Creating a mobile P2P file sharing environment over Bluetooth

Gerard McNamara[1], Yanyan Yang[1]
[1]*Department of Electronic and Computer Engineering, University of Portsmouth, United Kingdom*
gerard.mcnamara@port.ac.uk; linda.yang@port.ac.uk

Abstract

Peer to peer (P2P) file sharing on mobile device has attracted great attention in both academia and industry. Current P2P file sharing applications for mobile users is expensive and still remains a substantial problem for the transmission of the large shared files. In this paper, we discuss the current file sharing possibilities on a 3G network environment. A mobile P2P file sharing solution to facilitate people's social life and allow for cheap data transmission is presented. The design, architecture and implementation of JBPeer, a mobile P2P file sharing system based on Bluetooth radio technology, 3G technology and J2ME, is reported on here. We use scenarios to illustrate the suitability and usefulness of JBPeer for every-day file sharing applications in the mobile context.

Keywords: Mobile P2P, File sharing, 3G, Bluetooth.

1. Introduction

Peer-to-peer (P2P) has become one of the most widely discussed terms in information technology. The term *peer-to-peer* refers to the concept that in a network two or more nodes are able to spontaneously collaborate without necessarily needing central coordination [1]. P2P systems date back as far as 1970, when ARPANET used P2P architecture to distribute messages [2]. As the underlying technology evolved and computers become faster capable of storing more data, the evolution of P2P systems and applications began. Desktop P2P systems have become a major part of modern computing, and it is estimated that P2P file sharing makes up about 70% of all internet traffic [1] which is a clear indication of its popularity. Whilst the evolution of computer hardware was occurring, so was the evolution of mobile technology, mobile phones and Personal Digital Assistants (PDAs) were fast becoming powerful devices. In 2008 most mobile devices are more powerful then computers were back in the mid 1990's.

The modern mobile phone now possesses as much on board connectivity equipment as most Computers do, and in some cases, such as GPS (Global Positioning System), the mobile phone exceeds the computer. Devices not only include the basic GSM (Global System for Mobile Communications) system, that will enable voice and data traffic to be sent and received from the phone, but also include Infra-red to enable basic peer to peer wireless communication through Light waves, Bluetooth to enable wireless peer to peer connectivity through radio waves, Wi-Fi (Wireless Fidelity) to enable phone connectivity to wireless access points, and most recently GPS to enable satellite positioning of the device.

In 2006, it was reported that 79% of people in the United Kingdom owned and used a mobile phone [3]. When compared to Japan, China and South Korea, the UK is still a distance behind, however mobile phones are still an everyday device that people use and as mobile phone manufactures create devices with more and more connectivity, application developers are creating software that make the most of this connectivity and provide users with more and more integrated services. Mobile phones have changed the way people communicate, interact in the physical world, and coordinate their social activities [4].

As the mobile networks move into the 3rd Generation, the ability to share information through a P2P service over a GSM network has become possible. This ability is brought about because the 3G mobile network enables true IP-based communication for mobile phones on the telecommunication network which means that a phone will have an IP-address of its own. Mobile P2P file sharing has shown an increasing development in recent years. Examples include the recent adoption of Mobile Social Software (MoSoSo) which is primarily intended as communication portals but provide the ability to share files within this service.

The mobile network operators have also acknowledged this move towards mobile services by introducing price plan options that include data usage. Slowly these price plans have moved further and further away from expensive per kilobyte charges into blanket once a month charges that correspond to a download limit. On a mobile phone back in 2000, 1GB (Gigabyte) would have been a waste, now though with the emergence of mobile services, such as Video on Demand and Music on Demand this download cap has become far more of a restriction.

Current file sharing on a mobile device is very expensive, as large amounts of data are sent and received via the GSM networks. There is no application that exists to date that provides a P2P file sharing

978-1-4244-2020-9/08/$25.00 ©2008 IEEE

system that doesn't transfer files over the costly GSM networks.

The motivation for our work has been to provide cheap data transmission for mobile phone users. We have developed a Java Bluetooth Peer to Peer (JBPeer) system, which uses the Bluetooth transport medium to handle the transmission of the shared files and the GSM network to handle the transmission of control data. The JBPeer system provides cheap P2P file sharing service that no other mobile P2P software does.

