Mobile Information Systems 5 (2009) 233–253
DOI 10.3233/MIS-2009-0083
IOS Press

Virtual social networks online and mobile systems

Maytham Safar*, Hussain Sawwan, Mahmoud Taha and Talal Al-Fadhli
Department of Computer Engineering, Kuwait University, Kuwait

Abstract. Location-based applications are one of the most anticipated new segments of the mobile industry. These new applications are enabled by GPS-equipped phones (e.g., emergency applications, buddy finders, games, location-based advertising, etc.). These services are designed to give consumers instant access to personalized, local content of their immediate location. Some applications couple LBS with notification services, automatically alerting users when they are close to a pre-selected destination. With the advances in the Internet and communications/mobile technology, it became vital to analyze the effect of such technologies on human communications. This work studies how humans can construct social networks as a method for group communications using the available technologies. We constructed and analyzed a friends network using different parameters. The parameters that were calculated to analyze the network are the distribution sequence, characteristic path length, clustering coefficient and centrality measures. In addition, we built a PDA application that implements the concept of LBS using two system modules. In the first module, we have developed an application for entertainment purpose; an application program which enables end users to send their birth year and get their horoscope in return. The second part of the project was, to build an application, which helps people to stay in touch with their friends and family members (Find Friend). It helps users to find which of their buddies are within the same area they are in.

Keywords: Virtual communities, ubiquitous computing, small world, scale-free network, social networks, GIS system

1. Introduction and related work

With the increasing population of the world, the importance of modeling social networks increases [5, 19,27,28]. The reason is that such networks are crucially important for communication. Most human communication takes place directly between individuals. The spread of news, rumors, and diseases all take place by contact between individuals. And a rumor can spread from a region to another far faster over a social network in which the average degree of separation is low, than it can over a network in which the average degree is high.

The spread of disease also occurs by person-to-person contact. The structure of networks of such contacts has a huge impact on the nature of epidemics. For example, in a highly connected network, bird flu disease can spread far faster than in a network where the paths between individuals are relatively long.

Many works developed different techniques and algorithms for mining mobile data and providing different services. Those aimed at finding useful knowledge from the raw data produced by mobile users based on location [8,9] and/or finding frequency pattern and group pattern [8]. Other works focused on developing algorithm to answer queries on mobile devices more efficiently [1,2,13,21].

*Corresponding author: P.O. Box 5969, Safat 13060, Kuwait. Tel.: +1 965 24987962; Fax: +1 965 24839461; E-mail: maytham.safar@ku.edu.kw.

Similar mobile applications relevant to the context of social networks were developed in [10,19]. The work in [19] presents an enhanced chat-bot architecture (EHeBby). It is a humoristic conversational agent capable of generating, proposing, telling and answering to the user: humoristic expressions, riddles, and jokes. It is based on traditional rule-based knowledge representation, i.e. the standard AIML KB, which exploits both WordNet and the CMU pronouncing dictionary. It builds an LSA-inspired semantic space in which the sentences stored in the standard AIML KB.

The other work in [10] developed two tour guide systems for Brunel University; one with context-aware user feedback and one without. It studied the impact of using real-time context-aware information on system usability by assessing three metrics, effectiveness, efficiency and satisfaction.

The work in [20] presented an integrated middleware and service provision platform for delivering Location Based Services (LBS). It covers the full life cycle of a LBS and facilitates the creation and provision of service issues like specification, deployment, invocation, and maintenance. All that is accomplished with minimum effort the mobile service provider.

In our previous works [23], we studied and analyzed different social models and networks that are available on the web. The purpose of this paper is not to offer a new model for social networks on the web. The social networks developed on the web are relating people online and they have interactions between them based on email messages, blogs, and bulletins. For example, the work in [4] concerns with everyday life. It analyzes how mobile information systems, including social networking affects people's life and how they interact. It developed a framework for the analysis of processes in settings of private everyday life.

Those processes establish services based on mobile application and devices. It allows the assessment of alternatives to support these processes with mobile services. Hence, it facilitates the creation of recommendations for developing, launching and establishing complex services in private settings.

The application targeted in this paper is more into applications on mobile and that are location based mostly [10,19]. We start our work by constructing and analyzing a friends network using a member of a friendship web site. The main aim of such analysis is the measuring of relationships and information flows between individuals. We were able to model the friends network as a social network; as each individual user is an entity and their interactions imply relationships and flows. The friends network is modeled as a graph, consisting of a set of nodes and edges, where each node represents a user and an edge represents a relationship between a pair of nodes.

To illustrate the creation and analysis of friends network, in this paper, we developed a location-based application (i.e., Find Friend) as a game scenario. Location-based applications [10,19] are one type of application that helps set apart the wireless applications from wired applications. It involves the use of position information about a user's particular location with intelligent applications and solutions to provide relevant information and services. With automated acquisition of user location; location based services (LBS) scenarios are becoming more compelling and their applications are more useful.

LBS solutions [20] in their simplest form can include providing relevant maps, driving directions, proximity searches for Points of Interest (POI) such as a nearby ATM. It can also include more advanced enterprise scenarios such as intelligent dispatch, field service and geo-fencing. These solutions often involve multiple application types including web based or client/server and mobile applications providing LBS on Pocket PC, Tablet PC and other mobile devices.

We have developed two mobile applications providing location-based services. The first application is "Horoscope" application; which enables users to send their birth year and get their horoscope in return. The application consists of an application running on a Pocket PC device. It interacts with LBS scenario that we build for the purpose of processing application requests and hence, fetch the required data from the database.

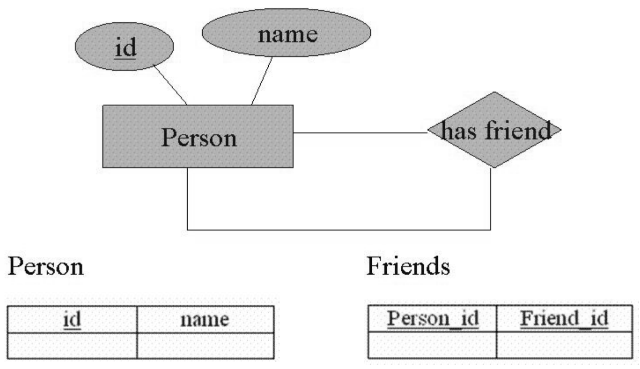


Fig. 1. Database design.

The second application is "Find Friend" application running on Pocket PC device. "Find Friend" application helps users to enter their location (i.e., Kuwait City) and send it as a message to the service provider. This in turns query for the list of friends and family members that exists in the same area. "Find Friend" application consists of an application running on a Pocket PC device which interacts with LBS scenario that we have built for the purpose of processing requests. The scenario then query for the required information, fetch the data from the database, and finally send it back to the user.

2. Friends network construction

In this work we construct and analyze a friends network using a member of a friendship web site (i.e., www.hi5.com). We analyzed this sample because it is a representation of the real world. We took a member and explored the friend's list of the member to include the friends' list of each friend and so on. We created a database that stored a list of each person with his/her unique id, and name and another table that has that persons' id and his/her friends' ids.

The database design is shown in Fig. 1. To construct the friends network G, we first created nodes representing each individual and the friends in his list. Directed edges were added between each individual and his friends who are in the list and we created a network as shown in the Fig. 2 in a random layout. Figure 3 shows a different view of the same network in a circular layout.

The parameters that were calculated to analyze the network are the distribution sequence, characteristic path length, clustering coefficient and centrality measures. Before giving the definition of the distribution sequence, we have to introduce an important definition which is the neighborhood of a vertex $\Gamma(v)$. "The neighborhood $\Gamma(v)$ of a vertex v is the sub graph that consists of the vertices adjacent to v" [27]. The distribution sequence for v is the sequence

$$\Lambda_j(v) = \sum_{i=0}^{j} \mid \Gamma^i(v) \mid \text{ for } 0 \leqslant j \leqslant jmax \tag{1}$$

The distribution sequence for G $\Lambda(v) = \Lambda_j$ over all $v \in V(G)$. It follows from these definitions that $maxv(jmax(v)) = D$ which is the diameter of the graph [27]. The Λj is an indicator of the rate at which information spreads throughout the network.

