

Experimenting an Indoor Bluetooth-based Positioning Service

Giuseppe Anastasi,

Computer Engineering Dept., University of Pisa, Italy

giuseppe.anastasi@iet.unipi.it

Renata Bandelloni,

Computer Science Dept., University of Trento, Italy

r.bandelloni@iei.pi.cnr.it

Marco Conti, Franca Delmastro, Enrico Gregori, and Giovanni Mainetto

CNR-IIT, Pisa Research Area, Italy

{firstname.lastname}@iit.cnr.it

Abstract

The Bluetooth wireless technology is an emerging technology originally designed as a short-range connectivity solution for personal, portable, and handheld electronic devices. This paper presents the functionality and the architecture of an indoor positioning service developed on the top of this technology. Most of the design choices for the service have been strongly influenced by the features of Bluetooth. The effectiveness of the indoor positioning service is critically analyzed. Experimental and simulation results used for defining the policy of mobile device discovery are shown.

1. Introduction

In current distributed systems, the notion of mobility is emerging in several forms and used in many applications. One of these forms is the mobility of users that arises naturally in wireless mobile computing [1]. Here, when a mobile user moves, his point of attachment to the fixed wired network changes. Current applications of wireless mobile computing cover diverse application areas, on different spatial scales, ranging from the very small dimension of sensor networks up to the world-wide size of satellite-based networks [2].

In the scale represented from corporate and academic office spaces, buildings, and campuses, the increasing presence of Bluetooth wireless technology¹ [3] [4] is witnessed from the wide availability of this technology on a large number of Personal Digital Assistants and Portable PCs [5]. In this context, the mobile users utilize the Bluetooth technology of their portable devices in office applications that are related to group work and to individual access to the corporate services. Accesses to the intranet infrastructure and services of a corporate are granted by disseminating the building of several Bluetooth access points.

In this paper, we present an experiment that we have performed in using Bluetooth and an Ethernet LAN as the enabling technologies of an indoor positioning service, named BIPS, covering a building. BIPS offers a service that allows a mobile user to visualize on his/her portable device the shortest path she/he has to follow in order to reach another mobile user inside the same building. The typical building could be an academic department and the mobile users can be students, visitors, professors, staff members.

The core of BIPS is devoted to tracking the mobile users that walk (or stand) inside a building at a low speed (eventually zero). To this aim, BIPS manages location information about moving users. A single room of the building is the granule of location information that BIPS considers. We define a single room as a space that can fit into a circle of 10 meters radius since this is the dimension of the greatest coverage area of a Bluetooth cell [3]. BIPS has a station architecture that consists of a set of Bluetooth cells, one for every meaningful room of the building, interconnected via an Ethernet LAN with a central server machine. On the central computer, the location database of BIPS operates.

In this experiment, we have chosen Bluetooth as wireless network infrastructure since this technology is emerging as a "de facto" standard for connecting several personal devices (with diverse computing capabilities) to the corporate or academic LANs. In the near future, rooms of offices will provide Bluetooth access points connected to the wired LAN. Since currently every room is usually already equipped with a LAN port and a desktop computer, Bluetooth connectivity is a cost-effective solution for several personal devices, and it can grow up smoothly (i.e., the investment for providing a building of Bluetooth connectivity can be planned in several years). Another main reason for constructing our system on Bluetooth technology is that the precision of a person position represented from the coverage area of a Bluetooth cell is adequate for an indoor positioning service.

¹ Bluetooth is a trademark owned by the Bluetooth SIG, Inc., USA.

The usage of other WLAN technologies such as those based on IEEE 802.11b [5] has been discarded since this solution is more complex to realize and it has a less general applicability. It is more complex because the tracking of a mobile user position requires the setup of a power triangulation procedure performed from at least three base stations that should be placed in the transmission range of the mobile users. It is less generally available since the devices involved in this kind of communications need sufficient computing power and battery capabilities.

The paper is organized as follows. In the next section, we summarize the network architecture and the main functionality of BIPS. A short description of the problems encountered and of their solution in BIPS is provided, too. Sect. 3 introduces the way in which the part of Bluetooth technology of our interest works and how it can be programmed. Sect. 4 describes the experiments and the simulations that we have undertaken and their resulting outcomes. These experiments have driven the implementation process of BIPS with particular reference to the policy of mobile device discovery from a fixed station. Sect. 5 concludes commenting the experiments and the simulation results.

