2009 Fourth International Conference on Computer Sciences and Convergence Information Technology

Generic Information System Using SMS Gateway

Muhammad Saleem[1] , Kyung-Goo Doh[2]
Department of Computer Science and Engineering
Hanyang University
Ansan, South Korea
[1]engr_saleemwazir@yahoo.com, [2]doh@hanyang.ac.kr

Abstract— **In the last few years, SMS (Short Message Service) has made a big impact on the way we communicate. Instead of communicating over the phone using voice, people rather prefer SMS not only for messaging but also for information exchange. This paper proposes a method of building an extendable generic application which can be used to provide various types of information services using mobile SMS. Mobile users send required information through an SMS to a mobile gateway that forwards it to the generic application. Given the user-provided information, the generic application automatically generates an appropriate query.**

Keywords- SMS (Short Message Service); S M S Gateway; Generic Information; Multiple Databases

I. INTRODUCTION

SMS has shown significant resilience in market that is bombarded with media that all add to the clutter of daily communications. SMS is a form of highly personal, immediate communication with high reach capability, low cost and high retention levels. With communications media converging, SMS is now accessible in many ways as a business tool. Consumers have been first in adopting SMS as a means of communication, popularizing the protocol with a specific language and creating the playing fields that are now being entered by businesses [1]. There are a variety of business applications used in industries such as:

- Mobile commerce [8,10] and transactions
- Internal communications among staffs
- Field services and engineering [9]
- Billing information updates

Notable applications based on SMS technology are Public Transport Service [2], Sales Reporting [3], Mobile-Quiz [4], and Payments [5]. Each of these applications was separately developed and did not take the extensibility and reusability into consideration. Thus if system administrators want to add some new functionalities to an existing system, they have either to make change to the existing system or rebuild the entire system from scratch.

This paper proposes a method for making a generic application which dynamically communicates with databases and extracts information based on the contents of SMS. This application can be used to provide information to users through SMS. The system administrator can add additional functionalities by simply providing the information of new features without making any change in the source code. Section 3 describes a detailed method of how our new SMS messaging service can be added by the system administrator.

In order to get specific information, a user will send an SMS to mobile Gateway which will forward it to the desktop application for necessary query execution. After collecting required information from a specific database, a prompt reply will be forwarded to the user in reverse order.

Since it is difficult to memorize formats of SMS to get different information, user can send a query SMS asking for exact format. A prompt reply will occur showing the format of SMS.

This paper is further organized as follows; section II shows an overview of the system, section III describes our proposed design model for implementing this generic information system, section IV contains an algorithm for generating a dynamic query, section V includes some practical examples of information's systems for which this model can be used, and section VI contains the implementation requirements.

II. OVERVIEW OF THE SYSTEM

In this section, we introduce our generic SMS information system. The communication scenario of the system is divided into five steps, as shown in Figure 1.

1. The user sends an SMS message to the SMS Gateway application having a specific starting identifier (@ in our case). Format of the SMS is automatically generated when administrator adds some new messaging service to the system.
2. The SMS Gateway application receives the message from the user and forwards it to the server application along with the phone number.
3. Server application then processes the SMS and makes a dynamic link with concerned database for information retrieval. The server application then executes a dynamic query on the database and then generates a prompt reply containing that record which comes as a result of the required query execution.