The neighborhood is also useful in computing the clustering coefficient of the network. The clustering coefficient γv of Γv characterizes the extent to which vertices adjacent to any vertex v are adjacent to

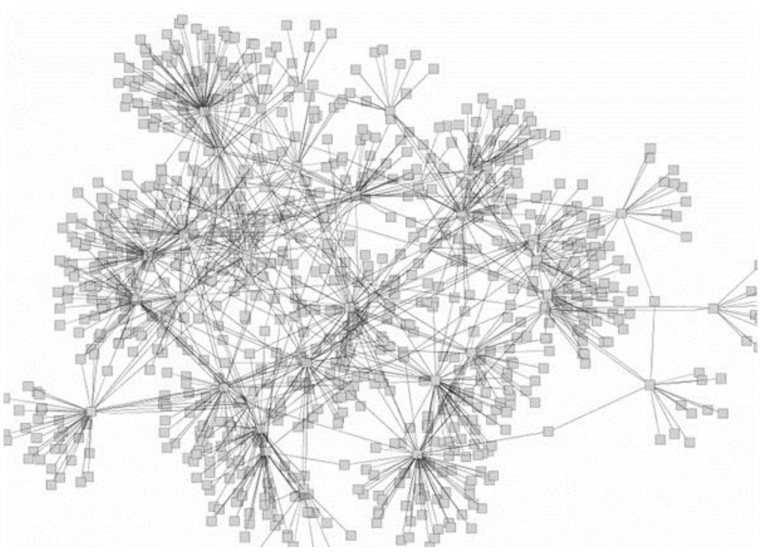


Fig. 2. Random layout.

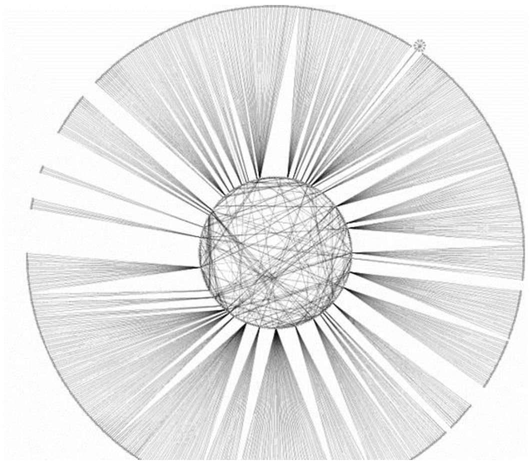


Fig. 3. Circular layout.

each other, where $\gamma_v = \frac{|E(\Gamma_v)|}{n(n-1)}$. $\mid E(\Gamma_v) \mid$ is the number of edges in the neighborhood of v and $n(n-1)$ is the total number of possible edges in Γv. This equation was modified from the one in Eq. (1) because here we are considering a directed network. The directed network is used here because each person has a friend but not necessarily visa versa. The clustering coefficient of G is $\gamma = \gamma v$ averaged over all $v \in V(G)$. In terms of my friends network the clustering coefficient will give me an indication to know how many of my friends know each other.

The characteristic path length L is the median of the means of the shortest path length connecting each vertex $v \in V(G)$ to all other vertices. Centrality deals with the role of an individual in a network.

Several centrality measures such as degree, closeness and betweenness can suggest the importance of a node [3]. The degree is the number of direct connections a node has. The degree implies the activity of an individual. The closeness is the sum of all the geodesics (shortest path between any two nodes)

Table 1
Network parameters

Parameter/Network	No added edges	Added edges
Number of nodes	893	893
Number of edges	1154	2878
Avg. clustering coefficient	8.446E-4	0.00884
Graph distribution sequence	56	687.0
Graph diameter	14	16
Characteristic path length	0	5.68

Table 2
Closeness and Betweenness

Parameter/Network	No added edges	Added edges
Minimum closeness	3427	1
Maximum closeness	100000	100000
Avg. closeness	15745.455	15540.98361
Minimum betweenness	0	0
Maximum betweenness	25869	69354
Avg. betweenness	307.9507279	3975.363942

between the particular node and every other node in the network.

The closeness implies how close the individual is to others. A node with a high closeness implies that an individual is in an excellent position to monitor the information flow of the network. The betweenness of a node is the number of geodesics passing through it. The betweenness implies the importance of an individual location in a network. A node with a high betweenness has great influence over what flows in the network due to its critical location.

3. Friends network analysis

The first thing that we tried to do is analyzing the type of network we have. There are three main network types: random networks, scale free networks and small world networks. As the name indicates, random networks have a random number of nodes and edges which are connected to each other in a random manner. This type of networks has two main characteristics that make them different from other types of networks. A random network has a small path length and a small clustering coefficient [3].

Many real-world networks are categorized as a second network type, named scale-free networks. It is a specific kind of complex networks that is characterized by a power-law distribution of a node's degree. In scale-free networks, some nodes act as highly connected hubs (high degree), although most nodes are of low degree. The small world network is a network in which every node can be reached from every other by a small number of steps. This network has a small path length and a high clustering coefficient [3]. To categorize our network we performed some analysis on the degree of a node and the number of nodes of that degree. The result is shown in Fig. 4 (at the top).

When calculating the proposed network parameters, as shown in Table 1, we noticed that the characteristic path length of some nodes is 0. This problem occurred because more than 50% of the nodes had an out degree of 0. This occurred because network expansion through the friend's list was terminated after a few links; to avoid infinite search for friends and loops. To solve this, we added 0 to 5 randomly generated outer edges to these nodes to connect them to other nodes in the network.

We had two sets of results the one for the original network and another to the added edges network. The new graph of the degree vs. number of nodes is shown in Fig. 4 (at the bottom). As noticed, the

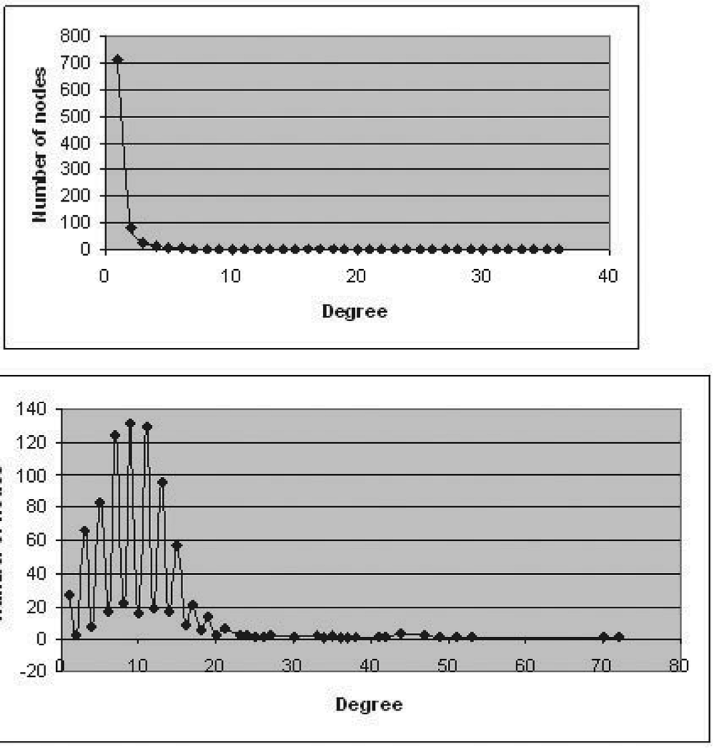


Fig. 4. Degree vs. number of nodes (original network and added edges network).

change happened in the area where the nodes had a small degree. The result of computing the distribution sequence, characteristic path length, clustering coefficient is shown in Table 1.

