A Middleware Based Approach to Dynamically Deploy Location Based Services Onto Heterogeneous Mobile Devices Using Bluetooth in Indoor Environment

Pampa Sadhukhan[1], Rijurekha Sen[2], and Pradip K. Das[3]

[1] School of Mobile Computing & Communication,
Jadavpur University, India 700032.
pampa.sadhukhan@gmail.com
[2] Dept. of CSE, IIT Bombay, Powai, Mumbai-400076.
riju@cse.iitb.ac.in
[3] Faculty of Engineering & Technology,
Mody Institute of Technology & Science, India 332311.
pkdas@ieee.org

Abstract. Several methods for providing location based service (LBS) to mobile devices in indoor environment using wireless technologies like WLAN, RFID and Bluetooth have been proposed, implemented and evaluated. However, most of them do not focus on heterogeneity of mobile platforms, memory constraint of mobile devices, the adaptability of client device to the new services it discovers whenever it reaches a new location. In this paper, we have proposed a Middleware based approach of LBS provision in the indoor environment, where a Bluetooth enabled Base Station (BS) detects Bluetooth enabled mobile devices and pushes a proper client application only to those devices that belong to some registered subscriber of LBS. This dynamic deployment enables the mobile clients to access any new service without having preinstalled interface to that service beforehand and thus the client′s memory consumption is reduced. Our proposed work also addresses the other issues like authenticating the clients before providing them LBSs and introducing paid services. We have evaluated its performance in term of file transfer time with respect to file size and throughput with respect to distance. Experimental results on service consumption time by the mobile client for different services are also presented.

Key words: Base Station (BS), Location Based Service (LBS), Middleware, Bluetooth.

1 Introduction and Related Work

The current advancement in wireless communication technology and mobile platform has led the researchers to work in the field of providing location based services (LBSs) to mobile clients. It is quite helpful to have LBS in indoor environments like a shopping mall, a railway station, an airport or a university

department building for sharing information of exciting purchase offers, schedule change of train and flight, direction to reach a location and so on. Several major problems faced by the service providers to provide aforementioned services are pointed out in the following sub sections.

1.1 Problem of localization and technology availability

Several approaches [1, 2] to provide context-aware services and LBS in ubiquitous environment have been proposed. These approaches depend on the integrated positioning technology employing Global Positioning System (GPS), Radio-frequency identification (RFID) and Wi-Fi to provide location-related information and context-aware services to the mobile users. However, one of the major challenges for providing LBS indoor is localization, as GPS does not work properly in indoor environment. Cell phone based localization is too coarse to localize within a building. Acoustic wave based localization as in [3] needs dense sensor deployment. On the other hand, a Bluetooth enabled Base Station (BS) can localize all Bluetooth enabled devices in its vicinity very easily using Bluetooth inquiry procedure [4]. Although several other technologies like Zigbee and RFID [5, 6, 7] can also be used for indoor localization, Bluetooth is the most widely available technology in mobile devices.

In this work, the position of a mobile user is determined by the position of the BS that detects the user within its Bluetooth connectivity range (20 meters only) and is provided to the user via a location service.

1.2 Problem of connectivity

In our previous work [8], we have presented a Middleware based approach and an LBS-Infrastructure that would enable service providers to deploy Location-aware application efficiently and to publish them to the mobile users. Mobile clients could consume these services over Bluetooth, GPRS or Wi-Fi connection. Wi-Fi based localization and LBS provision have been also been studied in [9,10,11]. Wi-Fi communication, however, is hindered by varying network connectivity and limited bandwidth. In this work, we do not use the infrastructure network of wireless access points, if any, inside the building and hence do not pose problem to the normal internet users. Instead we deploy some Bluetooth enabled BS inside the building, that provide LBS to Bluetooth enabled mobile devices in its vicinity. In Table 1, we compare various available technologies to validate the use of Bluetooth due to its precise indoor localization, good connectivity, low cost and easy availability in mobile devices. Some might argue that Bluetooth range is too limited. Nevertheless, in our kind of application, setting up a few BSs at strategic positions like baggage counter in shopping malls, ticket counters or information centers in railway stations and airports and departmental offices in university buildings will be sufficient, as almost all people visit these locations initially after reaching the place.

