Etherthreads: An Infrastructure for Location-based Messages

by
Bradford Lassey

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology
May 20, 2004 [June 2004]

Author . . . . [illegible]
Department of Electrical Engineering and Computer Science
May 20, 2004

Certified by. . . . . [illegible]
Patricia Maes
Associate Professor MIT Media Laboratory
Thesis Supervisor

Accepted by . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Etherthreads: An Infrastructure for Location-based Messages

by
Bradford Lassey

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2004, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

This thesis proposes an infrastructure for location-based services for Bluetooth enabled cellular phones. Specifically, it explores the use of this architecture in a location-based messaging application. A user can send or receive messages based on their current location, thereby increasing the relevance of the messages to the user's current context and activities. The messages are organized into threads, which embody a subject matter, a set of possible senders and a set of anticipated receivers. This is the primary means by which the messages that are filtered, to prevent users from being bombarded with irrelevant messages.

Acknowledgements

I would like to thank my advisor, Prof. Pattie Maes for her support and guidance throughout this project. Her encouragement was essential to see me through the many set backs I encountered.

The many talented people at the MIT Media Lab guided, inspired and pushed me since I worked here as an undergraduate. The Media Lab as a whole provided an environment which encouraged the interaction that helped me so much.

Nokia provided the phones used for this project and many others at the lab.

Finally, I would like to express the indebtedness I feel towards my family. I would like to dedicate this to my parents. To my mother for her strength and to my father for the work ethic he tried to instill in me.

Contents

1. Introduction

The combination of time, location and context can give information to when message delivery will have the most impact. Recently released mobile phones, frequently labeled "smart phones" have the computing power and sensory capabilities to acquire this information and make use of it.

Existing messaging technologies are not sensitive to their users' contexts. Cell phones should not ring in a movie theater, important email should be redirected to voice mail while the user is away and PDAs should display pertinent web content as the user walks down the street. This project does not attempt to solve these limitations of existing technology, but instead offers an alternative messaging technology, which pays particular attention to the location of the user.

Our modern high speed life style does not lend itself to our traditional local community messaging forums. Traditionally these include town hall meetings, conversations at the local barbershop, the neighborhood bar and the ladies' sowing circle. Some of these forums still exist, but to a much lesser extend than society requires. Another, still prominent messaging forum is the newspaper, particularly the op-ed section. Once again, though, society's increasing speed of life almost mandated that the majority of the population substitute TV news for their newspapers, which gives a less localized op-ed forum, which is less closely followed.

This project implements a location based messaging system, which can serve as the vehicle for local community messaging in the 21st century. Using their cell phone, a user can leave or receive messages related to their current location. The two main sub-problems, which this thesis depends on, are the determination of a client's location and the delivery of the messages. To determine location indoors, a Bluetooth location-sensing network was developed. Outdoors, GPS can be used to determine location. A protocol was constructed by which a Bluetooth enabled device, which may or may not have the ability to determine signal strength, may establish a connection to a network of servers over the L2CAP layer of Bluetooth and query for their signal strength. Position can then be determined based on a survey of signal strengths from three or more of those servers.

This thesis is organized as follows.

Chapter 2 describes several usage scenarios for the system. They will demonstrate the ability of the system to serve as a community-messaging forum, a reminder system, a conduit for organizational memories to propagate and a medium for distribution of authoritative information.

Chapter 3 focuses on the implementation of the system; touching on the client and its user interface, the location sensing Bluetooth network, the message servers and their database along with the filtering scheme.

The conclusions drawn from this project are expressed in Chapter 4. This chapter also goes into some depth about selected design decisions. Chapter 5 reviews the previous work done related to this project.

Finally, Chapter 6 describes some extensions and refinements that could be made by anyone interested in continuing this work. The system was made as flexible and extensible as possible. Great care was taken not to handcuff any future developers wishing to expand upon this research.

2. Scenarios

The following are several usage scenarios, which demonstrate the intended capabilities of the system.

2.1 Community Messaging: Lunch

In this scenario, the user is walking down the street in a new city and is looking to get some lunch. Unfortunately, the user knows nothing about the local restaurants. The user tells his or her mobile device that he or she is interested in community messages about restaurants. Now, as the user walks by each restaurant, messages about that restaurant appear on the device's screen. The messages were left by the community on a public thread about restaurants. A public thread is one that anyone can write and anyone can read. The subject matter is restaurant discussions and the context is the intent of finding a good place to eat.

As the user passes by restaurant number one a message is displayed from another user saying that he received terrible service at the restaurant. As the user keeps walking he or she receives several more messages about each restaurant he or she passes. Finally the user passes a restaurant with a message outside praising the chowder. The user likes chowder and decides to grab lunch there. At the end of the meal, the user leaves a message that the chowder is indeed tasty.

This was an example of community messaging . It is important to point out a few things. First, since this is a community process there is no assurance of the quality of the messages. To this end, the messages that the user received are all signed by the senders, although the signature may be anonymous. It is anticipated that the signed messages will carry more weight than the anonymous messages, thus giving some information suggesting quality. In other community discussions users may get to know each other, which would provide even more quality information. This scenario does not demonstrate that well as presumably the user does not know too many people in this city that he or she is new to. Note that the system does make it possible to only exchange messages between groups of users, much like and email list. As such, the user can request messages about restaurants left only by friends.

2.2 Self Reminder: Trash Day

The user must take out the trash every Wednesday morning. The user is forgetful so he or she needs a reminder system. The reminder can be associated with both a time and a place. The time is Wednesday mornings and the place is the front door, presumably to remind them before he or she leaves for work.

Traditional reminder systems can do either effectively, but not both. For instance most digital watches have the ability to set alarms. Thus, the user could set such an alarm to go off at 8:15 every morning, which is their best approximation to when he or she will be stepping out the door. However, the user may still be in the shower at 8:15 or they may already in the car heading to work. In either case the alarm is most likely not effective at all.

To encapsulate the location effectively, one could simply put a note on the front door. However, this has its drawbacks too. Most likely since the user will see that note every morning and not just Wednesday mornings it will loose some effectiveness as a reminder. Also it may not be the most aesthetically pleasing thing in the world.

However, with a personal reminder thread, which is a protected thread where only the user can read and write to, the user can encapsulate both the time and the place that the message would be most effective. Therefore, the user would leave a message at the front door, with a time parameter of Wednesday mornings. As the user approaches the front door on a Wednesday morning, the cell phone in his or her pocket will ring, alerting the user to the message that he or she had left there, and reminding him or her to take out the trash.

It should be noted that the ability to specify a time frame for a message is not implemented in the current version of the system, however it is one obvious extension of the system.

2.3 Organizational Memory: Lab Visit

Many organizations have frequent visitors, either internal or from outside. One such organization is the MIT Media Lab. Twice a year, the Media Lab holds a series of events which consist of talks, meetings and open houses in which sponsors are invited to come and see the research they are sponsoring. Since most sponsors have a long term relationship with the lab, they come every 6 months and are able to see the evolution of different projects. However, often the visiting party changes over time. Also, even with a static visiting party, memories fade.

The Etherthreads system can be helpful in two ways to these visitors. First, to help guide visitors during an event, organizers could scatter messages around the lab which describe each research group and the various projects being presented. The visitor could specify which areas of research they are interested in and the devices would then display the messages most pertinent to their interests, thus effectively directing them to the most interesting projects for them.

Also, as the sponsors explore the lab, they will undoubtedly have thoughts and opinions which are valuable to their organization as a whole. If they were to record these thoughts as they occurred they would leave their memories scattered around the lab. For the individual visitor, these notes would be valuable in constructing a trip report. Being associated with a time and place could also add context, which may aid in future recollection.

The true value of these notes comes when they are viewed as messages to the visitor's colleagues. Both on the day of the event and for future events, the messages can facilitate organizational memory. One visitor from the organization may view the project first and leave a message for his or her colleagues. When subsequent colleagues come by, they will have some context with which to think about the project they are looking at. In addition, when it comes time for the next meeting, they will be able to view past notes, which will allow them to start the conversation off with some knowledge of how a given project affects them. They will also be able to judge progress. Finally, since the messages are protected such that only members of the organization can read and write them, internal, sensitive and proprietary knowledge included in the messages. For instance if a student's project has application to an internal project, that may be noted; without leaking that the project exists to competitors.

This particular scenario could be implemented with a private thread for each company that visits the lab, which is configured to be readable and writable only by the employees of that company.

2.4 Competent Guide: A Tour of Boston

Many tourists visit Boston because of its rich history. However even the best guide books can't begin to share the wealth of knowledge about every important location in such a historically significant city. The Etherthreads system could be used as an alternative.

For instance if the Boston Historical Society were so inclined they could annotate every historically significant location in the city with information. A tourist could then walk the freedom trail and be privy to even more information than your average guide book provides, without the hassle of flipping through pages. Perhaps more exciting would be the ability to get off the beaten trail and explore the many side streets which are also rich with history. The experience could turn into an information treasure hunt.

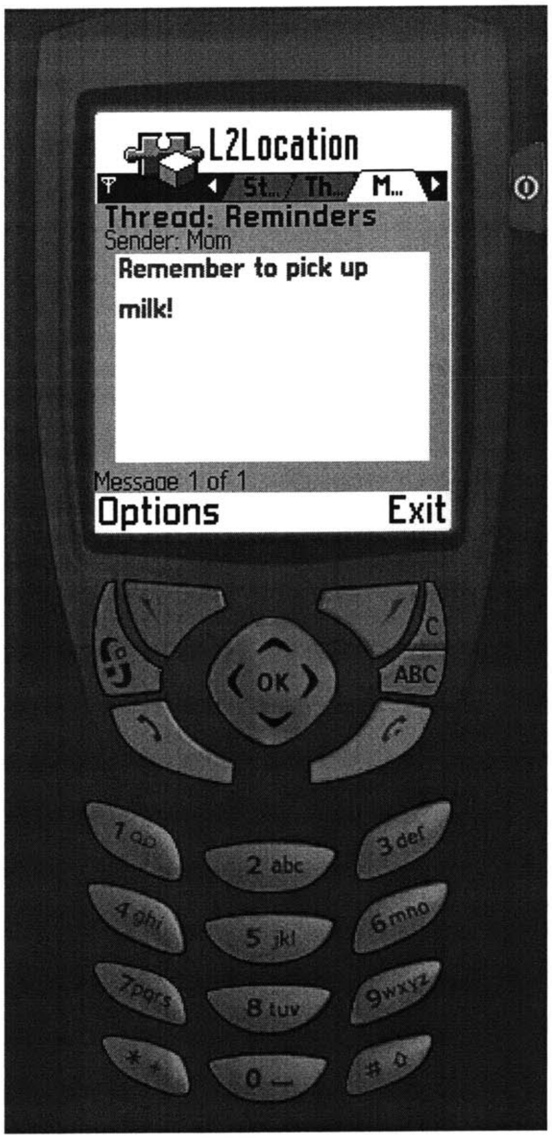
Along the same lines, park rangers could annotate hiking trails with important information, such as safety tips. The system could also be very helpful in guiding a hiker back toward the base camp.

In all of these examples, different threads could be used to convey different specialty interests. In the Boston tour example, threads could be tailored to provide information about architecture, history or the arts.

3. Implementation

3.1 Clients and UI

The client's UI is fairly simple. It is organized in a tabbed view. The user can navigate between tab views by rocking the five-way stick to the left or right. Also, the options menu lists each of the views available (see Figure 5). Selecting any of the views on the options menu list will jump to that view.



Remember to pick up milk!

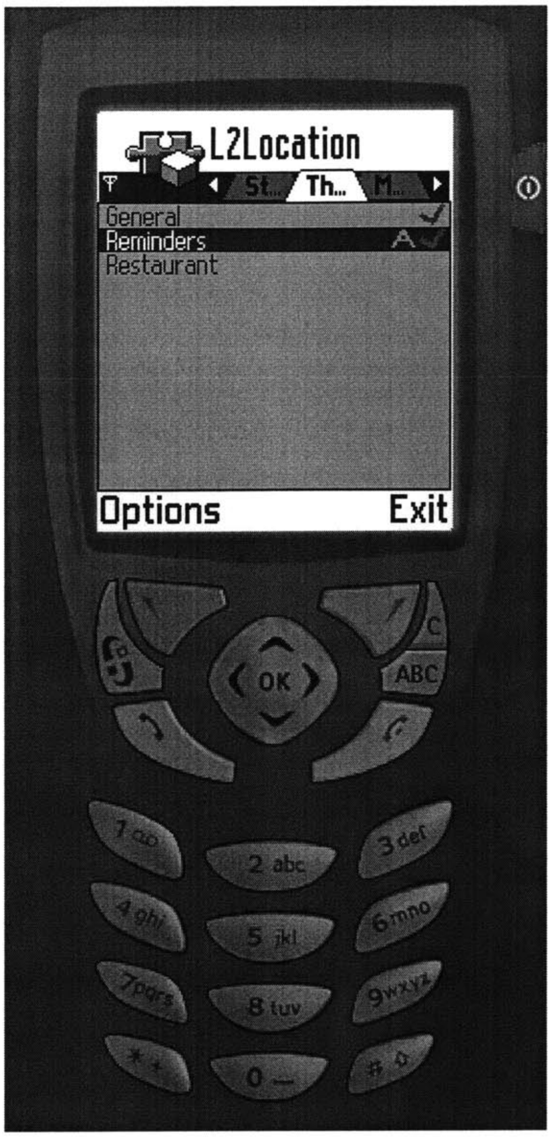
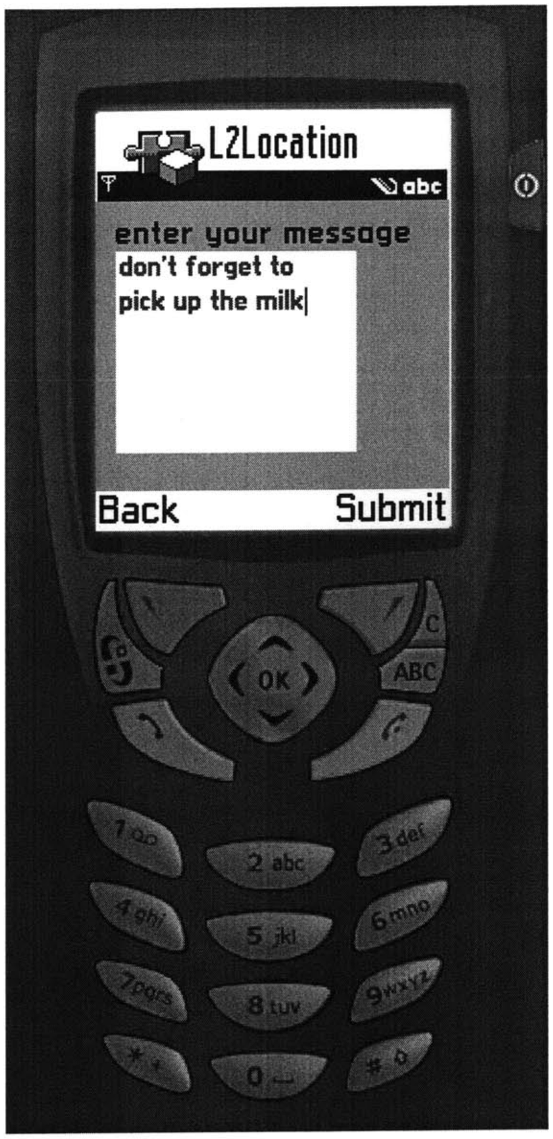


Figure 1 The Message Display View

Figure 2 The Thread Selection View

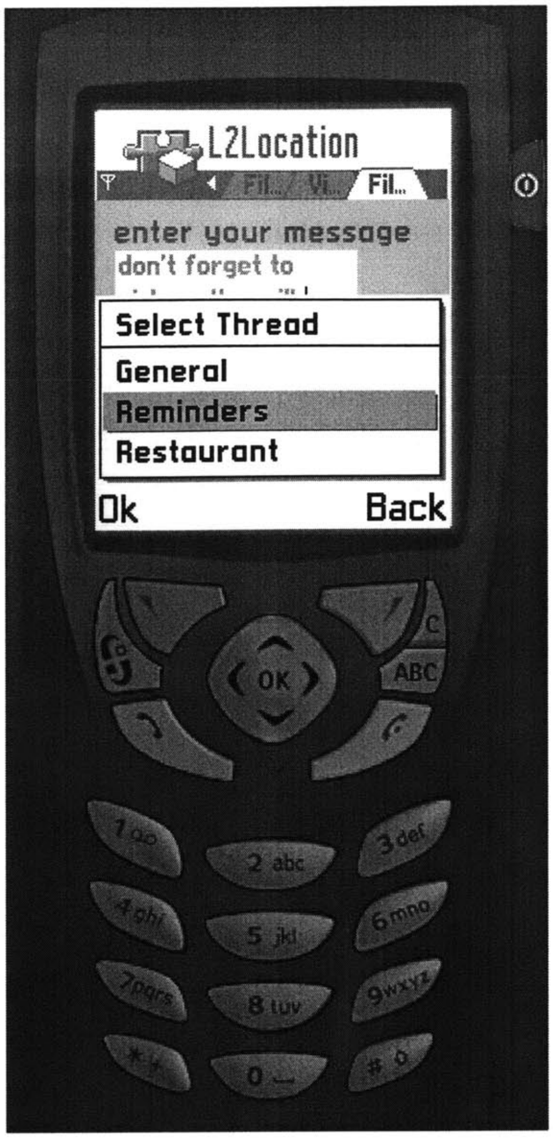
The first tab labeled "Start Up" has a start up screen with which, for debugging purposes, the user guides the client through the start up process. In a "productized" version, the start up process would be automated and this screen could be eliminated.



enter your message

don't forget to
pick up the milk

Figure 3 The Message Creation View.



enter your message

don't forget to

Select Thread

General

Reminders

Restaurant

Figure 4 Thread Selection for Message Creation

The second tab (see Figure 2) displays a list of threads that are available for the user to monitor. The user can scroll up and down the list with the five-way stick. Pressing down on the five-way stick will select and unselect the highlighted thread. A red check is placed next to the selected thread to signify its selection. Additionally, the user can select to be alerted to the existence of a message from that thread. A green 'A' is displayed next to the thread to show that the user has selected to be alerted to any messages from that thread. The alert consists of an audio beep upon receipt of a message. With the typical update rate of the system, this turns into a rapid string of beeps because the beep is played on every update as the message is retrieved repeatedly. Only the threads that the user has access to are displayed.

The third tab (see Figure 1) displays the current message. It is very simple and uncluttered, printing the message in green against a white background with a font that balances readability against the need to accommodate longer messages. At the top it shows the thread that the message was posted to in bold with the author of the message displayed below that. Below the white message display area, there is a line saying which message of a set of available messages are being viewed. The available messages are defined as the top ranked message, plus any message within 2 meters of it. Pressing up and down on the five-way stick scrolls through this set.

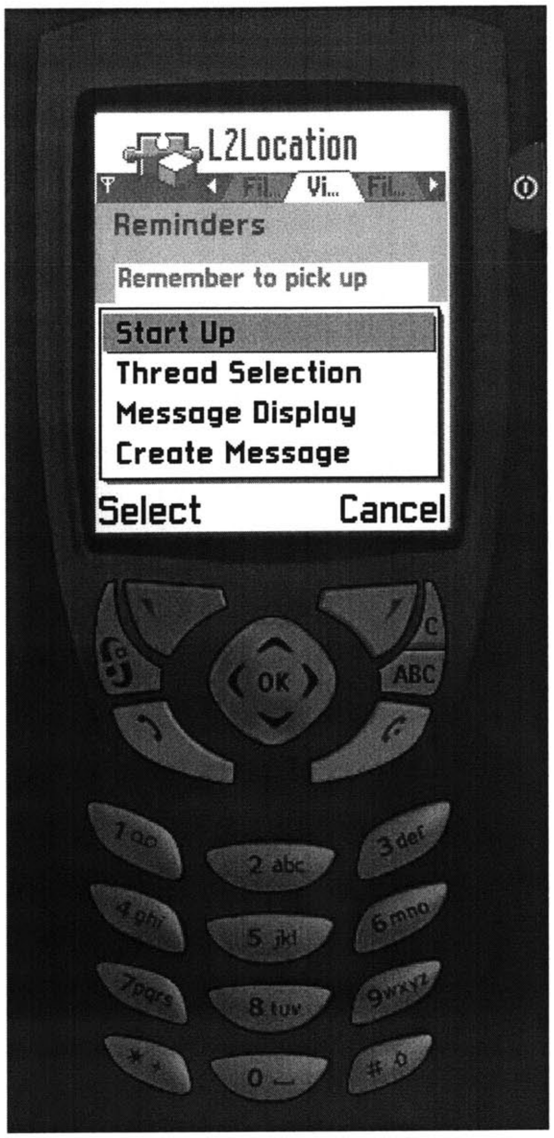


Figure 5 The Options Menu

Finally, the fourth tab (see Figure 3) is the message creation view. There is a text editor, in which the user can place their message. There are two buttons, "back" which brings the user back to the third tab (since the five-way stick is used for editing) and "submit." Upon pressing "submit," a popup list appears (see Figure 4), asking the user which thread they would like to post to. Again the permissions are checked to see if the user has permission to post. Once that is selected a final "your message has been sent" message is displayed.

3.2 Bluetooth Network

The simplest way to get location information is with an existing technology like GPS. Similarly, an existing data network such as GPRS would be the simplest way to get messages to the cell phone. Unfortunately, both these systems are largely limited to outdoor usage due to signal degradation indoors. The system uses exactly that solution for outdoors usage, but also provides an alternative for indoor usage.

The alternative used indoors is a network of Bluetooth enabled servers. These servers listen on an l2cap socket, which is a connection oriented Bluetooth protocol, waiting for a client to open a connection. Once a client opens a connection, it must send its Bluetooth address to the server. A Bluetooth address is similar to an IP address or a Mac address, but identifies a Bluetooth device.

Having the client report its own address to the servers, opens a security hole, but not as large as one might think. While it would be cleaner for the server to obtain the Bluetooth address of the client automatically through the protocol stack (it would reduce the length of the initial hand shake,) it does not compromise security, since the Bluetooth address is obtained in the protocol stack by sending a request to the connecting client device. In addition, in order for this type of address spoofing to work, both the tracked client and the tracking client must be connected to the same set of servers. In this case, they would be within sight of each other, so the value of tracking would be fairly small. For this reason, no attempt is made to secure the tracking system at this level.

It should be noted that in general securing the system is outside the scope of this thesis. The current implementation would allow even a relatively unskilled malicious user access to the messages of others. For example there is no authentication of the user, only the devices Bluetooth address is used to identify a user. This is easily spoofed. There are many other security holes.

Once connected, the server begins responding to simple commands. If the client sends an 'r', 'l', or 't' the server will respond with the RSSI, Link Quality or Total Power Level, respectively. Upon receiving an 'm' it begins processing a message request and a 'c' is the command to create a message. The protocol and the procedure the server follows for requesting and creating a message is described in the next section.

The server was implemented in Java. An API was implemented wrapping the Bluez stack. A similar wrapping already exists in JBluez API, but this does not allow the creation of sockets or streams from the Java side. JBluez was used for inspiration[1]. See Appendix C for the source code to this API. See Appendix B for the server's source code.

3.3 Location Algorithm

The algorithm used to determine location is a weighted average of the location each of the location servers. They are weighted according to the inverse of their RSSI (Received Signal Strength Indication).

$$x = \frac{\sum_{i=0}^{numberofservers} x_i * s_i}{\sum_{i=0}^{numberofservers} s_i} \qquad y = \frac{\sum_{i=0}^{numberofservers} y_i * s_i}{\sum_{i=0}^{numberofservers} s_i}$$

Figure 6 The equations used to determine location, where x_i and y_i are the coordinates of server i and s_i is the RSSI strength measurement of the connection between that server and the cell phone.

3.4 Message Servers

The messages are stored on a central MySQL database. The client polls for the messages, contacting the Bluetooth server with the strongest signal, and issuing the 'm' command to the server. Once the Bluetooth server receives the 'm' command, the server reads the stream until it encounters a new line character ('\n'). The string of characters it reads is formatted such that it can be appended to a SQL query of the form "SELECT message FROM etherthreads.messages." Thus, the string it reads may be of zero length, in which case the SQL server will return a result set of messages with no filtering and in no particular order. It can contain a "WHERE" clause, which would encapsulate the user selecting which threads they want to see. It can also limit the message 'search' to a certain period. It can also contain an "ORDER BY" clause, which will specify a heuristic used to determine which message of the filtered set is selected. The primary purpose of the heuristic is to obtain the message closest to the user. It can also give preference to new messages over older ones. By allowing the client to specify these two clauses, the system pushes that design decision out to them, and creates more flexibility in user preferences. The Bluetooth server issues the SQL query to the central message server. The first message in the result set is then written to the Bluetooth output stream to the device that originally requested it. A null character, '\0' signals the end of the message.

```
+---------+------------------+------+-----+---------+----------------+
| Field   | Type             | Null | Key | Default | Extra          |
+---------+------------------+------+-----+---------+----------------+
| message | varchar(64)      | YES  |     | NULL    |                |
| thread  | varchar(24)      | YES  |     | NULL    |                |
| sender  | varchar(26)      | YES  |     | NULL    |                |
| x       | int(11)          | YES  |     | NULL    |                |
| y       | int(11)          | YES  |     | NULL    |                |
| date    | timestamp(14)    | YES  |     | NULL    |                |
| id      | int(10) unsigned |      | PRI | NULL    | auto_increment |
+---------+------------------+------+-----+---------+----------------+
```

Figure 7 The messages table

The threads are also encapsulated by a SQL table. The table has fields for the thread's name, a user allowed to access the thread and a group allowed to access the thread. It also has an enumeration of the rights that user or group has for that thread. Finally, it has an auto-incremented id field, to allow unambiguous selection of a row in the table. Although not enforced, typically an entry in the user field and an entry in the group field are mutually exclusive. Said another way, usually if there is a value in the user field, the group field is null and visa versa. A separate row exists in the table for each user or group of users given access to a thread. The rights enumeration can have the values 'read', 'write' or 'both.' The client will make a request for the threads available to it in the system, the same way in which it requests messages, except with the command 't.' The system then executes a query such as:

```
select distinct name, rights from threads where user = 'user_name' || group
= 'group1' || group = 'group2'
```

Where group1 and group2 are groups to which the user belongs. This returns a result set, which contains a list of all threads that the user has either read or write privileges on, and what type of privileges the user has on that thread. They system then appends the result set into one common delineated string of the thread names with the character ‘r’, ‘w’ or ‘b’ appended to the front. The appended character signals to the client that the user has ‘read’, ‘write’ or ‘both’ privileges on that thread. The group ‘general’ is appended to everyone’s list with ‘both’ permissions. This is a sort of catchall general discussion thread.

```
+--------+-----------------------------+------+-----+---------+----------------+
| Field  | Type                        | Null | Key | Default | Extra          |
+--------+-----------------------------+------+-----+---------+----------------+
| name   | varchar(20)                 | YES  |     | NULL    |                |
| user   | varchar(20)                 | YES  |     | NULL    |                |
| groups | varchar(20)                 | YES  |     | NULL    |                |
| rights | enum('read','write','both') | YES  |     | NULL    |                |
| id     | int(10) unsigned            |      | PRI | NULL    | auto_increment |
+--------+-----------------------------+------+-----+---------+----------------+
```

Figure 8 The threads table

The groups mentioned above are described in another table, which contains three columns. The first column is the group name; the second is the user name of a member of that group. The third and final column is an auto-incremented id; again, this is used to unambiguously select a given row. Like the thread table, an entry (or row) in the table exists for every member of every group. To produce a list of groups to which a user exists, the following query would be executed:

```
select name from groups where member = 'user_name'
```

This list of the groups to which a user is a member is used to construct the query in the previous paragraph which reveals the threads that user has permissions on.

```
+--------+------------------+------+-----+---------+----------------+
| Field  | Type             | Null | Key | Default | Extra          |
+--------+------------------+------+-----+---------+----------------+
| name   | varchar(20)      | YES  |     | NULL    |                |
| member | varchar(20)      | YES  |     | NULL    |                |
| id     | int(10) unsigned |      | PRI | NULL    | auto_increment |
+--------+------------------+------+-----+---------+----------------+
```

