ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 4, June 2012

An Automated Mobile Applications Generator for Remote Database Access

K. Spandana[#1], K. Shirin Bhanu[#2] , G. Loshma[#3]

[#1] Student, Sri Vasavi Engineering College, Tadepalligudem, West Godavari(dt),

[#2] Senior Asst.professor, Sri Vasavi Engineering College, Tadepalligudem, West Godavari(dt),

[#3] Assoc.professor, Sri Vasavi Engineering College, Tadepalligudem, West Godavari(dt).

Abstract- **Due to the display area and resource restrictions of small mobile devices, currently, most reported database query systems developed for them are only offering a small set of pre-determined queries that can be posed by users. This not only limits the usage of such systems in their own domain of application, it also prevents them from being used to query other databases. Thus, the abilities to support different query types as well as unplanned queries are essential for these systems in order to make them generic. Hence, this paper presents a new free-form database query language that is suitable for these devices. The language which is used as the query formulation method in a database query system prototype for WAP enabled mobile phones has been found to be effective based on results from usability tests. It is possible to show the result in different PC by using MVC architecture and MIDP application of J2ME.**

1. Introduction

The ability to query for information from remote databases anytime anywhere has become increasingly indispensable for today's highly mobile society. Today, with the advancement of technology in both data communication networks and accessing devices, the above activity can be carried out using many small mobile devices such as the Personal Digital Assistants (PDAs), the palmtops and even the mobile phones. However, due to the small display size as well as the limited resources of these devices, providing such capability would be a challenge. Currently, majority of the database querying systems developed for mobile access, for examples [1], [2] and [3], are for use on the PDAs. Even for these devices, which can be considered as having considerable resources, the systems only provide minimal querying capabilities. Hence, possible queries that can be formulated on these systems are mostly pre-determined by the developers as sets of options provided on a menu. Such a measure tends to limit the usage of these systems. It leaves no room for users to issue any query of interest. Furthermore, this approach of accepting only precise queries also hinders the systems from being easily adopted to query other databases. Therefore, we believe that in order to have a generic database query system for small mobile devices, the system must be made supportive of different types of queries as well as unplanned queries. In doing so, the system should use minimal resources possible. Thus, in this paper, we shall introduce a new database query language that is suitable to be implemented as the query formulation method on display area and resource restricted mobile devices. In our study, we have implemented the language in a database query system prototype for mobile phones. We advocate that if the language works on mobile phones, then it should be able to be adopted by other "thicker" small devices. Hence, the remainder of this paper is organized as follows. Section 2 highlights some related works, Section 3 introduces the main concepts of the query language, Section 4 presents the database query system prototype for mobile phones, Section 5 discusses the usability tests conducted on the prototype, and Section 6 provides conclusions.

2. Related work

Database querying has been the focus of many database researchers for a long time. However, the capability of transacting queries while on the move using small mobile devices has only recently gained interest from the database community. Currently, the above interest is mainly targeted to mobile devices of considerable resources such as the PDAs, the palmtops and the notebooks. Even for these devices, the works reported are mostly application specific. For example, Hung and Zhang [1] presented a telemedicine system which can be used to access patient generalinformation and medical conditions such as BP reading and ECG diagramming on PDAs. Meanwhile, Koyama et al. [2] developed a system for education application. Their system can be accessed on PDAs by students who want to perform lesson's unit test. Boonsrimuang, Kobayashi and Paungma [3], on the other hand, presented a system for transportation application, also on PDAs. Even though the above applications and others are undeniably important, they are very limited in terms of their functionalities. In other words, their usefulness is confined to a single domain of application, and even within that particular domain, they are restricted to the functions (queries) that have been pre-defined by their developers. Thus, there is no possibility for unplanned queries to be formulated on such systems.

By using precise input query method as the above, unplanned queries are rather hard to implement. This is so, since they require almost all combinations of possible query terms to be thought of beforehand. These terms would need to be presented on the system interface for user selection. Especially for mobile devices, in particular, the mobile phones, this concept would be too expensive to implement since it requires a complex menu structure that needs considerable number of clicks and scrolls. A user experiment which was conducted by Ahmad and Abdul-Kareem [4] on two different interfaces, i.e., menu/GUI-based and keyword-based, has shown that users most of the times prefer query input method that requires direct inputs rather than several expansion of menus. Therefore, imprecise queries must be allowed as an alternative.

