Volume 2, Issue 8, August 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Q-Cache: A Data Management Technique for Mobile Location Based Services

Gobi.R*
Research Scholar
Dept. of Computer Science
Bharathidasan University
Tamil Nadu, INDIA

E.Kirubakaran
Additional General Manager
Outsourcing Dept
Bharat Heavy Electricals Ltd
Tamil Nadu, INDIA

E.George Dharma Prakash Raj
Assistant Professor
Dept. of Computer Science
Bharathidasan University
Tamil Nadu, INDIA

Abstract*— Recent research in mobile computing has proved that Location Based Services (LBS) are the upcoming thriving factors in the mobile environment besides voice and data services. The scope of LBS and mobile data management grows at a tremendous rate. This growth has resulted in some setbacks such has response delay etc. which affect the end to end experience. In order to overcome these setbacks, the potential of LBS and the challenges faced in data management should be addressed with a much more efficient mechanism. The main objective of this paper is to access location based service in an effective way using server side query caching. Accessing data from cache results in efficient retrieval, minimized communication cost, and reduced response time compared with traditional information access mechanism which is being followed at present in the mobile environment.

***Keywords*— LBS, Query Cache, Data Management, Mobile Device, Method**

I. INTRODUCTION

With the proliferation of mobile computing technologies, location based services have been identified as one of the most promising target application. Location based services can be defined as services that integrate a mobile device's location or position with other information so as to provide added value to a user. In the present world mobile customers require the services and information at their door step irrespective of their location. LBS data are highly dynamic and real time so there traditional information access technique may not well suited. So there should be data management models to handle the issues and to provide the potential to end user. Based on this, our existing work was proposed a communication framework for LBS. And our present work for this paper to utilize query caching techniques to address the data management issues even in the dynamic situations. The proposed work for this paper is a cache management model for enhancing the availability of the location based information's with respect to server side caching. Based on this model one can get access from cache if the query is available with our model in a reduced response time compared with traditional information access time. The rest of the paper is organized as follows. Section 2 provides a survey of related issues and research work. Section 3 presents our cache based model for data management. In Section 4, implementation of our proposed work in server and client environment. Section 5 deals facilitate the analysis and discussion with respect to our proposed system. Section 6 concludes the paper.

II. RELATED WORK

Location based services have the potential for becoming more pervasive and resulting in significant revenue for service providers, wireless carriers, and applications developers and integrators. [1]

A. Location Management

Location management deals with how to keep track of an active mobile station within the cellular network. The paging operation is performed by the cellular network. The location update operation is performed by an active mobile station. A location update scheme can be classified as either global or local. This scheme is global if all subscribers update their locations at the same set of cells, and a scheme is local if an individual subscriber is allowed to decide when and where to perform the location update.

Mobile computing is a generic term describing the application of small, portable, and wireless computing and communication devices. This includes devices like Laptops with Wireless Local Area Networks technology, mobile phones, and PDAs with Bluetooth or Infrared Data Association (IrDA) interfaces. The mobile computing focuses on the requirement of providing access to information, communications and services everywhere, anytime and by any available means [Litke 2004]. The technical solutions for achieving this are not always easy to implement. In fact, mobile computing requires the creation of communication infrastructures and the

modification of computer networks, operating systems, and application programs [Gai 1998]. The mobility issue implies some constraints that must be addressed since they limit the capability of a moving resource in contrast to a fixed one. [Chu 2004] discussed the various challenges in a wireless environment.

B. Distributed Mobile System

Middleware development in the growing and promising field of distributed mobile systems. With the advent of technology wireless as well as mobile computing systems are widely used now. A wireless/mobile computing system is a distributed system consisting of both Mobile Device (MD) and static Mobile Support Stations nodes [Neogy 2007]. The MSSs are connected with fixed static network and MDs are connected with wireless communication link [Saha 2007]. An MSS may communicate with a number of MDs but an MD at a time communicates with only one MSS. An MD communicates with the rest of the system via the MSS it is connected to. The links in the static network may support First in First out (FIFO) message communication. To manage a large number of MDs in a network, multiple MSSs is used, one may not be assigned to the network like in Figure 2.4 [Ahn 2007].