978-0-7695-3896-9/09 $26.00 © 2009 IEEE
DOI 10.1109/ICCIT.2009.99

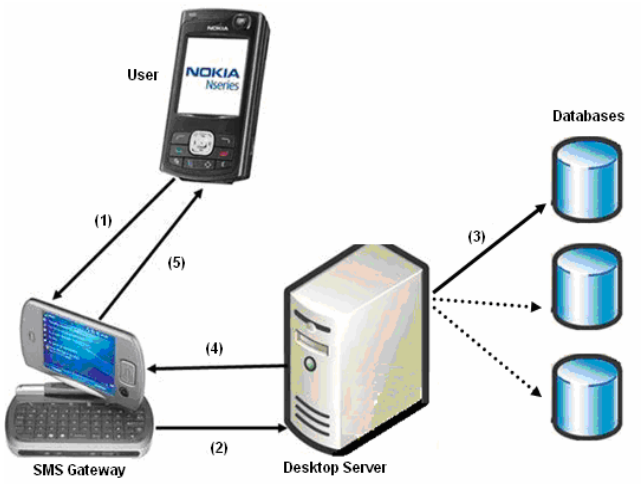

Figure 1. SMS Messaging Service

4. The result of the query execution is sent back to the SMS gateway along with the user number.
5. The SMS gateway then forwards the message to the user.

III. GENERIC INORMATION SYSTEM

Figure 2 shows our proposed design model for the required system which addresses two basic issues: first, how the system administrator extends the existing system by adding new SMS messaging services, and second, how the server application generates a dynamic query for required information's retrieval.

The automatic message format generating system (AMFGS) is used to add new SMS messaging services. For this the AMFGS takes specific database information, such as user-provided list, response list and link information. They will be provided by the system administrator.

The user-provided list contains those database table attributes that must be provided by the sender in the sending SMS. The response list contains those table attributes that are included in response of the SMS. The link information shows the database joins between those tables that are used during the query execution. The third input parameter of the link information between tables is either optional or mandatory. The system administrator provides this parameter only if the user-provided list or the response list contains table attributes selected from more than one table.

When the system administrator provides all input parameters for a specific SMS messaging service, the AMFGS generates the message format to be sent by the user. The general message format has a specific starting identifier "@" as follows:

@ <Service No.> <user-provided list>

Here "Service No." uniquely identifies each SMS messaging service provided by the generic information system. "Service No." starts from 1 and increases by 1 for every new service added by the administrator.

After generating message format, the AMFGS then stores the user-provided list, response list, link information's (if available), message format, database name (which stores all information of the new messaging service) , Service No. and tables used (name of database tables from where user-provided list or response list are selected) in a specific table called **"Hash Table"**, which contains the history of all applications provided by the generic information system. The information's stored in "**Hash Table"** are used during the query generating algorithm to generate the final query for required information's retrieval.

The user message has to observe the format made by the AMFGS. In order to know the format of the SMS, user can send a query SMS to the server application, asking for the

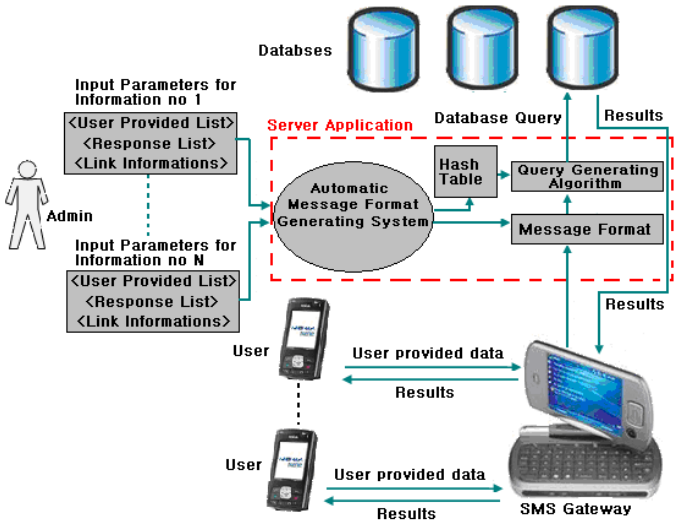

Figure 2. Our Proposed Model

format of the SMS. The server application then replies with an SMS containing the exact format. The system administrator can also provide formats of the SMS through a website or through some TV advertisement.

When the user message comes into the server application, the server application checks the message format first, and then applies the query generating algorithm on that message to generate final database query.

IV. QUERY GENERATING ALGORITHM

Once the SMS comes to the server application, the required results are extracted from a specific database by using the following algorithm.