Previously in this section we showed the degree vs. number of nodes results, which are part of the centrality measures. Another centrality measure is the closeness. For the closeness we gave the nodes that had no close nodes a high value (e.g., 100,000). The results of computing the closeness for both the original network and the network after adding the random edges are shown in Fig. 5. The results of computing betweenness for both the original network and the network after adding the random edges are shown in Fig. 6. The results of closeness and betweenness are shown in Table 2.

After calculating all the parameters in friends network that we tested, we concluded that the friends network is a scale free network like the Internet. We also found that as the number of edges increases in the network the clustering coefficient and the average betweenness also increases while the closeness decreases. As can be seen in Fig. 7 (at the top) when we combine the two resulted graphs of closeness C1 and C2 we find that the only nodes that get affected by these changes are the nodes that are near the average. This is also the case for betweenness as shown in Fig. 7 (at the bottom).

4. Location based services – LBS

In this age of significant telecommunication competition, location based services (LBS) [20] open new horizons for cellular operator for provisioning of innovative lucrative value added services. LBS are a growing technology field that focuses on providing GIS and spatial information via mobile and field

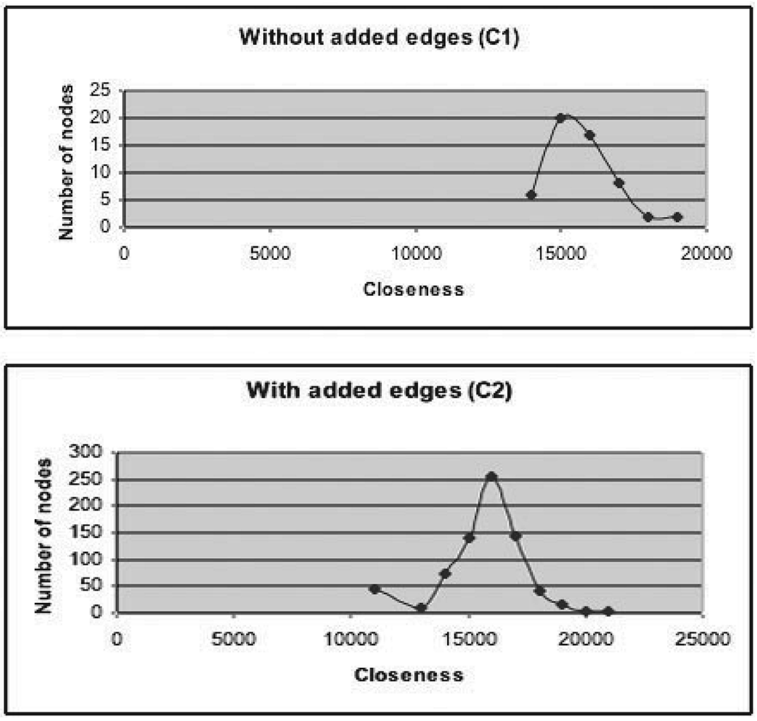


Fig. 5. Closeness vs. number of nodes (original network and added edges network).

units. Those services provided to the subscriber based on their current geographical location. Customer's position can be known by user entry or a GPS receiver. The GPS receiver can either be embedded in the mobile phone or a separated handheld device. Most often, user's location can be determined using a radiolocation function built into the cell network or handset. This uses triangulation between the known geographic coordinates of the base stations through which the communication takes place. Examples of LBS might be determining the nearest point of interest to the customer. For example this can be the nearest restaurant, petrol station, shopping mall, etc.

As an application of social networks, we focused on LBS that provides personalized services to the subscriber based on their current position. Furthermore, they employ accurate, real-time positioning to connect users to nearby points of interest It advises them of current conditions such as traffic and weather, or provide routing and tracking information, all via wireless devices. LBS can also be used to identify the location of a caller by using various positioning determination technologies.

For example, emergency number uses location based services for identifying the location of a caller. Mobile companies already maintain a subscriber database listing every assigned telephone number, the subscriber's name, address and billing information. The telephone system already identifies the telephone number for every call placed, in order to properly bill the subscriber each month. This is done through Automatic Number Identification (ANI). All user details are stored in a Master Street Address Guide (MSAG) database. This database cross-references every assigned telephone number, subscriber's address and the block number ranges for every street, in every jurisdiction served by the telephone company.

LBS traffic is protected from power failures and other system problems. So, whenever telephone company central office switch receives a call regarding position determination such as emergency call

Table 3
Year

Column	Type	Constraint
Year	Integer	Not null
Zodiac	Varchar2(20)	Not null

Table 4
Zodiac

Column	Type	Constraint
Zodiac	Varchar2(20)	Not null
Description	Varchar2(100)	Not null

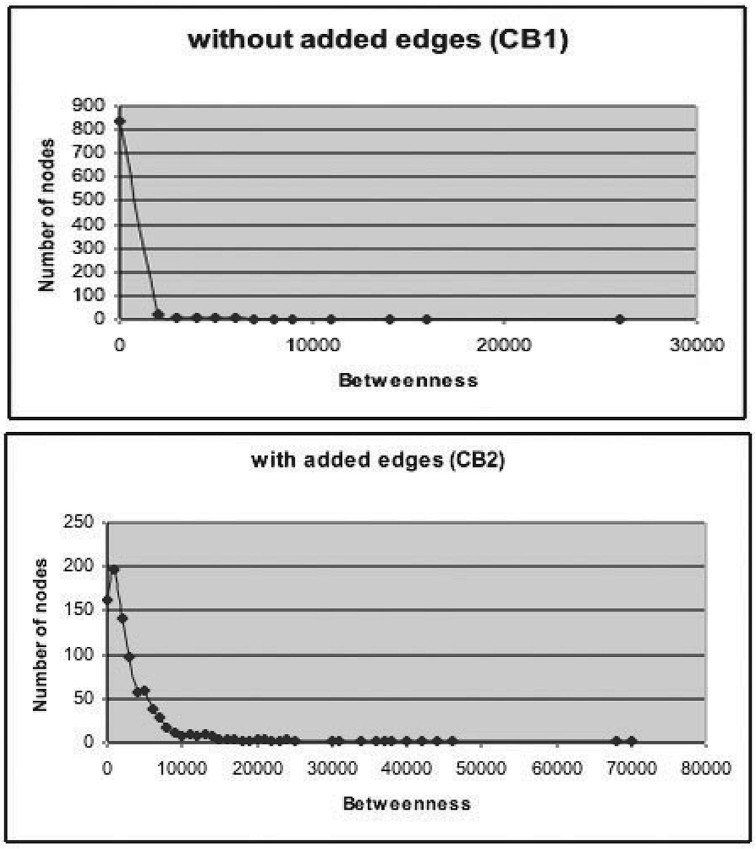
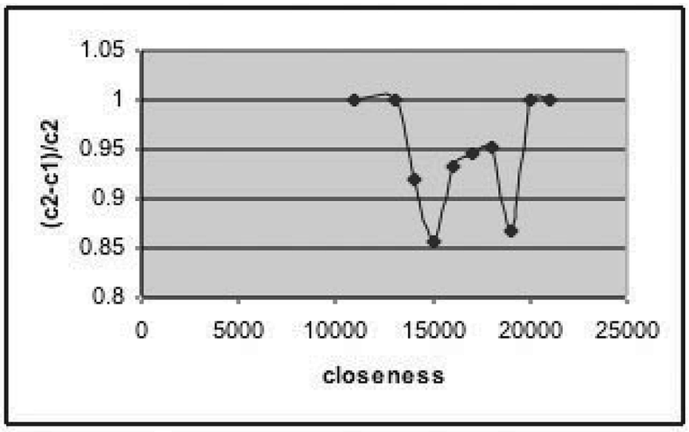


Fig. 6. Betweenness vs. number of nodes (original network and added edges network).

then it is routed to the LBS networks. The ANI (telephone number) information is decoded through a subscriber database to obtain the caller's address and other information. Next, the call is processed; sometimes simultaneously through the MSAG to obtain the ID code of the agency that should handle the call. The LBS network then routes the voice and ANI/ALI (automatic number information and automatic location Information) information to the correct agency.