Figure 9 The groups table

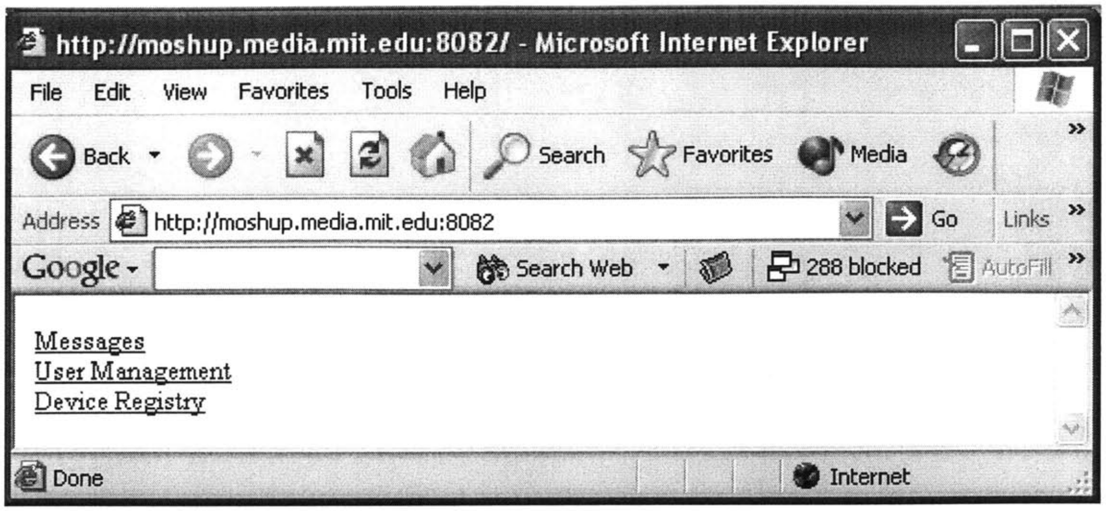
Finally, to identify the user without the user having to log in to every Bluetooth network server with which his or her device interacts, the system requires that the user register his or her device first. This essentially creates a mapping of devices, identified by their Bluetooth Addresses to users. It is possible for a user to register multiple devices to their user name; however registering multiple user names to a singular device would create an ambiguity in the system. This ambiguity would be masked by the natural ordering of the table's id field. The table consists of three columns: the first containing the Bluetooth address of devices, the second listing the user names associated with each device and the third being the id field mentioned above, which as in the other tables provides an unambiguous means to select an individual entry (or row in the table). As soon as the device connects to the server, it sends its Bluetooth address. This is used to identify the device in system calls on the server, such as the call to measure signal strength. It is also used right away to identify the user, with a query to this table.

```
+--------+------------------+------+-----+---------+----------------+
| Field  | Type             | Null | Key | Default | Extra          |
+--------+------------------+------+-----+---------+----------------+
| btaddr | varchar(17)      | YES  |     | NULL    |                |
| user   | varchar(20)      |      |     |         |                |
| id     | int(10) unsigned |      | PRI | NULL    | auto_increment |
+--------+------------------+------+-----+---------+----------------+
```

Figure 10 The device registry table

3.5 Web Administration Interface

To maintain this database messages, threads, users groups and devices, a web interface is used. Connecting to the central message server on port 8082 with an ordinary web browser will bring up a start page that links to the aspects of the database that can be managed from through the web interface. There is no log in or authentication.


http://moshup.media.mit.edu:8082/ - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address http://moshup.media.mit.edu:8082
Go
Links
Google
Search Web
288 blocked
AutoFill
Done
Internet

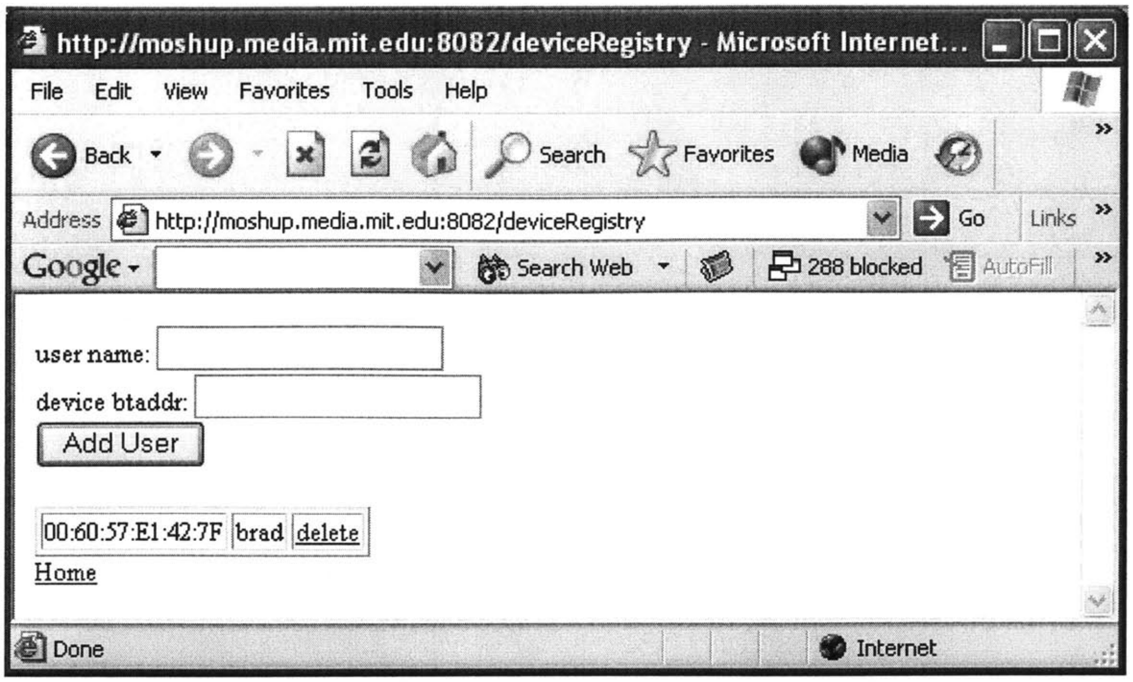
Messages
User Management
Device Registry

Figure 11 The Administration Start Page

The messages link brings the user to a page with fields corresponding to the user-supplied portions of a message (see Figure 13). Normally, on the device, the user name is entered automatically to be the user register to the device. However, on the web the user must manually enter their user name. Clicking the "Create Message" button inserts the entered message into the database as part of the thread specified at the location the user entered. Clicking the "delete" link next to any message will delete the message from the server. The page refreshes as soon as a message is either created or deleted, and the table reflects that change.

Following the "User Management" link allows the user to insert into the groups table described above the rows that give a particular user membership in a group (see Figure 14). Rows may also be removed from the table by clicking the "delete" link net to it. Upon clicking either the "Add User" button or a "delete" link, the action is taken and the page refreshes to show the result.

Finally, there is the "Device Registry" page (see Figure 12). Here the user can enter the Bluetooth address of their device and their user name. Clicking the button then enters that pairing into the system. Again, the delete link will remove the pairing.


http://moshup.media.mit.edu:8082/deviceRegistry - Microsoft Internet...
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://moshup.media.mit.edu:8082/deviceRegistry Go Links
Google Search Web 288 blocked AutoFill
Done Internet

user name:

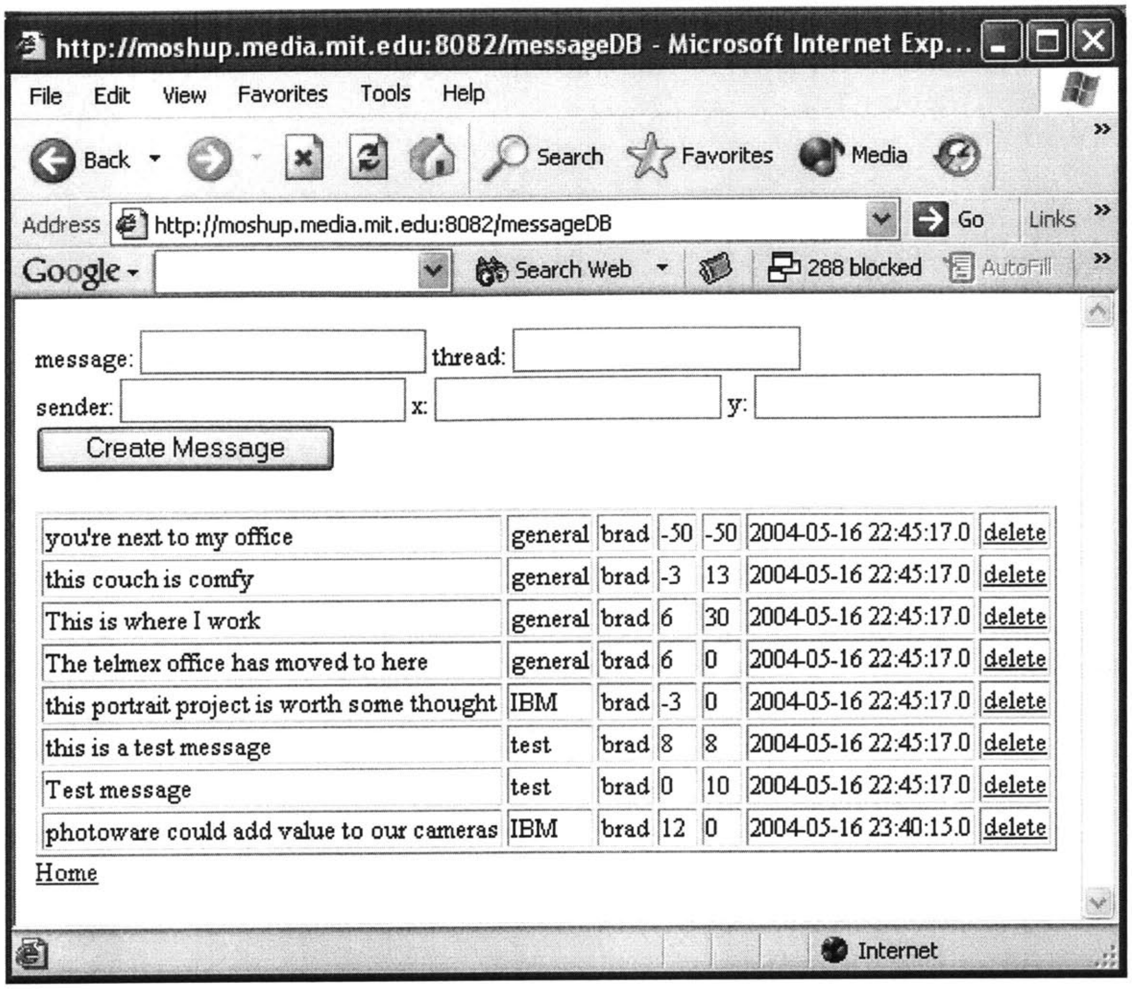
device btaddr:

Add User

00:60:57:E1:42:7F	brad	delete

Home

Figure 12 The Device Registration Page

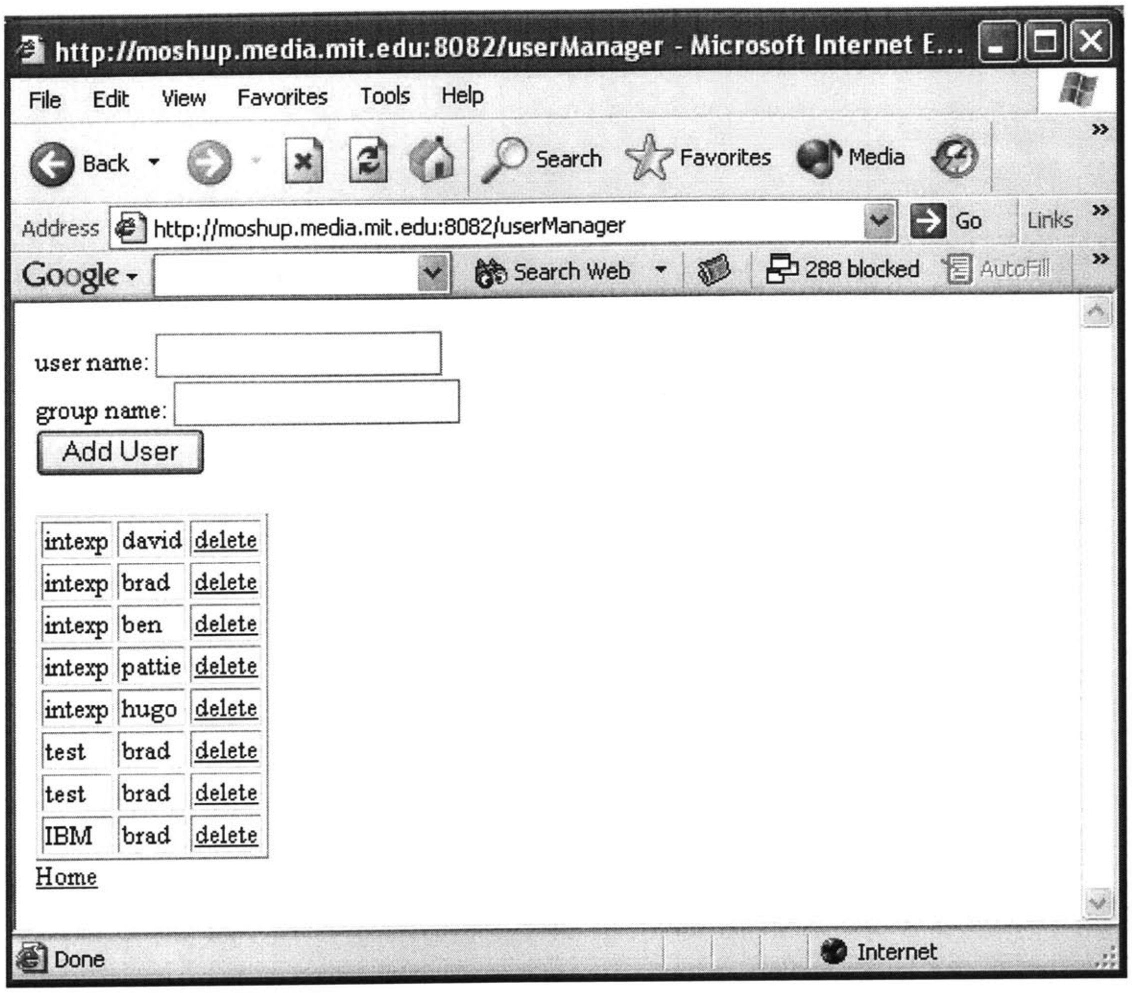

http://moshup.media.mit.edu:8082/messageDB - Microsoft Internet Exp...
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address http://moshup.media.mit.edu:8082/messageDB
Go
Links
Google
Search Web
288 blocked
AutoFill
Internet

message: thread:
sender: x: y:
Create Message

you're next to my office	general	brad	-50	-50	2004-05-16 22:45:17.0	delete
this couch is comfy	general	brad	-3	13	2004-05-16 22:45:17.0	delete
This is where I work	general	brad	6	30	2004-05-16 22:45:17.0	delete
The telmex office has moved to here	general	brad	6	0	2004-05-16 22:45:17.0	delete
this portrait project is worth some thought	IBM	brad	-3	0	2004-05-16 22:45:17.0	delete
this is a test message	test	brad	8	8	2004-05-16 22:45:17.0	delete
Test message	test	brad	0	10	2004-05-16 22:45:17.0	delete
photoware could add value to our cameras	IBM	brad	12	0	2004-05-16 23:40:15.0	delete

Home

Figure 13 The Message Administration Page


http://moshup.media.mit.edu:8082/userManager - Microsoft Internet E...
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://moshup.media.mit.edu:8082/userManager Go Links
Google Search Web 288 blocked AutoFill
Add User
Done Internet

user name:

group name:

intexp	david	delete
intexp	brad	delete
intexp	ben	delete
intexp	pattie	delete
intexp	hugo	delete
test	brad	delete
test	brad	delete
IBM	brad	delete

Home

Figure 14 User and Group Management Page

3.6 Filtering

The filtering of messages is accomplished through the user selecting which threads to subscribe to. Some threads are public and thus open the user to a greater possibility of unwanted messages. Messages remain on the server indefinitely. The client may select out the older messages or select for a particular time frame. When there are many messages in the same area, the client must provide a heuristic by which the most appropriate message will be selected. The current client's heuristic is the message's Euclidian distance from the user's location.

The user can scroll through a list of threads; depressing the five-way stick will select the thread for monitoring, which is signified by a red check mark place to the far right of the screen. This is expressed in the SQL clause that is passed as an argument to the message request as part of the where clause. The base of the where clause is "WHERE 1 = 0." Each selected thread is then appended as " OR thread = 'THREAD_NAME'."

The user can also select to be alerted of the presence of a message from a specific thread. This is selected by pressing the "2" key and is reflected on the interface by a green 'A' placed to the right of the thread name, just to the left of where a red check would exist. Once a message is received from a message request, the name of thread it belongs to is checked against the names of all threads selected to be alerted about. A short tone is then emitted. Since the messages are polled for frequently, the beeps repeat frequently. A vibration alert would be preferable, because it is not as obtrusive. Currently if the user would like to stop the beeping, he or she would have to turn off the alert to the thread.

3.7 Hardware Used

This project required Bluetooth enabled phones that provided programmatic access to certain functionality of their Bluetooth stack. Specifically, the phones needed to be able to discover other Bluetooth devices within range and establish a connection with them. The phones selected were the Nokia Series 60; specifically the 3650[2] and 6600[3]. They run the Symbian OS, versions 6.0 and 7s respectively. Both allow access to the native Bluetooth API in C++. Additionally, the 6600 allows access to certain Bluetooth functionality for applications written in Bluetooth with the JSR-82 API.

The Bluetooth network servers were given Bluetooth capabilities with BAFO USB Bluetooth dongles[4]. These were chosen primarily for their cost effectiveness. Additionally, they provide a great deal of functionality, which added flexibility to the architecture and was compatible with Windows and Linux. The servers, which use these dongles to make up the Bluetooth server network, run Knoppix[5] Debian[6] Linux version 2.4.22-xfs with the Bluez stack. These dongles were also used to upload code to the phones from Windows XP and Windows 2000 development machines.

To provide GPS functionality to the Nokia phones, the Socket Communications GPS Receiver with Bluetooth[7] was employed. This stand-alone device interfaces with the phone over a

RFCOMM Bluetooth serial port. It may be kept in the user's pocket while the user operates the phone, which combined with its small size and weight, provide the GPS capabilities to the phone without much of a burden to the user and without changing the phones' form factor.

4. Conclusions

The main conclusion that can be drawn from this work is that location can be determined with a fair amount of accuracy using Bluetooth signal strength. Roughly, this accuracy is approximately 1-3m. This accuracy depends on the amount of interference in the vicinity. The single greatest sources of this interference are people.

This aspect of the project alone has many applications well outside the scope of this project. It is particularly useful because, unlike most other location sensing techniques with similar accuracy (and many with worse accuracy), this system can work on fairly standard computing hardware, such as phones, PDAs and laptops, without any additional custom hardware.

The messaging system itself is potentially very useful, but it requires two things to evaluate that usefulness. First, there must be a user base of a certain size in order to make use of the community messaging and community discussion aspects of the system. The system does have a certain level of usefulness for an individual to provide incentive for early adopters before the user base has grown sufficiently. That usefulness is in personal reminders and institutional postings (announcements for events and such).

The second requirement for the indoor system to be useful is for the infrastructure of Bluetooth network servers to be in place. It is feasible that such a network would be set up in a business or university setting. There are certain productivity gains that can be foreseen by providing appropriate information to employees or students based on their location that could motivate a large company or university to install the network of beacons that the system requires. Of course, outdoors, GPS is sufficient, and this would be an important stepping-stone to build a user base with.

5. Related Work

Previous work related to this thesis fall into two main categories; location-base messaging applications and research into Bluetooth location sensing.

5.1 Location-based Messaging

Previous work in the area of location-based messaging was done by Emily Chang under the supervision of Pattie Maes in their Hanging Messages[8] project. The project used Pocket PC PDAs equipped with GPS receivers to obtain location information and display the messages. A large part of the project focused on filtering the messages. The system relied on a categorical hierarchy of messages, which the user could then filter into three categories: active, passive or ignored. Active messages would alert their user to their arrival, while the user would have to explicitly query the device to receive passive messages. The user could also create profiles based on location or time, for instance having a separate set of filters while at their place of employment during the hours of 9am to 5pm. The design was largely modeled after email, with the messages "arriving" at a specific time and only being presented at that time. Unlike email, the system also allowed messages to be sent to everyone, however if the user's profile allowed these messages, they were automatically classified as passive.

E-Graffiti[9], a project at Cornell studied how a context-aware, location-based messaging system would be received by the general student body. The response was promising, but the study also brought to light some important design considerations. The system implemented was for use on 802.11b-enabled laptops. It consisted of an interface that displayed a list of available messages and a window displaying the currently selected message from that list. The messages could either be privately addressed to a particular user, or publicly addressed to everyone. Location was determined to a building level granularity by the wireless access point that the computer was connected to. Many of the study subjects had trouble understanding the concept of location awareness. Many of the messages were private messages, which the subjects were using in a manor similar to IRC or AIM. This may be anticipated, as with any new communication medium, the first users tend to mimic the conventions and uses of a related, more common medium. The system allowed users to send messages to a location without being at that location. Some suggested that the message be tagged to indicate this, as it provides important contextual information. The subjects requested that the system be able to inform them if there were other subjects at a certain location (typically to see if there was anyone to read their message at their location.) The study points out the need for future implementers of location based messaging systems to educate users on the advantages of location based messaging and to guide their use of the system, perhaps with examples.

ActiveCampus[10], a project at UCSD also uses 802.11x to determine location. The project implements a SOAP interface, allowing other developers to produce their own location-based applications. The main application is a map of the user's location on campus, overlaid with links to information and the location of their friends and colleagues. The links can be information about events going on at that location, or general information about the location (like a department's home page), clicking on these links brings up a web page. Clicking on any of the people brings up a chat session with them. Finally, there is a graffiti feature too, but the user is required to specifically query for graffiti at a location in order to prevent clutter. The messages left in graffiti are readable by any user.

The ActiveCampus project interpolates between 802.11x access points to generate the location information. Unfortunately, since these access points are not arranged with location sensing in mind, the average error is 33 feet in doors and 75 feet out doors. To combat this, the application allows users to specify their location, which adds a virtual access point to the data set. The next time a user is in that location, he or she will presumably have a similar mix of signal strengths from the real access points and the system will recognize that as the virtual access point. The virtual access point will then dominate in the location-sensing algorithm, and provide a truer reading. This system has the desirable quality that it works from day one, all-be-it with the errors reported above, and will gradually learn to report more accurate data through regular use. Also, the users have an immediate incentive to create new virtual access points.

ComMotion[11] was a system designed to deliver personal location and context aware messages to a user from his or her self. The system was GPS based, and the user interaction was speech based. Certain features, such as displaying a map, also required a display; but they were not central to the design and could be considered optional. The system kept track of where the user went, and when it found a location that the user frequented, it asked the user to name it. The user could also specify classes of locations, such as grocery stores. Once the location was named, a "to-do" list was created for the location. The users could populate these to-do lists themselves, allow others to post to them, or subscribe to services such as news bulletins. The items stay on the list until the user deletes them or "checks them off." The system also allowed for other users to query for the location of a user. It implemented both per user and per location preferences to govern the location queries.

Location Linked Information[12], a project by Matt Mankins at the MIT Media Laboratory attempted to index information available on the web and in public databases by longitude and latitude. It held this information, in the form of URL pointers to the information, in a distributed system of databases. The information was accessed by client devices, using the Jabber XML protocol.

Finally, GeoNotes[13] surveys these and other related works and proposes a system to capture the social aspect of community location based messaging. A message server and a java client were implemented to demonstrate their proposed system. The system was very similar to e-Graffiti; it provided building level resolution, though the implementation relied on the user to specify their position. It also allowed propagation of the notes through a resolution hierarchy. For instance, if a user told the system that he/she was on campus, they would get all the notes from all the buildings on campus. The same is true of specifying a particular region of campus. This system was not implemented on any actual mobile platform.

5.2 Bluetooth Location Sensing

Location enabled services are viewed by many to be "the next big thing" in mobile computing. Cell phone manufacturers and service providers see it not only as yet another service to sell, but also as a way to distinguish themselves from the pack. For instance, Motorola is providing navigational services to Avis rental car customers[14].

The FCC in the US[15] and regulatory groups in other parts of the world have enacted laws requiring manufacturers and service providers to provide location information to emergency personnel when an emergency call is placed by a cellular phone. To accomplish this, carriers primarily rely on knowing which cellular tower the device is using, and may augment that with signal strength information. These types of services are very crude and are limited to resolutions on the order of hundreds of meters and up to a few miles. On the other hand it should be noted that governments have also taken notice of the possibility of such information being abused. Some have enacted laws restricting the use of location-based services[16].

GPS provides more fine-grained location data than the use of cellular towers. Motorola[17] and Nokia[18] both released cell phone models featuring GPS receivers. While GPS is finer-grained than E911 and other such standards, its 10m – 100m resolution is not fine enough for many potential applications. This is evidenced by the fact that despite GPS enabled phones have been available for a relatively long time now and still no "killer app" is associated with them. Their use is still extremely niche (they are mostly used by developers) and each company still only offers a few models featuring GPS, most without the consumer centric features that have pushed sales of other high end phones (such as built in cameras). Another important constraint on GPS is its frequent inability to get a fix on the satellites that the system depends on when indoors or in urban environments. Even when a fix can be made, the interference caused by the buildings can distort the location reading greatly.

Several people have looked into Bluetooth as a way to gather location data. It is intriguing because many of the latest cellular phones and PDAs are already equipped with the necessary hardware. Also, unlike IR, Bluetooth does not suffer from line of sight limitations. One company now sells a Bluetooth LAN access point that also doubles as a location sensor[19]. Another Company sells a Bluetooth based location-based services infrastructure[20]. Both of these systems are location based in the sense that they receive data from a Bluetooth or IR if the mobile device is within range. Work at Columbia[21] and Universidad de Vigo[22] has also produced systems which detected location based on which Bluetooth location beacons a device can "see." These differ from the approach taken in Etherthreads, because these systems can only determine that they are near a waypoint (a waypoint being one of the Bluetooth beacons) where as the Etherthreads system can interpolate between those way points, giving better resolution by more than an order of magnitude.

The University of Waterloo has also explored the possibilities of more advanced techniques which yielded higher resolution than simply which access point can be seen[23]. Bhoutika explored using signal strength, time of arrival, time difference of arrival and angle of arrival with IR, Bluetooth, 802.11x and Ultrasound systems theoretically, but no working system has been publicized. Work at the University of Pittsburgh has explored in depth using signal strength from 802.11 access points to determine location[24]. The work highlighted the multitude of algorithms available to determine location based on signal strength, and was part of the inspiration to push this algorithmic design decision out to the client, rather than force the clients to accept the pros and cons of a design decision made within the infrastructure. The paper specifically endorses a "finger-printing" approach, which learns the characteristic set of signal strengths which are unique to each location. Similar work was done at the MIT Media Lab[25]

Work at Minnesota State University has also explored the use of signal strength, link quality and bit error rate of Bluetooth connections in determining distance between devices[26]. The published research from this group essentially models the relationship between the signal strength and link quality of a connection and the distance between the devices. Their work could be helpful in developing new algorithms to determine the distance from Bluetooth servers, and therefore location.

6. Future Work

This project can be expanded in many directions, principally in the area of productization. There are two ways in which this could be productized; as a consumer product and as a industrial product.

For the consumer style productization, it is important to minimize the setup. The most likely scenario would be a location network set up kit, which would contain a number of extremely inexpensive Bluetooth beacons, which would replace the Bluetooth network server in the current design. These Bluetooth beacons would plug in to a typical wall socket, and once provided with power, ping out, looking for other beacons. Once they find each other, they can measure distance and orientation to each other, and with the assumption that each is in a wall socket, map out the home. The distance measuring and orientation can even be augmented with other, more specific circuitry (the many different techniques available are discussed in related work.) A home computer could then pair with any of the beacons and control the whole network in a point to point configuration.

The consumer Bluetooth network could also offer other services, such as providing IP phone services to cell phones and PDAs, to cut down on cellular minutes used. It could also provide networking support for cell phones, PDAs and even laptops. The speed would be less than that of an 802.11b system, but much greater than a GPRS connection.

In an industrial setting, the system could employ one dedicated server, which can control a number of Bluetooth beacons connected by some form of cable. The simplest solution would be to use USB Bluetooth dongles, with a series of USB cables and hubs. The limits of the USB standard limit this scenario to five hubs, each with 5m of cable between them[27], which means a dedicated server would cover a 25-meter radius, with more servers needed to cover more area. Another alternative would be to engineer a device, which could hook into a standard IP network and parasite the power it needs from that.

In more of a public implementation, possibly the alternative that uses GPS for location and GPRS for data, the public threads would have to be monitored to ensure the content is in keeping with the theme, and probably more likely, clean out the spam. Private threads are less susceptible to SPAM, but could still benefit from being cleaned, depending on how private they are. Significant work is being done in the area of determining which emails are spam and which are not[28]. Cleaning the threads would be a simplified version of the classic spam problem, because the subject matter would be known from an authoritative source. Anything not in keeping with the subject matter of the thread can be flagged for review, or simply removed. The rest of the messages on thread can be used as a training set for a thread cleaner.

The system could easily be adapted to handle the attachment of multimedia content such as pictures or audio messages. This could be accomplished by adding another column to the message table in the SQL database. This additional column could hold a URL pointing to the multimedia content. The difficulty surrounding the inclusion of multimedia content is the question of how integrate it into the user interface; when to display the content; how to alert the user to the existence of the content; and how to filter the content (this is more difficult than the text processing proposed for the thread cleaners). Also, there is the more practical question of where the multimedia content would be stored; should the users provide their own web space, or will it be provided as part of the general infrastructure.

To be able to handle the personal reminder scenario laid out in section 2.2, the system would have to be able express what dates and times the message should be active. There are two ways to do that. The first is to give a field which can contain a script that the message server can then interpret and evaluate. The difficulty with this approach is to construct a scripting language which is easily interpreted and can handle whatever criteria that the user needs. Also, this requires the message to post-process the results from its SQL query.

The second approach is two add fields to the message table which would enumerate the possible date and time combinations that a user might want to use. To this end, a field of the type 8 bit char, could enumerate all possible combinations of days. And two time fields could define the border times of the day. This would be enough to make the previously discussed scenario possible. This approach also allows the time selection of the messages with in the SQL query. The drawback is that the construction of these fields restricts what kinds of time specifications can be made.