Imprecise query has been the subject of many researchers. It has been widely used especially in information retrieval [5][6]. However, for database, imprecise query was only discussed in some keyword based schema-less related query systems such asthose of Agrawal, Chaudhuri and Das [7] and Calado et al. [8]. Furthermore, the above systems were developed for usage on conventional devices which have fixed network connection. As for small mobile devices, to the authors knowledge, no such querying concept is yet to be found. Furthermore, the word "unplanned" in database querying is rather subjective. This is because, like any other computer-related operations, there is always a limit to the kind of operations which can be executed. For database, this range of operations would depend on the level of expressiveness that a query language exhibits [9]. Chandra [10] mentioned in his work that a language can support relational level of expressiveness at the lowest level to computable expressiveness at the highest end; and these languages can be of different forms. For example SQL is a textual language which exhibits at least relational level of expressiveness (it supports the five basic operations of Selection, Projection, Join, Union and Set Difference) and QBE is a graphical language which is also capable of supporting relational expressiveness. For mobile devices, only one work which discussed the issue of language expressiveness can be found [11]. Polyviou, Samaras and Evripidou [11] discussed expressive queries in their work which implement directory-like interface for query formulation. However, their method is suitable only to be used with devices that have pen input mechanism. As the majority of mobile phones lack this type of input mechanism, we believe that a conventional way of input, i.e., input using normal phone's keypad and function keys, should be considered. Another interesting option to be considered for inputs would be voice input. However, as being highlighted in [12] and [13], this method has one major problem of voice recognition which needs further research.

3. Query language for mobile phones

Prior to the construction of the query language, a small survey involving 45 fourth year students from the Department of Computer and Information Sciences (CIS) at Universiti Teknologi PETRONAS, Malaysia (UTP) who had just returned from an eight-mont industrial internship program, was conducted for gathering the types of queries users would normally issue to a database. In the survey, a test database schema (of a university scenario) was presented to the respondents in a narrative form. Based on the scenario, each respondent was asked to provide at least five possible queries that they may want to issue to the database. As a result, 262 queries were returned. Based on the major query operation each query required, five different query groups were identified. The five groups were projection, selection, join, set difference and union. Figure 1 below shows the distribution of the returned queries among the five groups.

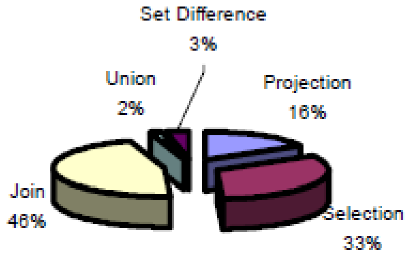


Fig1.Distribution of query types

Based on the above figure, it can be said that out of the five groups, majority of the queries received were of the join (46%) and the selection (33%) types. Furthermore, it can also be said that user queries, eventhough might be unplanned, were conformed to the relational level of query expressiveness as coined by E.F. Codds in the 1970's [9]. As mentioned in [9], five basic relational operations that make up relational expressiveness are the projection, selection, Cartesian products, union and set difference operations. Hence,our language is therefore, must support all five operations in order to be able to allow for acceptance of unplanned queries. To cater for the above requirement, a free-form query language is proposed. Free-form is a concept which is based on the universal relation [14]. It can be used to reduce the number of terms needed in a query.

Especially for queries of the join type, the number of query terms can be greatly reduced by not specifying the foreign key relationships between them. The universal relation concept has been known to beapplied in keyword-based querying [7][8]. However, incontrast to keyword-based querying which uses database instances as query terms, the approach that we opted for our language uses schema terms instead. This is so due to several reasons which are related to the accessing devices that we are using, i.e., the mobile phones. As mentioned earlier, works on keyword based querying were done on conventional devices, i.e., desktop over the fixed line networks. Hence, time, processing power and cost are not of major considerations for such systems. However, for mobile phones, the above factors are important. Therefore, using precise terms is one way to avoid costs induced by terms mismatches. Besides, schema-based approach can also reduce the number of possible query results due to minimal term ambiguity as compared to schema-less method .

Figure 2 below shows the structure of our proposed language, and Table 1 that follows present some examples of queries that can be composed using the language.