2. An overview of BIPS

BIPS is an indoor positioning service designed for tracking mobile users that can be in motion inside a corporate building. Every mobile BIPS user is represented from an handheld device equipped with a Bluetooth interface for interacting with the static part of the system. The static part of BIPS is a locally distributed system that consists of a centralized server machine and a set of workstations interconnected via an Ethernet LAN.

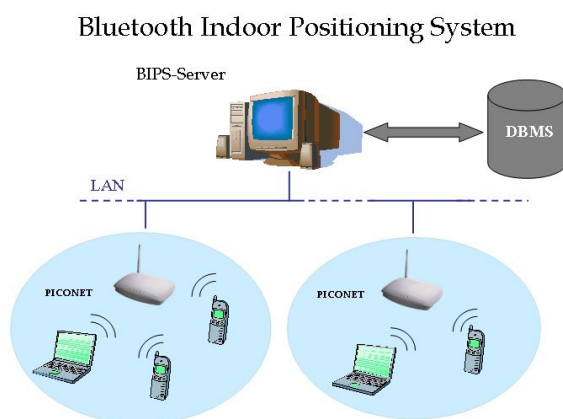


Figure 1. BIPS network architecture

Every workstation is provided of a Bluetooth interface that allows the static part of BIPS to communicate with handheld devices. The main task of every BIPS

workstation is the discovering of those mobile users that enter in the coverage area of the workstation's Bluetooth interface. Once an handheld device has been discovered, its position is communicated to the central server machine where the position is stored in a database for the sake of successive lookups. The network architecture of BIPS is illustrated in **Figure 1** (according to Bluetooth terminology, a cell is called a *piconet*).

An off-line procedure has been implemented for registering new users of BIPS. The procedure associates the name of a user with a user identifier (userid). In this phase, a password and a set of access rights are defined for enforcing security and privacy issues. BIPS allows to define the access rights of individual mobile users and group of mobile users to information related to a userid. The access rights of BIPS have a hierarchical structure that ensures a greater power of interaction to those mobile users in higher layers of the hierarchy.

When a registered user with a mobile device enters for the first time into a piconet of the building, the first action he/she should perform is logging in the BIPS system. The most important effect of this operation is the definition of a one-to-one correspondence between a userid and the Bluetooth device address (BD_ADDR) of the device. From this moment until the mobile user log out of the system, BIPS will constantly track all the movements of the mobile device inside the building.

From the handheld mobile device, the end-user can interact in several ways with BIPS. In this paper, we are interested in analyzing the action of asking for the shortest path to reach the position of another target mobile user with a given name. Answering this question involves two steps in BIPS: querying the centralized location database and generating a path between two piconets. The problem of presenting the result of the query can be solved by showing a map of the building where the current position of the querying mobile user, the position of the person to be reached, and the path are put in evidence.

Before generating the query, BIPS verifies that the target mobile user is logged in and that the querying user has the right to formulate this question. The formulated query is a very simple spatio-temporal query that takes the following form: "Select the target actual piconet of the mobile device BD_ADDR1 where BD_ADDR1 is associated with a userid1 and where userid1 is associated with the given user name". Since the position of the mobile device changes in space and time, the strategy of maintaining accurate position information is the most important technical issue of BIPS. As usual for this kind of systems, BIPS should provide a sufficient level of accuracy for position information, that is a good trade-off between the computational load and the precision in representing position information. BIPS does not impose very stringent requirements on position information maintenance since the mobile users move at a maximum

speed of 2 meters per second through circles of 10 meters radius. In BIPS, every workstation has the task of computing the presence of those mobile devices positioned inside the piconet. These presence are relieved at fixed intervals of time. In order to reduce the computational and communication burden of the system, a workstation updates the central location database only when it relieves a new presence or a new absence in its piconet.

Once the current piconet of the target mobile user has been identified, BIPS needs to determine the shortest path. To this aim, BIPS defines a weighted undirected connected graph that reflects the topology of workstations inside the building. There is a node in the graph for every BIPS workstation. An edge between two adjacent nodes is defined when there is a *direct* physical path in the building that connects the rooms where the two corresponding workstations are placed. A physical path between two rooms is *direct* when the physical path does not intersect the circle representing the coverage area of another Bluetooth workstation. Every edge is labelled with a weight, a positive integer proportional the average time needed to cover the distance between the two workstations. If there could be more than one weight for the same edge, than the smallest one is chosen. Thus the problem that BIPS has to solve is the classical shortest-path problem, i.e. to find the path in a weighted undirected connected graph connecting two given vertices with the property that the sum of the weights of all edges is minimized over all such paths. BIPS implements the Dijkstra algorithm for solving this problem [7]. We underline that the static nature of BIPS wired network allows to compute off-line all the shortest paths that connect all the possible pairs of two nodes. Hence the computation of the shortest path has no impact on BIPS online activities: given two nodes, the answer is immediate since it has been previously computed.