The implementation of the client does not search for new Bluetooth network servers. The has the drawback that if the user moves out of the range of the servers he or she is connected to, the client does not find new Bluetooth network servers to connect to. To make matters worse, the client does not handle the loss of the a connection gracefully. A few factors are behind this. First, the client will crash if it attempts to connect to more than three of the Bluetooth servers. There is no explanation for this. Second, the client has a much better success rate the closer it is to the beacon it connecting to. This is the reason behind the start up screen; it allows the user walk closer to each of the beacons before connecting to them. In order for the system to be practical, this hand off between servers, which includes the automatic discovery of and connection to new servers, and the graceful handling of the loss of connections. In this implementation, the servers' addresses and locations are hard coded into the client application. This information would have to be obtained in more of an automatic process; either with the information being in the database, or the servers reporting it on their own.

Works Cited

[1] "JBluez," [Online Document], 2002 Sept. 16, Available HTTP: http://jbluez.sourceforge.net/
[2] "Nokia 3650 Phone," [Online Document] Available HTTP: http://www.nokiausa.com/phones/3650
[3] "Nokia 6600 Phone," [Online Document] Available HTTP: http://www.nokiausa.com/phones/6600
[4] "BAFO USB Bluetooth Dongle," [Online Document], Available HTTP: http://www.bafo.com/bafo/productslists.asp?foproduct_id=1165
[5] "KNOPPIX," [Online Document], Available HTTP: http://knoppix.com/
[6] "Debian GNU/Linux," [Online Document], Available HTTP: http://www.debian.org/
[7] "Socket Communications: GPS Reciever with Bluetooth Wireless Technology," [Online Document], Available HTTP: http://www.socketcom.com/product/GP0804-405.asp
[8] E. Chang and P. Maes, "Hanging Messages: Using Context-Enhanced Messages for Just-In-Time Communication," [Online Document], 2001 May, Available HTTP: http://xenia.media.mit.edu/~elchang/HM/docs/hm_brief.pdf
[9] J. Burrell and G. Gay, "E-Graffiti: evaluating real-world use of a context-aware system," [Online Document] , 2002, Available HTTP: http://www.hci.cornell.edu/LabArticles/Egraffiti_Burrell.pdf
[10] W. G. Griswold, et. all, "UCSD ActiveCampus," [Online Document], 2004 April 28, HTTP : http://activecampus.ucsd.edu/
[11] N. Marmasse and C. Schmandt, "Location-aware information delivery with comMotion," [Online Document], 2000, Available HTTP: http://web.media.mit.edu/~nmarmas/cmHUC2k.pdf
[12] Matt Mankins, "Location Linked Information," [Online Document], Available HTTP: http://xenia.media.mit.edu/~mankins/lli/
[13] F.Espinoza et. all, "GeoNotes: Social and Navigational Aspects of Location-Based Information Systems," Available HTTP: http://www.sics.se/~espinoza/documents/GeoNotes_ubicomp_final.htm
[14] H. Wolinsky, "Avis institutes cell-phone-based navigation system from Motorola," [Online Document], 2003 Dec 9, Available HTTP: http://www.suntimes.com/output/business/cst-fin-moto09.html
[15] FCC, "Enhanced 911," [Online Document], Available HTTP: http://www.fcc.gov/911/enhanced/
[16] "South Korea to introduce tougher Location Laws," [Online Document], 2003 Aug 21, Available HTTP: http://www.cellular.co.za/news_2003/082203-south_korea_to_introduce_tougher.htm
[17] Motorola iDEN, "Motorola iDEN Phones," [Online Document], Available HTTP: http://www.idenstore.com/dr/v2/ec_MAIN.Entry10?V1=442714&PN=1&SP=10023&xid=39192&DSP=&CUR=840&PGRP=0&CACHE_ID=0
[18] "Nokia 3585i AMPS/CDMA 2000 GPS Phone," [Online Document], Available HTTP: http://www.cellular.co.za/phones/nokia/2002/nokia_3585i-cdma2000.htm
[19] Lesswire, "Lesswire AG," [Online Document], Available HTTP: http://www.lesswire.com/products/soft_comp/ln.htm
[20] BluePosition, "BLIS4Platform," [Online Document], Available HTTP: http://www.blueposition.com/platform.php
[21] H. Chu and S. Sidiroglou-Douskos, "Bluetooth Location Services," [Online Document], Available HTTP: http://www1.cs.columbia.edu/~hhc42/bluetooth_project/
[22] F. Gonzalez-Castano and J. Garcia-Reinoso, "Bluetooth Location Networks," [Online Document], Available HTTP: http://www-gist.det.uvigo.es/~manolo/ipc/0203/SAWN-07-8.pdf
[23] K. K. Bhoutika, "Bluetooth Indoor Location Management," [Online Document], 2003 May, Available HTTP: http://www.ccng.uwaterloo.ca/Seminars/Presentations/bilm.ppt
[24] P. Prasithsangaree, P. Krishnamurthy and P.K. Chrysanthis, "On Indoor Position Location With Wireless LANs," [Online Document], Available HTTP: http://netlab18.cis.nctu.edu.tw/html/paper/2003_02_11/ON INDOOR POSITION LOCATION WITH WIRELESS LANS.pdf
[25] Alisa Marshall, "Signal-Based Location Sensing Using 802.11b," [Online Document], March 4 2004, Available HTTP: http://web.media.mit.edu/~vemuri/alisa/MyWork/papers/aup-final-draft2.pdf
[26] M. Kamalapuri, "Using Standard Bluetooth APIs to Determine Relative Distance Between Bluetooth Devices," [Online Document], Available HTTP: http://www.wireless.mnsu.edu/2003 Symposium/Presentations/Using Standard Bluetooth APIs to Determine.pdf
[27] USB Implementers Forum, Inc., "USB Info: Frequently Asked Questions: USB Cables, Connectors, and Networking with USB, " [Online Document], Available HTTP: http://www.usb.org/faq/ans5
[28] Eric Horvitz, "A Bayesian Approach to Filtering Junk E-mail," [Online Document], Available HTTP: http://research.microsoft.com/~horvitz/junkfilter.htm