Table 1. Reasons for choosing Bluetooth

Technology	Indoor Localization	Connectivity	Availability	Cost
Bluetooth	good	good	good	low
Wi-Fi	good	load on Internet users	good	low
Zigbee	good	good	poor	low
RFID	good	N.A.	poor	infrastructure cost
Accoustic sensing	good	N.A.	good	infrastructure cost
GPS	poor	N.A.	good	low
SIM-based	poor	good	good	service provider charges
GPRS	N.A.	good	good	service provider charges

1.3 Problem of application preinstallation in mobile devices

The authors in [12, 13] present Bluetooth based LBS provision in indoor environment. In [12], authors have presented a Bluetooth and Java based context-aware system, UbiqMuseum, to provide the museum visitors the precise information about the arts that they are viewing inside the museum. In their proposed system, the mobile client obtains information on one or more pieces of arts via Museum Information Point (MIP) from the central data server where information related to all arts inside the museum are stored. The MIP provides that information to client over Bluetooth connection. However, this sort of context-aware information provisioning is limited to those mobile devices that have the support for the standard Java API for Bluetooth Wireless Technology [14] to communicate with MIP and they have the preinstalled client application running on the device. In UbiqMuseum, each MIP acts as a master in its own piconet [15] to communicate with multiple mobile clients acting as slave devices in that piconet. The authors have also proposed an algorithm to form Scatternet [16] in which a mobile client needs to act either as a master/slave bridge device in the MIPs piconet or to join to another piconet hosted by another mobile client acting as a bridge device in the MIPs piconet when it finds that MIP already has five active slave devices. This algorithm would fail to provide uninterrupted information delivery in case the mobile client acting as master/slave bridge device leaves the existing piconet to view some other art inside the museum.

In [13], authors have described a Bluetooth technology based system, SBIL, to provide location information for mobile users in indoor environment by using the Received Signal Strength (RSS) measurements. SBIL also provides service information from the server to the mobile users via Bluetooth beacons, which also help to track location of the mobile user by using the signal strength from the mobile device. To provide uninterrupted service to the mobile user in the SBIL system, the mobile client acts as a master and remain connected to at least four beacons all the time it wishes to navigate the service. The negativity in this approach is that it requires a high density of Bluetooth beacons around the area where SBIL is deployed. Moreover, mobile devices must have extended battery life to remain connected to at least four beacons all the time.

The issue of adapting a mobile device to new LBSs it is offered whenever it moves to a new location without having any prior installed client applications for those services has been highlighted by very few researchers. It is impossible to predict what applications might come in use and thus, preinstallation of the applications wastes a lot of memory. A sensible way to address the problem is providing all software and data needed to consume LBS to the device that chooses to consume that LBS. In [17], authors have proposed and given a prototype implementation of a comprehensive architecture that enables the heterogeneous mobile devices to discover the new services as it moves to a new location and then to invoke those services. Their proposed architecture exposes the interface of the services and accessible resources through an interface specification language (ISL) to avoid the difficulties in obtaining all the required custom user interfaces (UIs) for invoking all the services on different types of hardware. However only those mobile devices which have proper compiler support for that interface specification language, could be able to discover and access the services as they need to read the ISL file for the services to be invoked.

In our proposed system, the BS push the appropriate client application based on the profile of the devices and other data files needed to consume LBS to the devices. Thus, in our work, heterogeneous mobile devices can invoke LBSs dynamically irrespective of whether these devices have support for Java Runtime Environment or not.

1.4 Problem of authenticating the user and offering paid services

We have introduced a registration process to be followed by each mobile user willing to invoke LBSs, for the purpose of preventing malicious users from blocking network resources and offering the paid services to the users. In this work, the mobile user obtains a unique user name and password based on device′s profile after successful registration. The user has to set the Bluetooth device name to the user name given to him/her so that any BS can recognize it as a registered mobile device during the Bluetooth inquiry procedure. The password is used for authenticating the user before allowing him/her to consume LBSs.

2 System Design and Implementation

Our proposed system shown in fig. 1 consists of several BSs among which one is attached to a GSM box via a serial port to collect the registration request message from the mobile users. That BS is called central BS. The central is connected to other BSs through a LAN so that central BS can periodically broadcast the list of registered mobile users to the other BSs.

Each BS in our system is Bluetooth enabled and a Java based Middleware is deployed onto each BS. The registration process to be followed by mobile users to invoke LBSs is described in subsection 2.1. The sequences of events that happen when a new mobile device comes in vicinity of a BS are i) upon detecting a

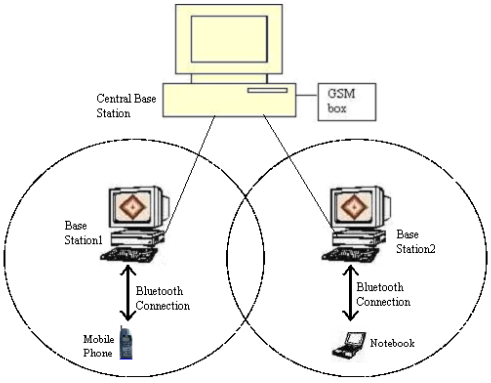

Fig. 1. Architecture of the system.

new device during Bluetooth inquiry operation, BS tries to send an appropriate client application to the device depending on the its profile if the Bluetooth device name of that device appears in the list of registered user name. ii) The user can run the client application and need to send the password to the BS over the Bluetooth connection. iii) After successful authentication, mobile client would receive the list of services available there and would be able to consume the LBSs.