3. Programming the Bluetooth technology for BIPS

The basic concept in Bluetooth communications is the *piconet*, a collection of Bluetooth devices that can communicate with each other by sharing a common channel. A piconet is a star-shaped wireless network in which the device in the center plays the role of the *master* and every other device plays the role of the *slave*. Slaves and master of a piconet are synchronized according to a channel-hopping sequence that is a function of master's BD_ADDR and clock. Bluetooth is a Frequency Hopping System where the clock cycle lasts for 312.5 μ s. Thus the clock rate is 3.2KHz.

In general, a piconet is formed in an ad hoc manner without any infrastructure assistance, and it lasts for as

long as the creator of it needs and is available to communicate with other devices. In BIPS, the role of master is always played from a fixed workstation which is also connected to the wired network, and the role of slave is always played from a portable device.

The process of establishing a communication channel is performed by executing two sequential phases. The first initial phase is named *inquiry* and it corresponds to device discovery from the master. The second final phase is named *paging* and it corresponds to initial connection setup. In both phases the master and the slaves perform different actions that we will briefly summarize in the following subsections. At the end of these phases, the devices are synchronized on the clock of the inquirer/pager that becomes the master of the piconet. Then the connection is established and the devices can begin to exchange data.

3.1. Inquiry phase

A master begins device discovery by entering in the inquiry state. In this state, the master broadcasts messages in the following way. The inquiry hopping sequence is split into two 16-hop parts, named train A and B. A single slot lasts for 625 μ s. On every even slot, the master sends two ID packets switching between two frequencies of the same train every 312.5 μ s. On odd numbered slots, the master listens for slave answers. Since a single train will last for 16*312.5 μ s and the slots for sending and listening are interleaved, two 10ms trains are defined. Each train must be repeated for at least $N_{inquiry} = 256$ times before a new train is used. In order to collect all responses in an error-free environment, at least three train switches must take place. As a result, the inquiry state may have to last for 10.24s.

A slave that wants to be discovered enters in inquiry scan state. In this state, it listens for ID packets on the same 32 dedicated frequencies used from the master. The slave changes listening frequencies every 1.28s. The $T_{w_inquiry_scan}$ time during which the slave listens on the same frequency must last enough to completely scan one train. $T_{inquiry_scan}$ is the time between the beginning of two consecutive inquiry scan cycles and it shall last at most 2.56s. If $T_{inquiry_scan} = T_{w_inquiry_scan}$ then the inquiry scan activity is continuous. Both intervals can be changed. Default values are:

$$T_{inquiry_scan} = 1.28s \quad T_{w_inquiry_scan} = 11.25ms$$

3.2. Page and connection phases

Armed with the knowledge of the identity of devices in its proximity, the master of a piconet explicitly pages devices to join its piconet. With the information sent by the paging device to the paged device, the paged device

can join as a slave the piconet whose master is the paging device. Once the paging procedure is completed, the slave has been *enrolled* in the piconet.

The master sends page messages on every frequency belonging to the two trains. Since the master does not know when the slave will enter into the page scan substate, each train must be repeated at least N_{page} times. N_{page} is defined depending on the $T_{\text{page_scan}}$ value, which is the interval between the beginning of two consecutive page scans. Default values for page scan parameters are equal to inquiry scan default values:

$$T_{\text{page_scan}} = 1.28\text{s} \quad T_{\text{w_page_scan}} = 11.25\text{ms}$$

After some other interactions, the master in page state and the slave in page scan state enter in a substate in which the clock input to the hop selection mechanism is frozen on the same value, and they can communicate using the same hopping sequence.

On completion of page, the master enters into the *connection* state and asks for opening a connection. If the slave agrees, it acknowledges the request, the connection is established and the two devices can begin to exchange data on this connection.

The best situation in terms of device discovery and synchronization time is that in which the slave enters into the inquiry scan state exactly in the same instant in which the master begins the inquiry procedure and they start their hopping sequence from the *same* train. Vice versa, the worst situation is that in which the master starts the inquiry procedure on a train while the slave is listening on the last two frequencies of the alternative train. In this case, the time spent in device discovery is greater than 5.12s.

