Collaborative Middleware for Bluetooth-based ad-hoc Wireless Networks on Symbian OS

FENG GAO, MARTIN HOPE
Informatics Research Institute
the University of Salford
Salford, M5 4WT, UK,

Abstract: - In this paper we explore the possibility of using Bluetooth in the development of a Mobile Ad-Hoc Network (MANET) suitable for transmitting data between Symbian OS based Smartphone's. We also analyse the problems that Bluetooth presents when considering existing MANET routing protocols. Then, we present the design of a collaborative application engine by making allowances for the restrictions associated with Bluetooth and finally we review our current progress and consider future work.

Key-Words: -Bluetooth, Wireless, Collaboration, Symbian OS.

1 Motivation

The result of research activity in the field of mobile ad-hoc networks (MANET) has made short to medium range radio transceivers very popular and inexpensive. Most of the practical work and implementations focus on IEEE 802.11 Wireless LAN as an underlying physical radio network and this has been used for simulations or testing. For use in wearable devices like PDAs or cell phones, 802.11 has the disadvantage of consuming more battery power than other technologies such as Bluetooth [1][4]. Furthermore, it seems that after some initial problems, Bluetooth has become a very common feature in cell phones and PDAs. The research presented in this paper focuses on a collaborative middleware platform running on mobile devices (Smartphones) in order to enhance communication and the exchange of data via Bluetooth.

PDAs and Smartphone's can run applications such as organizers, games, and communications programs (e.g. e-mail, Web browsers etc). Here the Smartphone's goal is to combine mobile telephony and computing technologies in a synergistic way. A simple example is the ability to pull up a person's contact information or even their picture, hit a button and automatically dial that person's phone number. Other examples include viewing a pdf document, dialing an international call via VoIP, or watching live TV and listening to music. However, central to Smartphone technology is the Symbian OS. Symbian OS is a full-featured mobile operating system that resides in most of today's Smartphones [8]. Since Symbian OS is an open system, users can download, install and uninstall applications written by third-party developers (or by the users themselves). No special carrier service or device manufacturer's agreement is required to distribute new Smartphone applications since they can be downloaded by the user from a PC to the Smartphone through a link such as USB or by using a wireless network such as Bluetooth.

The initial goal of the work presented in this paper was to consider if current Smartphone technology can be easily used in combination with Bluetooth, or alternatively what modifications would be required to enable Bluetooth-based Ad-hoc networking via the Symbian OS. Initially, we tried to find a scenario where most of the characteristics and problems of MANETs running over a Bluetooth connection will appear. Within this scenario we considered connecting a large number of Bluetooth-enabled Symbian smartphones to a (multi-hop) MANET. This includes a maximum of ten nodes within the network because this number would be more likely in reality, and is also the maximum number of nodes Bluetooth has the ability to support. After the restrictions and limitations were considered in Bluetooth networks, a design was proposed which when fully developed acts like a Symbian OS application engine, and meets all the predefined requirements for a collaborative middleware.

There are a numerous applications for the middleware platform proposed in the paper. For example: given a large office complex where people move around a lot between their office desk, conference rooms, or central areas like printer-rooms, fixed-line telephones don't provide a practical method of voice communications. Therefore, people tend to call their colleagues on their cell phone in order to find out their current location, or in order to reach them for urgent requests. Of course, the network carrier charges for these calls, however, the middleware proposed could possibly enable them to make the same calls at no cost at all. Another use of this system may be at large exhibits or fares where groups of people with Bluetooth enabled phones may assemble. Again the middleware proposed will enable them to share and relay data such as shared audio / video files or work based documents.

2 Bluetooth overview

In this section we provide a very brief overview of the relevant aspects of the Bluetooth standard. For detailed information please see [1][4].

When established in 1998, the original idea of Bluetooth was to create a cheap wireless replacement for the myriad of data cables that surround today's multimedia devices. Like many other communication technologies, Bluetooth is composed of a hierarchy of components [5], more commonly referred to as a stack and is shown in figure. 1. Bluetooth uses a protocol stack of several layers. The Radio Layer describes the physical radio system. The Baseband Layer is responsible for transmission and reception of data packets, error detection and encryption (if used). The Link Controller uses a state machine to control synchronization, connection setup and shutdown and the Host-Controller-Interface (HCI), separates the Bluetooth hardware from the part of the protocol stack that is usually implemented in software. Thanks to this standardised interface, the Bluetooth hardware and the Bluetooth stacks usually interoperate very well. The Logical Link Control and Adaption Protocol (L2CAP) layer multiplexes different data streams, manages different logical channels and controls fragmentation. Multiple higher layer modules may access the L2CAP layer in parallel. These higher layer modules may consist e.g. of RFCOMM for emulation of serial connections, OBEX for transmission of serialized data objects or SDP for service discovery.