C. Cache Management Model Existing approach

In the distributed mobile system a mobile device acts as client device as well as a data source for providing relevant information needed for business process. When one Mobile Device(say MD_Source) needs to communicate with other Mobile Device(say MD_Dest) for access the business information conventionally there would be direct communication established between the two MD through MSS. The following procedure describes the flow of data between the destination device and source device.

1. The MD_Source sends a query message to its local Mobile Support Stations (say MSS1).
2. MSS1 receives the query message and sends a request to the location server for the current location of MD_Dest.
3. Location server looks up the reference table for the entry of MD_Dest and returns the current location information (Assume, MD_Dest currently resides in MSS2) to MSS1 in the form of a message.
4. The query message is transferred from MSS1 to MSS2 if the location information is valid otherwise the error message is forwarded to MD_Source.
5. MSS2 accept the query message and check if MD_Dest is a registered Mobile Host. If it is a registered Mobile Devicethen the message is forwarded to MD_Dest otherwise an error message is returned to MSS1.
6. MD_Dest receives a query message from MSS2 and returns the requested business data in the form of reply message to MSS2.
7. MSS2, receives the reply message and forward it to MSS1.
8. MSS1, receives the message from MSS2 and forward it to MD_Source.

In the above algorithm the destination MSS (MSS2) happens to be the same as the source MSS (MSS1), therefore steps 2, 3, 4 and 7 are not required. As described in the algorithm above, eight packet transmissions over the network are required for the retrieval of a query result, four of those transmissions over the wired portion of the network and another four are over the wireless portion. This represents the average case for the number of packet transfers for this application [Janakiram 2005]. Normally the wireless communication is being unstable; the availability of mobile devices will be much felt during the business transaction. To increase the response time,performance,reducing cost and content availability the cache management is introduced.

III. PROPOSED MODEL

The cache management model acts as a representative of the MD. This bridges the MD and its support environment. The wireless network is highly unstable. This causes the Mobile Deviceto frequently connect and disconnect from the mobile network. The data will be available in cache even the traditional server is not available.

The major advantages of using the cache management architecture are:

1. It provides solution to the instability of mobile network in a distributed mobile system. The cache management model is a customizable approach, meaning that host specific and application specific constraints can be enforced like emergency applications.[Anastasi 2001].results displayed in the terminal with the recent logged information.

2. Cache management maintains the current location of MD there by solving the problem of identifying location of MD. It acts as a server for handling data dissemination to provide mobile data access, in both server-push and client-pull models.

3.The cache management can also cache Mobile Device specific data and reduce the response times for many client queries. It also supports disconnected operations of the MD by buffering client requests or using the cached data to handle them [Janakiram 2005].

4.It provides optimal utilization of wireless bandwidth, as the cache management knows the current network connectivity and other constraints of its corresponding host.

A. The Structure of Cache management Model

Cache Management is the server of the MD. Therefore the data structure of the cache management should be relevant to the MD. The service provider can also provide support services to various m-business applications by using the cache management. Hence the cache management is also designed to contain important valuable m-business data. The structure of a distributed mobile system with cache management is as shown in Figure

The Cache management structure also contains methods to process the MD and m-Business data. In the distributed environment each Mobile Deviceis has a cache management. Cache management Identifier is unique identifier assigned for each object. Methods are available to send and receive messages. A new cache management for a particular MD is created using getInstance() method. It contains methods to set and get the status of message.

IV. IMPLEMENTATION OF PROTOTYPE

The proposed architecture is designed based on two ways.The client side designed using Sun Java J2ME wireless Toolkit 2.5 simulator.J ava RMI is used for simulating distributed environment for location based web application done by Tomcat 5.0 and Oracle Database is used.

4.1 Technologies used for the Cache management Model

Several technologies are used by the cache management model. They include the RMI, web component and J2ME.