Appendix A. Client Code

```
/*
*
========================================================================
=====
*  Name     : L2Location.rss
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     This file contains all the resources for the L2Location.
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
========================================================================
=====
*/

//  RESOURCE IDENTIFIER
NAME    AWIZ // 4 letter ID

//  INCLUDES

#include <eikon.rh>
#include "l2location.hrh"
#include "l2location.loc"
#include <avkon.rsg>
#include <avkon.rh>
#include <avkon.mbg>
#include <avkon.loc>



//  RESOURCE DEFINITIONS

RESOURCE RSS_SIGNATURE { }

RESOURCE TBUF { buf = "L2Location"; }

RESOURCE EIK_APP_INFO
    {
    status_pane = r_l2location_status_pane;
    }

//----------------------------------------------------
//
//    r_l2location_hotkeys
//
//----------------------------------------------------
//
RESOURCE HOTKEYS r_l2location_hotkeys
    {
    control =
        {
```

```
        HOTKEY { command = EAknCmdExit; key = 'e'; }
        };
    }

//----------------------------------------------------
//
//    r_l2location_view1
//
//----------------------------------------------------
//
RESOURCE AVKON_VIEW r_l2location_view1
    {
    hotkeys = r_l2location_hotkeys;
    menubar = r_l2location_menubar_view1;
    //cba      = R_AVKON_SOFTKEYS_SELECTION_LIST;
      cba          = R_AVKON_SOFTKEYS_OPTIONS_EXIT;
    }

//----------------------------------------------------
//
//    r_l2location_menubar_view1
//
//----------------------------------------------------
//
RESOURCE MENU_BAR r_l2location_menubar_view1
    {
    titles =
        {
        MENU_TITLE { menu_pane = r_l2location_app_menu; txt = "App"; }
        //MENU_TITLE { menu_pane = r_l2location_view1_menu; txt =
"View"; }
        };
    }

//----------------------------------------------------
//
//    r_l2location_view1_menu
//
//----------------------------------------------------
//
RESOURCE MENU_PANE r_l2location_view1_menu
    {
    items =
        {
        MENU_ITEM { command = EL2LocationCmdAppTest; txt =
qtn_view1_option_item; }
        };
    }

//----------------------------------------------------
//    r_aknexeditor_view1_edwin
//    resource for edit window control on view1
//----------------------------------------------------
//
RESOURCE EDWIN r_aknexeditor_view1_edwin
    {
    flags = EAknEditorFlagNoEditIndicators;
    width = 150;
    lines= 5;
    maxlength = 32;
    }

//-----------------------------------------------------
//
//    r_l2location_view2
//
//-----------------------------------------------------
//
RESOURCE AVKON_VIEW r_l2location_view2
    {
    hotkeys = r_l2location_hotkeys;
    menubar = r_l2location_menubar_view2;
    //cba      = R_AVKON_SOFTKEYS_SELECTION_LIST;
      cba          = R_AVKON_SOFTKEYS_OPTIONS_EXIT;
    }

//-----------------------------------------------------
//
//    r_l2location_menubar_view2
//
//-----------------------------------------------------
//
RESOURCE MENU_BAR r_l2location_menubar_view2
    {
    titles =
        {
        MENU_TITLE { menu_pane = r_l2location_app_menu; txt = "App"; }
//        MENU_TITLE { menu_pane = r_l2location_view2_menu; txt =
"View"; }
        };
    }

//-----------------------------------------------------
//
//    r_l2location_view2_menu
//
//-----------------------------------------------------
//
RESOURCE MENU_PANE r_l2location_view2_menu
    {
    items =
        {
        MENU_ITEM { command = EL2LocationCmdAppTest; txt =
qtn_view2_option_item; }
        };
    }

RESOURCE ARRAY r_simplelist_items
      {
      items =
            {
            LBUF
                  {
                  txt = "item 1";
```

```
                },
            LBUF
                {
                txt = "item 2";
                },
            LBUF
                {
                txt = "item 3";
                }
            };
        }

//-----------------------------------------------------
//
//      r_l2location_view3_threads_list
//
//-----------------------------------------------------
//
RESOURCE LISTBOX r_l2location_view3_threads_list
      {
      //array_id = r_simplelist_items;
      flags = EAknListBoxMarkableList;
      }



//-----------------------------------------------------
//
//    r_l2location_view3
//
//-----------------------------------------------------
//
RESOURCE AVKON_VIEW r_l2location_view3
    {
    hotkeys = r_l2location_hotkeys;
    menubar = r_l2location_menubar_view3;
    //cba     = R_AVKON_SOFTKEYS_SELECTION_LIST;
      cba         = R_AVKON_SOFTKEYS_OPTIONS_EXIT;
    }

//-----------------------------------------------------
//
//    r_l2location_menubar_view3
//
//-----------------------------------------------------
//
RESOURCE MENU_BAR r_l2location_menubar_view3
    {
    titles =
        {
        MENU_TITLE { menu_pane = r_l2location_app_menu; txt = "App"; }
//        MENU_TITLE { menu_pane = r_l2location_view3_menu; txt =
"View"; }
        };
    }

//-----------------------------------------------------
```

```
//
//    r_l2location_view3_menu
//
//----------------------------------------------------
//
RESOURCE MENU_PANE r_l2location_view3_menu
    {
    items =
        {
        MENU_ITEM { command = EL2LocationCmdAppTest; txt =
qtn_view3_option_item; }
        };
    }




//----------------------------------------------------
//
//    r_l2location_app_menu
//
//----------------------------------------------------
//
RESOURCE MENU_PANE r_l2location_app_menu
    {
    items =
        {
        MENU_ITEM { command = EL2LocationCmdAppStartUp; txt = "Start
Up"; },
        MENU_ITEM { command = EL2LocationCmdAppThread; txt = "Thread
Selection"; },
        MENU_ITEM { command = EL2LocationCmdAppMessage; txt = "Message
Display"; },
        MENU_ITEM { command = EL2LocationCmdAppCreate; txt = "Create
Message"; }
        //MENU_ITEM { command = EAknCmdExit; txt = qtn_options_exit; }
        };
    }

RESOURCE STATUS_PANE_APP_MODEL r_l2location_status_pane
    {
      panes =
            {
            SPANE_PANE
                  {
                  id = EEikStatusPaneUidNavi;
                  type = EAknCtNaviPane;
                  resource = r_l2location_navi_decorator;
                  }
            };
    }

//----------------------------------------------------
//
//    r_l2location_navi_decorator
//
```

```
//---------------------------------------------------------
//
RESOURCE NAVI_DECORATOR r_l2location_navi_decorator
    {
    type = ENaviDecoratorControlTabGroup;
    control = TAB_GROUP
            {
            tab_width = EAknTabWidthWithThreeTabs;  // two tabs
            active = 0;
            tabs = {
                  TAB
                        {
                id = EL2LocationView1Tab; // from application hrh
                txt = "Start Up";
                },
                  TAB
                        {
                id = EL2LocationView3Tab;
                txt = "Thread Selection";
                        },
                  TAB
                        {
                id = EL2LocationView2Tab; // from application hrh
                txt = "Message Display";
                },
                  TAB
                        {
                id = EL2LocationView4Tab;
                txt = "Create Message";
                        }
                  };
            };
      }

// ---------------------------------------------------------
//
//    Strings used by the application
//
// ---------------------------------------------------------
//
RESOURCE TBUF r_icon_file_name
      {
      buf = ICON_FILE_NAME;
      }



RESOURCE AVKON_VIEW r_aknexlist_view_markview
    {
  //  menubar = r_aknexlist_menubar_main;
//    hotkeys = r_aknexlist_hotkeys;
    //cba = R_AVKON_SOFTKEYS_OPTIONS_BACK;
    }
//--------------------------------------------------------------------
--------
//
//    r_aknexlist_menupane_markable_list
//    Menu Pane
//
//-----------------------------------------------------------------------
--------
//
RESOURCE MENU_PANE r_aknexlist_menu_pane_markable_list
    {
    items =
        {
        MENU_ITEM
            {
            command = EAknMarkAll;
            flags = EEikMenuItemDimmed;
            txt = qtn_aknexlist_mark_all;
            },
        MENU_ITEM
            {
            command = EAknUnmarkAll;
            flags = EEikMenuItemDimmed;

            txt = qtn_aknexlist_unmark_all;
            }
        };
    }

RESOURCE TBUF256 r_aknexlist_mark_info
    {
    buf = qtn_aknexlist_mark_info;
    }

//-----------------------------------------------------------------------
--------
//
//    r_aknexlist_format
//    Information message.
//
//-----------------------------------------------------------------------
--------
//
RESOURCE TBUF256 r_aknexlist_format
    {
    buf = qtn_aknexlist_format;
    }

//-----------------------------------------------------------------------
--------
//
//    r_aknexlist_no_mark
//    Information message.
//
//-----------------------------------------------------------------------
--------
//
RESOURCE TBUF256 r_aknexlist_no_mark
    {
    buf = qtn_aknexlist_no_mark;
    }
```

```
//----------------------------------------------------
//    r_aknexeditor_view4_label1
//    resource for label control on view4
//----------------------------------------------------
//
RESOURCE TBUF32 r_aknexeditor_view4_label1
    {
    buf = "enter your message";
    }

//----------------------------------------------------
//    r_aknexeditor_view4_gtexted
//    resource for global text editor control on view4
//----------------------------------------------------
//
RESOURCE GTXTED r_aknexeditor_view4_gtexted
    {
    flags = EAknEditorFlagDefault;
    width = 120;
    height = 100;
    numlines = 5;
    textlimit= 50;
    fontcontrolflags = EGulFontControlAll;
    fontnameflags = EGulNoSymbolFonts;
    }


RESOURCE TBUF32 r_aknexeditor_container4_label_format_pt
    {
    buf = " %d pt";
    }


//----------------------------------------------------
//    r_aknexeditor_view4
//    view4 definition
//----------------------------------------------------
//
RESOURCE AVKON_VIEW r_aknexeditor_view4
    {
    menubar = r_aknexeditor_menubar_view4;
    cba = r_softkeys_options_next;
    }

//----------------------------------------------------
//    r_aknexeditor_menubar_view4
//    menu bar definition for view4
//----------------------------------------------------
//
RESOURCE MENU_BAR r_aknexeditor_menubar_view4
    {
    titles=
        {
```

```
        MENU_TITLE
            {
            menu_pane = r_aknexeditor_app_menu;
            txt=qtn_aknexeditor_menu_dummy;
            },
        MENU_TITLE
            {
            menu_pane=r_aknexeditor_view4_menu;
            txt = qtn_aknexeditor_menu_dummy;
            }
        };
    }


//-----------------------------------------------------
//
//    r_softkeys_options_next
//    CBA definition (OPTIONS - NEXT)
//
//-----------------------------------------------------
//
RESOURCE CBA r_softkeys_options_next
    {
    buttons =
        {
        CBA_BUTTON
            {
            id = EAknL2LocationCreateMessageBack;
            txt = "Back";
            },
        CBA_BUTTON
            {
            id = EAknL2LocationCreateMessageSubmit;
            txt = "Submit";
            }
        };
    }

//-----------------------------------------------------
//
//    r_aknexeditor_app_menu
//    application menu pane definition.
//
//-----------------------------------------------------
//
RESOURCE MENU_PANE r_aknexeditor_app_menu
    {
    items=
        {

        };
    }


//-----------------------------------------------------
//    r_aknexeditor_view4_menu
//    menu pane definition for view4
```

```
//----------------------------------------------------
//
RESOURCE MENU_PANE r_aknexeditor_view4_menu
    {
    items=
        {
        MENU_ITEM
            {
            command = EAknL2LocationCreateMessageBack;
            txt = "back";
            },

        MENU_ITEM
            {
            command = EAknL2LocationCreateMessageSubmit;
            txt = "submit";
            }

        };
    }


//----------------------------------------------------
//
//    r_aknexpoplist_menu_title_test1
//    heading pane text for test pattern 1
//
//----------------------------------------------------
//
RESOURCE TBUF32 r_aknexpoplist_menu_title_test1
    {
    buf = "Select Thread";
    }

//----------------------------------------------------
//
//    r_aknexpoplist_select_message
//    message for selection
//
//----------------------------------------------------
//
RESOURCE TBUF64 r_aknexpoplist_select_message
    {
    buf = "message";
    }



RESOURCE DIALOG r_aknexpoplist_select_message_note
    {
    flags = EAknDialogGenericQueryFlags;
    buttons = R_AVKON_SOFTKEYS_OK_EMPTY;
    items =
        {
        DLG_LINE
            {
            type = EAknCtNote;
            id = EAknExPopListSelectMessageNote;
            control = AVKON_NOTE
                {
                layout = ENoteWithImageLayout;
                singular_label =
qtn_aknexpoplist_select_message_singular;
                plural_label = qtn_aknexpoplist_select_message_plural;
                };
            }
        };
    }

//----------------------------------------------------
//
//    r_aknexpoplist_menu_items_test1
//    menu items definition for test pattern 1
//
//----------------------------------------------------
//
RESOURCE ARRAY r_aknexpoplist_menu_items_test1
    {
    items =
        {
        LBUF { txt = qtn_aknexpoplist_menu_test1_item1; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item2; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item3; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item4; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item5; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item1; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item2; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item3; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item4; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item5; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item1; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item2; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item3; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item4; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item5; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item1; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item2; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item3; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item4; },
        LBUF { txt = qtn_aknexpoplist_menu_test1_item5; }

        };
    }


// End of File
```

```
/*
*
======================================================================
=====
*  Name     : L2Location_caption.rss
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     Caption file for L2Location.
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
======================================================================
=====
*/

#include "l2location.loc"
#include <apcaptionfile.rh>

RESOURCE CAPTION_DATA
{
    caption = qtn_app_caption_string;
    shortcaption = qtn_app_short_caption_string;
}
```

```
/*
*
==========================================================================
=====
*  Name     : CAknExEditorContainer4 from CCoeControl,
MCoeControlObserver
*  Part of  : AknExEditor
*  Copyright (c) 2003 Nokia. All rights reserved.
*
==========================================================================
=====
*/

// INCLUDE FILES
#include <eiklabel.h>  // for example label control
#include <barsread.h>  // for resource reader
#include <eikedwin.h>  // for CEikEdwin
#include <eikgted.h>   // for CEikGlobalTextEditor
#include <aknutils.h>  // for Fonts.

#include <L2Location.rsg>
#include "L2Location.hrh"
#include "AknExEditorContainer4.h"

// Definitions
#define GTEXTED_LABEL_POS   TPoint(10, 10)
#define GTEXTED_POS         TPoint(10, 25)


// Constants
const TInt KNumberOfControls    = 2;
const TInt KBufLength           = 64;

// Enumarations
enum TControls
    {
    EGTextEdPrompt,
    EGTextEd
    };

// ================= MEMBER FUNCTIONS =======================

// C++ default constructor can NOT contain any code, that
// might leave.
//
CAknExEditorContainer4::CAknExEditorContainer4()
                         :iLabel(NULL), iGTextEd(NULL)
    {
    }

// EPOC default constructor can leave.
void CAknExEditorContainer4::ConstructL(const TRect& aRect)
    {
    CreateWindowL();

    TBuf<KBufLength> text;
    iCoeEnv->ReadResource(text, R_AKNEXEDITOR_VIEW4_LABEL1);
    iLabel = new (ELeave) CEikLabel;
    iLabel->SetContainerWindowL(*this);
    iLabel->SetTextL(text);
    iLabel->SetExtent(GTEXTED_LABEL_POS, iLabel->MinimumSize());

    TResourceReader reader;
    iCoeEnv->CreateResourceReaderLC(reader,
R_AKNEXEDITOR_VIEW4_GTEXTED);
    iGTextEd = new (ELeave) CEikGlobalTextEditor;
    iGTextEd->SetContainerWindowL(*this);
    iGTextEd->ConstructFromResourceL(reader);
    CleanupStack::PopAndDestroy();  // Resource reader

    iGTextEd->SetExtent(GTEXTED_POS, iGTextEd->MinimumSize());
    iGTextEd->SetAvkonWrap(ETrue);
    iGTextEd->SetFocus(ETrue);

    SetRect(aRect);
    ActivateL();
    }

// Destructor
CAknExEditorContainer4::~CAknExEditorContainer4()
    {
    delete iLabel;
    delete iGTextEd;
    }

// ---------------------------------------------------------
// void CAknExEditorContainer4::SetSystemFontL(TInt aFontEnum)
// Change font style of CEikGlobalTextEditor using system font.
// (other items were commented in a header).
// ---------------------------------------------------------
//
void CAknExEditorContainer4::SetSystemFontL(TInt aFontEnum)
    {
    const CFont* editorFont;


    switch (aFontEnum)
        {
        case ELatinPlain12:
            editorFont = LatinPlain12();
            break;
        case ELatinBold12:
            editorFont = LatinBold12();
            break;
        case ELatinBold13:
            editorFont = LatinBold13();
            break;
        case ELatinBold17:
            editorFont = LatinBold17();
            break;
        case ELatinBold19:
            editorFont = LatinBold19();
            break;
        case ENumberPlain5:
            editorFont = NumberPlain5();
            break;
        case EClockBold30:
            editorFont = ClockBold30();
            break;
        case ELatinClock14:
            editorFont = LatinClock14();
            break;
        default:
            return;
        }

    TCharFormat     charFormat;
    TCharFormatMask charFormatMask;

    charFormat.iFontSpec = editorFont->FontSpecInTwips();
    charFormatMask.SetAll();

    // Set font to GlobalTextEditor
    iGTextEd->ApplyCharFormatL(charFormat, charFormatMask);

    // Change label text.
    TBuf<KBufLength> sizeText;
    TBuf<KBufLength> formatText;
    iCoeEnv->ReadResource(formatText,
R_AKNEXEDITOR_CONTAINER4_LABEL_FORMAT_PT);
    sizeText.Format(formatText,
                    charFormat.iFontSpec.iHeight / KTwipsPerPoint);
    TBuf<KBufLength> labelText = charFormat.iFontSpec.iTypeface.iName;
    labelText.Append(sizeText);
    iLabel->SetTextL(labelText);
    iLabel->SetExtent(GTEXTED_LABEL_POS, iLabel->MinimumSize());
    }


// ---------------------------------------------------------
// void CAknExEditorContainer4::SetFontL(TInt aResourceID)
// Change font style of CEikGlobalTextEditor.
// (other items were commented in a header).
// ---------------------------------------------------------
//
void CAknExEditorContainer4::SetFontL(TInt aResourceID)
    {
    // Create font
    TFontSpec* fontSpec = CreateFontSpecFromResourceL(aResourceID);
    CleanupStack::PushL(fontSpec);

    TCharFormat     charFormat;
    TCharFormatMask charFormatMask;
    CFont*  editorFont = iCoeEnv->CreateScreenFontL(*fontSpec);
    CleanupStack::PushL(editorFont);

    charFormat.iFontSpec = editorFont->FontSpecInTwips();
    charFormatMask.SetAll();

    // Set font to GlobalTextEditor
    iGTextEd->ApplyCharFormatL(charFormat, charFormatMask);

    iCoeEnv->ReleaseScreenFont(editorFont); // Release editorFont
    CleanupStack::Pop();                    // editorFont

    // Change label text.
    TBuf<KBufLength> sizeText;
    TBuf<KBufLength> formatText;
    iCoeEnv->ReadResource(formatText,
R_AKNEXEDITOR_CONTAINER4_LABEL_FORMAT_PT);
    sizeText.Format(formatText,fontSpec->iHeight / KTwipsPerPoint);
    TBuf<KBufLength> labelText = fontSpec->iTypeface.iName;
    labelText.Append(sizeText);
    iLabel->SetTextL(labelText);
    iLabel->SetExtent(GTEXTED_LABEL_POS, iLabel->MinimumSize());


    CleanupStack::Pop();  // fontSpec


    delete fontSpec;    // Delete Font
    }

// ----------------------------------------------------------
// CAknExEditorContainer4::CreateFontSpecFromResourceL(TInt aResourceId
// Create fontSpec object
// (other items were commented in a header).
// ----------------------------------------------------------
//
TFontSpec* CAknExEditorContainer4::CreateFontSpecFromResourceL(TInt
aResourceId)
    {
    TResourceReader reader;
    iCoeEnv->CreateResourceReaderLC(reader, aResourceId);

    HBufC*   fontName;
    TInt    fontSize;

    fontName = reader.ReadHBufCL();
    fontSize = reader.ReadInt16();

    CleanupStack::PopAndDestroy();  // for reader

    CleanupStack::PushL(fontName);

    TFontSpec* fontSpec = new TFontSpec(*fontName, fontSize);
    CleanupStack::PopAndDestroy();  // for fontName
    return fontSpec;
    }

// ----------------------------------------------------------
// CAknExEditorContainer4::SizeChanged()
// Called by framework when the view size is changed
// (other items were commented in a header).
// ----------------------------------------------------------
//
void CAknExEditorContainer4::SizeChanged()
```

```
    {
    }

// -----------------------------------------------------------
// CAknExEditorContainer4::CountComponentControls() const
// (other items were commented in a header).
// -----------------------------------------------------------
//
TInt CAknExEditorContainer4::CountComponentControls() const
    {
    return KNumberOfControls; // return nbr of controls inside this
container
    }

// -----------------------------------------------------------
// CAknExEditorContainer4::ComponentControl(TInt aIndex) const
// (other items were commented in a header).
// -----------------------------------------------------------
//
CCoeControl* CAknExEditorContainer4::ComponentControl(TInt aIndex)
const
    {
    switch (aIndex)
        {
        case EGTextEdPrompt:
            return iLabel;
        case EGTextEd:
            return iGTextEd;
        default:
            return NULL;
        }
    }

// -----------------------------------------------------------
// CAknExEditorContainer4::Draw(const TRect& aRect) const
// (other items were commented in a header).
// -----------------------------------------------------------
//
void CAknExEditorContainer4::Draw(const TRect& aRect) const
    {
    CWindowGc& gc = SystemGc();
    gc.SetPenStyle(CGraphicsContext::ENullPen);
    gc.SetBrushColor(KRgbGray);
    gc.SetBrushStyle(CGraphicsContext::ESolidBrush);
    gc.DrawRect(aRect);
    }

// -----------------------------------------------------------
// CAknExEditorContainer4::OfferKeyEventL(...)
// Notify key events to the editor.
// (other items were commented in a header).
// -----------------------------------------------------------
//
TKeyResponse CAknExEditorContainer4::OfferKeyEventL(
    const TKeyEvent& aKeyEvent, TEventCode aType)
    {
    return iGTextEd->OfferKeyEventL(aKeyEvent, aType);
    }


// ---------------------------------------------------------
// CAknExEditorContainer4::HandleControlEventL(
//     CCoeControl* aControl,TCoeEvent aEventType)
// (other items were commented in a header).
// ---------------------------------------------------------
//
void CAknExEditorContainer4::HandleControlEventL(
    CCoeControl* /*aControl*/,TCoeEvent /*aEventType*/)
    {
    }


// End of File
```

```
/*
*
==========================================================================
=====
*  Name     : CAknExEditorView4 from CAknView
*  Part of  : AknExEditor
*  Copyright (c) 2003 Nokia. All rights reserved.
*
==========================================================================
=====
*/

// INCLUDE FILES
#include <aknviewappui.h>
#include <avkon.hrh>

#include <AknPopup.h>
#include <AknLists.h>
#include <AknNoteDialog.h>
#include <eikedwin.h>  // for CEikEdwin
#include <eikgted.h>   // for CEikGlobalTextEditor

#include <L2Location.rsg>
#include "L2Location.hrh"
#include "AknExEditorView4.h"
#include "AknExEditorContainer4.h"
#include "L2LocationClient.h"
#include "L2LocationThreadSelector.h"

// ================= MEMBER FUNCTIONS =======================

// C++ default constructor can NOT contain any code, that
// might leave.
//
CAknExEditorView4::CAknExEditorView4()
    {
    }

// EPOC default constructor can leave.
void CAknExEditorView4::ConstructL(CL2LocationClient* c,
CL2LocationThreadSelector* t)
    {
    BaseConstructL(R_AKNEXEDITOR_VIEW4);
      this->iClient = c;
      this->iThreads = t;
    }

// Destructor
CAknExEditorView4::~CAknExEditorView4()
    {
    if (iContainer)
        {
        AppUi()->RemoveFromStack(iContainer);
        }

    delete iContainer;
    }
```

```
// -----------------------------------------------------------
// TUid CAknExEditorView4::Id()
// Returns Id of the view.
// -----------------------------------------------------------
TUid CAknExEditorView4::Id() const
    {
    return KViewId4;
    }

// -----------------------------------------------------------
// CAknExEditorView4::HandleCommandL(TInt aCommand)
// Handles commands
// -----------------------------------------------------------
void CAknExEditorView4::HandleCommandL(TInt aCommand)
    {
    switch (aCommand)
        {
            case EAknL2LocationCreateMessageSubmit:
                  ShowSingleItemPopupListL();
                  break;
        default:
            AppUi()->HandleCommandL(aCommand);
            break;
        }
    }
//-------------------------------------------------------------------
// CAknExPopListView::ShowSingleItemPopupList(TBool aShowTitle)
// shows "List pane for single item"
//-------------------------------------------------------------------
void CAknExEditorView4::ShowSingleItemPopupListL(){
      TBool aShowTitle = true;

    // Create listbox and PUSH it.
    CEikTextListBox* list = new(ELeave)
CAknSinglePopupMenuStyleListBox;
    CleanupStack::PushL(list);

    // Create popup list and PUSH it.
    CAknPopupList* popupList = CAknPopupList::NewL(
                                        list, R_AVKON_SOFTKEYS_OK_BACK,
                                        AknPopupLayouts::EMenuWindow);
    CleanupStack::PushL(popupList);

    // initialize listbox.
    list->ConstructL(popupList, CEikListBox::ELeftDownInViewRect);
    list->CreateScrollBarFrameL(ETrue);
    list->ScrollBarFrame()->SetScrollBarVisibilityL(
                                CEikScrollBarFrame::EOff,
                                CEikScrollBarFrame::EAuto);

    // Make listitems. and PUSH it
    CDesCArrayFlat* items  = new CDesCArrayFlat(1);//= //NULL;
            //iCoeEnv-
>ReadDesCArrayResourceL(R_AKNEXPOPLIST_MENU_ITEMS_TEST1);
      //items->AppendL(_L("Item 1"));
      //items->AppendL(_L("Item 2"));
```

```
        for( int i = 0; i < this->iThreads->numWriteItems; i++){
              items->AppendL(this->iThreads->writeItems[i]);
        }


      CleanupStack::PushL(items);

      // Set listitems.
      CTextListBoxModel* model = list->Model();
      model->SetItemTextArray(items);
      model->SetOwnershipType(ELbmOwnsItemArray);
      CleanupStack::Pop();

      // Set title if it is needed.
      if (aShowTitle)
          {
          TBuf<KMaxTitleLength> title;
          iCoeEnv->ReadResource(title, R_AKNEXPOPLIST_MENU_TITLE_TEST1);
          popupList->SetTitleL(title);
          }

      // Show popup list and then show return value.
      TInt popupOk = popupList->ExecuteLD();
      CleanupStack::Pop();            // popuplist
      if (popupOk)
          {
          TInt index = list->CurrentItemIndex();
          TBuf<KMessageLength> msg;
          //TBuf<KMessageLength> format;
          //iCoeEnv->ReadResource(format, R_AKNEXPOPLIST_SELECT_MESSAGE);
          //msg.Format(format, index + 1);
              TBuf16<120> text;
              this->iContainer->iGTextEd->GetText(text);
              if(this->iClient->CreateMessage(text, this->iThreads-
>readItems[index] ) == 0){
                    msg.Append(_L("Message Created Succesfully"));
              }else{
                    msg.Append(_L("Error Creating Message"));
              }
          CAknNoteDialog* dlg = new( ELeave ) CAknNoteDialog(
              CAknNoteDialog::ENoTone, CAknNoteDialog::ENoTimeout);
          dlg->PrepareLC( R_AKNEXPOPLIST_SELECT_MESSAGE_NOTE );
          dlg->SetTextPluralityL( EFalse );
          dlg->SetTextL( msg );
          dlg->RunLD();
          }

      CleanupStack::PopAndDestroy();  // list
      }

// ----------------------------------------------------------
// CAknExEditorView4::HandleClientRectChange()
// Handles client rect change.
// ----------------------------------------------------------
void CAknExEditorView4::HandleClientRectChange()
    {
    if (iContainer)
```

```
        {
        iContainer->SetRect(ClientRect());
        }
    }

// ---------------------------------------------------------
// CAknExEditorView4::DoActivateL(...)
// Creates the Container class object.
// ---------------------------------------------------------
void CAknExEditorView4::DoActivateL(const TVwsViewId& /*aPrevViewId*/,
                                    TUid /*aCustomMessageId*/,
                                    const TDesC8& /*aCustomMessage*/)
    {
    iContainer = new (ELeave) CAknExEditorContainer4;
    iContainer->SetMopParent(this);
    iContainer->ConstructL(ClientRect());
    AppUi()->AddToStackL(*this, iContainer);
    }

// ---------------------------------------------------------
// CAknExEditorView4::DoDeactivate()
// Deletes the Container class object.
// ---------------------------------------------------------
void CAknExEditorView4::DoDeactivate()
    {
    if (iContainer)
        {
        AppUi()->RemoveFromStack(iContainer);
        }

    delete iContainer;
    iContainer = NULL;
    }

// End of File
```

```
/*
*
======================================================================
=====
*  Name     : CL2LocationApp from L2LocationApp.cpp
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Implementation notes:
*
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
======================================================================
=====
*/

// INCLUDE FILES
#include    "L2LocationApp.h"
#include    "L2LocationDocument.h"

// ================= MEMBER FUNCTIONS =======================

// ---------------------------------------------------------
// CL2LocationApp::AppDllUid()
// Returns application UID
// ---------------------------------------------------------
//
TUid CL2LocationApp::AppDllUid() const
    {
    return KUidL2Location;
    }


// ---------------------------------------------------------
// CL2LocationApp::CreateDocumentL()
// Creates CL2LocationDocument object
// ---------------------------------------------------------
//
CApaDocument* CL2LocationApp::CreateDocumentL()
    {
    return CL2LocationDocument::NewL( *this );
    }

// ================= OTHER EXPORTED FUNCTIONS ==============
//
// ---------------------------------------------------------
// NewApplication()
// Constructs CL2LocationApp
// Returns: created application object
// ---------------------------------------------------------
//
EXPORT_C CApaApplication* NewApplication()
    {
    return new CL2LocationApp;
    }
```

```
// ----------------------------------------------------------
// E32Dll(TDllReason)
// Entry point function for EPOC Apps
// Returns: KErrNone: No error
// ----------------------------------------------------------
//
GLDEF_C TInt E32Dll( TDllReason )
    {
    return KErrNone;
    }

// End of File
```

```
/*
*
======================================================================
=====
*  Name     : CL2LocationAppUi from L2LocationAppUi.cpp
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Implementation notes:
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
======================================================================
=====
*/

// INCLUDE FILES
#include "L2LocationAppUi.h"
//#include "L2LocationView2.h"
#include <L2Location.rsg>
#include "l2location.hrh"
#include "AknExEditorView4.h"
#include "toneadapter.h"


#include <avkon.hrh>

// ================= MEMBER FUNCTIONS =======================
//
// ----------------------------------------------------------
// CL2LocationAppUi::ConstructL()
//
// ----------------------------------------------------------
//
void CL2LocationAppUi::ConstructL()
    {
    BaseConstructL();

      RDebug::Print(_L("starting..."));



    // Show tabs for main views from resources
    CEikStatusPane* sp = StatusPane();

    // Fetch pointer to the default navi pane control
    iNaviPane = (CAknNavigationControlContainer*)sp->ControlL(
        TUid::Uid(EEikStatusPaneUidNavi));

      iToneAdapter     = CToneAdapter::NewL(*this);


    // Tabgroup has been read from resource and it were pushed to the
navi pane.
    // Get pointer to the navigation decorator with the
ResourceDecorator() function.
```

```
    // Application owns the decorator and it has responsibility to
delete the object.

    iDecoratedTabGroup = iNaviPane->ResourceDecorator();
    if (iDecoratedTabGroup)
        {
        iTabGroup = (CAknTabGroup*) iDecoratedTabGroup-
>DecoratedControl();
        }



      view1 = new (ELeave) CL2LocationView;

    CleanupStack::PushL( view1 );
    view1->ConstructL(this->iToneAdapter, &this->iClient);
    AddViewL( view1 );      // transfer ownership to CAknViewAppUi
    CleanupStack::Pop();    // view1

    view3 = new (ELeave) CL2LocationView3;

      CleanupStack::PushL( view3 );
    view3->ConstructL();
    AddViewL( view3 );      // transfer ownership to CAknViewAppUi
    CleanupStack::Pop();    // view3

      view2 = new (ELeave) CL2LocationView2;

    CleanupStack::PushL( view2 );
    view2->ConstructL(this->iToneAdapter, view3);
    AddViewL( view2 );      // transfer ownership to CAknViewAppUi
    CleanupStack::Pop();    // view2



      CAknExEditorView4* view4 = new (ELeave) CAknExEditorView4;
    CleanupStack::PushL(view4);
    view4->ConstructL(&this->iClient, this->view3);
    CleanupStack::Pop();    // view4
    AddViewL(view4);        // transfer ownership to CAknViewAppUi

      this->view3->SetItems(_L8("bGeneral,bReminders,rRestaurant,"));

      iClient.ConstructL(this->view1, this->view2, this->view3);

    SetDefaultViewL(*view1);
      this->view1->SetWritable(true);
      //iClient = new (ELeave) CL2LocationClient();
      //CleanupStack::PushL(iClient);
      //iClient->ConstructL();
      //CleanupStack::Pop();

    }

// ----------------------------------------------------
// CL2LocationAppUi::~CL2LocationAppUi()
// Destructor
// Frees reserved resources
// ----------------------------------------------------
//
CL2LocationAppUi::~CL2LocationAppUi()
    {
//    delete iClient;
    delete iDecoratedTabGroup;
      delete iToneAdapter;
    iToneAdapter =  NULL;
   }

// ------------------------------------------------------------------
----------
// CL2LocationAppUi::DynInitMenuPaneL(TInt aResourceId,CEikMenuPane*
aMenuPane)
//  This function is called by the EIKON framework just before it
displays
//  a menu pane. Its default implementation is empty, and by overriding
it,
//  the application can set the state of menu items dynamically
according
//  to the state of application data.
// ------------------------------------------------------------------
----------
//
void CL2LocationAppUi::DynInitMenuPaneL(
    TInt /*aResourceId*/,CEikMenuPane* /*aMenuPane*/)
    {
    }

// ----------------------------------------------------
// CL2LocationAppUi::HandleKeyEventL(
//     const TKeyEvent& aKeyEvent,TEventCode /*aType*/)
// takes care of key event handling
// ----------------------------------------------------
//
TKeyResponse CL2LocationAppUi::HandleKeyEventL(
    const TKeyEvent& aKeyEvent,TEventCode /*aType*/)
    {
    if ( iTabGroup == NULL )
        {
        return EKeyWasNotConsumed;
        }

    TInt active = iTabGroup->ActiveTabIndex();
    TInt count = iTabGroup->TabCount();
      TBuf16<16> tgt;
    switch ( aKeyEvent.iCode )
        {
            case EKeyLeftArrow:
            if ( active > 0 )
                {
                active--;
                        iTabGroup->SetActiveTabByIndex( active );
                ActivateLocalViewL(TUid::Uid(iTabGroup-
>TabIdFromIndex(active)));
                        if(active ==0){
                              this->view1->SetWritable(true);
                              this->view2->SetWritable(false);
                        }else if(active ==1){
                              this->view1->SetWritable(false);
                              this->view2->SetWritable(true);
                        }

                        }

            break;
        case EKeyRightArrow:
            if( (active + 1) < count )
                {
                active++;

                iTabGroup->SetActiveTabByIndex( active );
                ActivateLocalViewL(TUid::Uid(iTabGroup-
>TabIdFromIndex(active)));
                        if(active ==0){
                              this->view1->SetWritable(true);
                              this->view2->SetWritable(false);
                        }else if(active == 1){
                              this->view1->SetWritable(false);
                              this->view2->SetWritable(true);
                        }

                        }
                  break;
            case 48:


                  tgt.Append(_L("count =  "));
                  //tgt.AppendNum( conCount );
                  RDebug::Print(_L(""));

                  this->view2->SetText(_L("0 Press~test10"));

                  break;
/*        case 49:
                  //iEikonEnv->InfoMsg(_L("1"));
                  this->iClient.Init();
                  count = 0;
                  break;
        case 50:
                  //iEikonEnv->InfoMsg(_L("2"));
                  this->iClient.Open(conCount);
                  break;
        case 51:
                  //iEikonEnv->InfoMsg(_L("3"));
                  this->iClient.Connect(conCount);
                  //count++;
                  break;
        case 52:
                  //iEikonEnv->InfoMsg(_L("4"));
                  this->iClient.SendAddresses();
                  break;
        case 53:
```

```
                    //iEikonEnv->InfoMsg(_L("5"));
                    this->iClient.StartLoop();
                    break;
          case 54:
                    //iEikonEnv->InfoMsg(_L("6"));
                    //this->iClient.OpenAll();
                    this->iToneAdapter->PlayL();
                    break;
          case 55:
                    //iEikonEnv->InfoMsg(_L("7"));
                    this->iClient.ConnectAll();
                    break;
          case 56:
                    conCount--;
                    break;
          case 57:
                    //iEikonEnv->InfoMsg(_L("9"));
                    conCount++;
                    break;*/
          default:
              return EKeyWasNotConsumed;
                    //TBuf16<16> tgt;
                    //tgt.Append(_L("key pressed = "));
                    //tgt.AppendNum( aKeyEvent.iCode );

                    //this->view1->SetMessage(tgt ,4);
              break;
          }
        if(active ==0){
              this->view1->SetWritable(true);
              this->view2->SetWritable(false);
        }else{
              this->view1->SetWritable(false);
              this->view2->SetWritable(true);
        }

    return EKeyWasConsumed;
    }

// ---------------------------------------------------------
// CL2LocationAppUi::HandleCommandL(TInt aCommand)
// takes care of command handling
// ---------------------------------------------------------
//
void CL2LocationAppUi::HandleCommandL(TInt aCommand)
    {
      if ( iTabGroup == NULL )
        {
        return;
        }

            TInt active = iTabGroup->ActiveTabIndex();
            TInt count = iTabGroup->TabCount();
            if(count <4){
                  return;
            }
    switch ( aCommand )
```

```
        {
            case EAknL2LocationCreateMessageBack:
                  {
                        if ( active > 0 )
                {
                active--;
                iTabGroup->SetActiveTabByIndex( active );
                ActivateLocalViewL(TUid::Uid(iTabGroup-
>TabIdFromIndex(active)));

                }
                        break;
                  }
            case EL2LocationCmdAppStartUp:
                  {
                        iTabGroup->SetActiveTabByIndex( 0);
                ActivateLocalViewL(TUid::Uid(iTabGroup-
>TabIdFromIndex(0)));
                        this->view1->SetWritable(true);
                        this->view2->SetWritable(false);

                        break;
                  }
            case EL2LocationCmdAppThread:
                  {
                        iTabGroup->SetActiveTabByIndex( 1);
                ActivateLocalViewL(TUid::Uid(iTabGroup-
>TabIdFromIndex(1)));
                        this->view1->SetWritable(false);
                        this->view2->SetWritable(false);
                        break;
                  }
            case EL2LocationCmdAppMessage:
                  {
                        iTabGroup->SetActiveTabByIndex( 2);
                ActivateLocalViewL(TUid::Uid(iTabGroup-
>TabIdFromIndex(2)));
                        this->view1->SetWritable(false);
                        this->view2->SetWritable(true);
                        break;
                  }
            case EL2LocationCmdAppCreate:
                  {
                        iTabGroup->SetActiveTabByIndex( 3);
                ActivateLocalViewL(TUid::Uid(iTabGroup-
>TabIdFromIndex(3)));
                        break;
                  }
            case EAknSoftkeyExit:
        case EEikCmdExit:
            {
//                      CleanupStack::PushL(iClient);
                        //iClient->Close();
//                      CleanupStack::Pop();
            Exit();
            break;
            }
        case EL2LocationCmdAppTest:
            {
            iEikonEnv->InfoMsg(_L("test"));
//                      CleanupStack::PushL(iClient);
//                      iClient.start(view1);
//                      CleanupStack::Pop();

                        break;
            }
        // TODO: Add Your command handling code here

        default:
            break;
        }
    }



// End of File
```

```
#include "L2LocationClient.h"
#include <utf.h>

CL2LocationClient::CL2LocationClient():CActive(CActive::EPriorityIdle){
      //int i;
      //for(i = 0; i < numServers; i++){


      //}
      createMessage = false;



}
//    for(i = 0; i < numServers; i++){
            //User::LeaveIfError(iSock[i].Open(iSocketServer,
KBTAddrFamily, KSockSeqPacket,KL2CAP));
//          connected[i] = FALSE;
//    }
      //User::SetTrapHandler(this);

//}

CL2LocationClient::~CL2LocationClient(){
      CActive::Cancel();
}


void CL2LocationClient::updateLocation(){
            int valid;
            int pos[numServers][2];
            int str[numServers] ;
            valid = 0;
            float x = 0;
            float y = 0;

            for( int ctr = 0; ctr <numServers; ctr++){

                        if(strengths[ctr]<9){
                            pos[valid][0] = locs[ctr][0];
                            pos[valid][1] = locs[ctr][1];
                            str[valid] = this->strengths[ctr];
                            valid++;

                        }

            }
            if(valid == 0){

                  iListener->SetMessage(_L("middle"),5);
            }else{
                  float s = 0;
                  float st = 0;
                  int zero = -1;
                  for(int ctr = 0; ctr< valid; ctr++){
                        if(ctr ==0){
                              strongSock = &iSock[ctr];
                        }else{
                              if(str[ctr] < str[ctr-1]){
                                    strongSock = &iSock[ctr];
                              }
                        }
                        if(str[ctr] == 0){
                              zero = ctr;

                        }else{
                              st = 1/(float)(str[ctr]);
      //(10-(float)str[ctr]);
                              x += (float)pos[ctr][0]*st;
                              y += (float)pos[ctr][1]*st;
                              s += st;
                        }
                  }
                  if(zero != -1){
                        x = (float)pos[zero][0];
                        y = (float)pos[zero][1];
                  }else if(s != 0){
                        x/=s;
                        y/=s;
                  }
                  currentX = (int)x;
                  currentY = (int)y;
            }
            /*int *str = strengths;
                  float x = str[1]
==0?(float)0.0:(float)str[1]/(float)(str[1]+str[2])*(float)10.0;
                  float y = str[1]
==0?(float)0.0:(float)str[1]/(float)(str[1]+str[0])*(float)10.0;
                  currentX = (int) x;
                  currentY = (int) y;

                  int stSock = str[0] <str[1]?0:1;
                  stSock = str[stSock]<str[2]?stSock:2;
                  strongSock = &iSock[stSock];
                  */
            TBuf16<16> tgt;
            tgt.AppendNum(currentX);
            tgt.Append(_L(" , "));
            tgt.AppendNum(currentY);
            iListener->SetMessage(tgt ,5);
}

void CL2LocationClient::Close(){
      int i;
      for( i = 0; i < numServers; i++){
            iSock[i].CancelAll();
      }
      iSocketServer.Close();
      for(i = 0; i < numServers; i++){
            //iSock[i].Shutdown( iSock[i].EImmediate, iStatus);
            //User::WaitForRequest(iStatus);
            iSock[i].Close();
      }
```

```
}
void CL2LocationClient::start(CL2LocationClientListener* lcl){
      iListener = lcl;
      lcl->SetMessage(_L("unknown"),4);
      TInt64 pondloc1(0x0090,0x4B205C5A);
      TInt64 pondloc2(0x000E,0x2600077E);
      TInt64 pondloc4(0x000C,0x769A3C34);
      TInt64 pondloc5(0x000C,0x769A3E62);

      servers[0] = pondloc1;
      servers[1] = pondloc2;
      servers[2] = pondloc4;
      servers[3] = pondloc5;

      locs[0][0] = -10;
      locs[2][0] = locs[1][0] = 10;
      locs[1][1] = locs[0][1] = -7;
      locs[2][1] = 7;

/*    if(numServers == 4){
            locs[3][0] = -10;
            locs[3][1] = 7;
      }
      */
      /*connect(0);
      checkError(iStatus,3);
      sendAddress(0);
      checkError(iStatus,4);
      connected[0] = TRUE;
      count = 0;
      queryWrite(0);
      //checkError(iStatus,5);
      //queryRead(0);
      //checkError(iStatus,6);
      //process(0);
      */

      int numCons = 0;
      lcl->SetMessage(_L("unknown2"),4);
      for(int i = 0; i <numServers; i++){
            iSock[i].Open(iSocketServer, KBTAddrFamily,
KSockSeqPacket,KL2CAP);
            TBuf16<16> tgt;
            tgt.Append(_L("loop "));
            tgt.AppendNum(i );
            iListener->SetMessage(tgt ,4);
            //connected[i] = FALSE;
            if(iStatus != KErrNone){
                  TBuf16<16> tgt;
                  tgt.Append(_L("con error = "));
                  tgt.AppendNum(iStatus.Int() );
                  iListener->SetMessage(tgt ,i);
            }else{
                  lcl->SetMessage(_L("starting..."),i);
                  TBuf16<16> tgt;
                  tgt.Append(_L("loop in"));
                  tgt.AppendNum(i );
```

```
                    iListener->SetMessage(tgt ,4);
                    connect(i);
                    /*TRAPD(error,);
                    if(error != KErrNone){
                          TBuf16<16> tgt;
                          tgt.Append(_L("trap error = "));
                          tgt.AppendNum(error );
                          iListener->SetMessage(tgt ,i);
                    }else{
                          iListener->SetMessage(_L("ok"),i);
                    }
                    */
                    if(iStatus == KErrNone){
                          sendAddress(i);

                          if(iStatus == KErrNone){
                                connected[i] = TRUE;
                                iListener->SetMessage(_L("address sent")
,i);

                                numCons++;
                          }else{
                                TBuf16<16> tgt;
                                tgt.Append(_L("sa error = "));
                                tgt.AppendNum(iStatus.Int() );
                                iListener->SetMessage(tgt ,i);

                          }
                    }else{
                          TBuf16<16> tgt;
                          tgt.Append(_L("con error = "));
                          tgt.AppendNum(iStatus.Int() );
                          iListener->SetMessage(tgt ,i);

                    }


              }
              //else{
              //    numCons++;
              //}


      }
      //count = 0;

      //if(numCons == numServers){
            queryWrite(0);
      //}


      //queryRead();
      //process();
}

void CL2LocationClient::DoCancel(){
      CActive::Cancel();
      for(int i = 0; i < numServers; i++){
            iSock[i].CancelAll();
      }
      iSocketServer.Close();
}

void CL2LocationClient::process(int i){
      TBuf16<16> tgt;
      tgt.Append(_L("rssi = "));
      strengths[i] = (int)iQueryBuf[0];
      tgt.AppendNum(strengths[i]);
    iListener->SetMessage(tgt ,i);

}


int CL2LocationClient::CreateMessage(TPtrC msg, TPtrC thread){
      //if( strongSock){

                  //TBuf8<128> tgt;
                  cMsgBuf.Append(_L("c"));
                  cMsgBuf.Append((char)currentX);
                  cMsgBuf.Append((char)currentY);
                  cMsgBuf.Append(msg);
                  cMsgBuf.Append('~');
                  cMsgBuf.Append(thread);
                  createMessage = true;
                  return 0;
}
void CL2LocationClient::SendMessage(){
            iState = ESendingMessage;
                  strongSock->Write(cMsgBuf,iStatus);
                  SetActive();

}

void CL2LocationClient::RunL(){
      switch(iState){
      case ESendingMessage:
            createMessage = false;
            this->queryMessage();
            break;
      case EQueryingWrite:
            {
                  this->queryRead(iterator);
            }
            break;
      case EQueryingRead:
            {
                  process(iterator);
                  iterator++;
                  if(iterator < numServers){
                        this->queryWrite(iterator);
                  }else{
                        this->updateLocation();
                        iterator = 0;
                        if(createMessage){
                              createMessage = false;
                              this->SendMessage();
                        }else{
                              this->queryMessage();
                        }
                  }
            }
            break;
      case EQueryingMessage:
            {
                  this->readMessage();
            }
            break;
      case EReadingMessage:
            {
                  TBuf16<64> iBuf16;

      CnvUtfConverter::ConvertToUnicodeFromUtf8(iBuf16,this-
>iMessageBuf);
                  this->iDisplay->SetText(iBuf16);
                  this->queryWrite(iterator);
            }
            break;
      case EQueryingThreads:
            {
                  this->readThreads();
            }
            break;
      case EReadingThreads:
            {
                  this->iView3->SetItems(this->iMessageBuf);
                  this->queryWrite(iterator);
            }
            break;

      }

}

      /*
      //checkError(iStatus, 2);



      //if(connected){


            if(count >= numServers ){
                  if(connected[count-numServers]){
                        if(iStatus == KErrNone){
                              process(count-numServers);
                        }else{
                              //TBuf16<16> tgt;
                              //tgt.Append(_L("pr error = "));
                              //tgt.AppendNum(iStatus.Int() );
                              //iListener->SetMessage(tgt ,count-
numServers);
```

```
                        }
                }else{
                        iListener->SetMessage(_L("Not
Connected"),count-numServers);
                }
            }
            updateLocation();
            count++;
            //count%=2*numServers;
            if(count == 2*numServers+2){

                  //Get a iBuf16 from a iBuf8 (data are modified)
                  TBuf16<64> iBuf16;

      CnvUtfConverter::ConvertToUnicodeFromUtf8(iBuf16,this-
>iMessageBuf);
                  this->iDisplay->SetText(iBuf16);
                  count = 0;
            }
            if( count < numServers){

                        queryWrite(count);

            }else if(count <2*numServers){
                        queryRead(count-numServers);
            }else if(count <2*numServers+1){
                  queryMessage();
            }else{
                  readMessage();
            }
      //}

}
*/
void CL2LocationClient::queryWrite(int i){
      if(i>=0 && i<numServers){
            if(connected[i]){

                  _LIT8(KDataToWrite,"r");
                  iState = EQueryingWrite;
                  iSock[i].Write(KDataToWrite,iStatus);
                  SetActive();
            }
      }

}
void CL2LocationClient::queryRead(int i){
      //__ASSERT_ALWAYS(!IsActive(), User::Panic(_L"bad",_L"worse"));
      if(i>=0 &&i<numServers){

            if(connected[i]){
                  iState = EQueryingRead;
                  iSock[i].Read(iQueryBuf,iStatus);
                  SetActive();
            }
      }
```

```
}
void CL2LocationClient::queryMessage(){

            if( strongSock){

                  TBuf8<128> tgt;
                  tgt.Append(_L("m"));
                  tgt.Append((char)this->iDisplay->GetMsgNum());
                  tgt.Append(_L(" "));
                  tgt.Append(this->iView3->GetWhereClause());
                  tgt.Append(_L(" ORDER BY ( x - "));
                  tgt.AppendNum(currentX);
                  tgt.Append(_L(" ) * ( x - "));
                  tgt.AppendNum(currentX);
                  tgt.Append(_L(" ) + ( y - "));
                  tgt.AppendNum(currentY);
                  tgt.Append(_L(" ) * ( y - "));
                  tgt.AppendNum(currentY);
                  tgt.Append(_L(" )\0"));
                  iState = EQueryingMessage;
                  strongSock->Write(tgt,iStatus);
                  SetActive();
            }


}
void CL2LocationClient::readMessage(){


            if(strongSock){
                  iState = EReadingMessage;

                        strongSock->Read(iMessageBuf,iStatus);


                  SetActive();
            }


}

void CL2LocationClient::queryThreads(){
      if(!strongSock){
            strongSock = &iSock[0];
      }
            if( strongSock){

                  iListener->SetMessage(_L("querying for threads"),4);
                  TBuf8<64> tgt;
                  tgt.Append(_L("t"));
                  iState = EQueryingThreads;
                  strongSock->Write(tgt,iStatus);
                  SetActive();
            }


}
```

```
void CL2LocationClient::readThreads(){


            if(strongSock){
                  iListener->SetMessage(_L("reading for threads"),4);

                  iState = EReadingThreads;

                        strongSock->Read(iMessageBuf,iStatus);


                  SetActive();
            }


}


void CL2LocationClient::connect(int i){
      //iSock[i].CancelAll();

      //TInt64 tnumber(0x0090, 0x4b209b8c);
       //TInt64 tnumber(0x000E,0x2600077E);
       TBTDevAddr devaddr(servers[i]);
     TBTSockAddr address;
     address.SetBTAddr(devaddr);
     address.SetPort(12);
       iListener->SetMessage(_L("connecting......"),i);
      iSock[i].Connect(address, iStatus);
      //SetActive();



      User::WaitForRequest(iStatus);
      checkError( iStatus,1);
      iListener->SetMessage(_L("connected"),i);
}
void CL2LocationClient::sendAddress(int i){
      _LIT8(KDataToWrite,"00:60:57:e1:42:7f");
      iSock[i].Write(KDataToWrite,iStatus);
      iListener->SetMessage(_L("sending address"),i);
      User::WaitForRequest(iStatus);
      iListener->SetMessage(_L("address sent"),i);

      //SetActive();
}


/*

void CL2LocationClient::Trap(){
      this->iListener->SetMessage(_L("Trap"),4);
}
void CL2LocationClient::UnTrap(){
      this->iListener->SetMessage(_L("UnTrap"),4);
}
void CL2LocationClient::Leave(int i){
      this->iListener->SetMessage(_L("Leave"),4);
}
*/

void CL2LocationClient::ConstructL(CL2LocationClientListener* lcl,
CL2LocationClientDisplay* lcd,CL2LocationThreadSelector* l2c3){
      iListener = lcl;
      iDisplay = lcd;
      iView3 = l2c3;

}
void CL2LocationClient::Init(){
      CActiveScheduler::Add(this);
      int result = StartC32();
      if(result != KErrNone && result != KErrAlreadyExists){
            User::Leave(result);
      }
      User::LeaveIfError(iSocketServer.Connect());
      User::LeaveIfError(iSocketServer.FindProtocol(_L("BTLinkManager")
, protocolInfo));

      TInt64 pondloc1(0x0090,0x4B205C5A);
      TInt64 pondloc2(0x000E,0x2600077E);
      TInt64 pondloc4(0x000C,0x769A3C34);
      TInt64 pondloc5(0x000C,0x769A3E62);

      servers[0] = pondloc1;
      servers[1] = pondloc2;
      servers[2] = pondloc4;
      //servers[3] = pondloc5;

      locs[0][0] = 10;
      locs[2][0] = locs[1][0] = 0;

      locs[1][1] = locs[0][1] = 0;;
      locs[2][1] = 10;



      for(int i = 0; i < numServers; i++){

            TBTDevAddr devaddr(servers[i]);

            address[i].SetBTAddr(devaddr);
            address[i].SetPort(12);
      }
      this->iListener->SetMessage(_L("Initialized"),5);



}
void CL2LocationClient::OpenAll(){
      for(int i = 0; i < numServers; i++){
            Open(i);
      }
}
void CL2LocationClient::Open(TInt i){
```

```
            User::LeaveIfError(iSock[i].Open(iSocketServer,
KBTAddrFamily, KSockSeqPacket,KL2CAP));
            connected[i] = FALSE;
            TBuf16<16> tgt;
            tgt.Append(_L("Opened: "));
            tgt.AppendNum(i );
            iListener->SetMessage(tgt ,i);


}
void CL2LocationClient::ConnectAll(){
      for(int i = 0; i < numServers; i++){
            Connect(i);
      }
}
void CL2LocationClient::Connect(TInt i){


            iSock[i].Connect(address[i], iStatus);
            User::WaitForRequest(iStatus);
            if(iStatus == KErrNone){
                  iListener->SetMessage(_L("connected"),i);
                  connected[i] = TRUE;
            }else{
                  TBuf16<16> tgt;
                  tgt.Append(_L("con error = "));
                  tgt.AppendNum(iStatus.Int() );
                  iListener->SetMessage(tgt ,i);
            }


}
void CL2LocationClient::SendAddresses(){
      for(int i = 0; i < numServers; i++){
            sendAddress(i);
      }

}
void CL2LocationClient::StartLoop(){
      //queryWrite(0);
      iterator = 0;
      iState = EIdle;
      this->queryThreads();
      //this->queryWrite(0);

}
```

```
/*
*
========================================================================
=====
*  Name     : CL2LocationContainer2 from L2LocationContainer2.cpp
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Implementation notes:
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
========================================================================
=====
*/

// INCLUDE FILES
#include "L2LocationContainer2.h"

#include <eiklabel.h>  // for example label control
#include <barsread.h>  // for resource reader
#include <eikedwin.h>  // for CEikEdwin
#include <eikenv.h>

#include <L2Location.rsg>
#include "L2Location.hrh"

// ================= MEMBER FUNCTIONS =======================

// ---------------------------------------------------------
// CL2LocationContainer2::ConstructL(const TRect& aRect)
// EPOC two phased constructor
// ---------------------------------------------------------
//
void CL2LocationContainer2::ConstructL(const TRect& aRect,
CToneAdapter* t, CL2LocationThreadSelector* h)
    {
    CreateWindowL();

    iLabel = new (ELeave) CEikLabel;
    iLabel->SetContainerWindowL( *this );
    iLabel->SetTextL( _L("Example View 2") );

      iLabel2 = new (ELeave) CEikLabel;
    iLabel2->SetContainerWindowL( *this );
    iLabel2->SetTextL( _L("Example View 2") );
      iLabel2->SetFont(CEikonEnv::Static()->DenseFont());

      iLabel3 = new (ELeave) CEikLabel;
    iLabel3->SetContainerWindowL( *this );
    iLabel3->SetTextL( _L("Example View 2") );
      iLabel3->SetFont(CEikonEnv::Static()->DenseFont());


      TResourceReader reader;
    iCoeEnv->CreateResourceReaderLC(reader, R_AKNEXEDITOR_VIEW1_EDWIN);
    iEdwin = new (ELeave) CEikEdwin;
    iEdwin->SetContainerWindowL(*this);
    iEdwin->ConstructFromResourceL(reader);
    CleanupStack::PopAndDestroy();  // Resource reader
    iEdwin->SetExtent(EDWIN_POS, iEdwin->MinimumSize());


      this->iToneAdapter = t;
      this->iThread = h;

         //iEdwin->SetFocus(ETrue);
      //TBuf16<48> initBuf;
      //initBuf.Append((char)1);
      //initBuf.Append();

      msgNum = 1;
    SetText(_L("Remember to pick up milk!\1~Reminders~Mom"));
            //initBuf);
      //iEdwin->SetTextL(tgt );

    //iToDoLabel = new (ELeave) CEikLabel;
    //iToDoLabel->SetContainerWindowL( *this );
    //iToDoLabel->SetTextL( _L("Add Your controls\n here") );

    SetRect(aRect);
    ActivateL();
      }


CL2LocationContainer2::CL2LocationContainer2():iEdwin(NULL){}


// Destructor
CL2LocationContainer2::~CL2LocationContainer2()
    {
    delete iEdwin;
    delete iLabel;
      delete iLabel2;
    delete iLabel3;
    }

// ---------------------------------------------------------
// CL2LocationContainer2::SizeChanged()
// Called by framework when the view size is changed
// ---------------------------------------------------------
//
void CL2LocationContainer2::SizeChanged()
    {
    // TODO: Add here control resize code etc.
    iEdwin->SetExtent(TPoint(10,25), iEdwin->MinimumSize() );
    iLabel->SetExtent( TPoint(5,1), TSize(180,12) );
      iLabel2->SetExtent( TPoint(5,13), TSize(180,10) );
    iLabel3->SetExtent( TPoint(0,133), TSize(200,10) );
    }

// ---------------------------------------------------------
// CL2LocationContainer2::CountComponentControls() const
// ---------------------------------------------------------
```

```
//
TInt CL2LocationContainer2::CountComponentControls() const
    {
    return 4; // return nbr of controls inside this container
    }

// ---------------------------------------------------------
// CL2LocationContainer2::ComponentControl(TInt aIndex) const
// ---------------------------------------------------------
//
CCoeControl* CL2LocationContainer2::ComponentControl(TInt aIndex) const
    {
    switch ( aIndex )
        {
        case 0:
            return iEdwin;
        case 1:
            return iLabel;
        case 2:
            return iLabel2;
        case 3:
            return iLabel3;
        default:
            return NULL;
        }
    }

// ---------------------------------------------------------
// CL2LocationContainer2::Draw(const TRect& aRect) const
// ---------------------------------------------------------
//
void CL2LocationContainer2::Draw(const TRect& aRect) const
    {
    CWindowGc& gc = SystemGc();
    // TODO: Add your drawing code here
    // example code...
    gc.SetPenStyle( CGraphicsContext::ENullPen );
    gc.SetBrushColor( KRgbGray );
    gc.SetBrushStyle( CGraphicsContext::ESolidBrush );
    gc.DrawRect( aRect );
    }

// ---------------------------------------------------------
// CL2LocationContainer2::HandleControlEventL(
//     CCoeControl* aControl,TCoeEvent aEventType)
// ---------------------------------------------------------
//
void CL2LocationContainer2::HandleControlEventL(
    CCoeControl* /*aControl*/,TCoeEvent /*aEventType*/)
    {

    // TODO: Add your control event handler code here
    }


// ---------------------------------------------------------
// CAknExEditorContainer1::OfferKeyEventL(...)
// Notify key events to editors.
// (other items were commented in a header).
// --------------------------------------------------------
//
/*
TKeyResponse CL2LocationContainer2::OfferKeyEventL(
    const TKeyEvent& aKeyEvent, TEventCode aType)
    {



    return EKeyWasNotConsumed;
    }
*/
TKeyResponse CL2LocationContainer2::OfferKeyEventL(
    const TKeyEvent& aKeyEvent, TEventCode aType)
    {
            switch ( aKeyEvent.iCode )
        {
        case EKeyUpArrow:
                  msgNum++;
                  this->DrawNow();
                  break;
            case EKeyDownArrow:
                  msgNum--;
                  this->DrawNow();
                  break;
            }
            if (iEdwin->IsFocused())
            {
            return iEdwin->OfferKeyEventL(aKeyEvent, aType);
            }
        return EKeyWasNotConsumed;
}


void CL2LocationContainer2::SetText(TPtrC msg){
      int i=0;
      int f = msg.Length();
      int b = 0;
      int num = 1;

      if(msg.Length() >1)
      {

            for(i = 1; msg[i] != '~' &&  i < msg.Length()-1 && i< 128
;i++){
            }
            num = msg[i-1];
      }
      TBuf16<32> msgof;
      msgof.Append(_L("Message "));
      msgof.AppendNum(msgNum);
      msgof.Append(_L(" of "));
      msgof.AppendNum(num);
      this->iLabel3->SetTextL(msgof);
      f = i-1;
      i++;
      b = msg.Length()-i;
      if(f>=msg.Length()){
            iEdwin->SetTextL(&msg);
      }else if(f>0){
            iEdwin->SetTextL(&(msg.Left(f)) );
      }
      if(b>0){
            TPtrC16 right = msg.Right(b);
            if(right.Length() >0){
                  for(i = 0; right[i] != '~' &&  i < right.Length()-1
&& i< 128 ;i++){
                  }
            }else{
                  i = 0;
            }
            int t = i;
            i++;
            int s = right.Length() - i;

            TBuf16<48> thread;
            thread.Append(_L("Thread: "));
            if(t >0){
                  thread.Append(right.Left(t));
            }
            iLabel->SetTextL(thread);
            TBuf16<48> sender;

            sender.Append(_L("Sender: "));
            if(s > 0){
                  sender.Append(right.Right(s));
            }
            iLabel2->SetTextL(sender);

            for(int c = 0; c < this->iThread->numReadItems; c++){
                  if(this->iThread->alert[c]){
                        if(!right.Left(t).Compare( this->iThread-
>readItems[c])){

                              this->iToneAdapter->PlayL();
                        }else{
                              RDebug::Print(this->iThread-
>readItems[c]);


                        }
                  }

            }

      }

      if(writable){