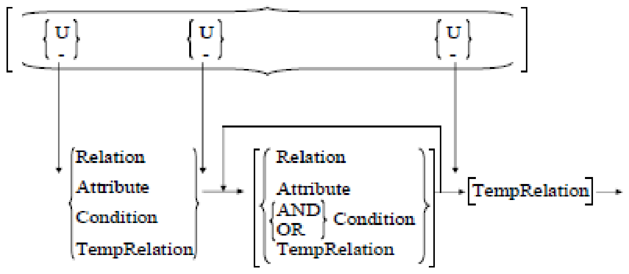


Figure 2. Syntax structure for query language

No	Queries
1	STUDENT SUBJECT
2	SUBJECT.name STAFF.name
3	SUBJECT STAFF.name
4	STAFF.ID=112971 SUBJECT.crhr>3
5	STUDENT.studname U STUDENT.studyear=3 STAFF.stafname

Table 1. Sample queries

Here we will see the results of each and every field by using Sun Wireless toolkit at the same time we can possible to see the result of each and every field updation result in different systems by using MVC Architecture. The following are the different modules.

1)server module
2)Connection module
3)Design of application
4)query generation module

1)Server Module

In this, we are using MySQL as the data server and Apache Tomcat 5.0 as the web server. These two are the main core of the server side programming. Our application has been deployed in the Apache Tomcat so that all kind of Http Requests and response can be handled easily. All the parameters passed from the request can be retrieved using the getParameter() method of HttpRequest Object.

2)Connection Module

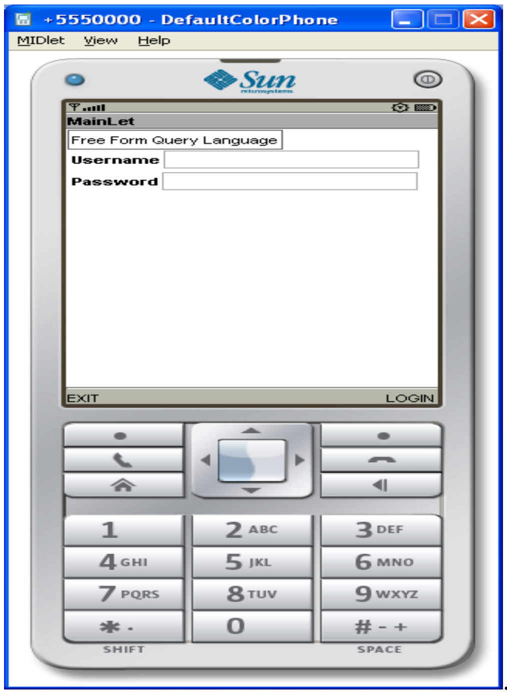
The Connection between the Client (i.e. J2ME application in mobile) and the Web Server is maintained by the object HttpConnection in javax.microedition.io.HttpConnection. Using this connection module we could able to retrieve the database information by passing the Http Request. The requesting attributes are sent as the parameters of the url built for HttpConnection.

3)Design of Application

According to the request sent by the client, the server process the data and it is responded to the client, which is further received by the help of openDataInputStream() in HttpConnection Object. So the client application is designed according to the Field information of the tables retrieved from the server. We would be using ChoiceGroup Object for designing Radio Buttons, Check Boxes and Dropdown list boxes.