The rest of this paper is organized as follows. In section 2, we describe the problem we are trying to solve and briefly review related work. Section 3 introduces our proposed mobile P2P file sharing system in details. In section 4, we describe JBPeer system architecture and introduce the main considerations of the design. In section 5 we report on the implementation status of our system. We conclude in section 6.

2. Problem Description and Related Work

The problem with all mobile P2P applications, whether part of a MoSoSo application or an independent one, is that they use the GSM networks to transmit and receive data.

Of the four major network operators in the United Kingdom, only one has begun to realise the growing importance of access to data on the move. Whilst operators such as Vodafone, O2 and Orange offer its customers mobile data packages, they are limited and expensive. T-Mobile however have begun to realise the market for the mobile devices wanting to access large amounts of data over the network and offers its customers a flat rate data service for the type of data they wish to transmit. The basic web browsing and email package is priced at £7.50 per month and allows unlimited access [5]. The medium package offers customers all the basic features and the ability to use the instant messaging services and VoIP (Voice over Internet protocol) services and costs £17.50 per month [5]. Its maximum package includes all the features of the previous two plus the ability to use your mobile device as a modem and connect your laptop to the internet through your mobile phone, allowing the full range of services provided by an ISP (Internet Service Provider), this service costs £22.50 per month [5].

It is clear the cost of sending and receiving data over the GSM networks is still expensive, and whilst operators have rolled out blanket charging, these services explicitly state that file sharing is not allowed, or partition the services into various bands, increasing in price. To create a P2P file sharing application for the mobile phone that uses the GSM data network to transmit the files is not an option if the application is to appeal to a wider user community.

In October 2006, a mobile torrent application called *SymTorrent* was released on the Nokia platform by the Department of Automation and Applied Informatics at Budapest University of Technology and Economics [7]. This application was technically advanced but useless for users who did not have the finances to be able to pay for the heavy data charges.

In April 2006, a mobile P2P application was released called *PeerBox Mobile* for the Nokia platform by Nareos [7]. Its concept was simply to allow users to upload files to a central server from their mobile device and provide them with the ability to search for files that have been uploaded and download them. The application is reliant on an Internet connection which is costly and slow if the user is using an older device not capable of using the 3G speeds, or simply costly if the user can use 3G but requires a data plan with monthly usage allowances and penalties for going over it.

It was suggested by Helsinki University of Technology [8] that the current Multimedia Message System be used as one way of realizing P2P functionality. MMS messaging content is distributed in a same way as in SMS messaging. Their solution was simple: a central server would allow users that are interested in sharing files to register and store their phone number, phone type and a list of files available for sharing on their mobile phone. The central server could store this information in a database, which could be used for searching by other registered file-sharers. If a search in the sharing database would get a hit, the phone number to the sharer would be given and the file could be requested by sending a MMS-request for the file to the sharing phone. If the sharer is interested in sharing the file, he would send it to the requester.

This solution removes the dependency on the GSM data network to send and receive files. Whilst a data connection is still required, the files themselves are transmitted over MMS. This solution however creates two problems. Once the user has found the file they want by querying a remote server, they must then send the user who has the file a MMS request using the number that the database has returned to them. This incurs a charge by the network operator for the cost of sending a SMS message. The second problem is that the user who is sharing the file must then send the file being requested to the requester in a MMS message. This will incur a charge to the file sharer. This means that this system will only be successful if the sharing party is willing to pay for the transmission of the file that the requesting party has made. It is unlikely this will be the case.

The problem is how to create a P2P file sharing system on a mobile phone that transmits and receives very small amounts of data over the GSM data networks and uses other technologies to handle the transmission of the shared files. Even if the data files were to be transmitted and received over the GSM data networks, the speed of upload and download will be very slow as all the operators packages (with the exception of T-Mobiles medium and maximum packages) only offer GPRS (General Packet Radio Service) and EDGE (Enhanced Data GSM Environment) speeds.

Adopting the concept of the MoSoSo applications and their ability to search local users using the devices Bluetooth hardware [9], and adopting the concept of some of the centralised desktop peer to peer file sharing systems, such as Napster [10], an application could be created that would allow users to store to a server the list of files that they wish to share, and then use Bluetooth to actually transmit those files to requesting nodes. Whilst this system would not be entirely free due to the data transmitted to a server over the GSM network, it would be substantially more affordable and the data that would be transmitted would be in the hundreds of kilobytes rather than thousands of kilobytes or even megabytes. This would open the application up to a much wider user base.