2.1 Registration Process

A mobile user who is willing to invoke LBSs has to send a SMS (Short Messaging Service) carrying his/her device′s profile and preferred user name to a well known phone number assigned to the GSM box of the central BS. The Middleware deployed onto the central BS would generate, at first, a unique user name and password for that user and then sends back to that user a SMS containing the user name, password and an instruction to set the Bluetooth device name to that user name generated by the central BS. How the user name is generated is described in subsection 2.3.2. The central BS periodically broadcasts the registered users′ user name, password and registration time along with their devices′ profile to the other BSs. The registration would remain valid for 24 hours.

The telecom operators can easily provide the paid services through wireless technology like Bluetooth to the mobile users in places like shopping mall, airport etc by adopting this registration process.

2.2 Middleware Software

The Middleware consists of several modules. Fig. 2 shows all the modules of the Middleware and how they work together to provide LBSs to the mobile user. The Main Module (MM) receives and handles the request from the clients sent over Bluetooth Connection. It invokes the Authentication Module (AM)

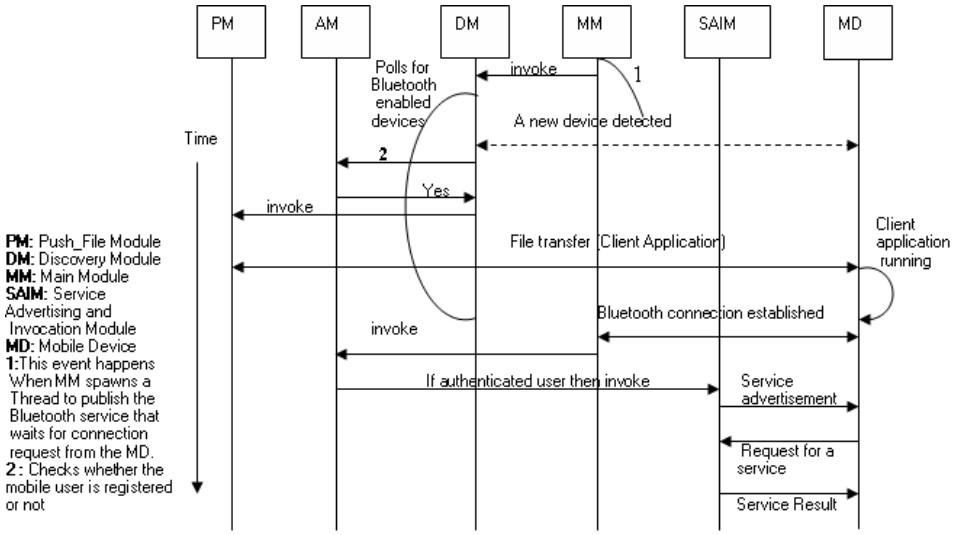

Fig. 2. Sequence diagram of interaction between different modules of Middleware and MD.

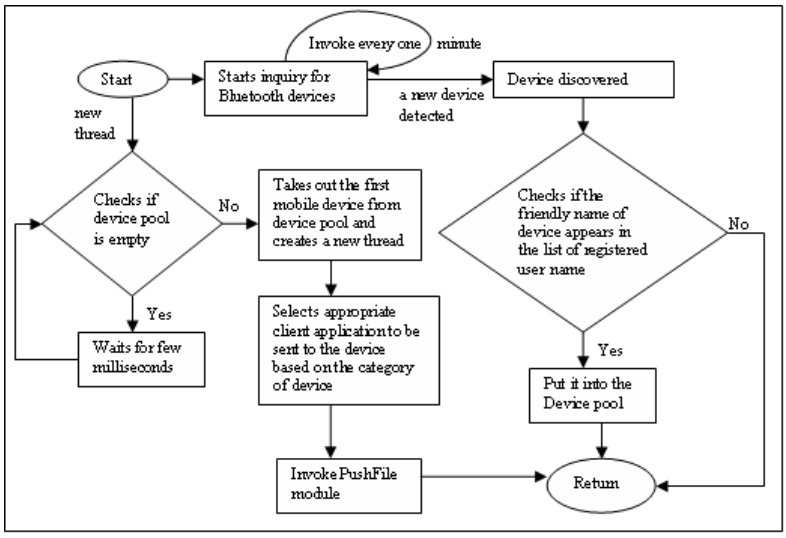

Fig. 3. Implementation details of Discovery Module(DM).

by providing it the user name and password received from the client over the Bluetooth connection. If client provided user name and password matches with those of a registered user, AM would invoke Service Advertising and Invocation Module (SAIM) to allow the mobile client to receive the service advertisement and to consume the LBSs also. The detailed design of MM and SAIM and the mobile client has been given in our previous work [8].

