- We introduced three tier mobile database architecture with a middle tier called GRS.
- This middle tier aims to minimize the shortcomings of typical two tier architecture.
- Fragmentation of server DB to GRSs plays the most significant role of the architecture.
- If data to be committed is local, only one time reconciliation is necessary.
- For shared data, two phase reconciliation is necessary, 1st phase in local GRS and 2nd phase in server.
- We proposed the necessary protocol along with the algorithms for two phase reconciliation.

References

- [1] S. H. Phatak and B. R. Badrinath, "Multiversion reconciliation for mobile databases" in *Proc. 15th International Conference on Data Engineering*, Mar. 1999, pp. 582–589.
- [2] M. T. Ozsu and P. Valduriez, "Principles of Distributed Database System". Prentice Hall Inc., 1991.
- [3] C. Gollmick, "Replication in Mobile Database Environments: A Client-Oriented Approach," in *Proc. 6th International Workshop on Mobility in Databases and Distributed Systems (MDDS 2003) at DEXA 2003*, Prague, September 2003.
- [4] N. Prabhu, V. Kumar, I. Ray and G.C. Yang, "Concurrency Control in Mobile Database Systems," in *Proc. 18th International Conf. on Advanced Information Networking and Applications (AINA'04)*, Fukuoka, Japan, March 2004.
- [5] L. YanSheng, S. XiongKai, "Improve Performance of Disconnected Operation through Submitting by Probability and Transferring Transactions in Groups," in *Proc. 2003 International Conference on Computer Networks and Mobile Computing (ICCNMC'03)*, Shanghai, China, October 20 - 23, 2003.

Thank You

