A Study of Application Recovery in Mobile Environment Using Log Management Scheme

A.Ashok, Harikrishnan.N, Thangavelu.V,

ashokannadurai@gmail.com, hariever4it@gmail.com,thangavelc@gmail.com,
Bit Campus, Anna University- Trichy

Abstract

The geographical mobility of the mobile devices makes the application recovery complex which needs to store application log for the recovery of the application.This paper focuses the application log management using the mobile agents.

Index Terms: *MDS, mobile agents, PCS, coordinators, log unification, application recovery*

1. INTRODUCTION

Wireless communication through PCS (Personal Communication Systems) or GSM (Global System for mobile Communications) has become the essentials of day-to-day life. Today all telecommunication companies improve the communication quality, availability, reliability, security to add the data management capabilities in the MDS (Mobile Database Systems).MDS is a database system which reach the data location and perform the processing and deliver the results since each mobile units will have high work load so these MDS will be a useful resource to perform transactions.

To perform the system-level functions MDS may require different transaction management techniques (concurrency control, database recovery, query processing), different caching schemes etc.

Application recovery includes number activities like transaction arrival, fragmentation and distribution of transaction to nodes, transaction execution, and updation of results from clients to server. The recovery recreates the module to the prior failure execution state of the application.

Application recovery is complex than the database recovery due to the following reasons:

1. Presence of multiple application states
2. Absence of last consistent state.
3. Large no of application required to manage database processing.