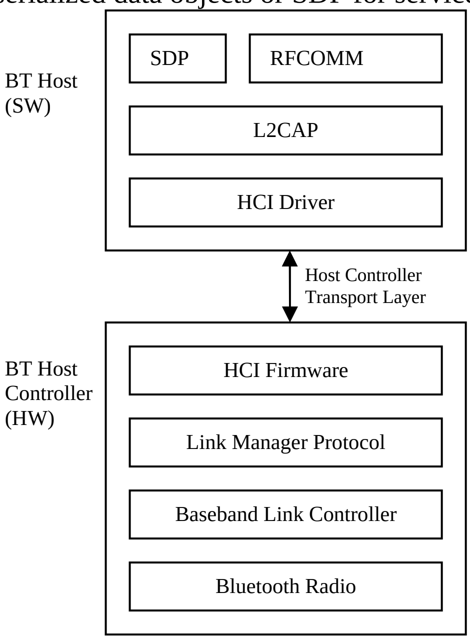

Figure 1: Bluetooth stack architecture

When a device wants to connect another device it first has to carry out an inquiry for its direct neighbors. After receiving the inquiry results it can contact another device using its unique Bluetooth address. When connected, the two devices form a so-called Piconet. The initiator of the connection becomes the Master of this Piconet; the other device becomes a Slave.

The difference between a master and a slave needs to be distinguished first in order to understand this research. A master is a device that establishes the connections to remote device, slaves. A slave can not establish any connections; it will act as a listener to incoming connections from the master device. The master discovers slave devices and their services and is capable of connecting to multiple slaves and holding these connections active simultaneously. In essence, the point-to-multipoint connectivity is a single master device (point) connecting to multiple slave devices (multiple points) and the slaves up to a maximum of seven. Using a Bluetooth connection between just two devices, it does not matter which one is Bluetooth master or slave. However, if we wish to connect more than two devices together in the same session (for example, a multiplayer game with more than two players), it is likely that we will have to consider how the Bluetooth master / slave roles impact upon the connection setup between the networked peers. Because once a master connected seven slaves, this master can not connect any more slaves; however this peer can be as slave connecting to another master peer, as show on Figure 2.The network topology will be complicated, and the requirement of routing protocol is higher as well.

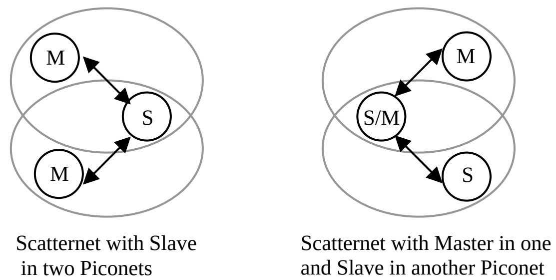

Figure 2: Bluetooth master and slave roles

3 Developing on Symbian OS S60 3rd platform

The Symbian OS is widely used on a number of Smartphone platforms such as the Series 60, the Series 80, and the Series 90, three of the UI (User Interface), platforms from Nokia, and UIQ (the UI platform from UIQ Technology). At the time of writing, the S60 variant resides on the majority of Smartphone platforms currently available and its third edition is based on Symbian OS v9.1[7]. Software applications running on the Symbian OS can be split into a UI (also known as a View) and an Engine (also known as a Model). The UI is the part that presents the data to the user. The engine is concerned with data manipulation and other operations independent of how these are eventually presented to the user. The engine can therefore be re-used by other applications (i.e. it can be built as a shared dll). It is the application engine that takes on the role of collaborative middleware and supplies a set of classes that can re-used by other applications. Like other OSs, Symbian provides the API to developers. Its Bluetooth API provides applications with access to RFCOMM, L2CAP, SDP, OBEX, and to a limited extent, HCI[4]. With a view to the way in which the collaborative middleware establishes communications, Bluetooth sockets were chosen to provide communications between devices, and OBEX was selected to finish a single-shot operation like transferring a file. These two methods of communication are discussed in following section.

4 Bluetooth sockets and the OBEX overview

In the Symbian OS, Bluetooth sockets are used to discover other Bluetooth devices and to read and write data over the Bluetooth network. The Bluetooth Socket API supports communications over both the L2CAP and RFCOMM layers of the Bluetooth protocol suite [5]. The API is based on the sockets client side API that provides a standard API, enabling a client to make a connection to a remote device. Alternatively, it also allows a device to act as a server and have a remote device connect to it. Once connected, the devices may send and receive data before they disconnect.