3.3. Slave programming

In BIPS the slave will be programmed in such a way that it begins alternating inquiry scan and page scan mode. Inquiry scan mode is necessary for the slave in order to be discovered by the master, while page scan mode enable the slave to synchronize itself with the master. After the reply to the master's page messages, the slave can accept a connection request. As soon as the device realizes it has been discovered, it exits from inquiry scan mode and remains in page scan mode. This behaviour avoids that the slave replies several times to inquiries either from different masters or from the same. The master periodically opens a connection with each slave in its area. Therefore, if a slave does not receive any connection request for a certain period of time, it assumes to be out of the previous master coverage area and it enters in inquiry scan/page scan mode.

3.4. Master programming

Master device periodically executes the inquiry procedure in order to discover devices moving in its coverage area. Information about discovered devices are kept in a local cache that is updated when necessary. Periodically the content of the cache is sent to BIPS-Server. Information about discovered devices are not immediately sent to the server after having collected inquiry results. It takes place after one polling cycle at least. This prevents updating database with inconsistent data. This could happen when there is a user moving inside the building and the master that discovers it immediately sends the position information to the server. The user could have meanwhile moved to another room. Moreover, the system keeps information on "in which room a user is", and if he is moving the system cannot precisely infer his current position. When the mobile device is discovered from the same master for at least one polling cycle, the system establishes its position and sends this information to the server. Every master keeps in communication with BIPS-Server and slave devices. On both sides it uses a client-server architectural model, playing two different roles in the two situations.

4. Initial measures and expected results

In this section, we briefly describe the environment used to develop and test BIPS and we provide some initial experimental and simulative results. The results concern the time and related probability of device discovery. The knowledge of these results is important for defining the scheduling policy of a master, which must dedicate a certain percentage of its working time to device discovery and the remaining time to serve slave applications.

4.1 The experiments

The whole system was developed on Linux platforms. Master role is played by a workstation equipped with a Pci-Pcmcia adapter from Texas Instruments and a 3COM Bluetooth card. Slave role is played by one laptop equipped with the same card. Linux operating system contains the *Official Bluetooth Protocol Stack (BlueZ)*, necessary to program card drivers. In all the experiments we have used the *f_{time}* routine of the Linux operating system for measuring the time spent by the master machine in discovering a new mobile device which enters into the coverage area of its piconet.

In all these experiments the master is completely dedicated to execute the discovery procedure, i.e. it is always in the inquiry phase. This situation represents the best case in the policy of device discovery, and it allows to verify if BIPS is able to track the mobile users.

In the various experiments we have only changed the working hypothesis of the slave. As already described,

this is the most advantageous situation for a moving device. The time that we measure in this first set of experiments is an interval that begins just before the master enters into the inquiry state and that ends when the master receives the answer from the slave to the inquiry message.

In the first experiment the slave promptly answers the inquiry since it is continuously listening for all the $T_{\text{inquiry_scan}} = 1.28\text{s}$ (see sect. 3.1). The outcome of the first experiment is illustrated in the following table.

Starting Train	Case No.	T_{average}
Same	255	0.3675s
Different	245	2.9273s
Mixed	500	1.6218s

The table shows the average discovery time that has been measured on 500 inquiry trials. The first column of the table shows the situation when the starting frequency trains used by master and slave are the same or not. The last row indicates the average result of all trials. It is worth noting that the probability that the master and the slave start on the same train is close to 50%. Our results are aligned with the recommendations specified in the Bluetooth standard.

In the second experiment the slave modifies its inquiry scan behaviour. The slave is not continuously listening for inquiry messages, but only for an interval which is one hundredth of all $T_{\text{inquiry_scan}}$, that is $T_{\text{w_inquiry_scan}} = 11.25\text{ms}$. The value of this time interval is the default value suggested from Bluetooth standard.

Starting Train	Case No.	T_{average}
Same	251	1.5931s
Different	249	4.1167s
Mixed	500	2.855s

The results shows that the average discovery time has considerably increased but not proportional to the percentage of reduction. In fact the increasing factor is about 5 times in the best case, that is when master and slave start on the same train, and about 1.5 times when master and slave start on different trains.

In the last experiment the slave alternates the period of inquiry scan and page scan. In both cases the slave listens for inquiry/page messages for an interval equal to 11.25ms, as long as the default values specified in the standard.