Input: User SMS in the general format ("@ <Service No.> <user-provided list >"), "Hash Table" which contains the history of all messaging services provided by the system.

Output: An SQL SELECT query which executes against a specific database to extract required information, sent back to the user in reply.

1. Make sub-parts of received SMS by splitting the SMS using spaces. Store the subparts of SMS in an array named "*SubFields*" such that *SubFields(0)* contains "@" , *SubFields(1)* contains "Service No.", and so on.

2. Execute a query on the table named "Hash Table" for the Service No ("*SubFields(1)*") sent by user in SMS, to extract user-provided list, response list, link information, database name and used table. The query is like:
   Select user-provided list, response list, link Information, database name, used table from Hash Table where Service_No= SubFields(1)
3. Store user-provided list in an array "*UList*" , reponse list in array "*RList*", link information in a variable "*linkinfo*", database name in a variable "*Db_Name*" and table used in a variable "*usedTable*".
4. Declare a string variable "*Condition*" which stores string of final query after "*where*" key word. Initialize that variable to empty string.
5. *for i = 0 to UBound (UList)*
   if i = 0 then
   Condition = Condition & UList(i) & " =' " & SubFields(2+i) & " ' "
   Else
   Condition = Condition & " and " & UList(i)
   & " =' " & SubFields(2+i) & " ' "
   End If
   End for
6. Make a link with the database using the database name stored in variable "*Bb_Name*".
7. *If* link information is empty then execute the following query to extract the results for response
   Select RList from usedTables where Condition
   Else
   Select RList from usedTables where Condition and linkInfo
   End If

V. CASE STUDIES

Our proposed model can be used to implement different useful information providing systems as in [2, 3, 6, 7]. Some of them are described below.

A. Exam Results

Suppose that the administrator wants to add a new feature "Viewing Exam Results through SMS" in which a student provides student ID and exam name/semester and gets grades in response to his/her SMS. The administrator can use a database named "Exam" which contains the results of all the semesters. The ERD (Entity Relationship Diagram) of the database is shown in Figure 3.

According to the requirements of the system, the system administrator selects table attributes **Student.St_id** and **Grade.Exam_name** in user-provided list, and **Subject.Sub_name**, **Grade.Marks** and **Grade.Gpa** in response list. As all these attributes belong to more than one table, i.e., **Student**, **Grade** and **Subject**, so the administrator must provide the link information among these tables.

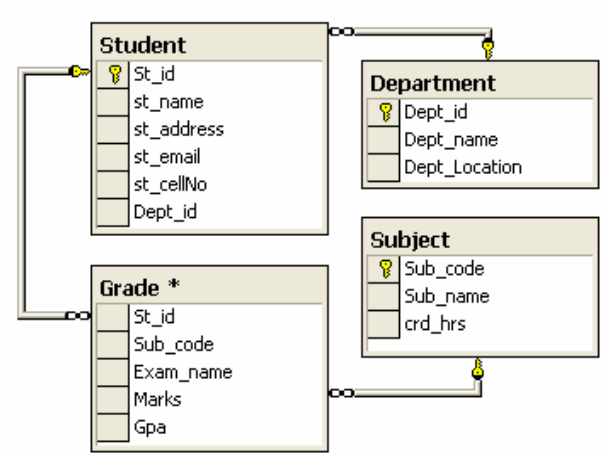

Figure 3. ERD of the **Exam** database

Figure 4 and 5 show the screen shots of the automatic message format generating system.