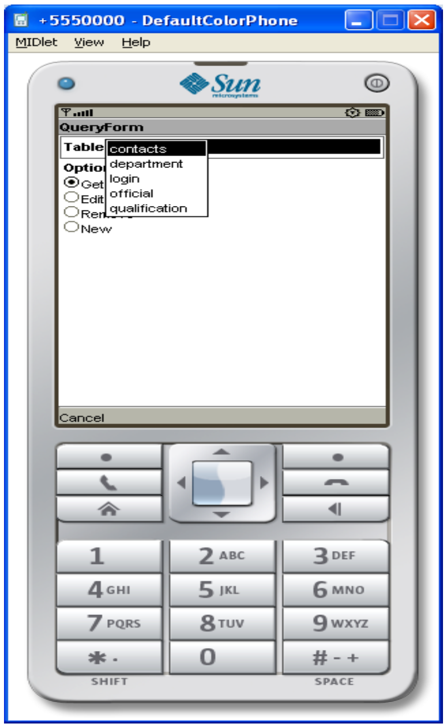
4)Query Generation Module

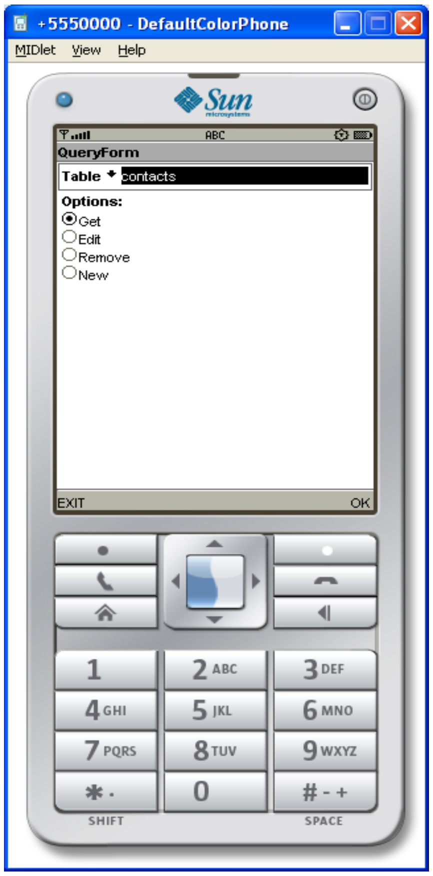
The design of the application is done by the data types which we used in the database. The choice which we are about to use, according the query will be generated behind the screen. After the complete operation of selecting the choices we are supposed to execute the query by passing it as the parameter to the server through Http Request

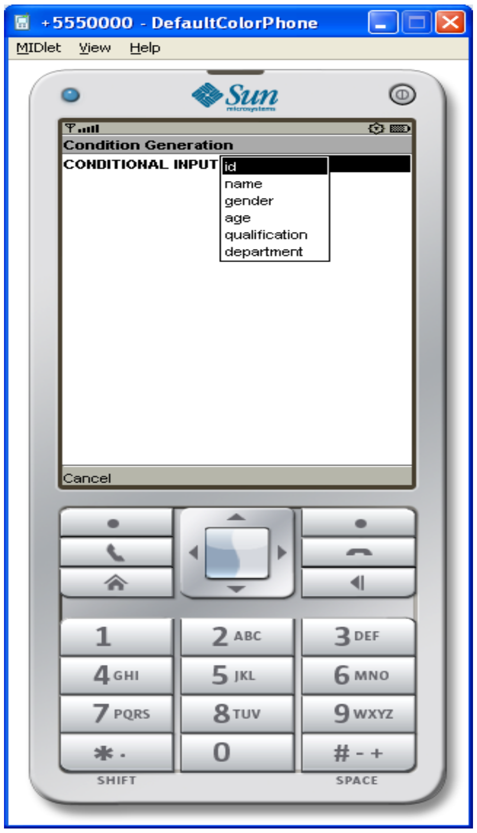


Free Form Query Language
Username
Password

.