4.1.1.Remote Method Invocation

Java Remote Method Invocation is a specification for building distributed object-oriented applications. RMI is designed primarily for use in conventional wired computing environments. Java RMI is a specification from Sun Microsystems that allows Java objects to invoke methods on objects in other address spaces while reserving the semantics of local method invocations. RMI uses sockets as the communication mechanism between the two Java Virtual Machine (JVMs) but abstracts the communication interface to the level of a method invocation, hiding the complexities of socket protocols from the programmer [Wall 2001]. This results in a much simpler and more intuitive means of building complex client-server systems.

Registration of New Mobile Device and Creation of Cache management

The new MD enters in the distributed mobile system; it registers itself in the MSS. In the registration process is shown in Figure 4.5.

.

The user enters his/her personal information and mobile number in the J2ME midlet applications. This information is sent to the registration server. The registration server is a servlet program which stores the information in the Oracle database then it invokes a remote method createMDID() for generating MD Identifier. The method createMDID() accept the mobile number and assign as MD Identifier and instantiate a cache management using getInstance() method and assign a unique Cache management Identifier. The various activities of MSS like creation of cache management, communicating with location server, message transfering are implemented using RMIServices.

Communication between the Mobile Hosts's within the Same Mobile Support Stations

The client side application is implemented using J2ME midlet applications. The MSS functions are implemented using RMI services. The client request and response are handled by Java servlet. The MD_Source wants to communicate with MD_Dest, the query message and the destination mobile number is forwarded as request. The message communication servlet accept the request and check the user validity. The Figure 4.7 shows the implementation of midlet to servlet communication.

The MD_Source and MD_Dest are valid users then the servlet uses the naming services of the middleware and gets the cache management reference of the corresponding MD_Dest using getInstance() method. If the cache management contains the requested information then the information is transferred to MD_Source as response otherwise the cache management of MD_Dest forward a query to MD_Dest. The

MD_Dest process the query and update a requested information to its cache management. The cache management is responsible for transferring the requested information to MD_Source..

TABLE I

Implementation Code

Implementation of RMI Procedure

```
import java.io.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;
import java.rmi.*;
import java.rmi.registry.*;
import java.net.*;
public class servletregister extends HttpServlet
{  String mobhostid=null;
  public void registerService(String mobhostid,String
currentlocation)
  {  try{
     String name=getInitParameter("registryName");
     String host=getInitParameter("registryHost");
      if(host==null)
        host="localhost";
        Registry registry=LocateRegistry.getRegistry
                (host,Registry.REGISTRY_PORT);

clientval=(clientoperation)Naming.lookup("rmi://"+host
+"/"+name);
        clientntinterface
clientval=(clientinterface)registry.lookup("mobhostid");
        String
id=clientval.register(mobhostid,currentlocation);
   }    catch(NotBoundException not)    {  }
  catch(RemoteException rem)   {  }
  }
try {
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection
c=DriverManager.getConnection("jdbc:odbc:serdb3",",s
ystem",",manager");
Statement st=c.createStatement();
String stmt="select * from MDid where
mobileid='"+sendto+"'";
ResultSet rs=st.executeQuery(stmt);
messageService(message,sendto,from);
out.println("You are in " GIS data);
}
```

Java ME Code for Client Side

```
Communicating with Servlet
String
url="http://localhost:8080/servlet/sendings
ervlet?message="+text+"&
sendto="+text1+"&from="+source;
String murl=encodeURL(url);
HttpConnection connection =
(HttpConnection)
Connector.open(url,Connector.READ_W
RITE, true);
connection.setRequestMethod(HttpConnec
tion.GET);
connection.setRequestProperty("Content-
Type","//text/plain");
```

```
public class Sendingservlet extends
HttpServlet
{
String MDid=null;
public  static cachemgmt do;
public void messageService(String
message,String sendto,String from)
{{
String
host=getInitParameter("registryHost");
if(host==null)
host="localhost";
Registry
registry=LocateRegistry.getRegistry(host,
Registry.REGISTRY_PORT);
clntinterface
cli=(clntinterface)registry.lookup("MDid")
;
so=cli.send(message,sendto,from);
System.out.println("Send servlet invoked
successfully");
System.out.println("RMI-Servlet");
System.out.println(so.message);
byte[] b;
InetAddress
a1=InetAddress.getByName("localhost");
DatagramSocket ds=new
DatagramSocket();
b=message.getBytes();
DatagramPacket dp=new
DatagramPacket(b,b.length,a1,4000);
ds.send(dp);
do.setmsg(mess);
do.setmsgstatus("yes");
}
```