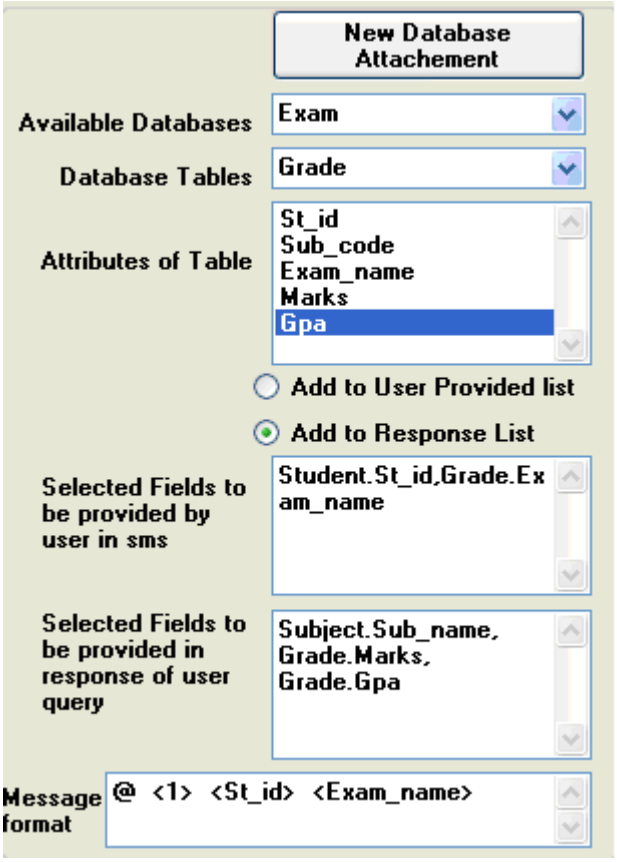

Figure 4. Screenshot of AMFGS for adding new information services

In Figure 4, the administrator selects two lists of attributes: one contains fields/attributes to be provided by the student in SMS and the other contains attributes to be provided in response to the user SMS/Query. After the selection of the user-provided list and response list, the third input parameter is link information among tables, which is shown in Figure 5

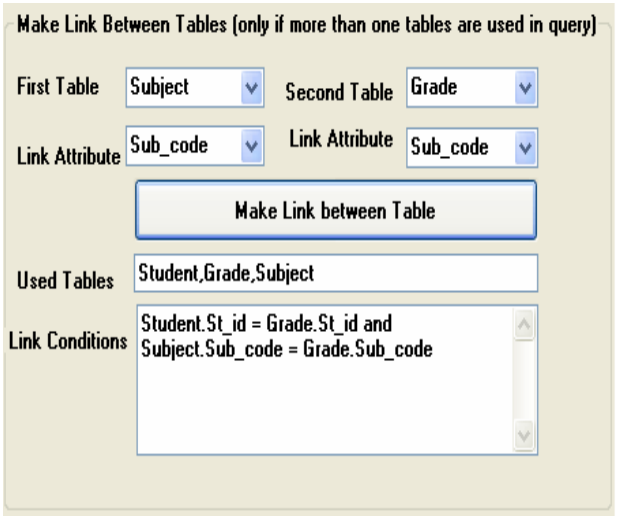

Figure 5. Screenshot of making links between tables (link information's)

After the administrator provides all these three input parameters, the automatic message format generating system produces the following output;

@ <1> <St_id> < Exam_name>.

This is the message format used by the student to view his/her results. Where <1> is the unique identifier for that SMS messaging service. The automatic message format generating system stores user provide list (Student.St_id, Grade.Exam_name), response list (Subject.Sub_name, Grade.Marks, Grade.Gpa), link information's (Student.St_id = Grade.St_id and Subject.Sub_code = Grade. Sub_code), message format (@ <1> <St_id> < Exam_name>), database name (Exam) and table used (Student, Grade, Subject) in "**Hash Table"**.

After sending SMS "**@ 1 200821 fall2008"** by the student having student id 200821 and exam name fall2008, the query generating algorithm produces the following SQL Query;

Select Subject.Sub_name, Grade.Marks, Grade.Gpa from Student , Grade , Subject where Student.St_id = '"& SubFields(2) &"' and Grade.Exam_name = '" & SubFields(3) &"' and Student.St_id = Grade.St_id and Subject.Sub_code = Grade.Sub_code.