The ANI/ALI information is displayed when the call-taker answers and, at some agencies, the call information is printed out when the call is completed.



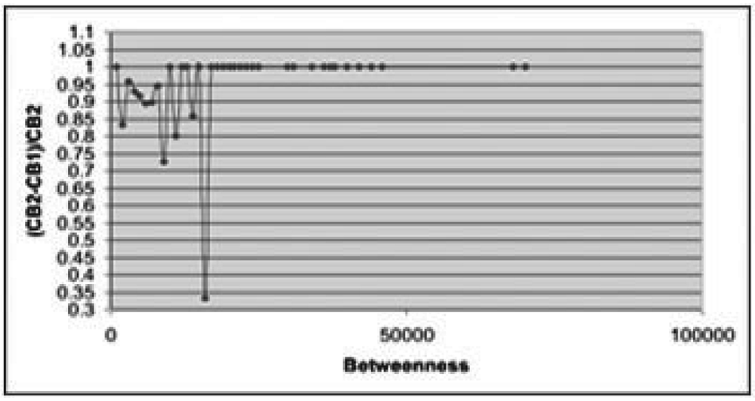


Fig. 7. Closeness and betweenness.

Examples of location based services includes:

– Destination guides with maps, directions
– Location-Based Traffic and weather Alerts
– Wireless advertising and electronic coupons
– Movie, theatre and restaurant location and booking
– Store locating applications
– Emerging buddy, child, or car finders.
– Personal Messaging (Live Chat with Friends)
– Information Services (News, Stocks, Sports)

5. LBS applications

We developed two Pocket PC applications, “Horoscope” and “Find Friend”. In this section we introduce the user interface of each application. We also outline Wingspan scenarios that we have implemented to handle the messaging between the user interface application and the database. We conclude with table structures of each application. Front-end applications have been developed using

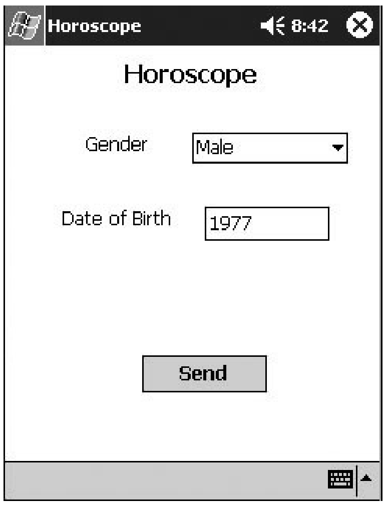


Fig. 8. Horoscope application user interface.

Microsoft Visual Studio VB.NET 2003 with Microsoft .NET Compact Framework 1.0. LBS scenarios have been designed using Wingspan service creation environment, and Oracle Database has been installed for the application backend. In order to implement a short message service the following requirements should be available:

– An account with a mobile service provider (e.g., MTC or Wataniya in Kuwait).
– Service creation environment (SCE) which will is provided by the service provider (e.g., Wingspan service from Wataniyah).
– GSM account for sending SMS.

Wingspan service creation environment (or scenario editor) is made up of different predefined GUI functional blocks (e.g. SMS and MMS). By connecting these blocks to each others we are can build a scenario that does one thing when a user makes a call requesting a service. To use the SCE:

1. Every scenario must have START block and it is the first block that is run in the scenario.
2. Every scenario must have and END block and it marks the end of the scenario.
3. There should be a call block that receives a call if the services that it provides is calling service or an SMS block if the services it provides an SMS service (Fax, FTP, DLL, EMAIL and TCP block).
4. There should be an assign block if the user sends an SMS that contains a word or multiple words.
5. There should be an Assign block to save the message sent by the user.
6. There should be a database block if the service requires storing or retrieving records.
7. After the scenario is created it should be compiled to make sure it contains no error.
8. To run the scenario it should be given to the operator in charge of the Wingspan engine to do what is required for the scenario.

These are the basic blocks that mostly used to provide a service to a mobile user.

5.1. HOROSCOPE application

To test the service creation environment (Wingspan) provided by the mobile service provider (Wataniyah), we developed a small application called “Horoscope”. Given the gender and the birth

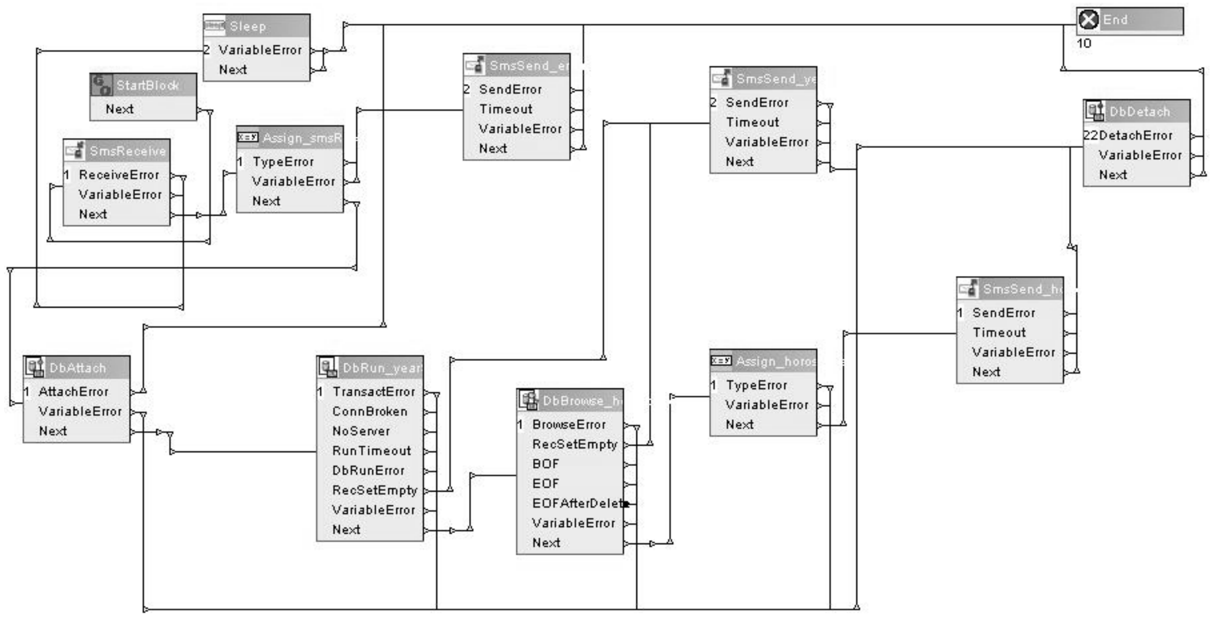


Fig. 9. Horoscope scenario.

date of the user, it returns an SMS with the users horoscope. After installing Horoscope.cab, the application can be started on the Pocket PC as follows: tap Start, tap Programs, and tap Horoscope. Figure 8 shows the main interface of the program.

In order to use the program, users should select their gender from the dropdown list (Male/Female) and enter their birth year manually. Calendar device control is still not available for smart device applications on VB.NET, so the user has to enter his/her birth date manually. Once the user enters the required information, he should click on send button. An SMS will be sent to the service provider with the user details. The “Horoscope” scenario handles the SMS by querying the database and fetches the zodiac, which matches the entered birth year. “Horoscope” scenario sends the result back to the user in a form of an SMS.

The scenario of answering a query for this application is described next. The user sends two pieces of information (gender and year of birth) to the service provider to request the horoscope of his year of birth according to the Chinese calendar. The Chinese calendar starts from year 1912 and it goes on until year 2007 and repeats itself. The user can enter a birth year from 1912–2007 to get the horoscope otherwise he/she will get a message saying that he should enter the message in this range. We have a database called “Users” and two tables one called “Zodiac” which contains the list of horoscopes and the other is called “Year” which contains the year and the Zodiac for that year. Figure 9, shows what building blocks of any Horoscope scenario. The descriptions of the blocks in “Horoscope” scenario are as follows:

1. Start block is the stating block of the scenario and it does nothing more than setting the begging of the scenario.
2. SmsReceive block is the block responsible for receiving SMS from the client or user requesting a service that this number (short code) is providing. We have a block called sleep which will be called in case of an SMS receive error. What this sleep block does is wait few seconds (15000 s)

to end the session in this scenario. Note that if we don't set this sleep block then the system might report an error before the SMS is received.