4.6.4 Implementation of Location Server and Communication between Mobile Hosts's in Different Mobile Support Stations

The location server activities are implemented using RMI services. When the MD_Source sends a message destined to MD_Dest, it will be forwarded to local MSS (say MSS1). When MSS1 receives the message and check if the cache management for MD_Dest is currently hosted. If not found, it sends a message to the location server for requesting the object reference of the cache management of MD_Dest. The location server looks up the reference table entry for the location of cache management of MD_Dest and returns the location information in the form of object reference to MSS1.

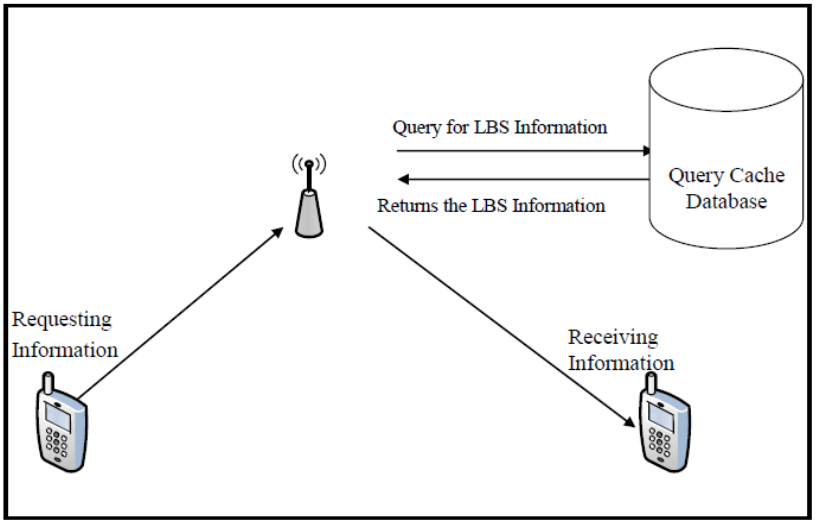


Figure 4.10 Implementation of Location Server

Assume that the MD_Dest's cache management currently resides in MSS2. An error in lookup results in an error message being returned. If the reply contains the location information, the message is sent to MSS2. The MSS2, upon receiving the query message checks if the cache management for MD_Dest is currently hosted. If yes, then the message is forwarded to the cache management. When the cache management for MD_Dest receives the message then it forwards the message to MD_Dest.

Implementation of Object movements with Response time

Object migration is the movement of objects from one application to another during execution. The serialization mechanism is used to migrate an object across a network connection. When the instance of a class is serialized, the only things that are saved are the non-static and non-transient instance data members. The class definitions are not saved. They must be available when the object is deserialized. The Java language supports a very general mechanism called object serialization that makes it possible to write any objects to a stream and read it again later. This means that any class that implements the Serializable interface can be migrated from one location to another location. There are no methods in the Serializable interface; the interface just serves as a marker to say that a class can be serialized.