After executing the above query on database **Exam** the message sent from student, and result from server application to SMS gateway application is shown in Figure 6. The SMS gateway application then forwards the data received form the server to the student as an SMS.

The administrator can provide other messaging service using the same database like "viewing details of a subject", in which the user send subject code and in response the subject name and total credit hours are returned to him. The user-provided list contains **Subject.Sub_code** and response list includes **Subject.Sub_name** and **Subject.crd_hrs**. As the entire attributes belongs to a single table "**Subject**", so there is no need of link information.

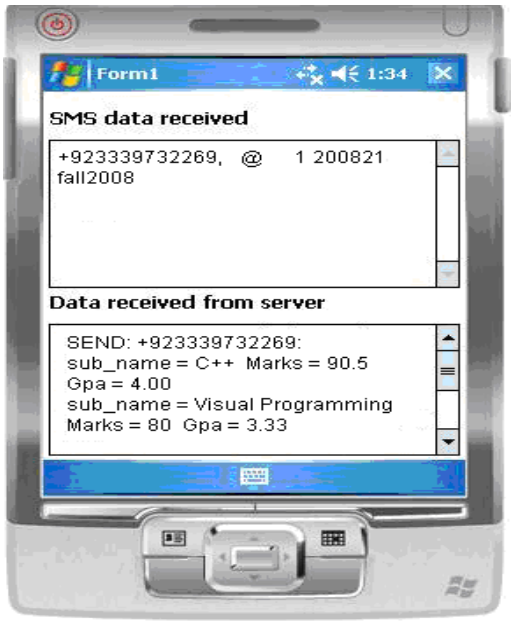

Figure 6. Screenshot Gateway Application

After providing these two input parameters, the format of the SMS to be provided by the user is:

@ <2> <Sub_code>.

The format of the query to be executed against the database is:

Select Subject.Sub_name, Subject.crd_hrs from Subject where Subject.Sub_code = '"& SubFields(2) & "'.

As it is difficult to memorize the format of the SMS to view results, a student may sent an SMS **@ <query> <Service No>** to the SMS gateway, in a reply the correct message format is provided to the student

B. Public Transport Services Information

Let the following information be provided by the system to the mobile users.

1. When a user provides a specific route number, the system responds with a list of bus numbers or bus services that are passing through this route along with their starting and destination station names.
2. When a user provides its destination stop number to which he/she wants to go, the system replies with a list of bus numbers that are passing through this stop.
3. When a user provides a bus number, the system sends all the names of stops through which this bus is passing.

The administrator may use the ERD named "**TransInfo**" given in Figure 7 to provide all these information.

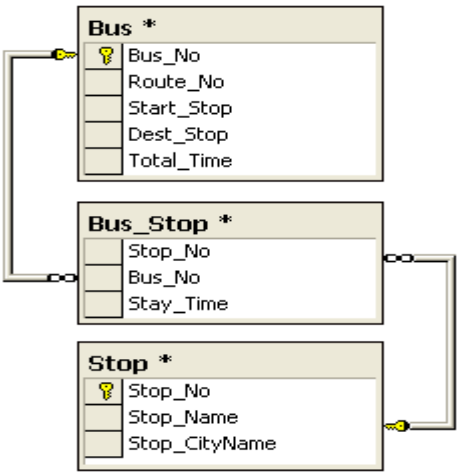

Figure 7. ERD of the Public Transport Services

To provide the first information, the administrator selects **Bus.Route_No** in user-provided list while **Bus.Bus_No** , **Bus.Start_Stop** and **Bus.Dest_Stop** in response list. As all the information are obtained from a single table name **Bus,** there is no need to provide link information. After providing these two input parameters, the format of the SMS given by the user is:

@ <3> <Route_no>.