3. Assign block that is used to save:
   (a) The phones of the user so we can send back an SMS or call him/her back. We saved the phone number as usermobile.
   (b) Any message(s) the user might send to this number, in our case is the gender and the year of birth (between 1912–2007).

   We have an error-trapping block in case of a variable error or type error. If anything goes wrong at this stage it will send back an SMS telling the user that something is wrong and calls the end block to end this session.
4. We call DBAttach block to connect to a database in this case is called "Users".
5. We call DBRun block to run an SQL query. In this block we run this SQL statement to retrieve the horoscope of this birth year. The SQL Statement is:
   "SELECT z.Description, z.Zodiac FROM ZODIAC z, YEAR y WHERE z.Zodiac = y.Zodiac AND y.YEAR = ' + @birthyear"
   We have set error trapping in case an error occurs.
6. We call DBBrowse block to select the record(s) returned. In this block it will have only one record if everything is goes well, otherwise a message is sent back to the user.
7. We call the Assign block to save the field(s) of the record(s) returned by the SQL statement in step 5 which it the your horoscope.
8. We call SMSSend block to send the message back to the user.
9. We Call DBDeattach to close the connection to the database.
10. We call End block to end this session.
11. Build and compile the scenario.
12. If no error give to the operator in charge at the Wataniya to deploy it.
13. The phones of the user so we can send back an SMS or call him/her back. We saved the phone number as usermobile.
14. Any message(s) the user might send to this number, in our case is the gender and the year of birth (between 1912–2007).

The Horoscope application backend consists of two tables. Table "Year"; which contains all the valid years associated with its corresponding zodiac. The other table "Zodiac" contains the list of zodiacs with their associated description. The following tables illustrate the structure of the tables used in the "Horoscope" application.

5.2. Find friend application

The second application we developed was an application called "Find Friend". Every user has to register to the system with a name and specifies the names of the other users in the system that he/she considers as friends. The system keeps tracking the locations of the users. Whenever the user pings the system, it records its new location and returns as an SMS the list of all his/her friends that are in the vicinity of the user. After installing FindFriend.cab, the application can be started on the Pocket PC as follows: tap Start, tap Programs, and tap Find Friend. Figure 10 shows the main interface of the program.

In order to use the program, user should select his area from the dropdown list which contains the list of all available areas in Kuwait and click on Send. Next, an SMS will be sent to the service provider.

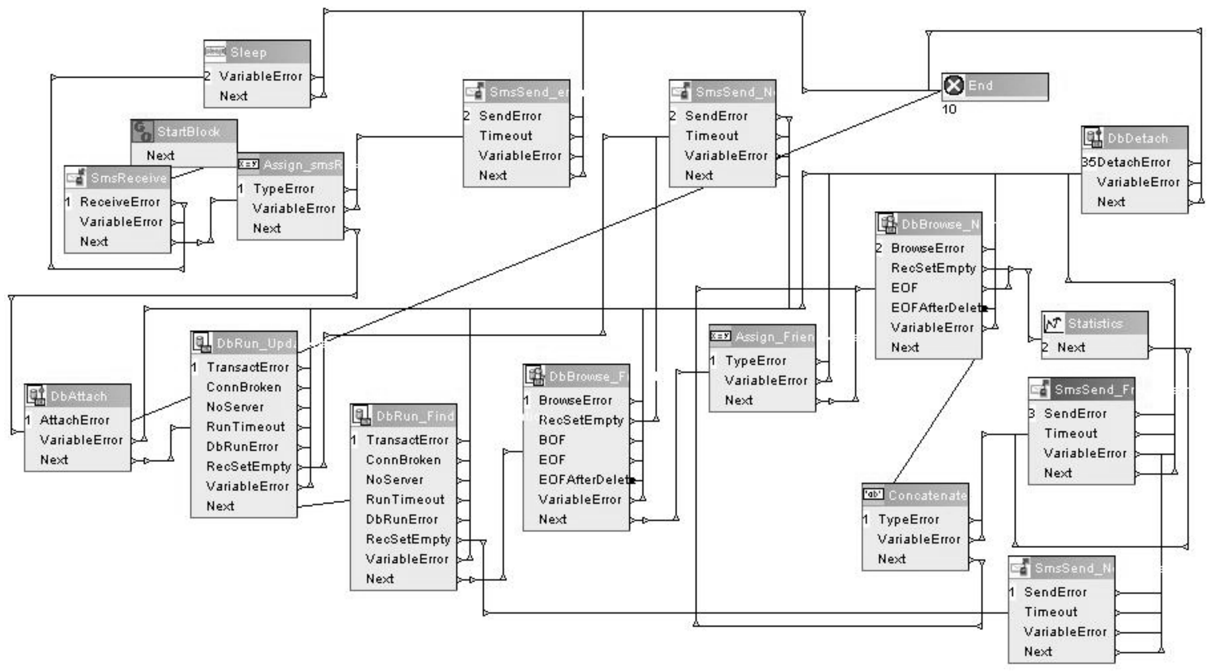


Fig. 10. Find friend application user interface.

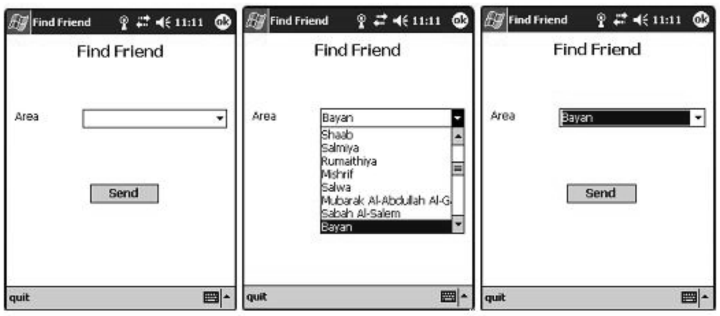


Fig. 11. Find friend scenario.

The “Find Friend” scenario handles the SMS by querying the database and fetches the list of available buddies that are available within the same area. “Find Friend” scenario sends the result back to the user in forms of SMS.

The scenario of answering a query that finds the friends of the user in the same area that he/she is in, is described next. The user sends an SMS to the service provider with the name of the area he is in. If there are friends in the same area SMS returned back with the names of you friends otherwise a message saying that no friends found in this area. To make sure that there are no misspells of area names we developed an application that allows the user to choose from drop down list. This list includes all the areas in Kuwait.

The first few blocks of this part is the same as the Horoscope part described above. We have a database called “Users” and two tables one called “Location” which saves the location of the SMS sender and the other is called “Friend” which has the phone of the SMS sender number and his/her friends’ numbers. Figure 11, shows what building blocks of any Find Friend scenario. Each user enters in the database his/her mobile number, name, the area they are in and their friends cell phone numbers. The following are the descriptions of the blocks in “Find Friend” scenario:

1. Start block is the stating block of the scenario and it does nothing more than setting the begging of the scenario.
2. SmsReceive block is the block responsible for receiving SMS from the client or user requesting a service that this number (short code) is providing. We have a block called sleep which will be called in case of an SMS receive error. What this sleep block does is wait few seconds (15000 s) to end the session in this scenario. If we don’t set this sleep block then the system might report an error before the SMS is received.
3. Assign block that is used to save:

   (a) The phones of the user so we can send back an SMS or call him/her back. We saved the phone number in a variable called usermobile.
   (b) Any message(s) the user might send to this number, in our case is the area he is in (Bayan). We saved the message in a variable called userarea.