Starting Train	Case No.	T_{average}
Same	236	1.6028s
Different	264	4.1320s
Mixed	500	2.865s

We notice that the average discovery times are close to the previous results. Since the slave cannot be completely dedicated to the task of being discovered but it should also spend some time in exchanging synchronization data with the master, the situation described in the last experiment represents the way in which the slaves are programmed. Furthermore, as the experiments clearly show, this implementation does not favour the discovery phase maintaining a correct balance among slave activities.

As already noted, the situation measured from this first set of experiments represents the best case in the policy of device discovery. A more realistic situation should take into account the fact that also the master cannot be completely dedicated to the task of discovering mobile devices, since it needs to dedicate time and resources to exchanging data with the slaves in its coverage area.

The real situation that we would like to measure is that where BIPS tracks several mobile devices so that we could evaluate the real master's performance in device discovery. For defining the master scheduling parameters before the actual implementation of BIPS, we have used a simulation environment to obtain some results for discovery phase with more than one slave. This is illustrated in the next subsection.

4.2 Simulation environment and results

To establish correct hypothesis for master behaviour in BIPS, we have developed a Bluetooth extension to the VINT project network simulator "ns2" [8]. Our extension is based on BlueHoc, the ns2-based simulator released by IBM [9]. In particular, in order to implement the operations of our solution, we have enriched BlueHoc with mechanism for handling collisions that might arise during the establishment of a link.

In the simulated scenarios, we consider a single Piconet in which only a device always assumes the role of master, while the others assume the role of slaves. In particular, slaves are always in inquiry scan mode and they start listening on frequencies of train A. Since discovery time depends on the number of slaves in the Piconet, we performed our simulations varying this number in the range [2,20].

In the first simulation we consider the best situation for discovering mobile devices. In fact, the master is completely dedicated to the inquiry phase and it transmits on train A, the same on which the slaves are listening. We have done 100 trials for each scenario and we have analysed the system every 0.25s. We can notice that when the number of slaves increases, the time needed to discover all devices (100%) increases too. It reaches 3 seconds for 20 slaves (**Figure 2**).

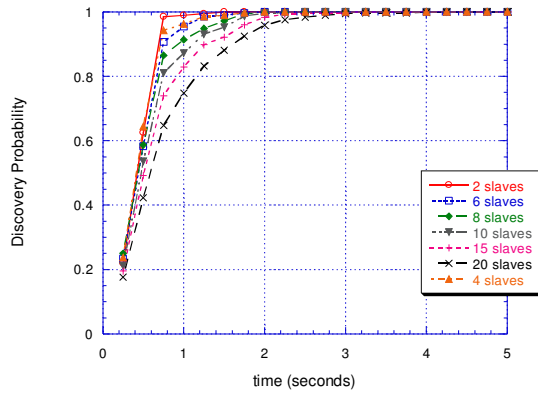


Figure 2. Inquiry using train A only.

In the second simulation, the master is still completely dedicated to the inquiry phase but here it transmits alternating between A and B trains while the slaves are listening on train A. In this case, we notice that the master has 50% probability to discover devices since the starting train is randomly chosen. If the master starts transmission on B train, the slaves have to wait at least 2.56 seconds before being discovered. This situation is clearly illustrated in Figure 3.

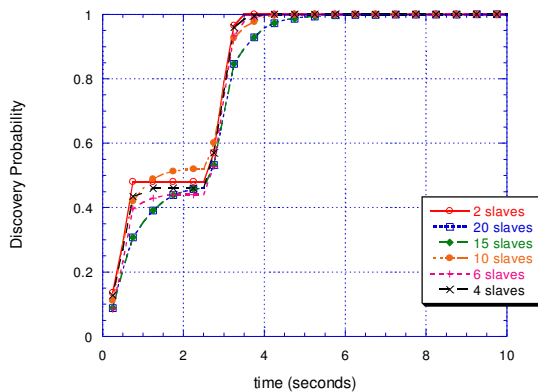


Figure 3. Inquiry using A and B train.

In the last simulation the master alternates device discovery and connection management and it transmits inquiry messages only using train A. It is reasonable to hypothesize that master is dedicated to device discovery for 20% of its operational cycle. For this reason we have fixed the length of inquiry phase to 1 second and the total period to 5 seconds.