2.3 Enhancements made to the Middleware

Two new modules, incorporated into the Middleware of our previous work, Discovery Module (DM) and PushFile Module (PM), the changes made to AM and the services deployed onto the BS are presented in this sub section. Each BS advertise two LBSs, namely location service and content-mapping service via SAIM to the mobile client over the establishd Bluetooth connection. The location service gives the location of the mobile user and the second one provides a map showing the direction from the current location to the destination user wants to reach upon receiving the name of destination from the user. Both of these services are quite useful to a newcomer in a departmental building.

2.3.1 Discovery Module (DM): It initiates Bluetooth inquiry procedure periodically (every 1 minute) and detects all the Bluetooth enabled devices set in discoverable mode in the neighborhood of the BS. Upon detecting a new device, it checks whether the Bluetooth device name of that device appears into the list of registered user name by invoking AM and sorts out those devices belonging to some registered user into a device pool, that is a data structure like java Vector [19]. Another thread created by this module examines the profile of each device stored into the device pool to determine the appropriate client application to be sent onto that device and invokes the PushFile module (PM). The implementation details and the work flow diagram for this module are depicted by fig. 3.

2.3.2 Authentication Module (AM): AM on the central BS receives the registration request message from the mobile users and inserts a record consisting of the user name, password, user′s phone number, registration time and the profile of the device belonging to the user into its database. The user name field is generated uniquely by the AM based on device model and the preferred user name provided by the user in the following way.

$$\textit{user name} = \textit{device model} + \textit{preferred user name} + \textit{3-digit random number}$$

The AM of central BS periodically (every three minutes) broadcasts the list of records for the users who have registered during the last period to other BSs.

The AM of other BSs also maintain that list of records received from the central BS into its database. It searches that list of records in two cases. When invoked by the DM, the AM searches the database to find a match for the Bluetooth device name obtained from some mobile client via the DM with the user name field in the registered users′ records. In the other case, it searches that list to find a match for the pair <user name, password> obtained from some client via the MM to authenticate that client. When the time of registration for a user becomes one day old, the corresponding record would be removed from the database in all the BSs.

2.3.3 PushFile Module (PM): It is invoked by AM and other modules of Middleware to send any type of file to a Bluetooth enabled mobile device using its OBEX object push service that begins to run whenever Bluetooth of the device is turned on. The implementation detail of this module is given below.

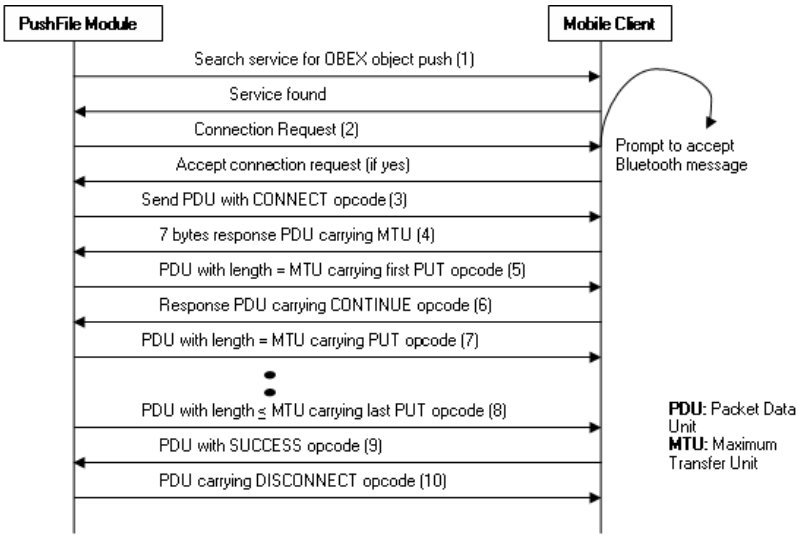

Fig. 4. Sequence diagram of pushing file from BS to mobile device.