The format of the query to be executed against the database is:

Select Bus.Bus_No , Bus.Start_Stop, Bus.Dest_Stop from Bus where Bus.Route_No = '" & SubFields(2) & "'

To provide the second information, the administrator selects **Bus_Stop.Stop_No** in user-provided list while **Bus_Stop.Bus_No** and in response list. As all the information is obtained from a single table name **Bus_Stop,** \ link information was not necessary.

After providing these two input parameters, the format of the SMS to be provided by the user is:

@ <4> <Stop_no>.

The format of the query to be executed against the database is:

Select Bus_Stop.Bus_No from Bus_Stop where Bus_Stop.Stop_No = '" & SubFields(2) & "'. The above query returns all the bus numbers which are making a stop on the user destination stop.

To provide the third information, the administrator selects **Bus_Stop.Bus_No** in user-provided list. As the response to the user SMS contains names or locations of all the stops through which a specific bus will go, administrator has to select **Stop.Stop_Name** instead of **Bus_Stop.Stop_No** in response list. As all the information are obtained from two tables **Bus_Stop** and **Stop**, link information between these two tables is necessary. The link information is:

" Bus_Stop.Stop_No = Stop.Stop_No"

which works as equi-join condition between the two tables.

After providing these three input parameters, the format of the SMS to be provided by the user is:

@ <5> <Bus_no>.

The format of the query to be executed against the database is:

Select Stop.Stop_Name from Bus_Stop, Stop where Bus_Stop.Bus_No = '" & SubFields(2) & "' and " Bus_Stop.Stop_No = Stop.Stop_No

The above query returns all stop names on which a specific bus is passing.

C. *Sales Information's*

Let the following information be provided by the system to the Customer.

1. When a customer provides its identification number along with starting date and ending date, then the system responds with a list of all the invoices of that customer between these two dates.
2. When a customer provides the invoice number of one of its invoices, then the system replies with a list of all the items along with quantity and unit prices of that specific invoice. Also the invoice date and discount value is returned to the customer.

The administrator may use the ERD named "SalesInfo" given in Figure 8 to provide both of these information.

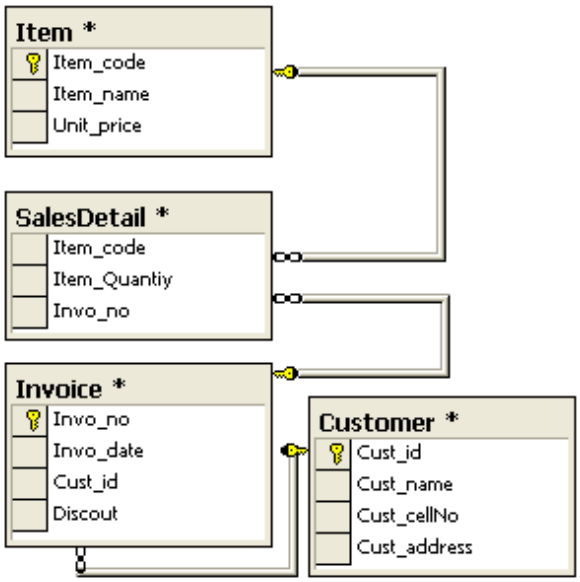

Figure 8. ERD of the Sales Information's System

To provide the first information, the administrator selects **Customer.Cust_id , Invo_date , Invo_date** in user-provided list. The first **Invo_date** is used as starting date while the later is used as ending date. The administrator selects **Invoice.Invo_no** in response list and **"Customer.Cust_id = Invoice.Cust_id"** as link information between tables **Customer** and **Invoice.**

After providing these three input parameters, the format of the SMS to be sent by the customer is:

@ <6> <Cust_id> < Invo_date> < Invo_date>

The format of the query to be executed against the database is:

Select Invoice.Invo_no from Invoice , Customer where Customer.Cust_id= '" & SubFields(2) & "' and Invoice.invo_date > = '" & SubFields(3) & "' and Invoice.invo_date < = '" & SubFields(3) & "' and Customer.Cust_id = Invoice.Cust_id

To provide the second information, the administrator selects **Invoice.Invo_no** in user-provided list while **SalesDetail.Item_code**, **Item.Item_name**, **Itme.Item_unitpriice**, **SalesDetail.Item_quantity**, **Invoice.Invo_date** in response list and **"Invoice.Invo_no = SalesDetail. Invo_no** and **SalesDetail.item_code = Item.Item_code"** as link information between tables, **Item**, **SalesDetail** and **Invoice**.