We can notice that, when the maximum number of slaves in the Piconet is up to 10, in the average the master succeeds in discovering about 90% of these slaves in the

first 1 second. Only in the second operational cycle it discovers the 100% of slaves. Increasing the number of slaves by 5 (i.e., 15-20), usually the slaves are all discovered in 2 cycles. (see Figure 4).

In a real scenario, the starting trains for master and slaves cannot be defined by the programmer, so they have 50% probability to start with the same train. For this reason, we have to fix the length of discovery phase to more than 3.84s. Indeed, the master spends 2.56s on the first train, discovering all the slaves listening on the same train. Then, as we have seen in the first simulation, 1.28s are enough to discover the 90% of the remaining slaves listening on the other train (if they are less than 10).

In summary, when there are 20 slaves in a master coverage area, if the master spends 3.84s on device discovery and the slaves are programmed for enrolling in the piconet, then the percentage of mobile device discovered is about 95% (19 slaves). In fact the 50% of the slaves belonging to the first train are completely discovered, and the 90% of the remaining 50% will be discovered in the second train.

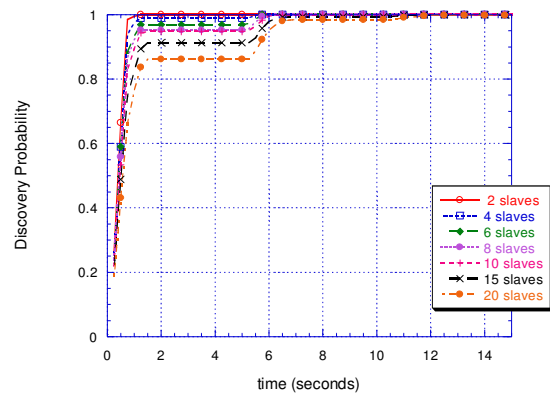


Figure 4. Inquiry and connection mngmnt

5. Conclusions and future developments

As shown in the previous section, when there are 20 slaves in a piconet, if the master spends 3.84s on device discovery and the slaves are programmed for enrolling in the piconet, then in the average case 19 slaves will be discovered. This represents a percentage that is satisfactory for BIPS.

To establish the percentage of working time that a master dedicates to device discovery, we have to estimate the average time that an average mobile user spends in crossing a piconet. Considering that a mobile user normally walks with a speed in the range [0 , 1.5] meters per second and that the diameter of the coverage area is

about 20m, we can estimate that an average walking user will spend $20\text{m} : 1.3\text{m/s} = 15.4\text{s}$ in the piconet.

This time represents the length of a complete operational cycle of a master. Hence, the master will dedicate a continuous slot of 3.84s for device discovery and the remaining 11.56s for servicing the slaves. The average load of tracking service in BIPS is about 24% of the operational cycle.

We are currently considering the following main direction of work. We will experiment a BIPS system where each logical Master is compound of two cards that cover the same physical area. One card will be completely devoted to the task of discovering new mobile users entering into the area, while the second card will be completely dedicated to the exchange of data among BIPS and those users on the same area that are already connected with the system. In this way, we will separate the burden of work on two parallel devices that do not interfere each other.

Acknowledgements. We are grateful to Bruno Raffaele for his help in setting up simulation environment.

6. References

[1] T. Imielinski and B.R. Badrinath, "Wireless Mobile Computing: Challenges in Data Management", *Comm. ACM*, vol. 37, no. 10, Oct. 1994, pp. 18-28.

[2] R. Malladi and D.P. Agrawal, "Current and Future Applications of Mobile and Wireless Networks", *Comm. ACM*, vol. 45, no. 10, Oct. 2002, pp. 144-146.

[3] C. Bisdikian, "An Overview of the Bluetooth Wireless Technology", *IEEE Comm. Magazine*, vol. 45, no. 10, Dec. 2001, pp. 86-94.

[4] Bluetooth web site: <http://www.bluetooth.com>.

[5] eBay site: <http://pages.ebay.com/catindex/catpda.html>.

[6] IEEE 802.11 web site: <http://grouper.ieee.org/groups/802/11/main.html>.

[7] E. W. Dijkstra, "A note of two problems in connexion with graphs", *Numerische Mathematik*, 1 (1959), pp. 269-271.

[8] The VINT Project, The ns Manual, <http://www.isi.edu/nsman/ns/>, 2002.

[9] IBM, BlueHoc: Bluetooth Ad Hoc Network simulator, Version 1.0, <http://www.124.ibm.com/developerworks/projects/bluehoc>, June 2001.