Implementation Details of PushFile Module : OBEX Object Push Service can be located into the Service Discovery Database (SDDB) available on every Bluetooth enabled device using Service Discovery Protocol (SDP) [20] that is part of Bluetooth Core Protocol stack. This service, identified by the 16-bit UUID value 0x1105, can be used by PM to push a file to the inbox of the mobile device. Fig. 4 shows how PM pushes the given file into the inbox of the Bluetooth enabled mobile device using OBEX Object Push Service. The structure of packets exchanged between the Middleware and the mobile device in course of file transfer is shown in fig. 5. Finding the structure of the packets required a lot of effort because no such document describing the exact structure of the packets required to exchange between two Bluetooth enabled devices during OBEX Object Push operation is available.

1. PM acts as SDP Client and mobile device acts as SDP Server.

2. Once the service is found, PM sends a connection request to it to establish a logical link between the BS and the mobile device.

3. In general connection request packet and connection response packet are 7 bytes long. Packet 1 and Packet 2 show the structures of these packets in fig. 5. Since Max OBEX Packet Length field is of two bytes, the maximum length of data packet can be 65,535 bytes.

4. SDP Server sends back to the SDP Client the response to CONNECT request. This response packet also contains the maximum packet size that can be received by the SDP Server. This maximum packet size, i.e, MTU (Maximum Transfer Unit) is determined by the minimum value among the Max OBEX Packet Length that sender supports and the Max OBEX Packet Length that the receiver supports.

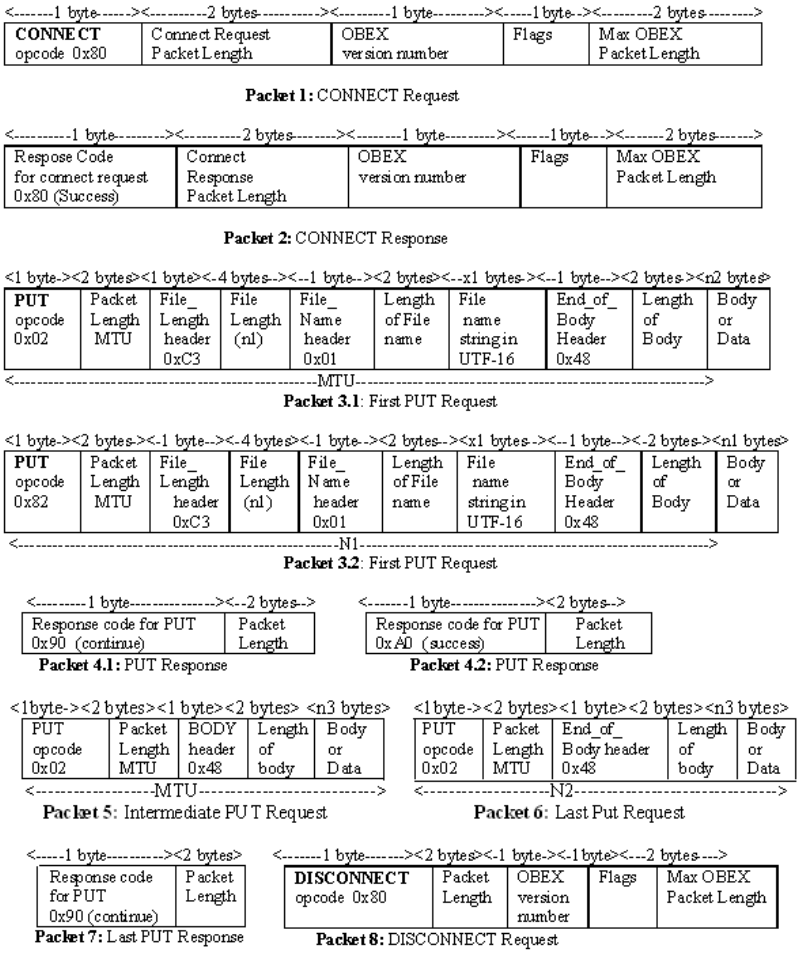

Fig. 5. Packet Structures.

5. If the first byte of Packet 2 contains opcode value for connection request accepted, that is 0xA0, SDP Client sends the PUT Request packet to indicate that it is about to send a file to the SDP server. The structure of the first PUT Request packet is depicted by packet 3.1 and packet 3.2. From those structures, it is evident that the body part of the first PUT request packet contains n2 data bytes, where $n2 = MTU - (14 + x1)$ and x1 is the length of file name string in UTF-16 format. If the file is longer than $n2$ bytes, multiple PUT request packets are required to send the entire file by the SDP client. In that case, all PUT request packets contain PUT opcode value 0x02 except the last PUT request packet that contain PUT opcode value 0x82 as shown by packet 3.1 and packet 6. On the other hand, the entire body of file can be sent in only one PUT request

packet when the $lengh\ of\ file \leq n2$ and PUT opcode value in that case will be 0x82 as shown by packet 3.2.