The ObjectOutputStream class uses to write objects to a stream and ObjectInputStream class uses to read them back. The basic process of serializing and deserializing an object is as follows (Table 4.11)

```
Public class cacheManagement  implements Remote,Serializable
{
// Data Members
// Methods
}
cacheManagement do = cacheManagement.getInstance();
ObjectOutputStream oos =  new ObjectOutputStream(anOutputStream );
Serializable s1=  oos.writeObject(do);
ObjectInputStream ois = new ObjectInputStream(anInputStream )
cacheManagement do1 = (cacheManagement)ois.readObject()
```

Figure 4.11 Procedure for Object Serialization and Deserialization

The other thing that is frequently explained about serialization is overriding the readObject and writeObject methods to change the default serialization behavior. By overriding these methods it can provide additional information to the output stream, to be read back in at deserialization time

In the proposed model need to migrate the cache managements whenever the MD is roaming. The entire cache management is serialized and sent to the other MSS. This causes a heavy network load and has an adverse effect on performance of the MSS. The serialization can be slow for larger objects [Horstmann 2008]. Hence the data filtration is applied. The data filtration reduces the amount of data to be serialized and transferred. The data filtration is referred as customization. The smaller version of cache management is created using createCuso() method then the object is migrated.

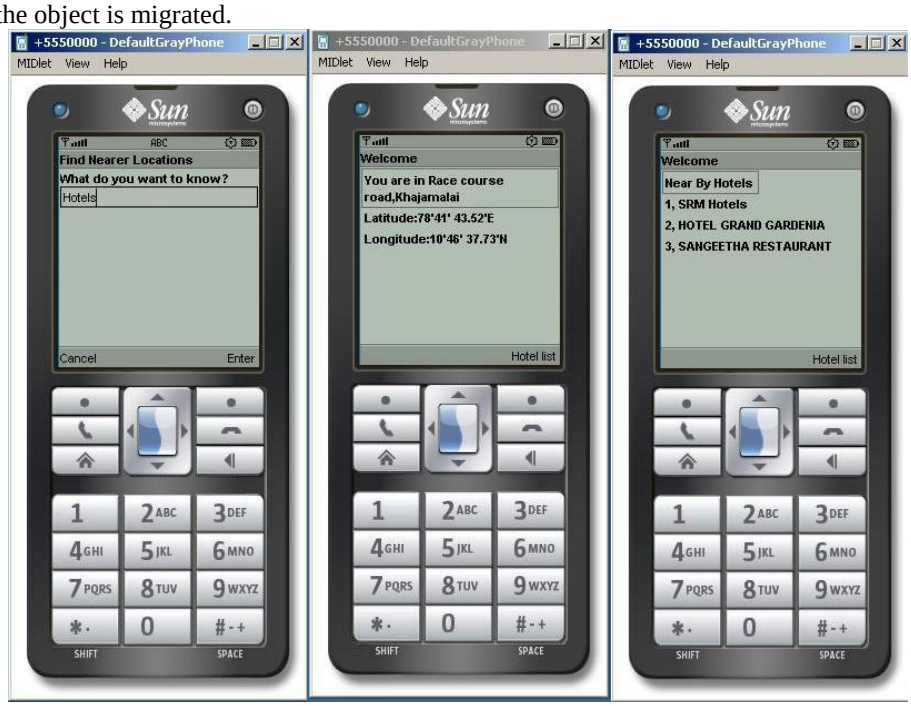


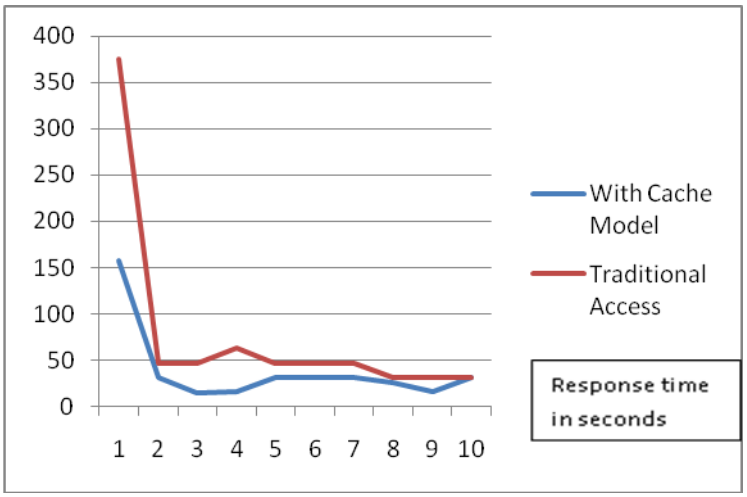
Fig. 1 Example of an image Transmission of Messages between MDs