The JBPeer system addresses this problem by generating a solution that uses the Bluetooth technology that is built into most modern mobile devices. Using Bluetooth to handle the actual sending and receiving of data files moves the application away from the heavy data charges. Bluetooth is a free to use short range wireless service that offers a minimum speed of 1Mbit/s (Megabits per second).

3. System Overview

The aim of the JBPeer system is to create an application that delivers a P2P file sharing service on a mobile device, using the Java programming language and the J2ME (Java 2 Platform, Micro Edition) mobile platform. The application uses the Bluetooth transmission medium to transmit and receive data files between devices and uses the GSM data network to transmit and receive small file lists.

The JBPeer client application, which is the software that is installed and run on the mobile device, is a MIDlet. A MIDlet is a mobile application written for the J2ME platform. J2ME provides a modular, scalable architecture to support a flexible deployment of Java technology to devices with diverse features and functions. The J2ME developer network currently lists 523 devices that support J2ME [11], spanning a wide range of manufacturers. Device manufacturers realised the potential for the J2ME and now millions of devices are shipped with J2ME software [12][13]. The J2ME adopts the same principles as the Java 2 Standard Edition, that being to provide a level of abstraction through a virtual machine. This means that the same code can be executed on multiple devices with little or no modification required.

The JBPeer system implements two optional API's (Application Program Interface) that are not part of the core J2ME platform. Small, resource limited devices implement the Connected, Limited Device Configuration (CLDC). The CLDC is a specialist set of API's designed to extend the functionality provided by the basic J2ME platform. However some APIs are left as optional by Sun Microsystems [14]. The two optional APIs that a device must implement if it is to run the JBPeer MIDlet is JSR-82 and JSR-75, which are for Bluetooth and File System access respectively.

From the users' perspective, they will download the application from the JBPeer website and install it on their mobile device. They are then required to register with the JBPeer server using a computer or even the device if it is capable. Once the user has a username and password set they are then able to log into client application running on their mobile device. The whole process is made as simple as possible so that non technical users can use the system as well.

The following real world scenarios are an indication of how the system could be used:

- A group of students are sitting together in the university union bar after just spending two days away on holiday together. Each student has a camera phone and took pictures over the two days of each other on holiday. There are some photos on each student's mobile phone that are of interest to the other students. Each student could ask each of his friends in turn "What photos would you like me to send you?" Using the JBPeer system, each student simply shares the location that they stored their photos and then the other students can select the files that they want to download.

- Whilst not its primary purpose, two colleagues working in close proximity to each other may chat via SMS (Short Message Service), which costs money every SMS that is sent. The colleagues could both use the JBPeer system to send each other Bluetooth messages instead which would be free.

- An up and coming music artist might use the JBPeer system as an alternative method to distribute their demos. As the transfer of the file is free because it is over Bluetooth more people will have access to the music and most MP3 players are now built into mobile phones. As more people download the music the more chances of finding a device hosting the music file and the wider the distribution.

- A secretary may use her company mobile phone to host the agenda of the next meeting. Whilst in the meeting, colleagues will be able to download the minutes and read them on their mobile devices on the journey home.

The JBPeer system provides a service that no other mobile P2P software does, and it does this by addressing the expense issue associated with all similar pieces of software.

4. JBPeer System Architecture and Design

Figure 4.1 shows the architecture that is used for the JBPeer system. The thin lines between each node represent the Bluetooth connectivity that is used to send/receive the requested files. These lines also represent free communication. The thick dashed lines between each node and the base stations represent the GSM data connectivity that is used to upload and download file lists. These lines also represent communication that costs the user at a rate determined by their network provider.