   We have an error-trapping block in case of a variable error or type error. If anything goes wrong at this stage it will send back an SMS telling the user that Something is wrong and calls the end block to end this session.
4. We call DBAttach block to connect to a database in this case is called “Users”.
5. We call DBRun block to run an SQL query. In this block we run this SQL statement to update the location table for this user. The SQL Statement is:
   “UPDATE Location SET Area = upper(”’ + @userarea + ”’) WHERE ID = ’ + @usermobile”.
   We have set error trapping in case an error occurs.
6. We call DBRun block to run an SQL query. In this block we run this SQL statement to retrieve my friends in friends table for this user. The SQL Statement is:
   “SELECT l.Name, l.area, f.FriendMobile FROM Friends f, Location l WHERE f.FriendMobile = l.ID AND f.MyMobile = ”’ + @usermobile + ”’ AND upper(l.area) = upper(”’ + @userarea + ”’)”.
   We have set error trapping in case an error occurs.
7. We call DBBrowse block to select the first record returned. In this block we may have more than one record since every user may have more friends that may visit the same area at the same time. We cannot do this step at one time because the SCE does not have the ability to select all record at once so we have to repeat this step using the concatenate block to accomplish this step. If something goes wrong at this point an error message will be sent to the user.
8. We call the Assign block to save the first record returned by the SQL statement in step 6.
9. We call DBBrowse again to get the next record that contains the next friend.
10. We call Concatenate to save this friend’s name.
11. We call 8 again until there are no more records in the returned records of the select statement in step 6.
12. We call SMSSend block to send the message back to the user.
13. We Call DBDeattach to close the connection to the database.
14. We call End block to end this session
15. Build and compile the scenario

Table 5
Location

Column	Type	Constraint
ID	Number	Not null
Area	Varchar2(20)	Not null
Name	Varcha2(30)	Not null

Table 6
Friends

Column	Type	Constraint
MyMobile	Number	Not null
FriendMobile	Number	Not null

16. If no error the scenario is given to the operator in charge at the Wataniya to deploy it.
17. The operator should give you a number (short code) that you can use to send the SMS
18. Send an SMS like: "Bayan" to this number and an SMS will be sent back with friends names in this area otherwise "there are no friends in this area" will be sent.

The "Find Friend" application backend consists of two tables. Table "Location" which contains the mobile numbers of all buddies associated with the area, which they exist in as well as the name of the buddies. The other table "Friends" associates the mobile number of the user with the mobile number(s) of his buddies. The relation between tables has been created to fetch only the list of friend buddies for a specific person. Tables 5 and 6 illustrate the structure of the tables used in the "Horoscope" application.

6. Find friend application enhancement

To make our Find Friend application more flexible and avoid using a specific service creation environment and service provider, we developed a second version of it. In the new application, users can use their Java, GPS internet enabled handheld devices to send their current location (longitude-latitude coordinates) through any cellular network provider. The current location will be processed by either a hosted service in the mobile operator or a third party company. Then it sends a response back to the customer through SMS containing a list of the nearest desired friends. Figure 12 shows the structure and building blocks of our new Find Friend application. Following are the hardware requirements to implement the new application:

- A J2me-enabled, Bluetooth, GPRS or EDGE mobile phone to send queries and receive SMS response.
- A Bluetooth GPS receiver to get the current user location.
- GPRS or EDGE connection with the mobile operator.
- A PC/Server with minimum 2GB of ram with either built-in Bluetooth support or external Bluetooth dongle.

The required tools that will be used in the design and implementation of the new application are:

- Microsoft windows XP SP2 with the latest Bluetooth stack.For the Bluetooth stack both Microsoft and win.
- Oracle Database Server 9i or 10g with Spatial support
- Any Java application server to host the admin pages. This can be Oracle Application Server, Sun application server, or BEA web logic.

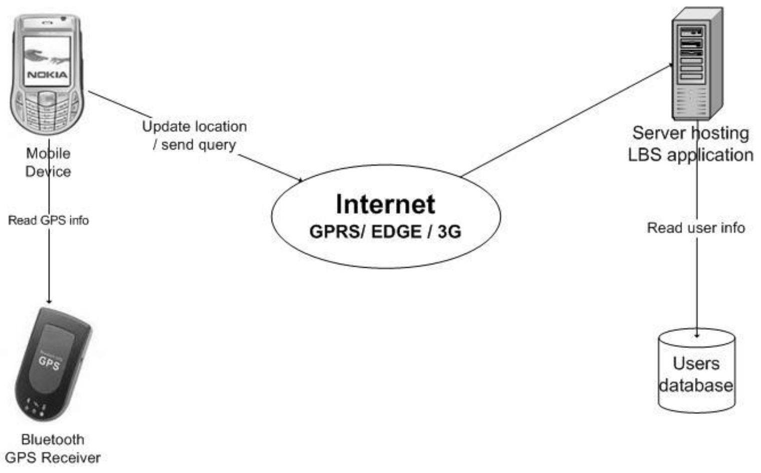
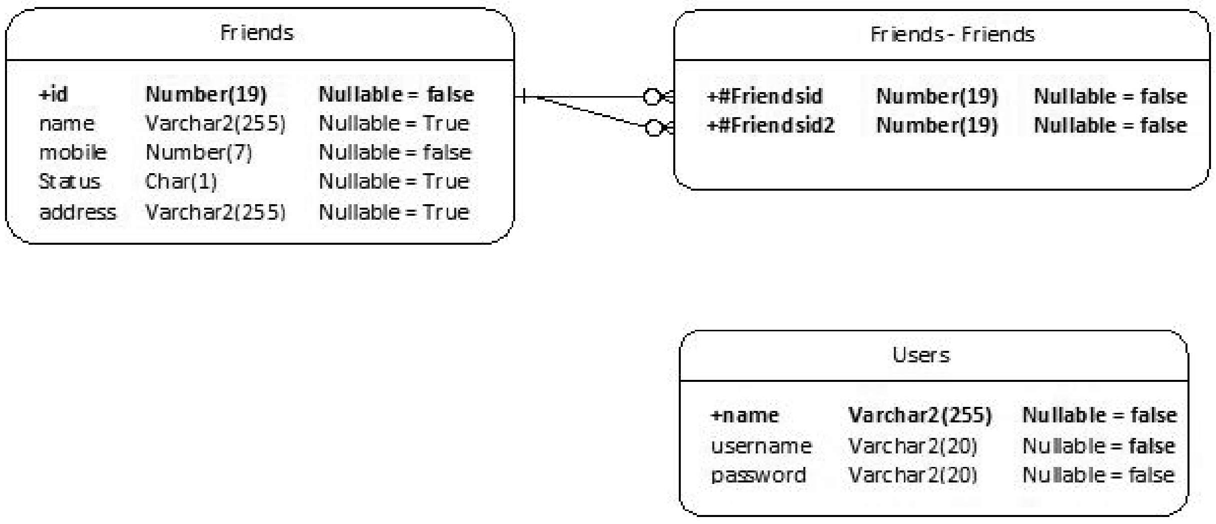


Fig. 12. New Find Friend application structure.



Friends

+id	**Number(19)**	**Nullable = false**
name	Varchar2(255)	Nullable = True
mobile	Number(7)	Nullable = false
Status	Char(1)	Nullable = True
address	Varchar2(255)	Nullable = True

Friends - Friends

+#Friendsid	**Number(19)**	**Nullable = false**
+#Friendsid2	**Number(19)**	**Nullable = false**

Users

+name	**Varchar2(255)**	**Nullable = false**
username	Varchar2(20)	Nullable = false
password	Varchar2(20)	Nullable = false

Fig. 13. ER model of the new Find Friend application.

– NetBeans 5.0 IDE with the mobility pack 5.0.
– Sun Java wireless toolkit 2.2.
– Avetana Java Bluetooth API for windows.
– Sun Java Studio creator v2.1.

The new “Find Friend” application backend consists of three tables. Table “Friends” contains the customer information. The next table “Friends_Friends” contains the relation between the customer and his friends. The last table “Users”, lists all the admin users of the system. Figure 13 illustrates the ER model of the database used in this application.