6. The structure of response packets sent by the SDP Server to the SDP Client corresponding to PUT requests in (5) are shown by packet 4.1 and packet 4.2. When the SDP Server has received part of file by examining the PUT opcode value and it wants to receive the remaining part, packet 4.1 is sent. In case, SDP Server has received the entire file, packet 4.2 is sent.

7. Packet 5 shows how the remaining part of file body is sent except the last PUT request packet. In these packets, length of body $n3 = MTU - 6$.

8. When n3 is greater than the remaining part of file content to be sent, the remaining part of the file is sent by last put request as shown by packet 6. In that case, packet length N2 is equal to $(n3 + 6)$.

9. SDP Server sends success response to SDP Client after receiving the entire file body as shown by packet 7.

10. SDP Client now disconnects from the SDP Server by sending the DISCONNECT packet as shown by packet 8.

3 Performance Evaluation

We did some experiments to evaluate the performance of PM in terms of time it takes to push file to the mobile clients with respect to different file size and the throughput with respect to distance between the BS and the device. Some preliminary experiments have been done to evaluate our proposed Middleware by measuring the time taken by the mobile devices to consume LBSs. To carry out the tests, we have used the following devices.

- **Base Station:** Work Station IBM XSERIES 206, Intel(R) Pentium(R) 4 CPU, 2.80 GHz, 512MB RAM with Bluetooth USB Dongle of Bluetooth Connection range 20 m and running Windows XP Professional with Service Pack 2.
- **Mobile Devices:** Tests were carried out with eight devices (five Nokia N70s, one Nokia 6600 and two HP iPAQs hw6515e).

3.1 Variation of push time with file size

Here the BS tries to push the files with different size, such as $2kb$, $11.5kb$, $62kb$ and $167kb$ to the eight devices as mentioned above simultaneously. Our experimentation shows that a device with more battery power is likely to be detected by the BS earlier than the others. Fig. 6 shows the time required to transfer the files of different sizes into the inbox of those devices. It is obvious that a device that is discovered earlier and prompt to accept message earlier, would be transferred the file before the others. In this experiment, Nokia N70 had more battery power than the other two devices like Nokia 6600 and HP iPaq and thus the file transfer time for N70 become short compared to other two devices.

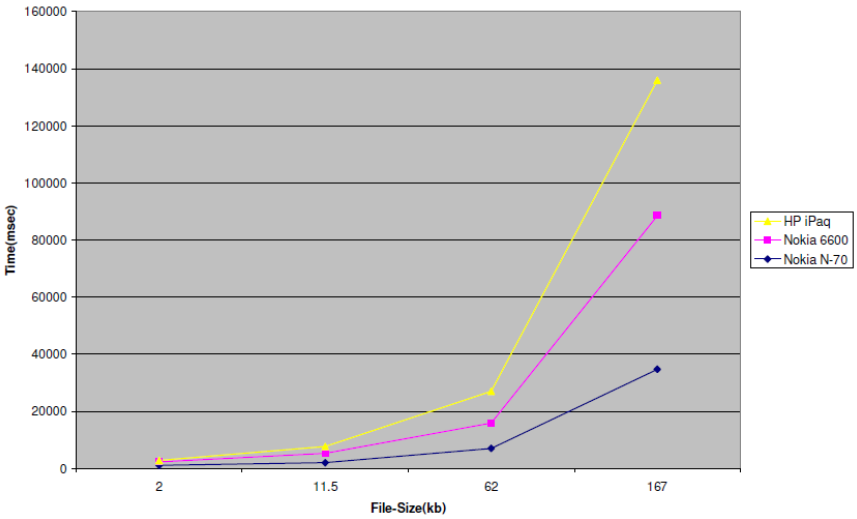

Fig. 6. File Size vs Push Time for different mobile devices.

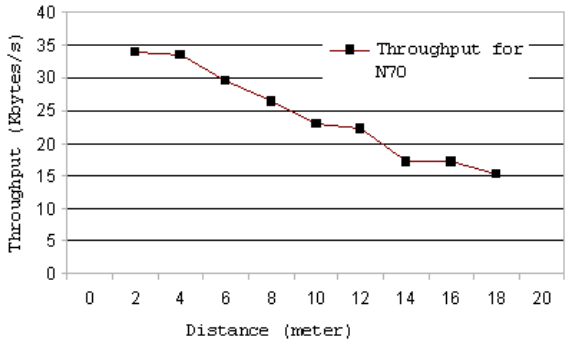

Fig. 7. Throughput achieved by Middleware vs separation distance between BS and mobile device