The API has five key concepts: socket address, remote device inquiry, RFCOMM commands and options, L2CAP commands, and HCI commands. This research concentrates on implement a Bluetooth connection using RFCOMM. Before using the Socket API to establish a connection, a Bluetooth device must locate a suitable device with which to connect to. This means that the Bluetooth device has to finish device and service discovery, before making a connection. The Bluetooth Service Discovery Protocol (SDP) performs this task.

The SDP can be broken down into two main parts: 1. The discovery of devices and services within the local area; 2. The advertisement of services from the local device.

As previously mentioned, Bluetooth sockets are the preferred choice when communicating between two devices. However, if the transmitting task is only a single file, like an image file or MP3, OBEX is more suitable. OBEX is a transfer protocol the operates on top of a number of different transport mediums, including IrDA and Bluetooth RFCOMM. It defines data objects and a communication protocol the two devices can use to exchange those objects. It also provides a method for transferring objects or chunks of data from one device to another. These chunks are typically files or other blocks of binary data. OBEX uses a client-server model and is independent of the transport mechanism and transport API. A Bluetooth enabled device wanting to set up an OBEX communication session with another device is considered to be the client device. The OBEX protocol also defines a folder-listing object, which is used to browse the contents of folders on remote device. The main purpose of the protocol is to support the sending and receiving of objects in a simple and spontaneous manner. For example: pushing business cards or synchronizing calendars on multiple devices is handled with this protocol. In conclusion, The OBEX protocol, in its simplest form, is quite compact and requires a small amount of code to implement, but is also at the same time a reliable transmission systems over Bluetooth networks.

5 Project analysis

Our research objective is to develop a collaborative middleware based on Symbian OS via Bluetooth. The following presents a discussion of some of the initial requirements and restrictions found to date.

When using Bluetooth as a physical layer for a MANET we have to consider a number of restrictions compared to the IEEE 802.11 wireless standards [9].

- Bluetooth is connection oriented. So, in order to send data to another node you have to setup and later tear down a connection.
- Bluetooth has no: ‘all neighbors’ broadcast capability (only point-to-multipoint within the Piconet). So, in order to flood a route request you have to first connect to all neighbors and then send a point-to multipoint packet to them. If there are more neighbors than are allowed in a Piconet [2] things get even more complicated.
- Bluetooth has very long inquiry and relatively long connection setup times.

When we want to implement a collaborative ad hoc network system using a Bluetooth, we have a number of requirements that contrast with above capabilities:

- Users want a short connection setup time similar to a normal phone dialing (a few seconds).
- We need to have relatively few hops. Otherwise, the delay will be too long, and the user will experience a significant pause and echo.
- In order to optimise the overall throughput and minimize interference we want to minimise the number of Piconets and shutdown unneeded connections.
- To avoid interrupts in data transmission we need a very efficient and fast route-maintenance-and-repair mechanism.

No current MANET routing protocols fulfill all of the above requirements. Therefore, we will design a new routing protocol with the focus on data transmission in Bluetooth Scatternets. However, the current protocols that do exist provide a number of interesting ideas which can be used as a basis for this new protocol: e.g. reactive operation and route maintenance / repair. Furthermore, there are other connection oriented networks that provide interesting concepts. For example our data packets don't carry any destination information in the header. Instead the intermediary nodes use the incoming (L2CAP or SCO) channel identifier to decide where to forward the data. This is very similar to the circuit switching approach of ATM. We will therefore try to take these ideas and integrate them.

6 Project design

In order to set up a collaborative Bluetooth network, the first thing to consider is how to establish and maintain a Bluetooth network. We then consider the design of some functional classes as an access point for the application. As previously discussed, there are three stages involved Bluetooth communications using a socket: device discovery, service discovery, and the communication itself [6]. Based on these stages a state machine model was developed as a design for the set-up of the Bluetooth network. See Figure 3. In our design, the application state starts from BTDormant, which is waiting for a user or client to request Bluetooth communications; when a user requests a connection, the application then searches for devices within range, the state is then BTDiscoveringDevices. If the application has not found any devices, the user can then restart the device discovery process; otherwise, the master is searching for the required service on each of the discovered devices. If no required services are found, the user can re-start the device discovery. After initiating a connection with a slave, and the master has established a connection with the server, the state is then BTWaitingToSend. When the master receives an enquiry message from the slave, the state then changes to BTWaitingToReceive. The Bluetooth network can then be set up following these processes.