The new application consisted of two major parts: the administration part, and the user part. They will be explained in the next two subsections, respectively.

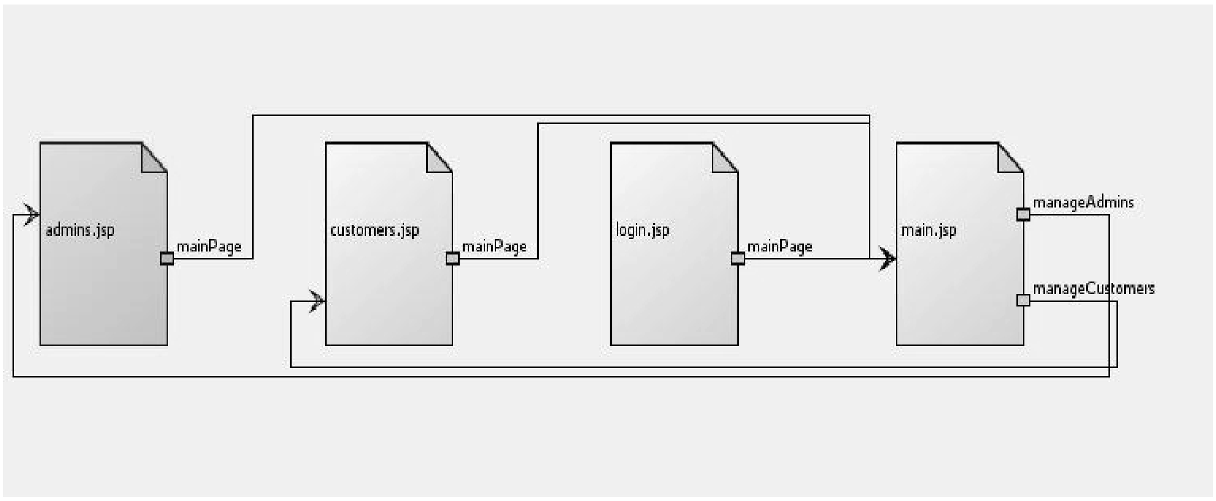


Fig. 14. Page navigation diagram of the admin part.

6.1. *Administration part*

In the administration part, a user with admin privilege can maintain the data resides in the databases about users and their locations. The administration part is a J2EE website built by Sun java studio creator using the latest java technology Java Server Faces JSF. The main admin functionalities are as follows:

1. Customers managements:
   - Add new user: adding a new user to the list of existing ones.
   - Delete an existing user: The site administrator should be able to delete any user that is no longer available.
   - View the set of users already defined in the system.
   - Update the status of an existing user (active, suspended).
2. Administrators managements:
   (a) Add new admin.
   (b) Delete an admin
   (c) View all administrators.
   (d) Update and admin information and password.

The LBS admin web applications consists of the following web pages:

1. Main.jsp: is the main page that includes links to other application pages.
2. Login.jsp: the login page of the application, where admin users can login to the system.
3. Admins.jsp: administrators management page, where admin user can add, edit, or delete admin users.
4. Customers.jsp: LBS application customer management page. Admin users here can modify, add or remove customer.

The page navigation and flow of the administration part of the system is shown in Fig. 14.

6.2. User part

The user part is related to the customer interactions with the system. The user part of our LBS is a mixture of a server application, J2EE and client application, j2me technologies. The main functionality is to send a query through SMS using Java-Enabled, web services device, and receive a response SMS about the nearest desired friend. User's can send a spatial nearest neighbor queries to find a list of friends that are nearest to the current customer location. This position is expressed in the international World Geodetic System, WGS84, and datum format. The customer location is being updated by connecting to GPS device over Bluetooth and using Web Services technology over mobile phones to update the location in the database server. The main user functionalities are as follows:

1. Server Application: J2EE web services are used to access and update the LBS users database. A web service is a standardized way of letting applications exchange data with other applications using the Internet as a communication media. The well-defined structure of a web service provides an easy way of sharing data between organizations and/or end point users without the need to adapt data or information for a specified usage.
   Compared to the HTTP/HTML based WWW a web service is providing a communication link at a computer to computer level instead of a text based human to computer interaction. Web service architecture is designed for dynamic application-to-application interactions. The standardized protocol stack allows easy adaptation of data flows between applications regardless of platforms, systems and implementations. In the LBS server application, a single web service is implemented, LBSWS. This web service provide two methods: – updateLocation: update the user location read from the GPS receiver in the LBS database, and – getNearestFriends: gets a list of the nearest friends on the customer.
2. Client Application: the client application is a J2ME Midlet application for mobile devices. The application uses the latest J2ME APIs and technology such as J2ME Wireless Messaging API (WMA), and J2ME Web Services APIs (WSA) and Bluetooth API. The Wireless Messaging API (WMA) is an optional package for J2ME that provides platform-independent access to wireless communication resources like Short Message Service (SMS).
   The J2ME Web Services API (WSA) extends J2ME to support web services in a Java environment. The WSA implements functionality for XML-parsing (JAXP package) and remote services needed to use web services (JAX-PRC package). The packages in WSA are optional to maintain scalability for the J2ME developer. Both the JAXP and the JAX-RPC are used in web service client implementation. In addition, we used J2ME Bluetooth APIs (JABWT). The J2ME APIs to add the Bluetooth connectivity function for J2ME applications. JABWT an optional package that can be used with any J2ME configuration or profile. The LBS client application contains the following three main packages:

   (a) Lbs.gps.gps: contains the gps Midlet, which keep connection to the GPS receiver and read GPS information and transfer it to the LBS database using the web services package.
   (b) Lbs.webservices: contains the necessary J2me web services classes and interfaces to connect to the LBS server application web service.
   (c) Lbs.sms: contains two functionality, a) smsSender: to send friend finding request by SMS to the LBS server, and b) smsReciever: is a simulation for the server SMS gateway, which receives SMS messages from customers and send the result back.

7. Conclusions and future work

This paper described an analysis on the friends network. This analysis is different due to the use of the available technology (i.e., internet) as a communication channel between the members of the network. The method presented in this paper can be applied to similar networks such as telecommunication networks.

From our analysis, we concluded that the friends network is a scale free network like the Internet. We also found that as the number of edges increases in the network the clustering coefficient and the average betweenness also increases while the closeness decreases.

We developed some location based services applications (e.g., Find Friend) to show how to create an actual friends network. The applications use the position information about a user's particular location to provide relevant information and services. We further enhanced our applications with the support of GPS (typically latitude and longitude.) This lead to the automation of the acquisition of a user's location using GPS devices, GPS enabled cell phones, location services from various mobile operators, positioning technologies with WiFi, and other wireless networks.

With automated acquisition of user location, LBS scenarios are becoming more compelling and useful. LBS solutions in their simplest form can include providing relevant maps, driving directions, proximity searches for Points of Interest (POI) such as a nearby ATM, and more advanced enterprise scenarios such as intelligent dispatch, field service and geo-fencing. These solutions often involve multiple application types including web based or client/server and mobile applications providing LBS on Pocket PC, Tablet PC and other mobile devices.

We believe that these services will create new markets and new revenue opportunities for device manufacturers, wireless providers, and application developers.

Future work will involve deploying the Find Friend application on a larger scale. We will monitor how the friends network will evolve and analyze its characteristics.

Acknowledgements

This research was funded by Research Administration at Kuwait University (Project No EO 08/06).