            if(IsReadyToDraw()){
                  this->DrawNow();
            }
      }
```

```
}
int CL2LocationContainer2::GetMsgNum(){
      return msgNum;
}
void CL2LocationContainer2::SetWritable(bool w){
      writable = w;
}
// End of File
```

```
/*
*
======================================================================
=====
*  Name      : CL2LocationContainer3 from L2LocationContainer3.cpp
*  Part of   : L2Location
*  Created   : 4/numItems/3004 by
*  Implementation notes:
*     Initial content was generated by Series 60 AppWizard.
*  Version   :
*  Copyright:
*
======================================================================
=====
*/

// INCLUDE FILES
#include "L2LocationContainer3.h"

#include <eiklabel.h>  // for example label control
#include <barsread.h>  // for resource reader
#include <eikedwin.h>  // for CEikEdwin
#include <eikenv.h>

#include <L2Location.rsg>
#include "L2Location.hrh"
//#include <coeaui.h >


// ================= MEMBER FUNCTIONS =======================

// ---------------------------------------------------------
// CL2LocationContainer3::ConstructL(const TRect& aRect)
// EPOC two phased constructor
// ---------------------------------------------------------
//
void CL2LocationContainer3::ConstructL(const TRect& aRect)
    {
    CreateWindowL();

      //view3->numItems = 2;
      //SetItems(_L8("general,IBM,"));
      this->view3->checked[0] = true;
      SetRect(aRect);
    ActivateL();
    }

CL2LocationContainer3::CL2LocationContainer3(CL2LocationThreadSelector*
ts){
      view3 = ts;
}

// Destructor
CL2LocationContainer3::~CL2LocationContainer3()
    {

    }
```

```
// -------------------------------------------------------
// CL2LocationContainer3::SizeChanged()
// Called by framework when the view size is changed
// -------------------------------------------------------
//
void CL2LocationContainer3::SizeChanged()
    {

    }

// -------------------------------------------------------
// CL2LocationContainer3::CountComponentControls() const
// -------------------------------------------------------
//
TInt CL2LocationContainer3::CountComponentControls() const
    {
    return 0; // return nbr of controls inside this container
    }

// -------------------------------------------------------
// CL2LocationContainer3::ComponentControl(TInt aIndex) const
// -------------------------------------------------------
//
CCoeControl* CL2LocationContainer3::ComponentControl(TInt ) const
    {


            return NULL;

    }

// -------------------------------------------------------
// CL2LocationContainer3::Draw(const TRect& aRect) const
// -------------------------------------------------------
//
void CL2LocationContainer3::Draw(const TRect& aRect) const
    {
            RDebug::Print(_L("Container3 Draw"));

            CWindowGc& gc = SystemGc();
            // TODO: Add your drawing code here
            // example code...
            gc.SetPenStyle( CGraphicsContext::ESolidPen );
            gc.SetBrushColor( KRgbGray );
            gc.SetBrushStyle( CGraphicsContext::ESolidBrush );
            gc.DrawRect( aRect );
            gc.SetBrushColor( KRgbBlue );
            int w = aRect.Width();
            gc.DrawRect(TRect(0,view3->selection*12+1,w,view3-
>selection*12+13));
            gc.UseFont(CEikonEnv::Static()->DenseFont());
            RDebug::Print(_L("Container3 Draw Text"));
            for(int i = 0; i <view3->numReadItems ; i++){
                gc.SetPenSize(TSize(3,3));

                if(view3->checked[i]){
                        gc.SetPenColor( KRgbRed );
                        gc.DrawLine(TPoint(w-15,i*12+7), TPoint(w-
10,i*12+10));

                        gc.DrawLine( TPoint(w-10,i*12+10), TPoint(w-
5,i*12+2));
                  }
                  gc.SetPenSize(TSize(2,1));

                  if(view3->alert[i]){
                        gc.SetPenColor( KRgbGreen );
                        gc.DrawLine( TPoint(w-25,i*12+2), TPoint(w-
20,i*12+10));

                        gc.DrawLine( TPoint(w-25,i*12+2), TPoint(w-
30,i*12+10));

                        gc.DrawLine( TPoint(w-22,i*12+7), TPoint(w-
27,i*12+7));

                  }
                  if(i == view3->selection){
                        gc.SetPenColor(KRgbWhite);

                  }else{
                        gc.SetPenColor( KRgbBlack );
                  }
                  RDebug::Print(_L("Container3 Draw t"));
                  gc.DrawText(view3->readItems[i], TPoint(5, i*12+12));



            }

    }


// ---------------------------------------------------------
// CAknExEditorContainer1::OfferKeyEventL(...)
// Notify key events to editors.
// (other items were commented in a header).
// ---------------------------------------------------------
//
TKeyResponse CL2LocationContainer3::OfferKeyEventL(
    const TKeyEvent& aKeyEvent, TEventCode )
    {
            switch ( aKeyEvent.iCode )
        {
        case EKeyUpArrow:
                  view3->selection--;
                  view3->selection%=view3->numReadItems;
                  this->DrawNow();
                  break;
            case EKeyDownArrow:
                  view3->selection++;
                  view3->selection%=view3->numReadItems;
                  this->DrawNow();
                  break;
            case 63557:
                    view3->checked[view3->selection] = !view3-
>checked[view3->selection];
                    this->DrawNow();
                    break;
              case 50:
                    view3->alert[view3->selection] = !view3->alert[view3-
>selection];
                    this->DrawNow();
                    break;
              default:
                    {
                          TBuf16<32> tgt;
                          tgt.Append(_L("Unhandled key event: "));
                          tgt.AppendNum(aKeyEvent.iCode );
                          RDebug::Print(tgt);

                          return EKeyWasNotConsumed;
                    }
              }
              return EKeyWasConsumed;
    }



// End of File
```

```
/*
*
========================================================================
=====
*  Name     : CL2LocationContainer from L2LocationContainer.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Implementation notes:
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
========================================================================
=====
*/

// INCLUDE FILES
#include "L2LocationContainer.h"

#include <eiklabel.h>  // for example label control


// ================= MEMBER FUNCTIONS =======================

// ---------------------------------------------------------
// CL2LocationContainer::ConstructL(const TRect& aRect)
// EPOC two phased constructor
// ---------------------------------------------------------
//
void CL2LocationContainer::ConstructL(const TRect& aRect)
    {
    CreateWindowL();

    TBuf16<16> tgt;
      tgt.Append(_L("startup________"));



    iLabel1 = new (ELeave) CEikLabel;
    iLabel1->SetContainerWindowL( *this );
    iLabel1->SetTextL( tgt);

      iLabel2 = new (ELeave) CEikLabel;
    iLabel2->SetContainerWindowL( *this );
    iLabel2->SetTextL( tgt );

      iLabel3 = new (ELeave) CEikLabel;
    iLabel3->SetContainerWindowL( *this );
    iLabel3->SetTextL(tgt );

      iLabel4 = new (ELeave) CEikLabel;
    iLabel4->SetContainerWindowL( *this );
    iLabel4->SetTextL(tgt);

      iLabel5 = new (ELeave) CEikLabel;
    iLabel5->SetContainerWindowL( *this );
    iLabel5->SetTextL(tgt);
```

```
      iLabel6 = new (ELeave) CEikLabel;
    iLabel6->SetContainerWindowL( *this );
    iLabel6->SetTextL(tgt);



      SetRect(aRect);
    ActivateL();
    }

// Destructor
CL2LocationContainer::~CL2LocationContainer()
    {
    delete iLabel1;
    delete iLabel2;
    delete iLabel3;
    delete iLabel4;
    delete iLabel5;
      delete iLabel6;
    }

// ---------------------------------------------------------
// CL2LocationContainer::SizeChanged()
// Called by framework when the view size is changed
// ---------------------------------------------------------
//
void CL2LocationContainer::SizeChanged()
    {
    iLabel1->SetExtent( TPoint(10,10), iLabel1->MinimumSize() );
    iLabel2->SetExtent( TPoint(10,30), iLabel2->MinimumSize() );
    iLabel3->SetExtent( TPoint(10,50), iLabel3->MinimumSize() );
    iLabel4->SetExtent( TPoint(10,70), iLabel4->MinimumSize() );
    iLabel5->SetExtent( TPoint(10,90), iLabel5->MinimumSize() );
    iLabel6->SetExtent( TPoint(10,110), iLabel6->MinimumSize() );

    }

// ---------------------------------------------------------
// CL2LocationContainer::CountComponentControls() const
// ---------------------------------------------------------
//
TInt CL2LocationContainer::CountComponentControls() const
    {
    return 6; // return nbr of controls inside this container
    }

// ---------------------------------------------------------
// CL2LocationContainer::ComponentControl(TInt aIndex) const
// ---------------------------------------------------------
//
CCoeControl* CL2LocationContainer::ComponentControl(TInt aIndex) const
    {
    switch ( aIndex )
        {
        case 0:
            return iLabel1;
        case 1:
            return iLabel2;
        case 2:
            return iLabel3;
        case 3:
            return iLabel4;
        case 4:
            return iLabel5;
        case 5:
            return iLabel6;
        default:
            return NULL;
        }
    }

// ---------------------------------------------------------
// CL2LocationContainer::Draw(const TRect& aRect) const
// ---------------------------------------------------------
//
void CL2LocationContainer::Draw(const TRect& aRect) const
    {
    CWindowGc& gc = SystemGc();
    // TODO: Add your drawing code here
    // example code...
    gc.SetPenStyle( CGraphicsContext::ENullPen );
    gc.SetBrushColor( KRgbGray );
    gc.SetBrushStyle( CGraphicsContext::ESolidBrush );
    gc.DrawRect( aRect );
    }

// ---------------------------------------------------------
// CL2LocationContainer::HandleControlEventL(
//     CCoeControl* aControl,TCoeEvent aEventType)
// ---------------------------------------------------------
//
void CL2LocationContainer::HandleControlEventL(
    CCoeControl* /*aControl*/,TCoeEvent /*aEventType*/)
    {
    // TODO: Add your control event handler code here
    }


// End of File

void CL2LocationContainer::SetMessage(TPtrC msg, int i){
      switch(i+1){
      case 1:
            iLabel1->SetTextL(msg);
            break;
      case 2:
            iLabel2->SetTextL(msg);
            break;
      case 3:
            iLabel3->SetTextL(msg);
            break;
      case 4:
            iLabel4->SetTextL(msg);
            break;
        case 5:
            iLabel5->SetTextL(msg);
            break;
        case 6:
            iLabel6->SetTextL(msg);
            break;
        }
        if(writable){
            if(IsReadyToDraw()){
                this->DrawNow();
            }
        }
}

void CL2LocationContainer::SetWritable(bool w){
      writable = w;
}

void CL2LocationContainer::SetToneAdapter(CToneAdapter* t){
      this->iToneAdapter = t;
}
void CL2LocationContainer::SetClient(CL2LocationClient* c){
      this->iClient = c;
}

TKeyResponse CL2LocationContainer::OfferKeyEventL(
    const TKeyEvent& aKeyEvent,TEventCode /*aType*/)
    {
      writable = true;
    TBuf16<16> tgt;
    switch ( aKeyEvent.iCode )
        {

            case 48:

                  tgt.Append(_L("count =  "));
                  tgt.AppendNum( conCount );
                  this->SetMessage(tgt ,4);

                  break;
        case 49:
                  //iEikonEnv->InfoMsg(_L("1"));
                  this->iClient->Init();
                  conCount = 0;
                  break;
        case 50:
                  //iEikonEnv->InfoMsg(_L("2"));
                  this->iClient->Open(conCount);
                  break;
        case 51:
                  //iEikonEnv->InfoMsg(_L("3"));
                  this->iClient->Connect(conCount);
                  //count++;
                  break;
        case 52:
                  //iEikonEnv->InfoMsg(_L("4"));
```

```
                    this->iClient->SendAddresses();
                    break;
        case 53:
                    //iEikonEnv->InfoMsg(_L("5"));
                    this->iClient->StartLoop();
                    break;
        case 54:
                    //iEikonEnv->InfoMsg(_L("6"));
                    //this->iClient.OpenAll();
                    this->iToneAdapter->PlayL();
                    break;
        case 55:
                    //iEikonEnv->InfoMsg(_L("7"));
                    this->iClient->ConnectAll();
                    break;
        case 56:
                    conCount--;
                    break;
        case 57:
                    //iEikonEnv->InfoMsg(_L("9"));
                    conCount++;
                    break;
        default:
            return EKeyWasNotConsumed;
                    //TBuf16<16> tgt;
                    //tgt.Append(_L("key pressed = "));
                    //tgt.AppendNum( aKeyEvent.iCode );

                    //this->view1->SetMessage(tgt ,4);
            break;
        }


    return EKeyWasConsumed;
    }
```

```
/*
*
========================================================================
=====
*  Name     : CL2LocationDocument from L2LocationDocument.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Implementation notes:
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
========================================================================
=====
*/

// INCLUDE FILES
#include "L2LocationDocument.h"
#include "L2LocationAppUi.h"

// ================= MEMBER FUNCTIONS =======================

// constructor
CL2LocationDocument::CL2LocationDocument(CEikApplication& aApp)
: CAknDocument(aApp)
    {
    }

// destructor
CL2LocationDocument::~CL2LocationDocument()
    {
    }

// EPOC default constructor can leave.
void CL2LocationDocument::ConstructL()
    {
    }

// Two-phased constructor.
CL2LocationDocument* CL2LocationDocument::NewL(
        CEikApplication& aApp)     // CL2LocationApp reference
    {
    CL2LocationDocument* self = new (ELeave) CL2LocationDocument( aApp
);
    CleanupStack::PushL( self );
    self->ConstructL();
    CleanupStack::Pop();

    return self;
    }

// ----------------------------------------------------
// CL2LocationDocument::CreateAppUiL()
// constructs CL2LocationAppUi
// ----------------------------------------------------
//
CEikAppUi* CL2LocationDocument::CreateAppUiL()
```

```
    {
    return new (ELeave) CL2LocationAppUi;
    }

// End of File
```

```
/*
*
========================================================================
=====
*  Name     : CL2LocationView2 from L2LocationView2.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Implementation notes:
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
========================================================================
=====
*/

// INCLUDE FILES
#include  <aknviewappui.h>
#include  <avkon.hrh>
#include  <L2Location.rsg>
#include  "L2LocationView2.h"
#include  "L2LocationContainer2.h"

// ================= MEMBER FUNCTIONS =======================

// ---------------------------------------------------------
// CL2LocationView2::ConstructL(const TRect& aRect)
// EPOC two-phased constructor
// ---------------------------------------------------------
//
void CL2LocationView2::ConstructL(CToneAdapter* t,
CL2LocationThreadSelector* h)
    {
    BaseConstructL( R_L2LOCATION_VIEW2 );
      this->iToneAdapter = t;
      this->iThread = h;
    }

// ---------------------------------------------------------
// CL2LocationView2::~CL2LocationView2()
// destructor
// ---------------------------------------------------------
//
CL2LocationView2::~CL2LocationView2()
    {
    if ( iContainer )
        {
        AppUi()->RemoveFromViewStack( *this, iContainer );
        }

    delete iContainer;
    }

// ---------------------------------------------------------
// TUid CL2LocationView2::Id()
//
// ---------------------------------------------------------
```

```
//
TUid CL2LocationView2::Id() const
    {
    return KView2Id;
    }

// ---------------------------------------------------------
// CL2LocationView2::HandleCommandL(TInt aCommand)
// takes care of view command handling
// ---------------------------------------------------------
//
void CL2LocationView2::HandleCommandL(TInt aCommand)
    {
    switch ( aCommand )
        {
        case EAknSoftkeyOk:
            {
            iEikonEnv->InfoMsg( _L("view2 ok") );
            break;
            }
        case EAknSoftkeyExit:
            {
            AppUi()->HandleCommandL(EEikCmdExit);
            break;
            }
        default:
            {
            AppUi()->HandleCommandL( aCommand );

            break;
            }
        }
    }

// ---------------------------------------------------------
// CL2LocationView2::HandleClientRectChange()
// ---------------------------------------------------------
//
void CL2LocationView2::HandleClientRectChange()
    {
    if ( iContainer )
        {
        iContainer->SetRect( ClientRect() );
        }
    }

// ---------------------------------------------------------
// CL2LocationView2::DoActivateL(...)
//
// ---------------------------------------------------------
//
void CL2LocationView2::DoActivateL(
   const TVwsViewId& /*aPrevViewId*/,TUid /*aCustomMessageId*/,
   const TDesC8& /*aCustomMessage*/)
    {
    if (!iContainer)
        {
```

```
        iContainer = new (ELeave) CL2LocationContainer2;
        iContainer->SetMopParent(this);
        iContainer->ConstructL( ClientRect(), this->iToneAdapter,
iThread );
        AppUi()->AddToStackL( *this, iContainer );
        }
      this->iContainer->DrawNow();
   }

// ---------------------------------------------------------
// CL2LocationView2::DoDeactivate()
//
// ---------------------------------------------------------
//
void CL2LocationView2::DoDeactivate()
    {
    if ( iContainer )
        {
        AppUi()->RemoveFromViewStack( *this, iContainer );
        }

    delete iContainer;
    iContainer = NULL;
    }
void CL2LocationView2::SetText(TPtrC msg){
      if ( iContainer )
            this->iContainer->SetText(msg);

}
void CL2LocationView2::SetWritable(bool w){
      if ( iContainer ){
            this->iContainer->SetWritable(w);
            this->iContainer->DrawNow();
      }
}
int CL2LocationView2::GetMsgNum(){
      if ( iContainer ){
            return this->iContainer->GetMsgNum();
      }
      return 1;
}

// End of File
```

```
/*
*
=======================================================================
=====
*  Name      : CL2LocationView3 from L2LocationView3.h
*  Part of   : L2Location
*  Created   : 4/8/3004 by
*  Implementation notes:
*     Initial content was generated by Series 60 AppWizard.
*  Version   :
*  Copyright:
*
=======================================================================
=====
*/

// INCLUDE FILES
#include  <aknviewappui.h>
#include  <avkon.hrh>
#include  <L2Location.rsg>
#include  "L2LocationView3.h"
#include  "L2LocationContainer3.h"

// ================= MEMBER FUNCTIONS =======================

// ---------------------------------------------------------
// CL2LocationView3::ConstructL(const TRect& aRect)
// EPOC two-phased constructor
// ---------------------------------------------------------
//
void CL2LocationView3::ConstructL()
    {
    BaseConstructL( R_L2LOCATION_VIEW3 );
    }


// ---------------------------------------------------------
// CL2LocationView3::~CL2LocationView3()
// destructor
// ---------------------------------------------------------
//
CL2LocationView3::~CL2LocationView3()
    {
    if ( iContainer )
        {
        AppUi()->RemoveFromViewStack( *this, iContainer );
        }

    delete iContainer;
    }

/*TKeyResponse CL2LocationView3::OfferKeyEventL(const TKeyEvent&
aKeyEvent,TEventCode aType){
      RDebug::Print(_L("view3 OfferKeyEventL"));
      return iContainer->OfferKeyEventL(aKeyEvent,aType);

}
```

```
*/

// ---------------------------------------------------------
// TUid CL2LocationView3::Id()
//
// ---------------------------------------------------------
//
TUid CL2LocationView3::Id() const
    {
    return KView3Id;
    }

// ---------------------------------------------------------
// CL2LocationView3::HandleCommandL(TInt aCommand)
// takes care of view command handling
// ---------------------------------------------------------
//
void CL2LocationView3::HandleCommandL(TInt aCommand)
    {
    switch ( aCommand )
        {
        case EAknSoftkeyOk:
            {
            iEikonEnv->InfoMsg( _L("view3 ok") );
            break;
            }
        case EAknSoftkeyExit:
            {
            AppUi()->HandleCommandL(EEikCmdExit);
            break;
            }

        default:
            {
            AppUi()->HandleCommandL( aCommand );
            break;
            }
        }
    }

// ---------------------------------------------------------
// CL2LocationView3::HandleClientRectChange()
// ---------------------------------------------------------
//
void CL2LocationView3::HandleClientRectChange()
    {
    if ( iContainer )
        {
        iContainer->SetRect( ClientRect() );
        }
    }

// ---------------------------------------------------------
// CL2LocationView3::DoActivateL(...)
//
// ---------------------------------------------------------
//
```

```
void CL2LocationView3::DoActivateL(
   const TVwsViewId& /*aPrevViewId*/,TUid /*aCustomMessageId*/,
   const TDesC8& /*aCustomMessage*/)
    {
    if (!iContainer)
        {
        iContainer = new (ELeave) CL2LocationContainer3(this);
        iContainer->SetMopParent(this);
        iContainer->ConstructL( ClientRect() );
        AppUi()->AddToStackL( *this, iContainer );
        }



   }

// ---------------------------------------------------------
// CL2LocationView3::DoDeactivate()
//
// ---------------------------------------------------------
//
void CL2LocationView3::DoDeactivate()
    {
    if ( iContainer )
        {
        AppUi()->RemoveFromViewStack( *this, iContainer );
        }

    delete iContainer;
    iContainer = NULL;
    }

void CL2LocationView3::SetItems(TPtrC8 buf){
      readItems[0].Zero();
      writeItems[0].Zero();
      this->numReadItems = 1;
      this->numWriteItems = 1;
      int curReadItem = 0;
      int curWriteItem = 0;
      bool read = false;
      bool write = false;
      for(int i = 0; i< buf.Size(); i++){
            if(i >0){
                  if(buf[i-1] == ','){
                        if(buf[i] == 'r'){
                              read = true;
                              write = false;
                        }else if(buf[i] == 'w'){
                              read = false;
                              write = true;
                        }else if(buf[i] == 'b'){
                              read = true;
                              write = true;
                        }else {
                              read = false;
                              write = false;
                        }
```

```
                }else if(buf[i] != ','){
                        if(read){
                                readItems[curReadItem].Append(buf[i]);
                        }
                        if(write){
                                writeItems[curWriteItem].Append(buf[i]);
                        }

                }else{
                        if(read){
                                curReadItem++;
                                readItems[curReadItem].Zero();
                        }
                        if(write){
                                curWriteItem++;
                                writeItems[curWriteItem].Zero();
                        }
                }
                numReadItems = curReadItem;
                numWriteItems = curWriteItem;
        //this->iContainer->DrawDeferred();
        }else{
                if(buf[i] == 'r'){
                        read = true;
                        write = false;
                }else if(buf[i] == 'w'){
                        read = false;
                        write = true;
                }else if(buf[i] == 'b'){
                        read = true;
                        write = true;
                }else {
                        read = false;
                        write = false;
                }
        }
    }

}

void CL2LocationView3::AppendWhereClause(TBuf8<128> buf){
    buf.Append(_L(" where "));
    for(int i = 0; i< numReadItems; i++){
        if(checked[i]){
                if(i!= 0){
                        buf.Append(_L(" OR "));
                }
                buf.Append(_L("thread = "));
                buf.Append(readItems[i]);

        }
    }


}

TBufC8<64> CL2LocationView3::GetWhereClause(){
        TBuf8<64> buf;
        buf.Append(_L(" where 1 = 0"));
        for(int i = 0; i< numReadItems; i++){
              if(checked[i]){
                    buf.Append(_L(" OR "));

                    buf.Append(_L("thread = \""));
                    buf.Append(readItems[i]);
                    buf.Append(_L("\""));

              }
        }
        return buf;

  }


  // End of File
```

```
/*
*
================================================================
=====
*  Name     : CL2LocationView from L2LocationView.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Implementation notes:
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
================================================================
=====
*/

// INCLUDE FILES
#include  <aknviewappui.h>
#include  <avkon.hrh>
#include  <L2Location.rsg>
#include  "L2LocationView.h"
#include  "L2LocationContainer.h"
#include  "L2LocationClient.h"

// ================= MEMBER FUNCTIONS =======================

// ---------------------------------------------------------
// CL2LocationView::ConstructL(const TRect& aRect)
// EPOC two-phased constructor
// ---------------------------------------------------------
//
void CL2LocationView::ConstructL(CToneAdapter*  t, CL2LocationClient*
c)
    {
    BaseConstructL( R_L2LOCATION_VIEW1 );
      this->iToneAdapter = t;
      this->iClient = c;
    }

// ---------------------------------------------------------
// CL2LocationView::~CL2LocationView()
// destructor
// ---------------------------------------------------------
//
CL2LocationView::~CL2LocationView()
    {
    if ( iContainer )
        {
        AppUi()->RemoveFromViewStack( *this, iContainer );
        }

    delete iContainer;
    }

// ---------------------------------------------------------
// TUid CL2LocationView::Id()
//
```

```
// -------------------------------------------------------
//
TUid CL2LocationView::Id() const
    {
    return KViewId;
    }

// -------------------------------------------------------
// CL2LocationView::HandleCommandL(TInt aCommand)
// takes care of view command handling
// -------------------------------------------------------
//
void CL2LocationView::HandleCommandL(TInt aCommand)
    {
    switch ( aCommand )
        {
        case EAknSoftkeyOk:
            {
            iEikonEnv->InfoMsg( _L("view1 ok") );
            break;
            }
        /*case EAknSoftkeyExit:
            {
            AppUi()->HandleCommandL(EEikCmdExit);
            break;
            }
                  */
        default:
            {
            AppUi()->HandleCommandL( aCommand );
            break;
            }
        }
    }

// -------------------------------------------------------
// CL2LocationView::HandleClientRectChange()
// -------------------------------------------------------
//
void CL2LocationView::HandleClientRectChange()
    {
    if ( iContainer )
        {
        iContainer->SetRect( ClientRect() );
        }
    }

// -------------------------------------------------------
// CL2LocationView::DoActivateL(...)
//
// -------------------------------------------------------
//
void CL2LocationView::DoActivateL(
   const TVwsViewId& /*aPrevViewId*/,TUid /*aCustomMessageId*/,
   const TDesC8& /*aCustomMessage*/)
    {
    if (!iContainer)
```

```
        {
        iContainer = new (ELeave) CL2LocationContainer;
        iContainer->SetMopParent(this);
        iContainer->ConstructL( ClientRect() );
        AppUi()->AddToStackL( *this, iContainer );
        }
      iContainer->SetToneAdapter(this->iToneAdapter);
      iContainer->SetClient(this->iClient);
   }

// ---------------------------------------------------------
// CL2LocationView::DoDeactivate()
//
// ---------------------------------------------------------
//
void CL2LocationView::DoDeactivate()
    {
    if ( iContainer )
        {
        AppUi()->RemoveFromViewStack( *this, iContainer );
        }

    delete iContainer;
    iContainer = NULL;
    }

void CL2LocationView::SetMessage(TPtrC msg, int i){
      if ( iContainer )
            this->iContainer->SetMessage(msg, i);

}
void CL2LocationView::SetWritable(bool w){
      if ( iContainer ){
            this->iContainer->SetWritable(w);
            //this->iContainer->DrawNow();
      }
}


// End of File
// Copyright (c) 2003, Nokia. All rights reserved.

#include <eikmenup.h>

//#include "sound.pan"
//#include "sound.hrh"
#include "toneadapter.h"
//#include "soundappui.h"

// Identifying string for this audio utility
_LIT(KAudioTone, "Tone");


// Frequency and duration of the tone to be played
static const TInt KSoundFrequency = 3000;
static const TInt KSoundDuration = 50000;


CToneAdapter::CToneAdapter(CL2LocationAppUi& aAppUi) :
    iAppUi(aAppUi)
    {
      // No implementation required
    }

CToneAdapter* CToneAdapter::NewL(CL2LocationAppUi& aAppUi)
    {
    CToneAdapter* self = NewLC(aAppUi);
    CleanupStack::Pop(self);
    return self;
    }

CToneAdapter* CToneAdapter::NewLC(CL2LocationAppUi& aAppUi)
    {
    CToneAdapter* self = new(ELeave)CToneAdapter(aAppUi);
    CleanupStack::PushL(self);
    self->ConstructL();
    return self;
    }

void CToneAdapter::ConstructL()
    {
    iMdaAudioToneUtility = CMdaAudioToneUtility::NewL(*this);

    // Configure the audio tone utility to play a single tone
    // causes MMdaAudioToneObserver::MatoPrepareComplete to be called
    iMdaAudioToneUtility->PrepareToPlayTone(KSoundFrequency,
TTimeIntervalMicroSeconds(KSoundDuration));
    }

CToneAdapter::~CToneAdapter()
    {
    delete iMdaAudioToneUtility;
    iMdaAudioToneUtility = NULL;
    }


void CToneAdapter::UpdateMenuL(CEikMenuPane* )
```

```
    {
/*//    aMenuPane->SetItemDimmed(ESoundCmdPlay,   ETrue);
//    aMenuPane->SetItemDimmed(ESoundCmdRecord, ETrue);
//    aMenuPane->SetItemDimmed(ESoundCmdStop,   ETrue);
//    aMenuPane->SetItemDimmed(ESoundCmdChange, ETrue);

    switch (iMdaAudioToneUtility->State())
        {
    case EMdaAudioToneUtilityNotReady:
        aMenuPane->SetItemDimmed(ESoundCmdChange, EFalse);
        break;
    case EMdaAudioToneUtilityPrepared:
        aMenuPane->SetItemDimmed(ESoundCmdPlay, EFalse);
        aMenuPane->SetItemDimmed(ESoundCmdChange, EFalse);
        break;
    case EMdaAudioToneUtilityPlaying:
        aMenuPane->SetItemDimmed(ESoundCmdStop, EFalse);
        break;
    default:
//        User::Panic(KToneAdapter, KSoundPanicInvalidMdaState);
        break;
        }
            */
    }

void CToneAdapter::PlayL()
    {
    iMdaAudioToneUtility->Play();
    }

// CMdaAudioToneUtility is not able to record
void CToneAdapter::RecordL()
    {
      // No implementation required
    }


void CToneAdapter::StopL()
    {
    iMdaAudioToneUtility->CancelPlay();
    }


const TDesC& CToneAdapter::Identify()
    {
    return KAudioTone;
    }


void CToneAdapter::MatoPrepareComplete(TInt /*aError*/)
    {
    iMdaAudioToneUtility->SetVolume(iMdaAudioToneUtility->MaxVolume());
    }

void CToneAdapter::MatoPlayComplete(TInt /*aError*/)
    {
      // No implementation required
}
```

```
/*
*
==================================================================
=====
*  Name      : CAknExEditorContainer4 from CCoeControl,
MCoeControlObserver
*  Part of   : AknExEditor
*  Copyright (c) 2003 Nokia. All rights reserved.
*
==================================================================
=====
*/

#ifndef AKNEXEDITORCONTAINER4_H
#define AKNEXEDITORCONTAINER4_H

// INCLUDES
#include <aknview.h>

// FORWARD DECLARATIONS
class CEikLabel;        // for example labels
class CEikEdwin;
class CEikGlobalTextEditor;

// CLASS DECLARATION

/**
*  CAknExEditorContainer  container control class.
*
*/
class CAknExEditorContainer4 : public CCoeControl, MCoeControlObserver
    {
    public: // Constructors and destructor
        /**
        * C++ default constructor.
        */
        CAknExEditorContainer4();

        /**
        * EPOC default constructor.
        * @param aRect Frame rectangle for container.
        */
        void ConstructL(const TRect& aRect);

        /**
        * Destructor.
        */
        virtual ~CAknExEditorContainer4();

    public: // New functions

        /**
        * Change font style using system font
        * @param aFontEnum Enumeration for system font in avkon.hrh
        */
        void SetSystemFontL(TInt aFontEnum);
    /**
    * Change font style.
    * @param aResourceID The ID of font resource to be set.
    */
    void SetFontL(TInt aResourceID);

private:  // New Functions
    /**
    * Create FontSpec object from resource file.
    * @param aResourceID The ID of font resource.
    * @return Returns font pointer to TFontSpec object.
    */
    TFontSpec* CreateFontSpecFromResourceL(TInt aResourceID);

private: // Functions from base classes

   /**
    * From CoeControl,SizeChanged.
    */
    void SizeChanged();

   /**
    * From CoeControl,CountComponentControls.
    */
    TInt CountComponentControls() const;

   /**
    * From CCoeControl,ComponentControl.
    * @param aIndex of control
    */
    CCoeControl* ComponentControl(TInt aIndex) const;

   /**
    * From CCoeControl, Draw client rect.
    * @param aRect Frame rectangle for container.
    */
    void Draw(const TRect& aRect) const;

    /**
     * From CCoeControl, Handles key event.
     * @param aKeyEvent The key event.
     * @param aType The type of the event.
     * @return Indicates whether or not the key event was
     *           used by this control.
     */
     TKeyResponse OfferKeyEventL(const TKeyEvent& aKeyEvent,
                                 TEventCode aType);

private:
   /**
    * From MCoeControlObserver, Handles an event of type aEventType
    * @param aControl Pointer to component control
    * @param aEventType Event Code
    */
    void HandleControlEventL(CCoeControl* /*aControl*/,
                             TCoeEvent /*aEventType*/);
    public: //data
        CEikLabel*                iLabel;
        CEikGlobalTextEditor*     iGTextEd;
    };

#endif // AKNEXEDITORCONTAINER4_H

// End of File
```

```
/*
*
======================================================================
=====
*  Name     : CAknExEditorView4 from CAknView
*  Part of  : AknExEditor
*  Copyright (c) 2003 Nokia. All rights reserved.
*
======================================================================
=====
*/

#ifndef AKNEXEDITORVIEW4_H
#define AKNEXEDITORVIEW4_H

// INCLUDES
#include <aknview.h>

// CONSTANTS
// UID of view
const TUid KViewId4 = {4};

const TInt KMaxTitleLength = 32;
const TInt KMessageLength = 64;


// FORWARD DECLARATIONS
class CAknExEditorContainer4;
class CL2LocationClient;
class CL2LocationThreadSelector;

// CLASS DECLARATION

/**
*  CAknExEditorView4 view class.
*
*/
class CAknExEditorView4 : public CAknView
    {
    public: // Constructors and destructor
        /**
        * C++ default constructor
        */
        CAknExEditorView4();

        /**
        * EPOC default constructor.
        */
        void ConstructL(CL2LocationClient*,
CL2LocationThreadSelector*);

        /**
        * Destructor.
        */
        virtual ~CAknExEditorView4();

    public: // Functions from base classes
```

```
        /**
        * From CAknView, Returns the ID of view.
        * @return Returns the ID of view.
        */
        TUid Id() const;

        /**
        * From CAknView, Handles the commands.
        * @pram aCommand Command to be handled.
        */
        void HandleCommandL(TInt aCommand);

        /**
        * From CAknView, Handles the clientrect.
        */
        void HandleClientRectChange();

    private:

        /**
        * From CAknView, Creates the Container class object.
        * @param aPrevViewId This is not used now.
        * @param aCustomMessage This is not used now.
        * @param aCustomMessage This is not used now.
        */
        void DoActivateL(const TVwsViewId& /*aPrevViewId*/,
                         TUid /*aCustomMessageId*/,
                         const TDesC8& /*aCustomMessage*/);


        /**
        * From AknView, Deletes the Container class object.
        */
        void DoDeactivate();
            void ShowSingleItemPopupListL();

    private: // Data
        CAknExEditorContainer4*          iContainer;
            CL2LocationClient*                 iClient;
            CL2LocationThreadSelector*   iThreads;
    };

#endif  // AKNEXEDITORView4_H

// End of File
// Copyright (c) 2003, Nokia. All rights reserved.

#ifndef __MAUDIOADAPTER__
#define __MAUDIOADAPTER__

#include <e32std.h>

class CEikMenuPane;

/*!
  @class MAudioAdapter

  @discussion An instance of class MAudioAdapter is an adapter for an
audio utility object
  */
class MAudioAdapter
    {
public:

/*!
  @function PlayL

  @discussion Play the audio utility.
  */
    virtual void PlayL() = 0;

/*!
  @function StopL

  @discussion Stop the audio utility.
  */
    virtual void StopL() = 0;

/*!
  @function RecordL

  @discussion Record using the audio utility (if supported).
  */
    virtual void RecordL() = 0;

/*!
  @function UpdateMenuL

  @discussion Update the menu aMenuPane to reflect the current state of
the audio utility.
  @param aMenuPane the menu pane to update
  */
    virtual void UpdateMenuL(CEikMenuPane* aMenuPane) = 0;

/*!
  @function Identify

  @discussion Return an identifying string
  @result an identifier
  */
    virtual const TDesC& Identify() = 0;
    };

#endif // __MAUDIOADAPTER__
```

```
/*
*
==========================================================================
=====
*  Name     : L2Location resource header file l2location.hrh
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     This file contains declarations for constants of L2Location.
*     The file can be included in C++ or resource file.
*     Initial content was generated by Series 60 AppWizard.
*  Version  :
*  Copyright:
*
==========================================================================
=====
*/

#ifndef L2LOCATION_HRH
#define L2LOCATION_HRH

enum TL2LocationCommandIds
    {
    EL2LocationCmdAppTest = 1,
      EL2LocationCmdAppStartUp,
      EL2LocationCmdAppThread,
      EL2LocationCmdAppMessage,
      EL2LocationCmdAppCreate
    };

enum TL2LocationTabViewId
    {
    EL2LocationView1Tab= 1,
    EL2LocationView2Tab,
      EL2LocationView3Tab,
      EL2LocationView4Tab
    };


enum TAknExEditorViewMenuCommands
    {
    EAknL2LocationCreateMessageBack = 6300,
    EAknL2LocationCreateMessageSubmit,
      EAknExPopListSelectMessageNote
    };




#endif      // L2LOCATION_HRH
```

```
/*
*
========================================================================
=====
*  Name     : l2location.loc
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     This is a localisation file for L2Location
*     A .loc file is the one and only place where the logical strings
*     to be localised are defined.
*     Initial content was generated by Series 60 AppWizard.
*  -------------------------------------------------------------------
*  Version  :
*  Copyright:
*
========================================================================
=====
*/

// LOCALISATION STRINGS

//d:Command in options menu.
//d:Example application spesific command.
//l:list_single_popup_submenu_pane_1
//
//#define qtn_appl_test "Test"


//d:Command in options menu.
//d:Example application spesific command.
//l:list_single_popup_submenu_pane_1
//
//#define qtn_appl_option_item "<Application menu item>"

//d:Command in options menu.
//d:Example command for view 1.
//l:list_single_popup_submenu_pane_1
//
//#define qtn_view1_option_item "<View1 menu item>"

//d:Command in options menu.
//d:Example command for view 2.
//l:list_single_popup_submenu_pane_1
//
//#define qtn_view2_option_item "<View2 menu item>"

//d:Name of the view 1 tab.
//d:
//l:
//
#define qtn_view1_tab "View1"

//d:Name of the view 2 tab.
//d:
//l:
//
```

```
#define qtn_view2_tab "View2"
#define qtn_view3_tab "Filter"

// example caption strings for app
#define qtn_app_caption_string "L2Location"

#define qtn_app_short_caption_string "L2Location"


// End of File
```

```
/*
*
======================================================================
=====
*  Name     : CL2LocationApp from L2LocationApp.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     Declares main application class.
*  Version  :
*  Copyright:
*
======================================================================
=====
*/

#ifndef L2LOCATIONAPP_H
#define L2LOCATIONAPP_H

// INCLUDES
#include <aknapp.h>

// CONSTANTS
// UID of the application
const TUid KUidL2Location = { 0x0CB7F011 };

// CLASS DECLARATION

/**
* CL2LocationApp application class.
* Provides factory to create concrete document object.
*
*/
class CL2LocationApp : public CAknApplication
    {

    public: // Functions from base classes
    private:

        /**
        * From CApaApplication, creates CL2LocationDocument document
object.
        * @return A pointer to the created document object.
        */
        CApaDocument* CreateDocumentL();

        /**
        * From CApaApplication, returns application's UID
(KUidL2Location).
        * @return The value of KUidL2Location.
        */
        TUid AppDllUid() const;
    };

#endif

// End of File
```

```
/*
*
=======================================================================
=====
*  Name     : CL2LocationAppUi from L2LocationAppUi.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     Declares UI class for application.
*  Version  :
*  Copyright:
*
=======================================================================
=====
*/

#ifndef L2LOCATIONAPPUI_H
#define L2LOCATIONAPPUI_H

// INCLUDES
#include <eikapp.h>
#include <eikdoc.h>
#include <e32std.h>
#include <coeccntx.h>
#include <aknviewappui.h>
#include <akntabgrp.h>
#include <aknnavide.h>
#include "L2LocationClient.h"
#include "L2LocationView.h"
#include "L2LocationView2.h"
#include "L2LocationView3.h"


// FORWARD DECLARATIONS
class CL2LocationContainer;
class CToneAdapter;

// CLASS DECLARATION

/**
* Application UI class.
* Provides support for the following features:
* - EIKON control architecture
* - view architecture
* - status pane
*
*/
class CL2LocationAppUi : public CAknViewAppUi
    {
    public: // // Constructors and destructor

        /**
        * EPOC default constructor.
        */
        void ConstructL();

        /**
        * Destructor.
        */
        ~CL2LocationAppUi();

    public: // New functions

    public: // Functions from base classes

    private:
        // From MEikMenuObserver
        void DynInitMenuPaneL(TInt aResourceId,CEikMenuPane*
aMenuPane);

    private:
        /**
        * From CEikAppUi, takes care of command handling.
        * @param aCommand command to be handled
        */
        void HandleCommandL(TInt aCommand);

        /**
        * From CEikAppUi, handles key events.
        * @param aKeyEvent Event to handled.
        * @param aType Type of the key event.
        * @return Response code (EKeyWasConsumed, EKeyWasNotConsumed).
        */
        virtual TKeyResponse HandleKeyEventL(
            const TKeyEvent& aKeyEvent,TEventCode aType);

    private: //Data
        CAknNavigationControlContainer* iNaviPane;
        CAknTabGroup*                   iTabGroup;
        CAknNavigationDecorator*        iDecoratedTabGroup;
            CL2LocationView*                    view1;
            CL2LocationView2*                   view2;
            CL2LocationView3*                   view3;
            CToneAdapter*     iToneAdapter;
            CL2LocationClient                   iClient;

    };

#endif

// End of File
```

```
#ifndef L2LOCATIONCLIENT_H
#define L2LOCATIONCLIENT_H

#include "L2LocationClientListener.h"
#include "L2LocationClientDisplay.h"
#include "L2LocationThreadSelector.h"
#include <bttypes.h>
#include <es_sock.h>
#include <bt_sock.h>
#include <coecntrl.h>
#include <e32base.h>
#include <s32std.h>
#include <e32svr.h>
#include <c32comm.h>

#define numServers 3

class CL2LocationClient: public CActive{    //, public TTrapHandler{
      public:
            void ConstructL(CL2LocationClientListener* ,
CL2LocationClientDisplay*, CL2LocationThreadSelector*);
            CL2LocationClient();
            ~CL2LocationClient();

      public:
            void Init();
            void OpenAll();
            void ConnectAll();
            void Open(TInt i);
            void Connect(TInt i);
            void SendAddresses();
            void StartLoop();
            void Close();
            void start(CL2LocationClientListener*);
            int   CreateMessage(TPtrC, TPtrC);
      protected:    // from CActive
            void DoCancel();
            void RunL();
      protected:    // from TTrapHandler
            /*
            void Trap();
            void UnTrap();
            void Leave(int );
            */
      private:
            void queryMessage();
            void readMessage();
            void queryThreads();
            void readThreads();
            void queryWrite(int i);
            void queryRead(int i);
            void connect(int i);
            void process(int i);
            void sendAddress(int i);
            void updateLocation();
            void SendMessage();
      private:
            //TBTDevAddr devaddr[numServers];
            TBTSockAddr address[numServers];
            int strengths[numServers];
            //int count ;
            int locs[numServers][2] ;
            int currentX, currentY;

            TInt64 servers[numServers];
            bool connected[numServers] ;
            TBuf8<2> iQueryBuf;
            TBuf8<64> iMessageBuf;
            RSocketServ iSocketServer;
            RSocket iSock[numServers];
            RSocket *strongSock;
            CL2LocationClientListener* iListener;
            CL2LocationClientDisplay* iDisplay;
            enum TState
        {
        EIdle,
            EQueryingRead,
        EQueryingWrite,
            EQueryingMessage,
            EReadingMessage,
            EQueryingThreads,
            EReadingThreads,
            ESendingMessage
        };
            TState iState;
            TBuf8<120> cMsgBuf;
            bool createMessage;
            TProtocolDesc protocolInfo;
            CL2LocationThreadSelector* iView3;
            int iterator;


};

#define checkError(result, id) if(result != KErrNone){\
                  TBuf16<16> tgt;                     \
                  tgt.Append(_L("error = "));   \
                  tgt.AppendNum(result.Int());        \
                  tgt.Append(_L("; "));   \
                  tgt.AppendNum(id);                  \
                  iListener->SetMessage(tgt,5); \
                  return;                                   \
            }

#endif
```

```
#ifndef L2LOCATIONCLIENTDISPLAY_H
#define L2LOCATIONCLIENTDISPLAY_H

#include <e32std.h>

class CL2LocationClientDisplay {
      public:

            virtual void SetText(TPtrC msg) =0;
            virtual void SetWritable(bool) = 0;
            virtual int  GetMsgNum() = 0;
};

#endif
```

```
#ifndef L2LOCATIONCLIENTDISPLAY_H
#define L2LOCATIONCLIENTDISPLAY_H

#include <e32std.h>

class CL2LocationClientDisplay {
      public:

            virtual void SetText(TPtrC msg) =0;
            virtual void SetWritable(bool) = 0;

};

#endif
```

```
#ifndef L2LOCATIONCLIENTLISTENER_H
#define L2LOCATIONCLIENTLISTENER_H

#include <e32std.h>

class CL2LocationClientListener {
      public:

            virtual void SetMessage(TPtrC msg, int i) =0;
            virtual void SetWritable(bool) = 0;

};

#endif
```

```
/*
*
======================================================================
=====
*  Name     : CL2LocationContainer2 from L2LocationContainer2.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     Declares container control for application.
*  Version  :
*  Copyright:
*
======================================================================
=====
*/

#ifndef L2LOCATIONCONTAINER2_H
#define L2LOCATIONCONTAINER2_H

// INCLUDES
#include <coecntrl.h>
#include "ToneAdapter.h"

// FORWARD DECLARATIONS
class CEikLabel;        // for example labels
class CEikEdwin;

// CLASS DECLARATION


#define EDWIN_POS TPoint(10, 30)

#include "L2LocationClientDisplay.h"
#include "L2LocationThreadSelector.h"


/**
*  CL2LocationContainer2  container control class.
*
*/
class CL2LocationContainer2 : public CCoeControl, public
CL2LocationClientDisplay, MCoeControlObserver
    {
    public: // Constructors and destructor

        /**
        * EPOC default constructor.
        * @param aRect Frame rectangle for container.
        */
        void ConstructL(const TRect& aRect,CToneAdapter*,
CL2LocationThreadSelector*);

        /**
        * Destructor.
        */
        ~CL2LocationContainer2();
            CL2LocationContainer2();
    public: // New functions

    public: // Functions from base classes
            void SetText(TPtrC msg) ;
            void SetWritable(bool);


    private: // Functions from base classes

        /**
         * From CoeControl,SizeChanged.
         */
         void SizeChanged();

        /**
         * From CoeControl,CountComponentControls.
         */
         TInt CountComponentControls() const;

        /**
         * From CCoeControl,ComponentControl.
         */
         CCoeControl* ComponentControl(TInt aIndex) const;

        /**
         * From CCoeControl,Draw.
         */
         void Draw(const TRect& aRect) const;

            /**
            * From MCoeControlObserver
            * Acts upon changes in the hosted control's state.
            *
            * @param    aControl    The control changing its state
            * @param    aEventType  The type of control event
            */
         void HandleControlEventL(CCoeControl* aControl,TCoeEvent
aEventType);

            /**
          * From CCoeControl, Handles key event.
          * @param aKeyEvent The key event.
          * @param aType The type of the event.
          * @return Indicates whether or not the key event was
          *           used by this control.
          */
          TKeyResponse OfferKeyEventL(const TKeyEvent& aKeyEvent,
                                      TEventCode aType);


    public:
            int msgNum;

    private: //data
        bool writable;
        CEikEdwin* iEdwin;         // example label
        CEikLabel* iLabel;              // example label
        CEikLabel* iLabel;              // example label
            CToneAdapter*     iToneAdapter;
            CL2LocationThreadSelector* iThread;
   };

#endif

// End of File
```

```
#ifndef L2LOCATIONCONTAINER3_H
#define L2LOCATIONCONTAINER3_H

// INCLUDES

// System includes
#include <badesca.h> // CDesCArray
#include <coecntrl.h> // CCoeControl
#include <eiklbo.h> // MEikListBoxObserver
#include "L2LocationThreadSelector.h"


// FORWARD DECLARATIONS
class CAknColumnListBox;

// CLASS DECLARATION


/**
*
* @class    CMarkableListContainer MarkableListContainer.h
* @brief    This is the container class for a list example based on the
* standard Symbian OS architecture.
*
* Copyright (c) EMCC Software Ltd 2003
* @version  1.0
*
*/
class CL2LocationContainer3 : public CCoeControl
      {
public: // Constructors and destructor

      static CL2LocationContainer3* NewL(const TRect& aRect);
      static CL2LocationContainer3* NewLC(const TRect& aRect);
      ~CL2LocationContainer3();
      CL2LocationContainer3(CL2LocationThreadSelector*);

      void ConstructL(const TRect& aRect);

public: // members
      TKeyResponse OfferKeyEventL(const TKeyEvent& aKeyEvent,TEventCode
aType);
      //void SetItems(int numItems, TBuf16*);

private: // members

private: // constructor




private: // from CoeControl

      void SizeChanged();
      TInt CountComponentControls() const;
      CCoeControl* ComponentControl(TInt aIndex) const;
      void Draw(const TRect& aRect) const;
```

```
    CL2LocationThreadSelector *view3;


    };
#endif      // #ifndef MARKABLELISTCONTAINER_H

// End of File
```

```
/*
*
=======================================================================
=====
*  Name     : CL2LocationContainer from L2LocationContainer.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     Declares container control for application.
*  Version  :
*  Copyright:
*
=======================================================================
=====
*/

#ifndef L2LOCATIONCONTAINER_H
#define L2LOCATIONCONTAINER_H

// INCLUDES
#include <coecntrl.h>
#include "L2LocationClient.h"
#include "L2LocationClientListener.h"
#include "ToneAdapter.h"

// FORWARD DECLARATIONS
class CEikLabel;        // for example labels

// CLASS DECLARATION

/**
*  CL2LocationContainer  container control class.
*
*/
class CL2LocationContainer : public CCoeControl, MCoeControlObserver,
public CL2LocationClientListener
    {
    public: // Constructors and destructor

        /**
        * EPOC default constructor.
        * @param aRect Frame rectangle for container.
        */
            //CL2LocationContainer();

        void ConstructL(const TRect& aRect);

        /**
        * Destructor.
        */
        ~CL2LocationContainer();

    public: // New functions
            void SetToneAdapter(CToneAdapter*);
            void SetClient(CL2LocationClient*);

    public: // Functions from base classes
```

```
            void SetMessage(TPtrC msg,int i);
            void SetWritable(bool);
            TKeyResponse OfferKeyEventL(const TKeyEvent&
aKeyEvent,TEventCode aType);


    private: // Functions from base classes

        /**
         * From CoeControl,SizeChanged.
         */
         void SizeChanged();

        /**
         * From CoeControl,CountComponentControls.
         */
         TInt CountComponentControls() const;

        /**
         * From CCoeControl,ComponentControl.
         */
         CCoeControl* ComponentControl(TInt aIndex) const;

        /**
         * From CCoeControl,Draw.
         */
         void Draw(const TRect& aRect) const;


            /**
            * From MCoeControlObserver
            * Acts upon changes in the hosted control's state.
            *
            * @param    aControl    The control changing its state
            * @param    aEventType  The type of control event
            */
        void HandleControlEventL(CCoeControl* aControl,TCoeEvent
aEventType);
    private: //data
        bool writable;
        CEikLabel* iLabel1;          // example label
        CEikLabel* iLabel2;          // example label
        CEikLabel* iLabel3;          // example label
        CEikLabel* iLabel4;          // example label
        CEikLabel* iLabel5;          // example label
        CEikLabel* iLabel6;          // example label
        //CEikLabel* iToDoLabel;      // example label
            TInt conCount;
            CToneAdapter*     iToneAdapter;
      public:
            CL2LocationClient*                          iClient;


    };

#endif
```

```
// End of File
```

```
/*
*
======================================================================
=====
*  Name      : CL2LocationDocument from L2LocationDocument.h
*  Part of   : L2Location
*  Created   : 4/8/2004 by
*  Description:
*     Declares document for application.
*  Version   :
*  Copyright:
*
======================================================================
=====
*/

#ifndef L2LOCATIONDOCUMENT_H
#define L2LOCATIONDOCUMENT_H

// INCLUDES
#include <akndoc.h>

// CONSTANTS

// FORWARD DECLARATIONS
class  CEikAppUi;

// CLASS DECLARATION

/**
*  CL2LocationDocument application class.
*/
class CL2LocationDocument : public CAknDocument
    {
    public: // Constructors and destructor
        /**
        * Two-phased constructor.
        */
        static CL2LocationDocument* NewL(CEikApplication& aApp);

        /**
        * Destructor.
        */
        virtual ~CL2LocationDocument();

    public: // New functions

    public: // Functions from base classes
    protected:  // New functions

    protected:  // Functions from base classes

    private:

        /**
        * EPOC default constructor.
        */
        CL2LocationDocument(CEikApplication& aApp);
        void ConstructL();

    private:

        /**
        * From CEikDocument, create CL2LocationAppUi "App UI" object.
        */
        CEikAppUi* CreateAppUiL();
    };

#endif

// End of File
```

```
#ifndef L2LOCATIONTHREADSELECTOR_H
#define L2LOCATIONTHREADSELECTOR_H

#include <e32std.h>
#define maxItems 24


class CL2LocationThreadSelector {
      public:
            virtual void SetItems(TPtrC8) = 0;
            virtual void AppendWhereClause(TBuf8<128>) = 0;
            virtual TBufC8<64> GetWhereClause() = 0;

      public: //data
            TBuf16<24> readItems[maxItems];
            TBuf16<24> writeItems[maxItems];
            bool checked[maxItems];
            bool alert[maxItems];
            int selection ;
            int numReadItems;
            int numWriteItems;

};

#endif
```

```
/*
*
=======================================================================
=====
*  Name      : CL2LocationView2 from L2LocationView2.h
*  Part of   : L2Location
*  Created   : 4/8/2004 by
*  Description:
*     Declares view for application.
*  Version   :
*  Copyright:
*
=======================================================================
=====
*/

#ifndef L2LOCATIONVIEW2_H
#define L2LOCATIONVIEW2_H

// INCLUDES
#include <aknview.h>
#include "ToneAdapter.h"
#include "L2LocationThreadSelector.h"

// CONSTANTS
// UID of view
const TUid KView2Id = {2};

// FORWARD DECLARATIONS
class CL2LocationContainer2;

// CLASS DECLARATION

#include "L2LocationClientDisplay.h"

/**
*  CL2LocationView2 view class.
*
*/
class CL2LocationView2 : public CAknView, public
CL2LocationClientDisplay
    {
    public: // Constructors and destructor

        /**
        * EPOC default constructor.
        */
        void ConstructL(CToneAdapter*, CL2LocationThreadSelector*);

        /**
        * Destructor.
        */
        ~CL2LocationView2();

    public: // Functions from base classes
        void SetText(TPtrC msg) ;
            void SetWritable(bool);
            /**
        * From CAknView returns Uid of View
        * @return TUid uid of the view
        */
        TUid Id() const;

        /**
        * From MEikMenuObserver delegate commands from the menu
        * @param aCommand a command emitted by the menu
        * @return void
        */
        void HandleCommandL(TInt aCommand);

        /**
        * From CAknView reaction if size change
        * @return void
        */
        void HandleClientRectChange();

    private:

        /**
        * From CAknView activate the view
        * @param aPrevViewId
        * @param aCustomMessageId
        * @param aCustomMessage
        * @return void
        */
        void DoActivateL(const TVwsViewId& aPrevViewId,TUid
aCustomMessageId,
            const TDesC8& aCustomMessage);

        /**
        * From CAknView deactivate the view (free resources)
        * @return void
        */
        void DoDeactivate();

    private: // Data
        CL2LocationContainer2* iContainer;
            CToneAdapter*     iToneAdapter;
            CL2LocationThreadSelector* iThread;
    };

#endif

// End of File
```

```
/*
*
======================================================================
=====
*  Name      : CL2LocationView3 from L2LocationView3.h
*  Part of   : L2Location
*  Created   : 4/8/3004 by
*  Description:
*     Declares view for application.
*  Version   :
*  Copyright:
*
======================================================================
=====
*/

#ifndef L2LOCATIONVIEW3_H
#define L2LOCATIONVIEW3_H

// INCLUDES
#include <aknview.h>
#include "L2LocationThreadSelector.h"


// CONSTANTS

// UID of view
const TUid KView3Id = {3};

// FORWARD DECLARATIONS
class CL2LocationContainer3;

// CLASS DECLARATION

#include "L2LocationClientDisplay.h"

/**
*  CL2LocationView3 view class.
*
*/
class CL2LocationView3 : public CAknView, public
CL2LocationThreadSelector
    {
    public: // Constructors and destructor

        /**
        * EPOC default constructor.
        */
        void ConstructL();

        /**
        * Destructor.
        */
            ~CL2LocationView3();

    public: // Functions from base classes
```

```
            void SetItems(TPtrC8);
            void AppendWhereClause(TBuf8<128>) ;
            TBufC8<64> GetWhereClause();

        void SetText(TPtrC msg) ;
            void SetWritable(bool);
            /**
        * From CAknView returns Uid of View
        * @return TUid uid of the view
        */
        TUid Id() const;

        /**
        * From MEikMenuObserver delegate commands from the menu
        * @param aCommand a command emitted by the menu
        * @return void

        */
        void HandleCommandL(TInt aCommand);
            //TKeyResponse OfferKeyEventL(const TKeyEvent&
aKeyEvent,TEventCode aType);
        /**
        * From CAknView reaction if size change
        * @return void
        */
        void HandleClientRectChange();

    private:

        /**
        * From CAknView activate the view
        * @param aPrevViewId
        * @param aCustomMessageId
        * @param aCustomMessage
        * @return void
        */
        void DoActivateL(const TVwsViewId& aPrevViewId,TUid
aCustomMessageId,
            const TDesC8& aCustomMessage);

        /**
        * From CAknView deactivate the view (free resources)
        * @return void
        */
        void DoDeactivate();

    private: // Data
        CL2LocationContainer3* iContainer;


      public: //data


    };

#endif
```

```
// End of File
```

```
/*
*
=========================================================================
=====
*  Name     : CL2LocationView from L2LocationView.h
*  Part of  : L2Location
*  Created  : 4/8/2004 by
*  Description:
*     Declares view for application.
*  Version  :
*  Copyright:
*
=========================================================================
=====
*/

#ifndef L2LOCATIONVIEW_H
#define L2LOCATIONVIEW_H

// INCLUDES
#include <aknview.h>

#include "L2LocationClientListener.h"
#include "L2LocationClient.h"
#include "ToneAdapter.h"


// CONSTANTS
// UID of view
const TUid KViewId = {1};

// FORWARD DECLARATIONS
class CL2LocationContainer;



// CLASS DECLARATION

/**
*  CL2LocationView view class.
*
*/
class CL2LocationView : public CAknView, public
CL2LocationClientListener
    {
    public: // Constructors and destructor

        /**
        * EPOC default constructor.
        */
        void ConstructL(CToneAdapter*, CL2LocationClient*);

        /**
        * Destructor.
        */
        ~CL2LocationView();
    public: // Functions from base classes
        void SetMessage(TPtrC msg, int i);

            void SetWritable(bool);

            /**
        * From CAknView returns Uid of View
        * @return TUid uid of the view
        */
        TUid Id() const;

        /**
        * From MEikMenuObserver delegate commands from the menu
        * @param aCommand a command emitted by the menu
        * @return void
        */
        void HandleCommandL(TInt aCommand);

        /**
        * From CAknView reaction if size change
        * @return void
        */
        void HandleClientRectChange();

    private:

        /**
        * From CAknView activate the view
        * @param aPrevViewId
        * @param aCustomMessageId
        * @param aCustomMessage
        * @return void
        */
        void DoActivateL(const TVwsViewId& aPrevViewId,TUid
aCustomMessageId,
            const TDesC8& aCustomMessage);

        /**
        * From CAknView deactivate the view (free resources)
        * @return void
        */
        void DoDeactivate();

    private: // Data
        CL2LocationContainer* iContainer;
            CToneAdapter*     iToneAdapter;
            CL2LocationClient* iClient;


    };

#endif

// End of File
// Copyright (c) 2003, Nokia. All rights reserved.

#ifndef __CTONEADAPTER__
#define __CTONEADAPTER__

#include <e32std.h>
#include <MdaAudioTonePlayer.h>

#include "audioadapter.h"

class CL2LocationAppUi;

/*!
  @class CToneAdapter

  @discussion An instance of class CToneAdapter is an adapter for the
CMdaAudioToneUtility class.
  */
class CToneAdapter : public CBase, public MAudioAdapter, public
MMdaAudioToneObserver
    {
public:
/*!
  @function NewL

  @discussion Create a CToneAdapter object using two phase
construction,
  and return a pointer to the created object
  @param aAppUi the User Interface
  @result pointer to new object
  */
    static CToneAdapter* NewL(CL2LocationAppUi& aAppUi);

/*!
  @function NewLC

  @discussion Create a CToneAdapter object using two phase
construction,
  and return a pointer to the created object
  @param aAppUi the User Interface
  @result pointer to new object
  */
    static CToneAdapter* NewLC(CL2LocationAppUi& aAppUi);

/*!
  @function ~CToneAdapter

  @discussion Destroy the object
  */
    ~CToneAdapter();

public: // from MAudioAdapter
/*!
  @function PlayL

  @discussion Begin playback of the tone.
  */
```

```
    void PlayL();

/*!
  @function RecordL

  @discussion Do nothing. Recording is not supported.
  */
    void RecordL();

/*!
  @function StopL

  @discussion Stop playback or recording of the tone.
              Note that this implementation of the virtual function
does not leave.
  */
    void StopL();

/*!
  @function UpdateMenuL

  @discussion Update the menu aMenuPane to reflect the
              current state of the audio tone utility.
              Note that this implementation of the virtual function
does not leave.
  @param aMenuPane the menu pane to update
  */
    void UpdateMenuL(CEikMenuPane* aMenuPane);

/*!
  @function Identify

  @discussion Return an identifying string
  @result An identification string
  */
    const TDesC& Identify();

public: // from MMdaAudioToneObserver
/*!
  @function MatoPrepareComplete

  @discussion Handle the event when a tone utility initialisation
operation has completed.
  @param aError indicates whether an error occurred.
  */
    void MatoPrepareComplete(TInt aError);

/*!
  @function MatoPlayComplete

  @discussion Handle the event when a tone playing operation has
completed.
  @param aError indicates whether an error occurred.
  */
    void MatoPlayComplete(TInt aError);

private:
```

```
/*!
  @function CToneAdapter

  @discussion Perform the first phase of two phase construction
  @param aAppUi the Ui to use
  */
    CToneAdapter(CL2LocationAppUi& aAppUi);

/*!
  @function ConstructL

  @discussion Perform the second phase construction of a CToneAdapter
object
  */

/*!
      @semantics Perform the second phase of two phase construction.
  */
    void ConstructL();

private:

/*! @var iMdaAudioToneUtility The audio tone utility object. */
    CMdaAudioToneUtility* iMdaAudioToneUtility;

/*! @var iAppUi Reference to the application's UI object. */
    CL2LocationAppUi& iAppUi;

    };

#endif // __CTONEADAPTER__
```