After providing these three input parameters, the format of the SMS to be sent by the customer is:

@ <7> <Invo_no>.

The format of the query to be executed against the database is:

Select SalesDetail.Item_code, Item.Item_name, Itme.Item_unitpriice, SalesDetail.Item_quantity, Invoice.Invo_date from Invoice , SalesDetail , Item where Invoice.Invo_no = '" & SubFields(2) & "' and Invoice.Invo_no = SalesDetail.Invo_no and SalesDetail.item_code = Item.Item_code

VI. IMPLEMENTATION

The following software and hardware are used during the development and implementation of this project:

Software Requirements

- Programming Language – VB.Net
- Database- SQL Server 2000
- Windows Mobile 5.0 Pocket PC SDK
- Active Sync 4.2

Hardware Requirements

- Desktop PC
- Windows Mobile 5.0 Pocket PC
- SIM Card
- Bluetooth USB Adapter

CONCLUTION

In this paper, we present a multipurpose information system which can be successfully used to provide various information in different firms. It is a cheaper way of providing useful information to users in those areas where there is no internet facility.

This system can further be extended to a system which not only provides information but can also does transactions based on user SMS. The user can make changes in the database records. But the security is the main issue to such a system. The message must be sent in a secure way between a user and a system.

ACKNOWLEDGMENT

This work was supported by the Engineering Research Center of Excellence Program of Korea Ministry of Education, Science and Technology (MEST) / Korea Science and Engineering Foundation (KOSEF), R11-2008-007-01003-0.

REFERENCES

[1] http://www.palowireless.com/sms/resources.asp.

[2] Lim Tai Ching; Garg, H.K "Designing SMS applications for public transport service system in Singapore" The 8th International Conference on Communication Systems, 2002. ICCS 2002. Volume 2, 25-28 Nov. 2002 Page(s):706 - 710 vol.2.

[3] Ke Wan "An SMS-based sales reporting system for a fashion-clothes franchising company" Engineering Management Conference, 2003. IEMC "03. Managing Technologically Driven Organizations: The Human Side of Innovation and Change 2-4 Nov. 2003 page(s):330 – 334

[4] Shahreza, M.S " M-Quiz by SMS". Sixth international conference on advance Technologies, 2006 05-07 July 2006 Page(s):726 – 729

[5] Aziz, Q "Payments through Mobile Phone" 6th International Conference on emerging Technologies 2006, ICET

[6] Carl collins, Amy Grude, Mathew Scholl, Robert Thompson "txt bus: wait time information on demand" , Conference on Human Factors in Computing Systems CHI '07 , April 2007 , Pages:2049-2054.

[7] Felix-Robinson Aschoff, Jasminko Novak "The mobile forum: real-time information exchange in mobile sms communities", Conference on Human Factors in Computing Systems CHI '08, April 2008, pages: 3489-3494

[8] Hua Wang, Xiaodi Huang, Goutham Reddy Dodda "Ticket-based mobile commerce system and its implementation". 2nd International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems, October 2006, pages: 119 - 122

[9] Archana Prasad, Sean Blagsvedt, Kentaro Toyama "SMSBlogging: Blog-on-the-street Public Art Project". 15th international conference on Multimedia, September 2007, pages: 501 - 504

[10] Md. Subrun Jamil, Fouzia Ashraf Mousumi1 "Short Messaging Service (SMS) Based m-Banking System in context of bangladesh" .11th International Conference on Computer and Information Technology (ICCIT 2008), 24-27 Dec. 2008 Page(s):599 - 604