References

[1] R. Agrawal, S. Dar and H.V. Jagadish, Direct transitive closure algorithms: design and performance evaluation, *ACM Trans Database Syst* **15**(3) (1990), 427–458.
[2] R. Agrawal and H. Jagadish, Algorithms for Searching Massive Graphs, *IEEE Transactions on Knowledge and Data Engineering* **6**(2) (1994), 225–238.
[3] R. Albert and A.L. Barabasi, Statistical mechanics of complex networks, *Reviews of Modern Physics* **74**(1).
[4] O. Bohl, S. Manouchehri and U. Winand, Mobile information systems for the private everyday life, *Mob Inf Syst* **3**(3,4) (2007), 135–152.
[5] J. Breslin and S. Decker, The Future of Social Networks on the Internet: The Need for Semantics, *IEEE Internet Computing* **11**(6) (2007), 86–90.
[6] S.C., K.M. and S.M., A Road Network Embedding Technique for k-Nearest Neighbor Search in Moving Object Databases, ACMGIS, McLean, VA, USA.
[7] R.W. Floyd, ACM Algorithm 97: Shortest Path, *Communications of the ACM* **5**(6) (1962), 345.
[8] J. Goh and D. Taniar, Mining Frequency Pattern from Mobile Users, in: Proceedings of the 8th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Part III, *Lecture Notes in Computer Science* **3215** (2004), 795–801.

[9] J.Y. Goh and D. Taniar, *Mobile Data Mining by Location Dependencies*, in: IDEAL, pp. 225–231.
[10] S.R. Gulliver, G. Ghinea, M. Patel and T. Serif, A context-aware Tour Guide: User implications, *Mob Inf Syst* **3**(2) (2007), 71–88.
[11] M.A.W. Houtsma, P.M.G. Apers and S. Ceri, *Distributed Transitive Closure Computations: The Disconnection Set Approach*, VLDB '90: Proceedings of the 16th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 335–346.
[12] M.A.W. Houtsma, P.M.G. Apers and G.L.V. Schipper, *Data fRagmentation for Parallel Transitive Closure Strategies*, in: Proceedings of the Ninth International Conference on Data Engineering, IEEE Computer Society, Washington, DC, USA, pp. 447–456.
[13] M.A.W. Houtsma, A.N. Wilschut and J. Flokstra, *Implementation and Performance Evaluation of a Parallel Transitive Closure Algorithm on PRISMA/DB*, in: VLDB '93: Proceedings of the 19th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 206–217.
[14] N. Jing, Y.-W. Huang and E.A. Rundensteiner, *Hierarchical Optimization of Optimal Path Finding for Transportation Applications*, in: CIKM '96: Proceedings of the fifth international conference on Information and knowledge management, ACM, New York, NY, USA, pp. 261–268.
[15] D.B. Johnson, Efficient Algorithms for Shortest Paths in Sparse Networks, *J ACM* **24**(1) (1977), 1–13.
[16] S. Jung and S. Pramanik, A HiTi Graph Model of Topographical Road Maps in Navigation Systems, *Data Engineering, International Conference on* **0** (1996), 76–84.
[17] S. Jung and S. Pramanik, An efficient path computation model for hierarchically structured topographical road maps, in: Knowledge and Data Engineering, *IEEE Transactions on* **14**(5) (2002), 1029–1046.
[18] W. O'Connell, I.T. Ieong, D. Schrader, C. Watson, G. Au, A. Biliris, S. Choo, P. Colin, G. Linderman, E. Panagos, J. Wang, and T. Walter, A Teradata content-based multimedia object manager for massively parallel architectures, *SIGMOD Rec* **25**(2) (1996), 68–78.
[19] G. Pilato, A. Augello, G. Vassallo, and S. Gaglio, EHeBby: An evocative humorist chat-bot, *Mob Inf Syst* **4**(3) (2008), 165–181.
[20] I. Priggouris, D. Spanoudakis, M. Spanoudakis and S. Hadjiefthymiades, A generic framework for Location-Based Services (LBS) provisioning, *Mobile Information Systems* **2**(2,3) (206), 111–133.
[21] B. R., On a routing problem, *Quarterly of Applied Mathematics* **16**(1), (1958), 87–90.
[22] S. Shekhar, A. Kohli and M. Coyle, *Path cOmputation Algorithms for Advanced Traveller Information System* (*ATIS*), Proceedings of the Ninth International Conference on Data Engineering, pp. 31–39.
[23] I. Sorkhoh, M. Safar and K. Mahdi, *Classification of Social Networks*, Proceedings of the IADIS (International Association for Development of Information Society) International Conference on WWW/Internet.
[24] D.E. W., A note on two problems in connection with graph theory, *Numerische Mathematik* **1** (1959), 269–271.
[25] H. W., J. N. and R. E., *A Semi-Materialized View Approach for Route Maintenance in Ivhs*, in: Proc. of the 2nd ACM Workshop on Geographic Information Systems, ACM, pp. 144–151.
[26] S. Warshall, A Theorem on Boolean Matrices, *J ACM* **9**(1) (1962), 11–12.
[27] D.J. Watts, *Small Worlds: the Dynamics of Networks between Order and Randomness*, Princeton University Press, 1999.
[28] A.C. Weaver and B B. Morrison, Social Networking, *Computer* **41**(2) (2008), 97–100.
[29] Y. wu Huang, N. Jing, and E.A. Rundensteiner, *Effective Graph Clustering for Path Queries in Digital Map Databases*, in: In Fifth international conference on information and knowledge management (CIKM'96, ACM, pp. 215–222.

Maytham Safar is currently an Associate Professor at the Computer Engineering Department at Kuwait University. He received his Ph.D. degree in Computer Science from the University of Southern California in 2000. He has one book, two book chapters, and over fifty articles and conference/journal. Dr. Safarfls current research interests include social networks, sensor networks, location based services, image retrieval, and geographic information systems. He is an ACM Professional member since 2002, and a Senior Member of IEEE since 2008. He is also a member of IEEE Standards Association, IEEE Computer Society, IEEE Geoscience & Remote Sensing Society, and International Association for Development of the Information Society (IADIS), International Organization for Information Integration and Web-based Applications & Services (@WAS). He is currently participating in several academic initiative programs offered by Sun Microsystems, Oracle, IBM, and Symbian. He managed to advise over seventeen graduate student projects, supervised three Master Theses, and co supervised two doctoral dissertations, and was on the examination committee of thee Master Theses, and two doctoral dissertations. He was granted over eleven research grants from research administration at Kuwait University, and Kuwait Foundation for the Advancement of Sciences (KFAS). He served on over forty-seven conference committees as a reviewer and/or a scientific program committee member such as ICDCS, IEEE ICME, , IADIS WWW/Internet, IEEE AINA, iiWAS, IEEE ICDIM, U-Media, IEEE ICCCN, IEEE CSTST, ICADIWT, WSKS, ACM SAC, and IEEE MMEDIA. He also served as a member of steering committee, organizing committee, publicity committee of over seven conferences. He served as a reviewer for over nineteen journals such as IEEE Transactions on Multimedia Journal, ACM Computing Reviews, JDIM, MTAP, IEEE Transactions on Pattern Analysis and

Machine Intelligence, ACM Multimedia Systems Journal, Computing and Informatics Journal, IJWIS. He also served as a member on the editorial board and committees on over nine journals such as JDIM, IJWIS, IAJIT, JUCS, and IJEC. He serves as an associate editor for JDET journal.

Hussein Sawwan is currently working as a Senior Oracle Database Administrator at Aayan Leasing & Investment Co. His responsibilities include database security, advanced performance tuning, capacity planning, and physical/logical database design in Unix and Linux environments. He received his Masterfls degree in Computer Engineering from Kuwait University in 2006. Husseinfls current research interests include Geographic Information Systems, Spatial Databases, XML Databases, Data Privacy and Security.

Mahmoud Taha is currently a technical support supervisor employee in the Information System Center at the ministry of information in Kuwait. He received his Master Degree in Computer Engineering from Kuwait University in 2006. He has one IEEE conference in Taiwan (SMC2006) in the area of Private Information e-Wallet.

Talal Al-Fadhli is currently a senior programmer in electronic channels department at Kuwait Finance House (KFH) Bank in Kuwait. He is responsible for on-line banking, credit card system, and intranet web based application in Java and Microsoft .Net. He received his Master Degree in Computer Engineering from Kuwait University in 2006. His current research interests include Web services, mobile applications, and integration systems.