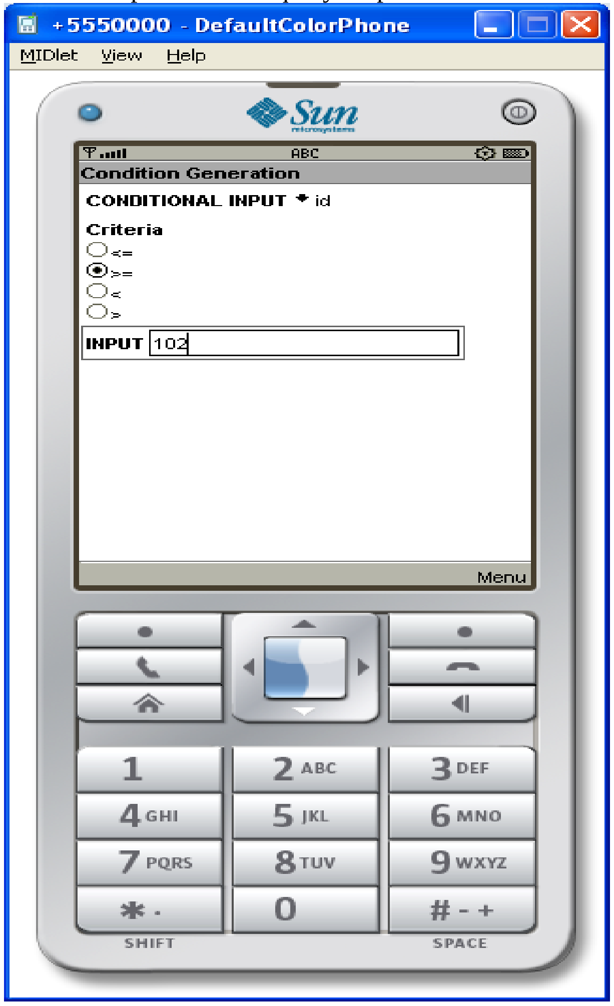
4. System prototype for mobile phones

In order to show that our language can support its intended capabilities, a prototype was developed. This prototype consists of a J2ME midlet for the interface on the Java phone emulator, and several Java servletsfor the execution of queries. The interface developed follows the guidelines given in most references on J2ME such as those of Mahmoud [15] which suggested that an interface for small devices should be simple anduse as many as possible high-level APIs. Figure 3below shows a screen-shot of our prototype's interface for composing free-form queries.
Figure 3. Interface for composing queries A pull-down list is used to present all query terms, i.e., schema information and operators. These terms are presented as a linear list. Composing a query is then, just a matter of clicking on as many query terms as needed and supplying values if required.

5. Usability test on prototype

In order to ensure that the language is effective in supporting different query types as well as unplanned queries, usability tests were conducted on the system prototype. 15 test subjects were identified. Five of them were first year first semester students at the CIS department of UTP who at the time of the tests took a database course, another five were first year secondsemester students of the same department who had taken a database course in the previous semester, and the rests were users from the public. All of the participants were observed individually in separate sessions as they were using the prototype for formulating database query tasks. Each session took around one-and-a-half hour to complete. Below are the details of an observation session. **Observation setup**. A notebook was used as both the application and database servers for the usability tests. A Java phone emulator and a real phone, Nokia 6681,which were loaded with the interface of the prototype, were used as possible accessing devices. Besides the prototype, a test database on a university domain was used for the first two groups, and a database on a sporting event was used for the third. The server wasconnected to the Internet for networking environment(Note: due to organization's firewall, another mobilephone, Motorola V360 was used as data modem).**Observation materials & approach**. There were four parts to an observation exercise. The first part beganwith the observer performing five query tasks on the prototype. The participant was asked to observe each query formulation process and to pose questions if he/she needed further clarification. Part two of the usability test proceeded with the participant being asked to provide and execute five database query tasks of his/her own choice. The participant was allowed to seek for assistance from the observer if needed. After part two has been completed, part three of the usability test started. This time, the participant was given the same query tasks as part two, and the observation started. Behavior of the participant was recorded and the number of attempts, the number of mistakes and assistance were jotted down. The participant was also required to identify the status of the query task, 3 if completed and sure the outputs were correct, 2 if completed but unsure of the outputs correctness, and 1 for incomplete after the end of each task. Part four of the usability test continued with the participant was given a totally new set of query tasks and observation as part three was carried out. Again, at the end of each task, the participant was asked to state the completion status of the task.

From the data collected during each observation session, the effectiveness of the language was calculated. This calculation was based on the simple method suggested by Neilsen [16] for measuring prototype effectiveness. Neilsen [16] said that partial credits should be given to tasks which were not completed as per expectation. Hence, different credit scores (in percentage value) will be given to denote the successfulness of tasks. For example, a 100% credit will be given to tasks which were completed accurately, a 0% credit to failed tasks, and a 50% credit for partly successful tasks. Overall effectiveness measurement is then derived by calculating the average of all given credits. In our study, five possible credits were used to denote task's completion status, i.e., 1,0.75, 0.5, 0.25 and 0; and they were assigned to each task based on the following collected data: The correctness of the query outputs as expected by the observer, and the rating on the scale of 1 to 3 which was given by test subject depicting his/her acceptance of the query outputs.



Condition Generation

CONDITIONAL INPUT ✦ id

Criteria

○ <=
● >=
○ <
○ >

INPUT 102

6. Conclusion

As a conclusion, it is possible to develop a generic database query system for mobile phones which accepts different types of queries as well as unplanned queries, by allowing imprecise inputs. Since mobile phones, are poor in terms of resources as compared to other mobile devices, the successfulness of implementing such a method on them would mean it is applicable to the other devices. Free-form language can provide a much simpler interface for users to formulate queries. The language helps in reducing the number of query inputs especially in cases where joins of relations are needed. Since the majority of queries which might be issued are of this type (as seen by the queries given by respondents), providing such a method would benefit users of resource-poor devices. Usability tests on the prototype have also shown that the language is effective even when used by novice users. For future work, we plan to integrate recommender systems into our work so that lesser and only relevant query terms will be presented to users for selection on the pull-down list. In future it can possible to access the images from the database with mobile with high speed enabled GPRS connection.