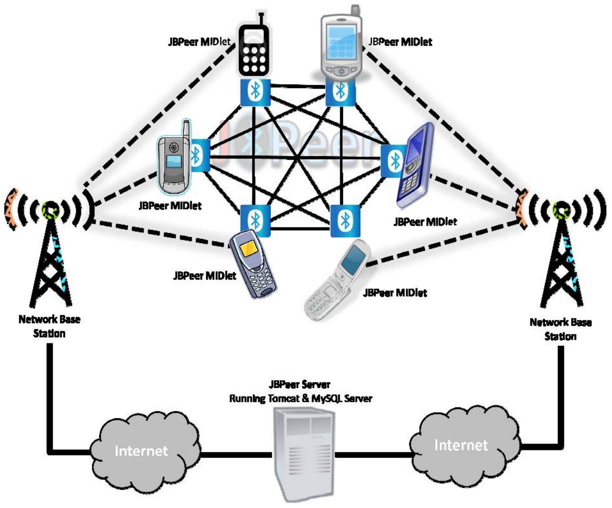


Figure 4.1 – JBPeer System Architecture

The architecture of the system follows that of most desktop based peer to peer systems in that each node is equal and performs both client and server functionality. However the existence of the server creates a slightly different architecture from the standard P2P architecture in which no central server exists [15].

The purpose of a central server in this system is to allow the users to let the server know what resources they want to share and allow other users to search the server for device file lists. Without a central server, each device is responsible for searching the locally available Bluetooth devices and then attempting to query each device for its file list which would create a huge amount of communication overhead. This would cripple the application, making it too slow to use and perhaps even run. The use of a central server will remove the need to implement a P2P protocol on a resource limited platform.

On the central server a MySQL database is used to store information about each user and the files that each device is sharing. If a node wants to join the P2P network the user will first download the JBPeer MIDlet from the website. Once they have installed the application they are required to register an account with the server. During the account creation process the user is asked to enter the Bluetooth MAC address of their device, which is shown on the JBPeer MIDlet login screen. This MAC address is used along with the username as a hash key to hash out the users password. This information is then stored to the database.

In adopting this technique, the user can only authenticate onto the JBPeer system if they do so from the device they registered with. This makes it almost impossible for malicious users to gain access to another person’s account.

Once the user has created themselves an account they will then be able to log into the system through the JBPeer MIDlet. This authentication will require the transmission of a small amount of data over the GSM network. If authentication is successful then the MIDlet immediately ask the user to set a directory, in their mobile devices file structure, that they want to share. If the user has previously logged into the system before then the location they set previous is remembered.

With the shared directory set, the MIDlet then starts the Bluetooth server and creates a list of all the filenames that are located in the shared folder and sends that list to the server. This communication requires the transmission of data over the GSM network. At the server end, each filename that is received is added to a file table and the Bluetooth MAC address of the device that sent it is also added. The purpose of adding the MAC address to each file is so that when a query is performed, only the filenames of the device that is discovered are returned.

The user is then able to perform a Search. In doing so the MIDlet attempts to locate any device within its range that is currently running the JBPeer MIDlet as well. A list of discovered devices is then presented to the user. The list itself is simply the Bluetooth MAC addresses of the devices found. The reason for making the list display MAC address rather than the ‘Friendly name’ of the device is to ensure anonymity which is provided by most desktop P2P systems. The user can then select a device whose file list they want to see. This action creates a query to the Central server which requires the transmission of data over the GSM network.

Once the user has the file list of the device, they are then able to select a particular file that they want to download. This sends a request to the device for the file. As the process is automated the device will automatically respond with the file that the user has requested. It does this by directly connecting to the requesting device over Bluetooth and sending across the file which is stored in the requesting devices shared folder.

The content that can be shared can be as restricted or open as desired. Whilst the device sends all the file names found in the shared directory because it requires the least amount of time and processor power, the server is capable of filtering. Implementing a filter on the server will stop files with key words or of a particular format being added to the database and so remaining unavailable for other users to download.

The J2ME stack that is being implemented for the JBPeer system supports HTTPS, however HTTPS only provides encryption for channels. The JBPeer system works by sending messages over an insecure transport, HTTP, to a particular socket. In this case the J2ME and CLDC APIs are insufficient and a different method of cryptography is required to secure the password that is sent to the server. An open-source API exists called the Bouncy Castle Cryptography package, created by BouncyCastle.org [16]. This organisation creates a cryptography toolkit specifically aimed towards the J2ME platform. This API is used in the JBPeer to create a hash of the user's password, using their username and Bluetooth MAC address as keys.

The Bluetooth communication between the peers is far more secure due to the underlying technology as implemented in the Bluetooth stack. The underlying frequency hopping of the Bluetooth radio is implemented in the JBPeer MIDlet, which provide enough protection from eavesdropping and interception.