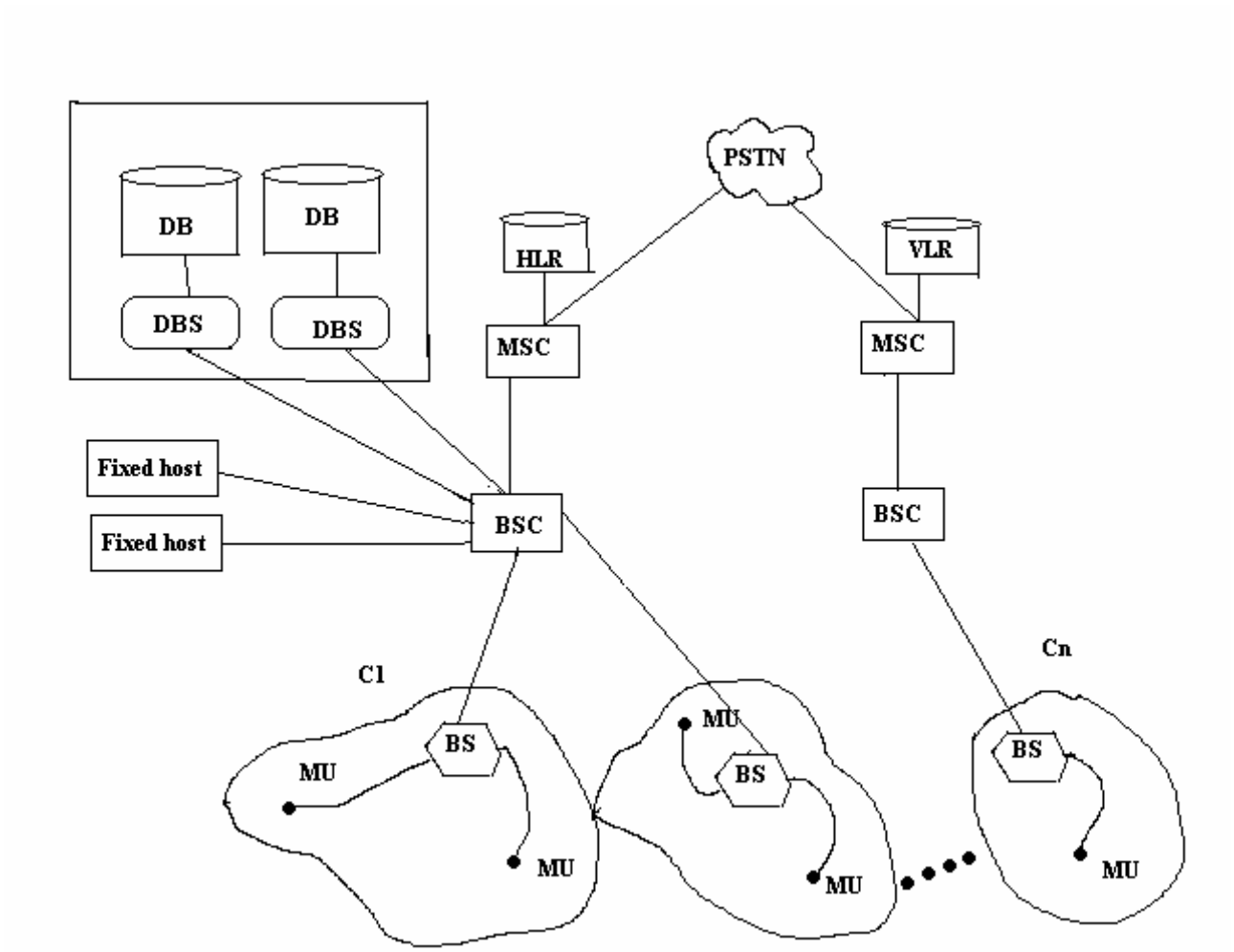
This becomes more complex in MDS due to

1. Processing demands of mobile units.
2. Existence of random handoffs.
3. Presence of operations connected, disconnected, intermittent connected modes.

In this logging scheme we store, retrieve, unify fragments of application log for recovery with in the constraints of the MDS.

2. REFERENCE ARCHITECTURE OF MOBILE DATABASE SYSTEM AND TRANSACTION EXECUTION

Fig. 1 shows the reference architecture of MDS which is based on PCS or GSM.



A set of general purpose computers are categorized as fixed hosts and BS (Base Station).one or more BS are connected to the BSC (Base Station Controller).BSC coordinates operations commanded by the MSC (Mobile Switching Center).

The size of the cell depends on the power of the BS and also the bandwidth of wireless communication channels.. A DBS provides full database services it communicates with MU through the BS and it is installed in either BS or FH.A MU cannot provide a reliable storage so for recovery it relies on either BS or FH. But to support these activities BS has to be entirely modified so we create DBS as separate nodes in wired network that could be reached by any BS at any time.

The mobile transaction model is referred to as "*mobilaction*". Mobil action is defined as $Ti = \{e1, e2.., en\}$ where a transaction T_i is requested at MU is called as H-MU (home MU) and is fragmented and are execute at MU. In a distributed environment a coordinator (CO) is used to commit, abort transaction. When the H-MU suffers handoff means the CO also changes this increases the time to commit. So the CO module can be housed either in BS, MU, DBS, and MSC. But selection of MU is not a good choice due to the following limitations:

1. Limited storage.
2. Limited power supply.
3. Unpredictable handoffs.
4. Limited availability.

3. RECOVERY PROBLEM SPECIFICATION

MDS recovery is complex due to the following reasons:

1. **MU's stability:** The MU may encounter some of the following failures limited battery power, run out of disk space, user physically drop the MU. So any of these events may affect the MU functionality and recovery algorithm must take all these in to consideration.
2. **Limited wireless bandwidth:** During the recovery MU have to communicate with the BS or with other MU so in that time if there are no free channels it affects the communication quality.
3. **Random Handoff:** This affects the recovery since the location of the MU is not available immediately for the communication.

3.1 Application Log Management

An efficient recovery scheme should consume minimized resources and recreate the execution environment as soon as possible after MU reboots. Messages that change the contents of the log are called write events. The H-MU records the events like

1. The arrival of Ti
2. Fragmentation of Ti
3. Assignment of CO to a T_i
4. Mobility history of H-MU.
5. Dispatch of the updates to the DBS.

The objective of using mobile agents with the MDS is to

1. Reduce communication overhead.
2. Recovery time should be minimal.
3. Easy implementation of recovery schemes.

4. A MOBILE AGENT-BASED LOG MANAGEMENT SCHEME

A Mobile agent is an autonomous program that suspend the execution at any point and resume execution from where is stopped in the new machine. An agent carries both code and the application state. Some of the advantages of the mobile agents are:

1. **Protocol Encapsulation:** Mobile agents contain own protocols in their code itself.
2. **Robustness and Fault Tolerance:** When failures are detected host system can easily dispatch the agents to the other hosts for the recovery.
3. **Asynchronous and Autonomous Execution:** Once agents are dispatched they can make the decisions independently.

These agents do have disadvantages where high migration leads to machine overhead. So this must be minimized.

In this architecture we have the following agents and their logical functions:

- **Bootstrap Agent (BsAg):** This agent addresses a BS failure. When a BS fails it registers in bootstrap agent. Once loaded this agent starts all the agents that have registered with it. When the agents are started they read the log information created and act accordingly.
- **Base Agent (BaAg):** This agent decides which logging scheme to be used for the current environment. The BA creates an instance of an agent to handle recovery of mobilactions for each MU.
- **Home Agent (HoAg):** This agent handles Mobilactions for each H-MU. It is responsible for maintaining log and recovery of H-MU.
- **Coordinator Agent (CoAg):** This is a coordinator residing in BS.
- **Event Agent (EvAg):** There are various events taking place like registration of a MU, failure of MU, handoff of MU etc. When any MU suffers from handoff it can be known by the HoAg through this EvAg only to take necessary action.
- **Driver Agent (DrAg):** During the migration of the mobile agents we need to transfer the code and actual data this increases the overhead to manage this we use driver agent (DrAg).