3.2 Throughput Evaluation

Fig. 7 shows the impact on throughput with varying the distance between the device and the BS. Throughput is estimated as follows.

$$Throughput\,(Kbytes/s) = \frac{Size\,of\;file\,that\,has\,been\,sent\,to\,the\,device\,in\,Kilobytes}{Time\,to\,send\,the\;file\,in\,seconds}.$$

The tests were carried out by placing a N70 at different distances such as 2 meter, 4 meter,$\cdots$, 12 meter, etc. from the BS and measuring the transfer time of a file with size 96 Kbytes to the N70 at the BS. Fig. 7 depicts that the throughput achieved by PM falls with growing distance between the device and the BS.

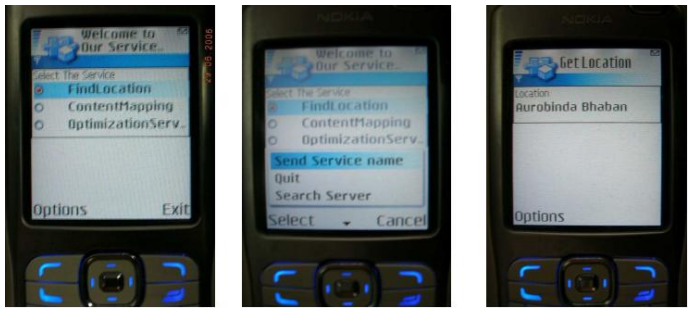

Fig. 8. Screen shot for location service

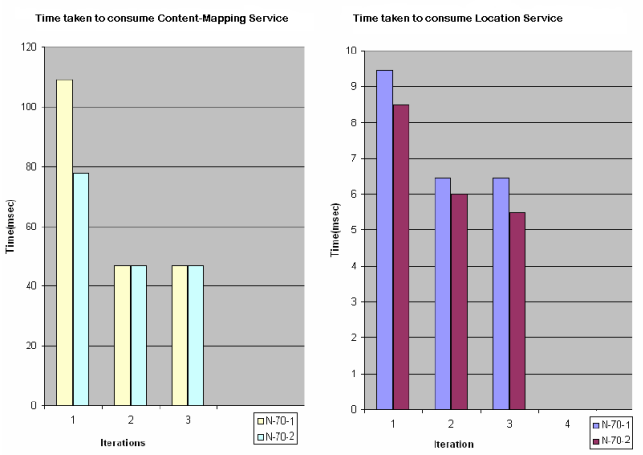

Fig. 9. Service consumption time for content-mapping and location service vs Iteration

Tests were also carried out with that N70 by keeping it out of Bluetooth connectivity range of the BS for 20, 23 and 25 seconds and then brought it back near the BS. In the first case, N70 successfully received the file, but in the remaining two cases, BS showed *Connection Closed. Exception: Failed to write* because the N70 remained out of BS′s Bluetooth range for a long time. According to Bluetooth specification version 2.0, when two Bluetooth devices have established a connection, the Link Manager(LM) in the Logical Layer of the Bluetooth Protocol Stack on the both side master and slave periodically (time period SupervisionTO [21]) send packet to detect link loss. Any time master or slave has not received a valid packet within time SupervisionTO, it breaks down the logical link.

3.3 Test results on LBS consumption time by mobile devices

Tests were carried out using two Nokia N70s to estimate the service consumption time for two LBSs, namely location service and content-mapping service at the

devices. The service consumption time is defined as the difference between the time at which client obtains the service result and the time at which it sends the request for that service. Fig. 8 shows the screenshot for location service.

The execution of LBS for the first time always takes more time than its future execution as shown in fig. 9. Because stubs (that send the SOAP request message to the service) uses some caching mechanism to store the service result and send the cached result when it receives a request for a service that matches with some previously executed service with same input data. Fig. 9 also shows that N70-1 takes larger time than N70-2 to consume the content-mapping service in the first iteration. The BS acquires knowledge about mobile device like its Bluetooth address, the URL of OBEX Object Push Service only after it sends the device the initial client application. N70-1 received the client application from some other BS and directly requested service. The BS needed some time to obtain the URL of OBEX Object Push Service for N70-1 before initiating the file transfer operation. The subsequent iterations take same time for the two devices as BS has now the entire device details about those two devices.

4 Conclusions and Future Work

This work was motivated by the recent trends in the area of providing location based services and information to the mobile user in public places. In the context of Middleware design, the implementation of a module to push location based information and client application onto a low-cost device that has no other wireless communication facility except a Bluetooth radio, has been shown. User has to run the client application on the device to interact with the Middleware to invoke LBS. The experimental results related to sending different size of files to multiple mobile devices, possibly indicate a new way of downloading multimedia data to the mobile devices from a nearby BS or Server using Bluetooth technology.