Appendix B. Bluetooth Network Server Code

```
import java.net.*;
import java.util.*;
import java.io.*;
import com.appliancestudio.jbluez.l2.*;
import java.sql.*;


public class L2LocationServer{
    public static void main(String a[]){
      //    while(true){
      L2Socket ss = null;
      try{
          exec(new String[]{"hciconfig", "hci0", "down"});
          exec(new String[]{"hciconfig", "hci0", "up"});
          ss = new L2Socket((short)12);
          ss.listen(2);
          L2ClientSocket cs=null;
          while(true){
            try{
                String user = "anonymous";
                System.out.println("waiting for connection");
                cs = ss.accept();
                //cs.getInputStream().read();
                OutputStream os = cs.getOutputStream();
                InputStream is = cs.getInputStream();
                int i = 0;

                byte b[] = new byte[17];
                while((i+=is.read(b,i,17-i)) <17);

                String btaddr = new String(b);
                System.out.println("read address: "+btaddr);

                byte buf[] = new byte[128];
                {
                  Class.forName("com.mysql.jdbc.Driver");
                  String connectionStr =
"jdbc:mysql://moshup.media.mit.edu:3306/etherthreads";

                  Connection con =
DriverManager.getConnection(connectionStr, "et","et");


                  PreparedStatement ps = con.prepareStatement("select
user from device_reg where btaddr = ?");
                  ps.setString(1,btaddr);
                  ResultSet rs = ps.executeQuery();
                  if(rs.next()){
                      user = rs.getString(1);
                  }
                }
                System.out.println("recieved connection from "+user);
                while( (i = is.read(buf)) >0 ){

                  System.out.println("read a "+new String(buf,0,i));
                  if( buf[0] == 'r' || buf[0] == 'R'){
                      String s = exec(new String[]{"hcitool",
"rssi",btaddr});
                      int rssi = 127;
                      try{
                        rssi =
Integer.valueOf(s.substring(19)).intValue();
                        rssi*=-1;
                        System.out.print('.');
                      }catch(Exception re){
                        re.printStackTrace();
                      }
                      os.write(rssi);
                  }else if( buf[0] == 'c' || buf[0] == 'C'){
                      try{
                        Class.forName("com.mysql.jdbc.Driver");
                        String connectionStr =
"jdbc:mysql://moshup.media.mit.edu:3306/etherthreads";

                        Connection con =
DriverManager.getConnection(connectionStr, "et","et");


                        PreparedStatement ps =
con.prepareStatement("insert into messages values(?,?,\"brad\",
?,?,NULL,NULL)");
                        ps.setInt(3,buf[1]);
                        ps.setInt(4,buf[2]);
                        int c;
                        for(c = 3; c <i && buf[c] != '~'&& buf[c] !=0 ;
c++);
                        System.out.println("c = "+c);
                        System.out.println("buf[c] = "+buf[c]);

                        ps.setString(1,new String(buf,3,c-3));
                        ps.setString(2,new String(buf,c+1 ,i-c-1));
                        ps.execute();
                        /*){
                            os.write(0);
                        }else{
                            os.write(1);
                            }*/

                      } catch (Exception sqle) {
                        sqle.printStackTrace();
                      }

                  }else if(buf[0] == 'm' || buf[0] == 'm'){
                      System.out.println("message request");
                      int c = 0;
```

```
                        String where = new String(buf,2,i);
                        int msgNum = buf[1];
                        System.out.println(where);

                        try{
                          Class.forName("com.mysql.jdbc.Driver");
                          String connectionStr =
"jdbc:mysql://moshup.media.mit.edu:3306/etherthreads";

                          Connection con =
DriverManager.getConnection(connectionStr, "et","et");


                          StringBuffer sb = new StringBuffer();
                          Statement st = con.createStatement();
                          ResultSet rs = st.executeQuery("select * from
messages"+where);
                          int count = 0;
                          String message = "No Message";
                          String thread = "";
                          String sender = "";
                          boolean range = true;
                          int firstx=0, firsty=0;
                          try{
                              while(rs.next() && range){
                                count++;
                                if(count == msgNum){
                                    message = rs.getString("message");

                                    thread = rs.getString("thread");

                                    sender = rs.getString("sender");
                                }
                                int x = rs.getInt("x");
                                int y = rs.getInt("y");
                                if(i == 1){
                                    firstx = x;
                                    firsty = y;
                                }
                                if((x-firstx)*(x-firstx) +(y-firsty)*(y-
firsty) >4){

                                    range = false;
                                }
                              }
                          }catch(Exception sqle){
                              sqle.printStackTrace();
                          }
                          sb.append(message);
                          sb.append((char)count);
                          sb.append('~');
                          sb.append(thread);
                          sb.append('~');
                          sb.append(sender);
                        //sb.append('\0');
                        os.write(sb.toString().getBytes());
                        System.out.println(sb.toString());
                      } catch (Exception sqle) {
                        sqle.printStackTrace();
                      }

                  }else if(buf[0] == 't' || buf[0] == 'T'){
                      System.out.println("Thread Request");
                      try{
                        Class.forName("com.mysql.jdbc.Driver");
                        String connectionStr =
"jdbc:mysql://moshup.media.mit.edu:3306/etherthreads";
                        Connection con =
DriverManager.getConnection(connectionStr, "et","et");


                        Statement st = con.createStatement();

                        ResultSet rs = st.executeQuery("select name
from groups where member = \""+user+"\"" );
                        StringBuffer groupClause = new StringBuffer();
                        while(rs.next()){
                            groupClause.append(" || groups = \"");
                            groupClause.append(rs.getString(1));
                            groupClause.append('\"');
                        }
                        rs = st.executeQuery("select distinct name,
rights from threads where user = \""+user+"\" "+groupClause);

                        StringBuffer sb = new StringBuffer();
                        sb.append("bgeneral,");
                        while(rs.next()){
                            sb.append(rs.getString(2).charAt(0));
                            sb.append(rs.getString(1));
                            sb.append(',');
                        }
                        os.write(sb.toString().getBytes());
                      } catch (Exception e) {
                        e.printStackTrace();
                      }
                  }else{
                      System.out.println("read \""+i+'\"');
                      os.write(-1);
                  }
                  Arrays.fill(buf,(byte)0);
                }
                System.out.println("read a "+i);
            }catch(Exception e){
                e.printStackTrace();
            }finally{
                if(cs != null){
                  cs.close();
                }
            }
          }
      }catch(Exception e){
            e.printStackTrace();
        }finally{
            if(ss != null){
              ss.close();
            }
        }
        //}
    }
    static String exec(String a[]){
        try {
            String ls_str;

            Process ls_proc = Runtime.getRuntime().exec(a);

            // get its output (your input) stream

            DataInputStream ls_in = new DataInputStream(
                                          ls_proc.getInputStream());

            try {
                StringBuffer buf= new StringBuffer();
                while ((ls_str = ls_in.readLine()) != null) {
                   buf.append(ls_str);
                }
                return buf.toString();
            } catch (IOException e) {
                System.exit(0);
                return null;
            }
        } catch (IOException e1) {
            System.err.println(e1);
            System.exit(1);
            return null;
        }
    }

}
```