7. References

[1] K. Hung, and Y-T. Zhang, "Implementation of a WAPBased Telemedicine System for Patient Monitoring", *IEEE Transactions on Information Technology in Biomedicine*, Vol. 7, No. 2, 2003, pp. 101-107.

[2] A. Koyama, N. Takayama, L. Barolli, Z. Cheng, and N. Kamibayashi,"An agent based campus information providing system for cellular phone", *in Proc. of the 1st International Symposium on Cyber Worlds*, 2002, pp. 339-345.

[3] P. Boonsrimuang, H. Kobayashi, and T. Paungma, "Mobile Internet Navigation System" , *in Proc. of the 5th IEEE International Conference on High Speed Networks and Multimedia Communications*, 2002, pp. 325-328. 283 Authorized licensed use limited to: VELLORE INSTITUTE OF TECHNOLOGY. Downloaded on August 31, 2009 at 03:35 from IEEE Xplore. Restrictions apply.

[4] R. Ahmad, S. Abdul-Kareem, "Keyword-driven Interface for Mobile Phone's Database Query: A Feasibility Study", *in Proc. of the International Conference on IT and Multimedia*, 2005.

[5] A. Bergstrom, P. Jaksetic, and P. Nordin, "Enhancing Information Retrieval by Automatic Acquisition of Textual Relations using Genetic Programming", *in Proc. of IUI 2000,*
2000, pp. 29-32.

[6] H-M. Lee, S-K. Lin, and C-W. Huang, "Interactive Query Expansion Based on Fuzzy Association Thesaurus for Web Information Retrieval," *in Proc. of IEEE International Fuzzy Systems Conference*, 2001, pp. 724-727.

[7] S. Agrawal, S. Chaudhuri, and G. Das, "DBXplorer: A System for Keyword-Based Search over Relational Databases", *in Proc. of IEEE 18th International Conference on Data Engineering (ICDE'02)*, 2002, pp. 5-16.

[8] P. Calado, A.S. da Silva, A.H.F. Laender, B.A. Ribeiro- Neto, and R.C. Vieira, "A Bayesian Network Approach to Searching Web Databases through Keyword-Based Queries", *Information Processing and Management*, Vol. 40, No. 5, 2004, pp. 773-790.

[9] R. Ramakrishnan, and J. Gehrke, *Database Management Systems*, McGraw-Hill, New York, 2000.

[10] A. Chandra, "Theory of database queries", *in Proc. of the Seventh ACM Symposium on Principles of Database Systems*, 1988, pp. 1-9.

[11] S. Polyviou, G. Samaras, and P. Evripidou, "A Relationally Complete Visual Query Language for heterogeneous data sources and Pervasive Querying", *in Proc. of the 21st International Conference on Data Engineering (ICDE 2005)*, 2005, pp. 471-482.

[12] E. Chang, F. Seide, H.M. Meng, Z. Chen, Y. Shi, and Y.C. Li, "A system for spoken query information retrieval on mobile devices", *IEEE Transactions on Speech and Audio Processing*, Vol. 10, No. 8, 2002, pp. 531-541.

[13] B.R. Bai, C.L. Chen, L.F. Chien, and L.S. Lee, "Intelligent Retrieval of dynamic networked information from mobile terminals using spoken natural language queries", *IEEE Transactions on Consumer Electronics*, Vol. 44, No. 1, 1998, pp. 62-72.

[14] D. Maier, J.D., Ullman, and M.Y., Vardi, "On the Foundations of the Universal Relation Model", *ACM Transactions on Database Systems*, Vol. 9, No. 2, June 1984, pp. 283-308.

[15] Q. Mahmoud, *Wireless Java*, O'Reilly, 2002. [E-book].Available: O'Reilly e-book

[16] J. Nielsen., "Success Rate: The Simplest Usability Metric", Jacob Nielsen's AlertBox, Feb. 18, 2001.