5. System Implementation

At the time of writing the JBPeer system has entered its development phase. The JBPeer server hosts a website that offers the user an interface into the backend of the system. When the user logs into the server they are able to manually control their file list, edit personal details such as email address and phone number and the ability for the user to delete their own account should they change their mobile device. The website also provides an administrative system that allows privileged users to reset, deactivate and delete user accounts. The Client application is currently being run in the Java Wireless Toolkit simulator. Our preliminary tests have confirmed that the application will run so far on a Nokia N95, Sony Ericsson W800i and HTC Kaiser II device. The main GUI aspects have been completed and the authentication, file list synchronization, and message sending capabilities have been implemented, tested and are working successfully.

The information flow between the JBPeer MIDlet and the JBPeer server happens as follows:

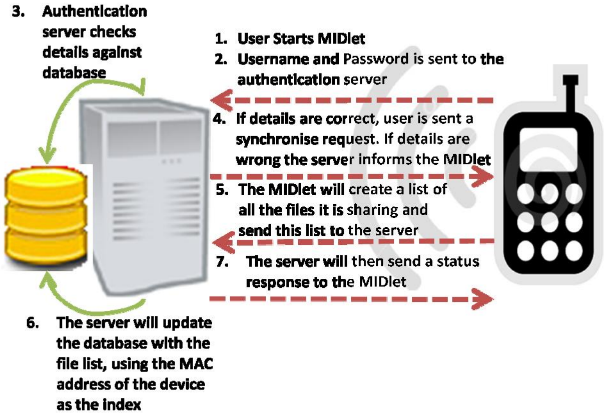

3. Authentication server checks details against database
1. User Starts MIDlet
2. Username and Password is sent to the authentication server
4. If details are correct, user is sent a synchronise request. If details are wrong the server informs the MIDlet
5. The MIDlet will create a list of all the files it is sharing and send this list to the server
7. The server will then send a status response to the MIDlet
6. The server will update the database with the file list, using the MAC address of the device as the index

Figure 5.1 – Information Flow between MIDlet and Server

The peer interaction follows the steps below:

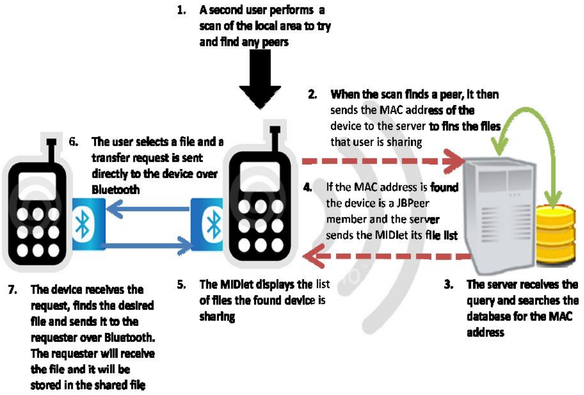
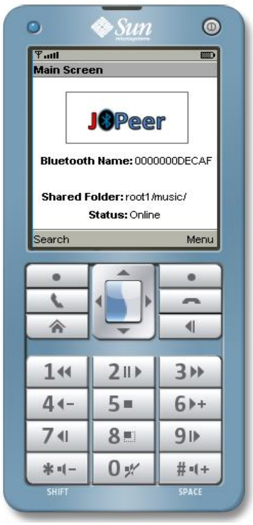

1. A second user performs a scan of the local area to try and find any peers
2. When the scan finds a peer, it then sends the MAC address of the device to the server to fins the files that user is sharing
6. The user selects a file and a transfer request is sent directly to the device over Bluetooth
4. If the MAC address is found the device is a JBPeer member and the server sends the MIDlet its file list
7. The device receives the request, finds the desired file and sends it to the requester over Bluetooth. The requester will receive the file and it will be stored in the shared file
5. The MIDlet displays the list of files the found device is sharing
3. The server receives the query and searches the database for the MAC address

Figure 5.2 – Peer Interaction

Figure 5.3 shows the Main Screen of the JBPeer MIDlet. This screen shows the currently shared directory and the status of the user. The status is used to either allow or disallow the reception of Bluetooth messages. From this screen the users can access all the MIDlet functions. During search process the JBPeer MIDlet will initiate a Bluetooth device query and attempt to find any device that is running the JBPeer system. The MIDlet will only return devices that are running JBPeer; it doesn't return every Bluetooth device that is found to avoid user confusion.