Appendix C. L2CAP Java Wrapper Code

```
package com.appliancestudio.jbluez.l2;

import java.net.*;
import java.io.*;
import com.appliancestudio.jbluez.*;

public class L2ClientSocket{

    static{
        try{
            System.loadLibrary("jbluez");
        }catch(UnsatisfiedLinkError e){
            e.printStackTrace();
            System.out.print(System.getProperty("java.library.path"));
            System.exit(-1);
        }
    }

    public void close(){
      close(client_sock);
    }
    private native void close(int s);
    int client_sock;
    String addr;
    public L2ClientSocket(int s, String a){
      System.out.println("new client sock "+s);
      client_sock = s;
      addr = a;
    }
    public L2ClientSocket(String s, short psm){
      client_sock = connect(s,psm);
      addr = s;
    }
    public L2ClientSocket(int s){
      System.out.println("new client sock "+s);
      client_sock = s;
      addr = null;
    }
    private native int connect(String s, short psm);
    public InputStream getInputStream(){
      return new L2InputStream(client_sock);
      }
    public OutputStream getOutputStream(){
      return new L2OutputStream(client_sock);
    }
    public String getAddr(){
      return addr;
    }


}
```

```
package com.appliancestudio.jbluez.l2;

import java.net.*;
import java.io.*;
import com.appliancestudio.jbluez.*;


public class L2InputStream extends InputStream{
    static{
        try{
            System.loadLibrary("jbluez");
        }catch(UnsatisfiedLinkError e){
            e.printStackTrace();
            System.out.print(System.getProperty("java.library.path"));
            System.exit(-1);
        }
    }



    int sock;
    public L2InputStream(int s){
      sock = s;
    }
    public int read(byte[] b, int off, int len) throws Exception{
      return read(b,off,len,sock);
    }
    public int read(byte b[]) throws Exception{
      return read(b,0,b.length,sock);
    }
    public int read() throws Exception{
      byte b[] = new byte[1];
      int ret = read(b,0,1,sock);

      if(ret <=0){
          System.out.println("returning error code");
          return ret;
      }else{
          System.out.println("returning char");
          return (int) b[0];
      }
    }
    private native int read(byte[] b, int off, int len,int s)throws
Exception;

}
```

```
package com.appliancestudio.jbluez.l2;

import java.net.*;
import java.io.*;
import com.appliancestudio.jbluez.*;


public class L2OutputStream extends OutputStream{

    static{
        try{
            System.loadLibrary("jbluez");
        }catch(UnsatisfiedLinkError e){
            e.printStackTrace();
            System.out.print(System.getProperty("java.library.path"));
            System.exit(-1);
        }
    }



    int sock;
    public L2OutputStream(int s){
      sock = s;

    }
    public void write(byte[] b, int off, int len) throws Exception{
      write(b,off,len,sock);
    }
    public void write(byte[] b) throws Exception{
      write(b,0,b.length);
    }
    public void write(int b) throws Exception{
      write( new byte[]{(byte)b},0,1);
    }
    private native void write(byte[] b, int off, int len, int s)throws
Exception;

}
```

```
package com.appliancestudio.jbluez.l2;

import java.net.*;
import java.io.*;
import com.appliancestudio.jbluez.*;

public class L2Socket {


    static{
        try{
            System.loadLibrary("jbluez");
        }catch(UnsatisfiedLinkError e){
            e.printStackTrace();
            System.out.print(System.getProperty("java.library.path"));
            System.exit(-1);
        }
    }

    private int sock;

    public L2Socket(short psm)throws Exception{
      sock = init(psm);
    }
    public void close(){
      close(sock);
    }
    public native void close(int s);
    private native int init(short psm) throws Exception;
    public L2ClientSocket accept() throws Exception{

      long l  = accept_long(sock);
      System.out.println(l);
      String str = Long.toHexString(l);
      System.out.println(str);

      int len = str.length();
      String addr = "00:60:57:E1:42:7F";
      int s =6;
      if(len >2){
         addr = str.substring(0,len-2);
      }
      if(len >0){
          try{
            s = Integer.valueOf(str.substring( len-
1,len),16).intValue();
          }catch(Exception e){
            e.printStackTrace();
          }
      }
      if(s<0){
          throw new Exception("invalid file descriptor");
      }else{
          System.out.println("got client socket: "+s);
          L2ClientSocket l2cs = new L2ClientSocket(s,addr);
          return l2cs;
      }
    }

    private native long accept_long(int s);
    public void listen(int backlog)throws Exception{
      listen(sock,backlog);
    }
    private native void listen(int s, int backlog) throws Exception;



}
```

```
/*
   JBlueZ - Bluetooth Java Interface for Linux Using BlueZ

   Copyright (c) 2002 The Appliance Studio Limited.
   Written by Edward Kay <ed.kay@appliancestudio.com>
   http://www.appliancestudio.com

   This program is free software; you can redistribute it and/or modify
it under
   the terms of the GNU General Public License version 2 as published
by the
   Free Software Foundation.

   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR
   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY,
   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY
RIGHTS.

   IN NO EVENT SHALL THE COPYRIGHT HOLDER(S) AND AUTHOR(S) BE LIABLE
FOR ANY
   CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY
DAMAGES
   WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN
AN ACTION
   OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR
IN
   CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

   ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY PATENTS,
   COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS, RELATING TO USE OF THIS
SOFTWARE IS
   DISCLAIMED.

   ---

   This file, bluez.c contains the native (C) code for implementing the
Java
   native methods defined in BlueZ.java. These are called from Java
using the
   Java Native Interface (JNI).

   The associated header file,  com_appliancestudio_jbluez_BlueZ.h, is
generated
   by Java using the 'javah' tool. Do not edit
   com_appliancestudio_jbluez_BlueZ.h - if you wish to make changes,
make them
   in BlueZ.java, compile using 'javac', and then use the 'javah' tool.
Further
   information regarding this process can be found in any good JNI
tutorial or
   reference, or see the included Makefile.

   The purpose of this file is to expose the many functions provided by
the
```

```
   BlueZ libraries to Java. For more information on what each of the
functions
   do, see the BlueZ documentation (for C) or the associated Javadoc
for
   BlueZ.java.

   $Id:$
*/

/* Standard includes */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

/* JNI includes */
#include <jni.h>

#include "com_appliancestudio_jbluez_BlueZ.h"
#include "com_appliancestudio_jbluez_l2_L2Socket.h"
#include "com_appliancestudio_jbluez_l2_L2ClientSocket.h"
#include "com_appliancestudio_jbluez_l2_L2OutputStream.h"
#include "com_appliancestudio_jbluez_l2_L2InputStream.h"
#include "btio.h"
/* Bluetooth includes */
#include <sys/ioctl.h>
#include <sys/socket.h>
#include <errno.h>
#include <bluetooth/bluetooth.h>
#include <bluetooth/hci.h>
#include <bluetooth/hci_lib.h>
#include <bluetooth/l2cap.h>

void throwByName(JNIEnv *env,char *ex, char *msg)
{

      /* Throw a BlueZException in Java, with the given message */
      jclass exception_cls;

      exception_cls = (*env)->FindClass(env,ex);
      if ((*env)->ThrowNew(env, exception_cls, msg) < 0)
      {
            /* If there was a problem even throwing the exception,
something */
            /* big must be wrong. Print out any info available and
exit.     */
            printf("** Error throwing BlueZ exception - exiting **\n");
            printf("Message:\n%s\n", msg);
            exit(1);
      }
      return;
}


void throwIOExceptionWithLastError(JNIEnv *env, char *msg){
  throwByName( env,  "java/io/IOException", msg);
}
```

```
void throwOutOfMemoryError(JNIEnv *env, char *msg){
  throwByName( env,  "java/lang/IOException", msg);
}

void throwException(JNIEnv *env, char *msg)
{
  throwByName( env,  "com/appliancestudio/jbluez/BlueZException", msg);
}




/*
 * Class:     com_appliancestudio_jbluez_BlueZ
 * Method:    hciGetRSSI
 * Signature: (IILjava/lang/String;)I
 */
JNIEXPORT jint JNICALL Java_com_appliancestudio_jbluez_BlueZ_hciGetRSSI
(JNIEnv *env, jobject o, jint hciID, jint dd, jstring bdaddr_jstr){
  char *bdaddr_str;
  bdaddr_t bdaddr;

  /* Convert Java String Bluetooth address to a bdaddr_t type */
  bdaddr_str = (char*) (*env)->GetStringUTFChars(env, bdaddr_jstr,
NULL);
  baswap(&bdaddr, strtoba(bdaddr_str));
  (*env)->ReleaseStringUTFChars(env, bdaddr_jstr, bdaddr_str);

  struct hci_conn_info_req *cr;
      struct hci_request rq;
      read_rssi_rp rp;

      int opt;


      if (dd < 0) {
            perror("HCI device open failed");
            //          exit(1);
            return;
      }

      cr = malloc(sizeof(*cr) + sizeof(struct hci_conn_info));
      if (!cr)
            return;

      bacpy(&cr->bdaddr, &bdaddr);
      cr->type = ACL_LINK;

      if (ioctl(dd, HCIGETCONNINFO, (unsigned long) cr) < 0) {
            perror("Get connection info failed");
            //exit(1);
            return ;
      }

      memset(&rq, 0, sizeof(rq));
      rq.ogf    = OGF_STATUS_PARAM;
      rq.ocf    = OCF_READ_RSSI;
      rq.cparam = &cr->conn_info->handle;
      rq.clen   = 2;
      rq.rparam = &rp;
      rq.rlen   = READ_RSSI_RP_SIZE;

      if (hci_send_req(dd, &rq, 100) < 0) {

            perror("Read RSSI failed");
            exit(1);
      }

      if (rp.status) {
            printf("Read RSSI returned (error) status 0x%2.2X\n",
rp.status);

            return;
      }

      return rp.rssi;

      free(cr);


}



JNIEXPORT jint JNICALL Java_com_appliancestudio_jbluez_BlueZ_hciGetLQ
(JNIEnv *env, jobject o, jint hciID, jint dd, jstring bdaddr_jstr){
  char *bdaddr_str;
  bdaddr_t bdaddr;

  /* Convert Java String Bluetooth address to a bdaddr_t type */
  bdaddr_str = (char*) (*env)->GetStringUTFChars(env, bdaddr_jstr,
NULL);
  baswap(&bdaddr, strtoba(bdaddr_str));
  (*env)->ReleaseStringUTFChars(env, bdaddr_jstr, bdaddr_str);

  struct hci_conn_info_req *cr;
      struct hci_request rq;
      get_link_quality_rp rp;
      int opt;


      if (dd < 0) {
            perror("HCI device open failed");
            //          exit(1);
            return;
      }
      cr = malloc(sizeof(*cr) + sizeof(struct hci_conn_info));
      if (!cr)
            return;

      bacpy(&cr->bdaddr, &bdaddr);
      cr->type = ACL_LINK;
      if (ioctl(dd, HCIGETCONNINFO, (unsigned long) cr) < 0) {
            perror("Get connection info failed");
            //exit(1);
            return ;
      }




      memset(&rq, 0, sizeof(rq));
      rq.ogf     = OGF_STATUS_PARAM;
      rq.ocf     = OCF_GET_LINK_QUALITY;
      rq.cparam = &cr->conn_info->handle;
      rq.clen    = 2;
      rq.rparam = &rp;
      rq.rlen    = GET_LINK_QUALITY_RP_SIZE;

      if (hci_send_req(dd, &rq, 100) < 0) {
            perror("Read RSSI failed");
            exit(1);
      }

      if (rp.status) {
            printf("Read RSSI returned (error) status 0x%2.2X\n",
rp.status);

            return;
      }

      return rp.link_quality;

      free(cr);


}



JNIEXPORT jint JNICALL Java_com_appliancestudio_jbluez_BlueZ_hciGetTPL
(JNIEnv *env, jobject o, jint hciID, jint dd, jstring bdaddr_jstr){
  char *bdaddr_str;
  bdaddr_t bdaddr;

  /* Convert Java String Bluetooth address to a bdaddr_t type */
  bdaddr_str = (char*) (*env)->GetStringUTFChars(env, bdaddr_jstr,
NULL);
  baswap(&bdaddr, strtoba(bdaddr_str));
  (*env)->ReleaseStringUTFChars(env, bdaddr_jstr, bdaddr_str);

  struct hci_conn_info_req *cr;
      struct hci_request rq;
      read_transmit_power_level_cp cp;
      read_transmit_power_level_rp rp;


      int opt;
```

```
        if (dd < 0) {
              perror("HCI device open failed");
              //          exit(1);
              return;
        }

        cr = malloc(sizeof(*cr) + sizeof(struct hci_conn_info));
        if (!cr)
              return;

        bacpy(&cr->bdaddr, &bdaddr);
        cr->type = ACL_LINK;
              cp.type = 0;

        if (ioctl(dd, HCIGETCONNINFO, (unsigned long) cr) < 0) {
              perror("Get connection info failed");
              //exit(1);
              return ;
        }
        cp.handle = cr->conn_info->handle;

        memset(&rq, 0, sizeof(rq));
        rq.ogf    = OGF_HOST_CTL;
        rq.ocf    = OCF_READ_TRANSMIT_POWER_LEVEL;
        rq.cparam = &cr->conn_info->handle;
        rq.clen   = READ_TRANSMIT_POWER_LEVEL_CP_SIZE;
        rq.rparam = &rp;
        rq.rlen   = READ_TRANSMIT_POWER_LEVEL_RP_SIZE;

        if (hci_send_req(dd, &rq, 100) < 0) {
              perror("Read RSSI failed");
              exit(1);
        }

        if (rp.status) {
              printf("Read RSSI returned (error) status 0x%2.2X\n",
rp.status);

              return;
        }

        return rp.level;

        free(cr);


}


bdaddr_t bdaddr;
int socktype = SOCK_SEQPACKET;


extern JNIEXPORT jint JNICALL
Java_com_appliancestudio_jbluez_l2_L2Socket_init (JNIEnv *env, jobject
o, jshort psm){
  printf("\nhello.........\n");
  int imtu = 672;


  int opt;
  struct l2cap_options opts;
  int s;
  struct sockaddr_l2 loc_addr;
  printf("1\n");
  if( (s = socket(AF_BLUETOOTH, socktype, BTPROTO_L2CAP)) < 0 ) {
    char exp[40]; sprintf(exp,  "Can't create socket. %s(%d)",
strerror(errno), errno);
    throwException(env,exp);
  }
  printf("2\n");

  loc_addr.l2_family = AF_BLUETOOTH;
  loc_addr.l2_bdaddr = bdaddr;
  loc_addr.l2_psm    = htobs(psm);
  printf("3\n");
  if( bind(s, (struct sockaddr *) &loc_addr, sizeof(loc_addr)) < 0 ) {
    char exp[40]; sprintf(exp,  "Can't bind socket. %s(%d)",
strerror(errno), errno);
    throwException(env,exp);
  }

  printf("4\n");
  /* Set link mode */
  opt = 0;
  printf("4\n");

  if (setsockopt(s, SOL_L2CAP, L2CAP_LM, &opt, sizeof(opt)) < 0) {
    char exp[40]; sprintf(exp,  "Can't set  L2CAP link mode. %s(%d)",
strerror(errno), errno);
    throwException(env,exp);
  }
  printf("5\n");

  /* Get default options */
  opt = sizeof(opts);
   if (getsockopt(s, SOL_L2CAP, L2CAP_OPTIONS, &opts, &opt) < 0) {
    char exp[40]; sprintf(exp,  "Can't get default L2CAP options.
%s(%d)", strerror(errno), errno);
    throwException(env,exp);
  }
   printf("6\n");

  /* Set new options */
  opts.imtu = imtu;
  if (setsockopt(s, SOL_L2CAP, L2CAP_OPTIONS, &opts, opt) < 0) {
    char exp[40]; sprintf(exp,  "Can't set L2CAP options. %s(%d)",
strerror(errno), errno);
    throwException(env,exp);
  }
  printf("7\n");

  return s;
```

```
}



extern JNIEXPORT jlong JNICALL
Java_com_appliancestudio_jbluez_l2_L2Socket_accept_1long (JNIEnv *env,
jobject o, jint s)
{
  jvalue args[1];
  jclass l2cs_cls;
  jmethodID l2cs_con_id;
  jobject l2cs;


  printf("\n");
  printf(".");
  jint s1;
  printf(".");
  struct sockaddr_l2 rem_addr;
  printf(".");
  int opt;
  printf(".");
  struct l2cap_options opts;
  printf(".");
  opt = sizeof(opts);
  if (getsockopt(s, SOL_L2CAP, L2CAP_OPTIONS, &opts, &opt) < 0) {
    printf("-");
    char exp[40]; sprintf(exp,  "Can't get default L2CAP options.
%s(%d)", strerror(errno), errno);
    printf("-");

    throwException(env,exp);

    return -3;
  }
  printf(".");

  if( (s1 = accept(s, (struct sockaddr *)&rem_addr, &opt)) < 0 ){
    printf("-");
    char exp[40];

    printf("-");
    sprintf(exp,  "Accept failed. %s(%d)", strerror(errno), errno);
    printf("-");

    throwException(env,exp);
    printf("-");
    return -1;
  }
  unsigned long ret = 0;
  int i = 0;
  for(i = 5; i>=0; i--){
    ret+=(unsigned long)rem_addr.l2_bdaddr.b[i];
    ret*=(unsigned long)256;
  }

  ret+= (unsigned long)s1;
  return ret;
}
extern JNIEXPORT void JNICALL
Java_com_appliancestudio_jbluez_l2_L2Socket_listen (JNIEnv *env,
jobject o, jint s, jint backlog){
  listen(s, backlog);
}
#define BUF_SIZE 8192
#define JVM_IO_ERR (-1)
#define JVM_IO_INTR (-2)

extern JNIEXPORT jint JNICALL
Java_com_appliancestudio_jbluez_l2_L2InputStream_read___3BIII (JNIEnv
*env, jobject this, jbyteArray bytes,
          jint off, jint len, int fd)
{
    int  nread, datalen;
    char stackBuf[BUF_SIZE];
    char *buf = 0;

    /*if (IS_NULL(bytes)) {
        throwNullPointerException(env, 0);
        return -1;
    }
    */
    datalen = (*env)->GetArrayLength(env, bytes);

    if ((off < 0) || (off > datalen) ||
        (len < 0) || ((off + len) > datalen) || ((off + len) < 0)) {
        throwByName(env, "java/lang/IndexOutOfBoundsException", 0);
        return -1;
    }

    if (len == 0) {
        return 0;
    } else if (len > BUF_SIZE) {
        buf = malloc(len);
        if (buf == 0) {
            throwOutOfMemoryError(env, 0);
            return 0;
        }
    } else {
        buf = stackBuf;
    }


    nread = JVM_Read(fd, buf, len);
    if (nread > 0) {
        (*env)->SetByteArrayRegion(env, bytes, off, nread, (jbyte
*)buf);
    } else if (nread == JVM_IO_ERR){
      nread = -2;
      throwIOExceptionWithLastError(env, "Read error");
    } else if (nread == JVM_IO_INTR){  /* EOF */
      nread = -3;
        throwByName(env, "java/io/InterruptedIOException", 0);
    } else { /* EOF */
```

```
        nread = -1;
    }
    if (buf != stackBuf) {
        free(buf);
    }
    return nread;
}




extern JNIEXPORT void JNICALL
Java_com_appliancestudio_jbluez_l2_L2OutputStream_write___3BIII (JNIEnv
*env, jobject this, jbyteArray bytes,
          jint off, jint len, jint fd)
{
    int  n, datalen;
    char stackBuf[BUF_SIZE];
    char *buf = 0;

    /*if (IS_NULL(bytes)) {
        throwNullPointerException(env, 0);
        return;
    }
    */
    datalen = (*env)->GetArrayLength(env, bytes);

    if ((off < 0) || (off > datalen) ||
        (len < 0) || ((off + len) > datalen) || ((off + len) < 0)) {
        throwByName(env, "java/lang/IndexOutOfBoundsException", 0);
        return;
    }

    if (len == 0) {
        return;
    } else if (len > BUF_SIZE) {
        buf = malloc(len);
        if (buf == 0) {
            throwOutOfMemoryError(env, 0);
            return;
        }
    } else {
        buf = stackBuf;
    }


    (*env)->GetByteArrayRegion(env, bytes, off, len, (jbyte *)buf);

    if (!(*env)->ExceptionOccurred(env)) {
        off = 0;
        while (len > 0) {
            n = JVM_Write(fd, buf+off, len);
            if (n == JVM_IO_ERR) {
                throwIOExceptionWithLastError(env, "Write error");
                break;
            } else if (n == JVM_IO_INTR) {
```

```
            throwByName(env, "java/io/InterruptedIOException", 0);
                break;
            }
            off += n;
            len -= n;
        }
    }
    if (buf != stackBuf) {
        free(buf);
    }
}



extern JNIEXPORT jint JNICALL
Java_com_appliancestudio_jbluez_l2_L2ClientSocket_connect(JNIEnv *env,
jobject o, jstring bdaddr_jstr, jshort psm){
  int imtu = 672;
  char *svr;
  jint s;

  svr = (char*) (*env)->GetStringUTFChars(env, bdaddr_jstr, NULL);

  struct sockaddr_l2 rem_addr, loc_addr;
        struct l2cap_options opts;
        int  opt;

        if( (s = socket(PF_BLUETOOTH, socktype, BTPROTO_L2CAP)) < 0 ) {
        char exp[40]; sprintf(exp, "Can't create socket. %s(%d)",
strerror(errno), errno);
                return -1;
        }

        memset(&loc_addr, 0, sizeof(loc_addr));
        loc_addr.l2_family = AF_BLUETOOTH;
        loc_addr.l2_bdaddr = bdaddr;
        if( bind(s, (struct sockaddr *) &loc_addr, sizeof(loc_addr)) <
0 ) {
                char exp[40]; sprintf(exp,  "Can't bind socket.
%s(%d)", strerror(errno), errno);
                throwByName(env, "java/io/IOException",exp);
                exit(1);
        }

        /* Get default options */
        opt = sizeof(opts);
        if( getsockopt(s, SOL_L2CAP, L2CAP_OPTIONS, &opts, &opt) < 0 )
{
                char exp[40]; sprintf(exp,  "Can't get default L2CAP
options. %s(%d)", strerror(errno), errno);
                throwByName(env, "java/io/IOException",exp);
                return -1;
        }

        /* Set new options */
        opts.omtu = 672;
        opts.imtu = imtu;
        if( setsockopt(s, SOL_L2CAP, L2CAP_OPTIONS, &opts, opt) < 0 ) {
                char exp[40]; sprintf(exp,  "Can't set L2CAP options.
%s(%d)", strerror(errno), errno);
                throwByName(env, "java/io/IOException",exp);
                return -1;
        }

        memset(&rem_addr, 0, sizeof(rem_addr));
        rem_addr.l2_family = AF_BLUETOOTH;
        baswap(&rem_addr.l2_bdaddr, strtoba(svr));
        rem_addr.l2_psm = htobs(psm);
        if( connect(s, (struct sockaddr *)&rem_addr, sizeof(rem_addr))
< 0 ){
                char exp[40]; sprintf(exp,  "Can't connect. %s(%d)",
strerror(errno), errno);
                close(s);
                throwByName(env, "java/io/IOException",exp);
                return -1;
        }

        opt = sizeof(opts);
        if( getsockopt(s, SOL_L2CAP, L2CAP_OPTIONS, &opts, &opt) < 0 ){
                char exp[40]; sprintf(exp,  "Can't get L2CAP options.
%s(%d)", strerror(errno), errno);
                close(s);
            throwByName(env, "java/io/IOException",exp);
                return -1;
        }



      return s;

}
extern void Java_com_appliancestudio_jbluez_l2_L2Socket_close__I
(JNIEnv *env, jobject o, jint sock){
  close(sock);
}
extern void Java_com_appliancestudio_jbluez_l2_L2ClientSocket_close__I
(JNIEnv *env, jobject o, jint sock){
  close(sock);
}
```