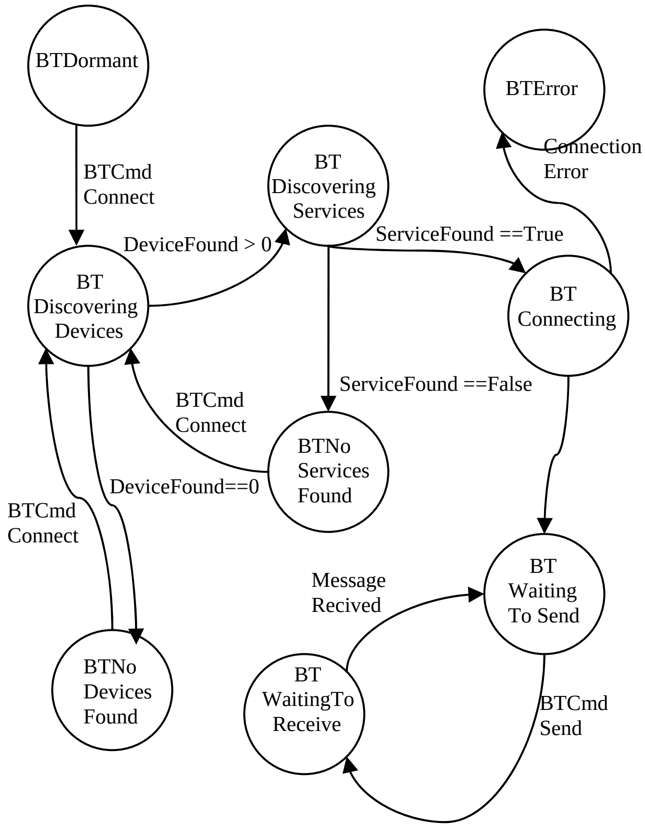

Figure 3: State machine model

According to requirement analysis, a set of classes was designed for the application Engine. However, an explanation of these classes is beyond the scope of this paper, please see [3]

7 Current progress

The research presented in this paper has been on-going for the last year. We have completed an extensive literature survey in this field. Related work such as XMiddle and Syn have been discussed and listed by advantage and disadvantage. Learning to program the Symbian OS has taken slightly longer than anticipated and the application structure and naming conventions is quite different from traditional programming. Unfortunately, we do not have the facilities to debug code on a target device, only on the emulator, which does not support Bluetooth. Bluetooth programming in Symbian OS is far from simple, so we have found that the key is to develop the application slowly and testing as often as possible. However, the initial design work has been completed, and we are able to program two Bluetooth devices which can communicate with each other via a Bluetooth socket and OBEX. This means files and messages can be sent between two Bluetooth devices using our application Engine. A shared file list, like a shared folder in the Microsoft Windows OS, is currently under test. Users will soon know what files are shared by others from this list, and when the device gets authorisation from the files owner, the downloading can begin.

8 Future work

In our on-going work we will try to optimise our design as well as build a real-world prototype in order to fully test our application engine as soon as possible. For a more efficient application of our research there needs to be a number of changes. First of all, the connection setup times need to be reduced drastically. Secondly, a routing protocol for enhancing collaborative communication between Bluetooth nodes need to be designed based on the transmitting protocols within Bluetooth. Finally, how to design and develop a test bed to fully test the application engine needs to be considered.

9 Conclusions

With the design presented in this paper we have tried to address the question whether it is reasonable to build a Bluetooth based MANETs on Smartphone technology and thus to use it to share data. This might depend largely on the intended use and environment, however; this has yet to be addressed. A number of real-world applications for this type of system was outlined including the potential benefits possible. In addition, the implications of using Bluetooth in this way has been discussed, and a state machine model that enables the establishment and maintenance of a Bluetooth network with a view to collaborating with other users has been presented.

References

[1] Core Specification v2.0 + EDR, http://bluetooth.com/Bluetooth/Learn/Technology/Specifications/ Specifications, 2004, 20/09/07.

[2] E.M. Royer, C.-K. Toh: A Review of Current Routing Protocols for Ad-Hoc Mobile Networks. *IEEE Personal Communications 6(2)*, April 1999; p. 46-55.

[3] F.Gao, The Collabrative Engine design http://www.freewebs.com/gaofeng/. 2007, 20/09/07

[4] J. Bray, C.F. Sturman: *Bluetooth - Connect without Cables*. Prentice Hall, New York, 2001,p63-66

[5] Nokia, S60 Platform Bluetooth API Developers Guide v2.0 en.pdf 2006,p18-23

[6] P.Coulton, R.Edwards *S60 Programming: A Tutorial Guide*, Symbian, 2007, p242-244

[7] Nokia, S60 3rd Edition SDK for Symbian OS, 2006

[8] S.Babin, *Developing Software for Symbian OS, Symbian*,John Wiley and Sons, 2006, p57-59

[9] S. Corson, J. Macker: Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations. IETF 1999, RFC 2501.