4.1 Action of Agents When Handoff Occurs

The HoAg moves along with the MU to the new BS instead of sending all its code we send driver agent when handoff occurs. The DrAg checks whether the code for the HoAg is present in new BS or not .If code is present it requests the BaAg in new BS to create an instance of HoAg for this MU. If it is not present DrAg sends request to previous BS's BaAg to take clone of HoAg and send the copy to the new BS to continue its operation.

When MU moves out the BS its information are not deleted automatically until it is notified by the agents of the MU.

5 FORWARD STRATEGY

The time duration between the MU failing and rebooting is called as Expected Failure Time (EFT). When a MU suffers from a handoff the communication with the last BS is disconnected only after the connection with the new BS so this allows the MDS to detect the failure of the MU. MU is monitored by the BS and any change in the situation of the MU is registered in EvAg this makes the EFT value appropriate. When the new BS does not know the location of old Bs it is obtained from the HLR. If the HLR is far away it is obtained from the VLR if both BS fall same under the control of VLR.If different VLR then HLR should be queried. If a MU crashes then the recovery of different BS needs the new BS to wait until the log unification in the old BS finishes.

In this we provide two schemes to reduce the recovery time by unifying the log information periodically when the number handoffs occurred crosses a predefined handoff threshold .when a handoff occurs the trace is transferred from old BS to new BS. The trace contains BS_ID of the new BS and log size. The log size updated when MU presents information. The EFT value is stored in a variable that is accessible by HoAg or if it is not supported then when MU fails is intimated by the agent framework through the event agent interface which starts EFT clock and stopped to get recovery time. When a MU log is unified at a BS a *garbage collection* message is sent to the entire BS's hosting the MU logs in the BS_ID list.

5.1 Forward Log Unification Scheme

When a MU fails the trace information contains the log size stored in different BS's. The unification time of the log depends on the network link speed and the log size. This is called as Estimated Log Unification Time (ELUT)

$$Max\{BS_i\text{-}LogSize/network\ link\ \ speed+propagation\ Delay\}$$

It also depends on the factors like whether BS is located in the same or different VLR. If the MU reboots in a different BS while the log is unified in the previous BS then it has to wait for unification to complete.

5.2 Forward Notification Scheme

This calculates the time spent in getting BS information from the HLR. Each VLR stores MU's status information (normal, failed, forwarded).when a MU fails the BS informs the VLR.VLR changes the status of the MU in database from normal to failed. The VLR sends the information to all adjacent VLR they store the message until they receive the denotify messages. There are three situations where the MU may recover in different locations and the VLR updates the contents of it by the following way.

Case 1: The MU reboots in the same BS where it failed.

VLR issues denotify message to all VLRs.

Case 2: The MU reboots in a different BS but in same VLR.

MU register in Bs and notified to VLR.

Case 3: The MU reboots in a different BS and a Different VLR.

The MU requests for the registration then the VLR returns the identity of the previous BS and the identity of the VLR to the HoAg of the MU in recovered BS. The BS perform log unification from the previous BS and new VLR sends recovery message to previous VLR and registration

message of the MU in the new location to the old HLR. After receiving recovered message VLR sends denotify message to all adjacent VLRs except the MU recovered and removes its registration.

6 CONCLUSION

In this paper we present the application recovery using mobile agents. Forward strategy reduces the recovery time with consistent behavior in all parameters.

REFERENCES

[1] A.Acharya and B.R. Badrinath,"Checkpointing Distributed Applications on mobile computers."*Proc. Third Int'l conf. parallel and Distributed Information Systems*, pp. 73-80,1994.

[2] J.Kiniry and D.Zimmerman,"A Hands-on Look at Java Mobile Agents,"*IEEE Internet computing*,vol. 1,no. 4,pp. 21-30,1997.

[3] D.B.Lange and M.Oshima,"Seben Good reasons for Mobile Agents,"*comm.. ACM*,vol. 42,no. 3,1999.

[4] C.Perkins,"Mobile Networking through Mobile IP,"*IEEE Internet Computing*,pp. 58-69,Jan.1998.

[5] P. Krishna,N.H.Vaidya,and D.K.Pradhan,"Recovery in Distributed Mobile Environments,"*proc.IEEE workshop Advances in parallel and Distributed Systems*,Oct.1993.

Article received: **2009-01-07**