Appendix D. Database Web Interface Code

```
import java.sql.*;
import java.io.*;
import java.net.*;
import java.io.*;
import java.util.*;


public class dbtest{
    static Connection con;
    public static void main(String a[]){
       //     PrintStream out = System.out;
       try{

           Class.forName("com.mysql.jdbc.Driver");
           String connectionStr =
"jdbc:mysql://moshup.media.mit.edu:3306/etherthreads";
           con = DriverManager.getConnection(connectionStr,
"et","et");
           ServerSocket ss = new ServerSocket(8082,10);
           while(true){
              Socket s = ss.accept();
              InputStream is = s.getInputStream();
              BufferedReader reader = new BufferedReader(new
InputStreamReader(is));
              OutputStream os = s.getOutputStream();
              PrintStream out = new PrintStream(os);


              String string = reader.readLine();
              while(string != null){
                  try{
                     if(string.indexOf("GET") != -1){
                         Map m = getArgs(string);
                         int i;
                         if(string.indexOf("messageDB") !=-1){
                            messageDB(out);
                         }else if(string.indexOf("deleteMessage")
!=-1){
                            deleteMessage(m);
                            messageDB(out);
                         }else if(string.indexOf("createMessage")
!=-1){
                            createMessage(m);
                            messageDB(out);
                         }else if(string.indexOf("userManager") !=-
1){
                         userManager(out);
                         }else if(string.indexOf("deleteUser") !=-
1){
                            deleteUser(m);
                            userManager(out);
                         }else if(string.indexOf("addUser") !=-1){
                            addUser(m);
                            userManager(out);
                         }else if(string.indexOf("deviceRegistry")
!=-1){
                            deviceRegistry(out);
                         }else if(string.indexOf("deleteDevice")
!=-1){
```

```
                              uregisterDevice(m);
                              deviceRegistry(out);
                           }else if(string.indexOf("registerDevice")
!=-1){
                              registerDevice(m);
                              deviceRegistry(out);
                           }else{
                              frontPage(out);
                           }
                           string = null;
                       }else{
                           string = reader.readLine();

                       }
                       out.flush();
                       out.close();
                    }catch(SQLException sqle){
                       try{
                           con.close();
                       }catch(Exception e2){
                           e2.printStackTrace();
                       }
                       try{
                           con = null;
                           con =
DriverManager.getConnection(connectionStr, "et","et");
                       }catch(Exception e2){
                           e2.printStackTrace();
                       }
                    }
                }
             }
             //con.close();
        }catch(Exception e){
             e.printStackTrace();
        }

     }
     static Map getArgs(String string) throws Exception{
        System.out.println("getting args in string: "+string);
        HashMap map = new HashMap();
        int a = 0;
        int e = 0;
        a = string.indexOf('?');
        while(a != -1){
             System.out.println("a = "+a);
             e = string.indexOf('=',a);
             if(e != -1){
                System.out.println("e = "+e);
                String var =
URLDecoder.decode(string.substring(a+1,e),"UTF-8");
                a = string.indexOf('&',e);
                if(a != -1){
                     String val =
URLDecoder.decode(string.substring(e+1,a),"UTF-8");
                     System.out.println(var+" = "+val);
                     map.put(var, val);
                }else{
                     a = string.indexOf(' ',e);
                     if( a != -1){
                        String val =
URLDecoder.decode(string.substring(e+1,a),"UTF-8");
                        System.out.println(var+" = "+val);
                    map.put(var, val);
                    a = -1;
                 }
              }
           }else{
              a = -1;
           }

        }
        return map;
     }
     static void  frontPage(PrintStream out){



           StringBuffer sb = new StringBuffer();
           sb.append("<html>");
           sb.append("<a href = \"messageDB\" > Messages </a>");
           sb.append("<br>");
           sb.append("<a href = \"userManager\" > User Management
</a>");
           sb.append("<br>");
           sb.append("<a href = \"deviceRegistry\" > Device
Registry </a>");
           sb.append("</html>");
           out.print(sb.toString());


     }
     static void messageDB(PrintStream out) throws SQLException{



                 StringBuffer sb = new StringBuffer();

                 sb.append("<html>");
                 sb.append(" <form action=\"createMessage\"
method=\"get\">");

                 sb.append("message:  <input type=\"text\"
name=\"message\" maxlength=\"128\">");
                 sb.append("   thread: <input type=\"text\"
name=\"thread\"  >");
                 sb.append("<br>sender: <input type=\"text\"
name=\"sender\"  >");
                 sb.append("    x: <input type=\"text\" name=\"x\"
>");
                 sb.append("    y: <input type=\"text\" name=\"y\"
>");
                 sb.append("<br><input type=\"submit\"
value=\"Create Message\" /></form>");

                 Statement st = con.createStatement();
                 ResultSet rs = st.executeQuery("select * from
messages");
                 sb.append("<table border = \"1\">");
                 while(rs.next()){
                    sb.append("<tr>");
                    for(int i = 1; i< 6; i++){
                       sb.append("<td>");
                       sb.append(rs.getString(i));
```

```
                        sb.append("</td>");
                    }
                    sb.append("<td>");
                    sb.append(rs.getTimestamp(6));
                    sb.append("</td>");

                    sb.append("<td>");
                    sb.append("<a href =
\"deleteMessage?id="+rs.getString(7)+"\" > delete </a>");
                    sb.append("</td>");
                    sb.append("</tr>");
                }
                sb.append("</table>");
                sb.append("<a href = \"/\" > Home </a>");
                sb.append("</html>");
                out.print(sb.toString());



    }
    static void userManager(PrintStream out) throws SQLException{




                StringBuffer sb = new StringBuffer();

                sb.append("<html>");
                sb.append(" <form action=\"addUser\"
method=\"get\">");

                sb.append("user name:  <input type=\"text\"
name=\"user\" /><br />group name: <input type=\"text\"
name=\"group\"  />");
                sb.append("<br /><input type=\"submit\"
value=\"Add User\" /></form>");


                Statement st = con.createStatement();
                ResultSet rs = st.executeQuery("select * from
groups");
                sb.append("<table border = \"1\">");
                while(rs.next()){
                    sb.append("<tr>");
                    for(int i = 1; i< 3; i++){
                        sb.append("<td>");
                        sb.append(rs.getString(i));
                        sb.append("</td>");
                    }
                    sb.append("<td>");
                    sb.append("<a href =
\"deleteUser?id="+rs.getString(3)+"\" > delete </a>");
                    sb.append("</td>");
                    sb.append("</tr>");
                }
                sb.append("</table>");
                sb.append("<a href = \"/\" > Home </a>");
                sb.append("</html>");
                out.print(sb.toString());


    }
    static void threadManager(PrintStream out) throws
SQLException{




                 StringBuffer sb = new StringBuffer();

                 sb.append("<html>");
                 /*sb.append(" <form action=\"addUser\"
method=\"get\">");

                 sb.append("user name:  <input type=\"text\"
name=\"user\" /><br />group name: <input type=\"text\"
name=\"group\"  />");
                 sb.append("<br /><input type=\"submit\"
value=\"Add User\" /></form>");
                 */

                 Statement st = con.createStatement();
                 ResultSet rs = st.executeQuery("select * from
groups");
                 sb.append("<table border = \"1\">");
                 while(rs.next()){
                    sb.append("<tr>");
                    for(int i = 1; i< 5; i++){
                        sb.append("<td>");
                        sb.append(rs.getString(i));
                        sb.append("</td>");
                    }
                    sb.append("<td>");
                    sb.append("<a href =
\"deleteThread?id="+rs.getString(5)+"\" > delete </a>");
                    sb.append("</td>");
                    sb.append("</tr>");
                 }
                 sb.append("</table>");
                 sb.append("<a href = \"/\" > Home </a>");
                 sb.append("</html>");
                 out.print(sb.toString());

    }
    static void deviceRegistry(PrintStream out) throws
SQLException{




                 StringBuffer sb = new StringBuffer();

                 sb.append("<html>");
                 sb.append(" <form action=\"registerDevice\"
method=\"get\">");

                 sb.append("user name:     <input type=\"text\"
name=\"user\" /><br />device btaddr: <input type=\"text\"
name=\"btaddr\"  />");
                 sb.append("<br /><input type=\"submit\"
value=\"Register Device\" /></form>");


                 Statement st = con.createStatement();
                 ResultSet rs = st.executeQuery("select * from
device_reg");
                 sb.append("<table border = \"1\">");
                 while(rs.next()){
                    sb.append("<tr>");
                    for(int i = 1; i< 3; i++){
                         sb.append("<td>");
                         sb.append(rs.getString(i));
                         sb.append("</td>");
                    }
                    sb.append("<td>");
                    sb.append("<a href =
\"deleteDevice?id="+rs.getString(3)+"\" > delete </a>");
                    sb.append("</td>");
                    sb.append("</tr>");
                 }
                 sb.append("</table>");
                 sb.append("<a href = \"/\" > Home </a>");
                 sb.append("</html>");
                 out.print(sb.toString());


     }

     public static void deleteUser(Map m)throws SQLException{
        PreparedStatement ps = con.prepareStatement("delete from
groups where id = ?");
        ps.setString(1, m.get("id").toString());
        ps.execute();

     }
     public static void uregisterDevice(Map m)throws SQLException{
        PreparedStatement ps = con.prepareStatement("delete from
device_reg where id = ?");
        ps.setString(1, m.get("id").toString());
        ps.execute();

     }
     public static void deleteMessage(Map m)throws SQLException{
        PreparedStatement ps = con.prepareStatement("delete from
messages where id = ?");
        ps.setString(1, m.get("id").toString());
        ps.execute();

     }
     static void addUser(Map map) throws SQLException{
        PreparedStatement ps = con.prepareStatement("Insert into
groups values(?,?,?)");
        ps.setInt(3,0);
        ps.setString(2,map.get("user").toString());
        ps.setString(1,map.get("group").toString());
        ps.execute();

     }
     static void registerDevice(Map map) throws SQLException{
        PreparedStatement ps = con.prepareStatement("Insert into
device_reg values(?,?,?)");
        ps.setInt(3,0);
        ps.setString(2,map.get("user").toString());
        ps.setString(1,map.get("btaddr").toString());
        ps.execute();

     }
```

```
    static void createMessage(Map map) throws SQLException{
       PreparedStatement ps = con.prepareStatement("insert into
messages values(?,?,?, ?,?,NULL,NULL)");
       ps.setInt(4,Integer.parseInt(map.get("x").toString()));
       ps.setInt(5,Integer.parseInt(map.get("y").toString()));
       ps.setString(1,map.get("message").toString());
       ps.setString(2,map.get("thread").toString());
       ps.setString(3,map.get("sender").toString());
       ps.execute();


    }


}
```