Main Screen

Bluetooth Name: 0000000DECAF

Shared Folder: root1/music/

Status: Online

Figure 5.3 – JBPeer MIDlet Main Screen

We are currently developing new applications to further test the system and we will continue to improve the system based on these tests.

6. Conclusions

The JBPeer system is unique in that all other mobile applications that offer file sharing services do so with the involvement of a file server. This involves heavy download and uploads to a file server and because this is being carried out over the GSM networks, the users are charged. By transferring all the files over Bluetooth, this system removes that huge cost, but introduces the limitation of distance. This can be seen as a limitation or in the case of MoSoSo applications; this could be exploited to help bring strangers together through their common interest in files.

With a unique solution to the current mobile file sharing applications, the platform independence of the J2ME and its large adoption by phone manufacturers and finally an identity, this project has the potential to become a marketable solution and certainly an application that users should find both fun and cheap to use.

References

[1] D.Schoder, K.Fischbach and C.Schmitt, "Core Concepts in Peer to Peer Networking," in *The Evolution of Disruptive Technology*. PA:2005, pp. 1-9.

[2] R.Clarke, "Peer-to-Peer (P2P) - An Overview," Novemeber, 2004. [Online]. Available: http://www.anu.edu.au/people/Roger.Clarke/EC/P2POview.html [Accessed April. 10, 2008].

[3] Mobile Herald, "2006 Mobile phone statistics and projections," *Mobile Herald*, December, 2006. [Online]. Available: http://www.mobileherald.com/2006-mobile-phone-statistics-and-projections/. [Accessed March 12, 2008].

[4] H.Rheingold , "Smart Mobs - The Next Social Revolution." Perseus Book Groups, ISBN: 0738206083, 2002.

[5] T-Mobile (UK) Ltd, "Pay monthly plans - web'n'walk. (2008)," *T-mobile.co.uk*, 2008. [Online]. Available: http://www.t-mobile.co.uk/shop/mobile-phones/price-plans/pay-monthly/webnwalk/plans/. [Accessed April. 10, 2008].

[6] Department of Automation and Applied Informatics, "SymTorrent," October, 2006. [Online]. Available: http://symtorrent.aut.bme.hu. [Accesed April. 10, 2008].

[7] T.O. Whenham, "PeerBox brings P2P to your phone," *Mobile Magazine*, 2006. [Online]. Available: http://www.mobilemag.com/content/100/103/C7390/. [Accessed April. 10, 2008].

[8] J.Bistrom and V.Partanen, "Mobile P2P – Creating a Mobile File Sharing Environment," presented at T-111.590 Research Seminar on Digital Media, 2004.

[9] N.Eagle and A.Pentland, "Social serendipity: mobilizing social software," *IEEE Pervasive Computing, Special Issue: The Smart Phone*, pp. 28-34, April-June 2005.

[10] Jeff Tyson, "How the Old Napster Worked," *Howstuffworks.com*, October, 2000. [Online]. Available: http://computer.howstuffworks.com/napster.htm. [Accessed April. 10, 2008].

[11] Sun Microsystems, Inc, "The Java ME Device Table," *Sun Developer Network*, 2008. [Online]. Available: http://developers.sun.com/mobility/device/pub/device/list.do?sort=manufacturer&page=1. [Accessed March. 15, 2008].

[12] 3G.co.uk, "Accelerated Adoption of Next Generation 3G Wireless Services," *3G.co.uk*, March, 2000. [Online]. Available: http://www.3g.co.uk/PR/March2002/3020.htm. [Accessed March. 17, 2008].

[13] B.Day, "The J2ME™ Platform at Five:Where We've Been, and Where We'll Be at Ten," *BillDay.com*, 2004. [Online]. Available: http://billday.com/Work/J2ME/BillDay_j2meAtFive.pdf. [Accessed March. 13, 2008].

[14] Sun Microsystems, Inc, "J2ME Optional Packages," *Sun Developer Network*, 2008. [Online]. Available: http://developers.sun.com/mobility/apis/. [Accessed March. 15, 2008].

[15] D.S Milojicic *et al., "Peer-to-Peer Computing,"* HP Laboratories, Palo Alto. HPL-2002-57 (R.1), 2003.

[16] Travis Winters, "The Legion of the Bouncy Castle," *BouncyCastle.org*, 2008. [Online]. Available: http://www.bouncycastle.org/java.html. [Accessed April. 10, 2008].