V. ANALYSIS AND DISCUSSION

The major factor that analysis the performance is response time (The time taken for moving an object from one MSS to another MSS). Figure 4.14 illustrates the comparison of response time for the cache management model and customized cache management model. The migration starting time is plotted against the migrated time of the object. Figure 4.14 Response time Comparison for Cache management Model and Customized Cache management Model. The average response time is evaluated in terms of milli seconds which is shown in the table

Table 4.1 Average Response Time

Methods	Response Time (in seconds)
Data Access with Cache Management	39.3
Data Access without Cache Management	76.7



The study shows the response time is quicker for the customized cache management model than the cache management model. Hence, the mobile customers are able to get the required information in a shorter time.

VI. CONCLUSION

This paper discusses in detail about the distributed mobile architecture and its implementation using the customized cache management model. Performance analysis is carried out comparing the cache management migration and customized cache management migration. It reduces the data transferring time and increase the performance of handling the client requests. The customized cache management migration model elegantly solves a whole set of problems in distributed mobile systems: location management, mobile data access in client-server systems, etc. It also provides an ideal placeholder for host specific information, thus enabling application or host specific constraints to be enforced. This facilitates building customizable applications over distributed mobile systems.

REFERENCES

1.[Lima 2008] Lima, L. Jr. and Calsavara, A., 'A Paradigm Shift in the Design of Mobile Applications' in Proceedings of the 22nd International Conference on Advanced Information Networking and Applications – Workshops, pp. 1631–1635, 2008.

2.[Horstmann 2008] Horstmann, C.S. and Cornell, G., 'Core Java Volume II – Advanced Features', Eighth Edition, Prentice Hall, 2008.

3. Neogy 2007] Neogy, S., 'WTMR – A New Fault Tolerance Technique for Wireless and Mobile Computing Systems', in Proceedings of the 11th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS'07), pp. 130–137, 2007.

4. Saha 2007] Saha, S.B. and Neogy, S., 'A Low Overhead Checkpointing Scheme for Mobile Computing Systems', in 15th International Conference on Advanced Computing and Communications, pp. 700–705, 2007.

5. [Ahn 2007] Ahn, J.H., 'On Tolerating Failures of Mobile Hosts and Mobile Support Stations', International Journal of Computer Science and Network Security, Vol. 7, No. 6, pp. 99–106, 2007.

6.[Janakiram 2005] Janakiram, D., Mohamed, M.A.M., Vijay Srinivas, A. and Chakraborty, M., 'Surrogate Object Model: Paradigm for Distributed Mobile System', in ACM International Conference on Information Systems Technology and its Applications, pp. 124–138, 2005.

7.Litke 2004] Litke, A., Skoutas, D., Varvarigou, T., 'Mobile Grid Computing: Changes and Challenges of Resource Management in a Mobile Grid Environment', in Workshop of Access to Knowledge through Grid in a Mobile Word, 2004.

8.[Chu 2004] Chu, H., You, C.W. and Teng, C.M., 'Challenges: Wireless Web Services', IEEE International Conference on Parallel and Distributed Systems, pp. 657–664, 2004.

9. [Anastasi 2001] Anastasi, G., Bartoli, A. and Spadoni, F., 'A Reliable Multicast Protocol for Distributed Mobile Systems: Design and Evaluations', IEEE Transaction on Parallel and Distributed Systems, Vol. 12, No. 10, pp. 1009–1022, 2001.

10.[Wall 2001] Wall, T. and Cahill, V., 'Mobile RMI: Supporting Remote Access to Java Server Objects on Mobile Hosts', in Proceedings of 3rd International symposium on distributed objects and applications, pp. 41–51, 2001.

11.Gai 1998] Gai, S., 'Internetworking IPv6 with Cisco Routers', pp. 448, Mcgraw-hill Professional, 1998. See also http://www.ip6.com.