There are some challenging issues that we hope to implement in near future. Due to mobility, it may happen that the mobile device goes out of the Bluetooth range of the BS while it is receiving the file from BS. Based on the experimentation on the estimation of SupervisionTO, we can think of creating another module that would help the LM of the current BS to transfer the device details including the connection properties of the OBEX object push service of the device to the LM of neighboring BS to continue with the interrupted file transfer service running on the mobile device.

Some rogue BS might try to mimic our BS, sending virus and malware in the guise of LBS advertisement. Thus, some method of authenticating the BS by mobile client is needed to be incorporated into our present work.

Acknowledgments. The authors gratefully acknowledge the facilities and support provided by the Director and all other staff members of the School of Mobile Computing and Communication, Jadavpur University, a Centre of Excellence set up under the ″University with a potential for Excellence″ Scheme of the UGC.

References

1. S. Martin, E. S. Cristobal, R. Gil, M. Castro G. Diaz, and J. Peire, ″A context-aware application based on ubiquitous location,″ Proc. of Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies(UBICOMM′08), Sept. 29 - Oct. 4 2008, pp. 83-88.
2. Y. Xia and H. Y. Bae, ″General platform of location based services in ubiquitous environment,″ Proc. of IEEE International Conference on Multimedia and Ubiquitous Engineering (MUE'07), 2007, pp. 791-795.
3. http://cricket.csail.mit.edu/.
4. Bluetooth Inquiry procedure, Bluetooth specification version 2.0 + EDR, volume 1, pp. 53.
5. P. Bahl and V.N. Padmanabhan, ″Radar: an in-building RF-based user location and tracking system,″ in INFOCOM 2000, The Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies Proceedings, vol. 2, pp. 775-784.
6. Y. Ohta M. Sugano, T. Kawazoe and M. Murata, ″Indoor localization system using RSSI measurement of wireless sensor network based on zigbee standard,″ Proc. of Wireless Sensor Network, 2006.
7. X. Lu G. Jin and M. Park, ″An indoor localization mechanism using active RFID tag,″ Proc. of IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, 5-7 June, 2006.
8. Pampa Sadhukhan, P. K. Das, Rijurekha Sen, Niladrish Chatterjee, and Arijit Das, ″A middleware-based approach to mobile web services,″ Proceeding of Asian Mobile Computing Conference (AMOC-2007): 5th international conference, Kolkata, India, January 3-6, pp. 167-175, 2007.
9. http://newsroom.cisco.com/dlls/partners/news/2006/prprod05-02.html.
10. Cristiano di Flora and Marion Hermersdorf, ″A practical implementation of indoor location-based services using simple Wi-Fi positioning,″ Journal of Location Based Services,vol. 2,Issue 2, June 2008, pp. 87-111.
11. Paul Castro, Patrick Chiu, Ted Kremenek, and Richard Muntz, ″A probabilistic room location service for wireless networked environments,″ Proc. of the 3rd international conference on Ubiquitous Computing(UBICOMM′01), LNCS Vol. 2201,2001, pp. 18-34.
12. Juan-Carlos Cano, Pietro Manzoni, and C.K.Toh, ″Ubiqmuseum: A bluetooth and java based context-aware system for ubiquitous computing,″ Wireless Personal Communications (2006) 38,Springer Science+Business Media B.V, pp. 187-202.
13. S. P. Subramanian, J. Sommer, S. Schmitt, and W. Rosenstiel, ″Sbil: Scalable indoor localization and navigation service,″ Proc. of Third International Conference on Wireless Communications and Sensor Networks (WCSN), 2007.
14. JABWT: Java APIs for Bluetooth Available at http://www.jcp.org/en/jsr/detail?id=82.
15. Piconet: Bluetooth specification version 2.0 + EDR, volume 1, pp. 51.
16. Scatternet: Bluetooth specification version 2.0 + EDR, volume 1, pp. 177.
17. T. D. Hodes and R. H. Katz, ″Composable ad hoc location-based services for heterogeneous mobile clients,″ Wireless Networks 5 (1999) pp. 411-427.
18. Class vector,http://java.sun.com/j2se/1.4.2/docs/api/java/util/vector.html.
19. Obex object push profile: Specification of the bluetooth system, specification volume 2, pp. 331-353.
20. SDP:Service Discovery Protocol,specification of the bluetooth system, volume 2, pp 66-80.
21. SupervisionTO,Bluetooth specification version 2.0 + EDR, volume 1, pp. 261, 374.