Telecommunication Systems 28:3,4, 333–347, 2005
© 2005 Springer Science + Business Media, Inc. Manufactured in The Netherlands.

Evaluating Bluetooth Performance as the Support for Context-Aware Applications

JUAN-CARLOS CANO, DAVID FERRÁNDEZ-BELL and PIETRO MANZONI
{jucano;pmanzoni}@disca.upv.es, dferry@ieee.org
Department of Computer Engineering, Polytechnic University of Valencia, Camino de Vera s/n, 46071 Valencia, Spain

Abstract. We present an experiment relative to the use of the Bluetooth wireless technology to provide network support for context-aware applications. We describe an approach to provide network interconnection using a combination of wireless and wired network technologies. We also describe the steps taken to create a Bluetooth based context-aware application. We, finally, evaluate, using a small test-bed and simulation the overall performance of this technology when adopted in the area of context-aware and ubiquitous computing.

Keywords: mobile ad hoc networks, ubiquitous computing, bluetooth

1. Introduction

Context-aware applications try to offer a flexible and adaptable service being totally conscious of the client, what surrounds him, and of its environment [Weiser, 23]. Much progress has been made in the human–computer interactions area to help the definition of these types of applications, but still, context-aware computing remains an unexplored area with much to be investigated and developed [Chen and Kotz, 7]. Context-aware applications require mobile wireless network technology. Many possible wireless networking technologies are available, ranging from 3rd generation wireless networks to personal area network (PANs) [Mäntyjärvi et al., 14].

Unlike other interesting proposals, like [Siegemund, 20; Nigel Davies et al., 16; Kargl et al., 11] or [Weatherall, 22] we concentrated on a bounded environment and based our proposal on the use of the Bluetooth technology [4]. Bluetooth is a versatile and flexible short-range wireless network technology with low power consumption [Beutel and Kasten, 2]. Above all, we are interested in its ability to locate neighbor devices and discover the type of services they could offer. Nodes that are close-by can find their neighbors using the *inquiry* procedure. After discovering close-by devices, a node can decide to page and to connect to them. A dedicated protocol called *Service Discovery Protocol* (SDP) is then used to interchange information about all the available services at each node. For a good overview on Bluetooth, please refer to [Chatschik, 6].

Our experimental applications provides context depended information to the visitors of a museum to enhance their experience. The system will give to visitors precise information about what they are viewing, at their level of knowledge, in their natural language. It will also give the possibility to have a graphical user interface (GUI) adapted to their device, e.g., mobile phone, PDA, laptop. The application also aims to help the maintainers of the museum by reducing costs in guiding their visitors, in keeping track of what are the preferred pieces of art, and so on.

We organized the network on the combination of an *edge* wireless network and a *core* wireless/wired network. The edge side is solely based on the Bluetooth technology. The application is basically built around the Bluetooth's Inquiry process and the Service Discovery Protocol. Each object will be provided with a Bluetooth device, that will be used as the information source Point of Access (POA). A user, while wondering around the museum, will continuously search for new POAs using Bluetooth's inquiry. When a new POA is found the user device will check if it can offer any information of interest through a SDP search. The core network is based on the integration of a fixed Ethernet local area network and a IEEE 802.11b WLAN. The edge network integrates one or more close-by users' devices. From the user perspective, information sources associated to any object are detected without intervention, obtaining new information about what they are viewing.

We evaluate our proposed application in a small test-bed and by using simulation. The goal of the test-bed experiment is not only to confirm the correct behavior of the designed application but also to acquire experimental data about how well is Bluetooth suited for context aware networking. Simulation allowed us to adjust the behavior of the application in many different scenarios.

The rest of this paper is organized as follows. Section 2 describes the end application and the overall system organization. Section 3 presents the details of the implementation prototype and section 4 illustrates the evaluation of the proposed design. Finally, section 5 describes the final comments about the work.

2. The experimental application

The experimental application considered in this paper is an hypothetical museum enhancement that provides visitors with context-aware information. The visitors should either possess or hire a Bluetooth enabled device. After downloading (if needed) the client application and gone through a brief configuration process, the users can use the system as a guide or as an information source of everything they see.

Thanks to Bluetooth, the museum system can find out where every user is, through the logging of the users and their interaction with the system, as it will provide information of only what the user has in front of him. Knowing what the user has in front, the system can provide the user with detailed historic and cultural information of the piece of art and any other information considered interesting. As many different types of devices exist, all with different technology and features, the system should be able to detect the type of device and present the information adapted to the technical capabilities of that device. For example, it could minimize the graphical performance for smaller devices like cellular phones to make better use of the audio system.

2.1. Overall system description

The overall network architecture is based on the cooperation of an *edge* wireless network and a *core* wireless/wired network. The edge side is solely based on the Bluetooth technology. The core network is based on the integration of a fixed Ethernet local area network and a wireless IEEE 802.11b LAN.

The interference issue between Bluetooth and IEEE 802.11 and between Bluetooth piconets is a known problem. In [de Morais Cordeiro et al., 8], for example, the authors demonstrate the impact of the interferences on the throughput. They show that the ideal conditions cannot always be achieved and hence applications must be designed to take Bluetooth bandwidth limitations and fluctuations into consideration. In our work we therefore assume that interferences would only decrease the available bandwidth without limiting the functionalities of our architecture.

The system considers three types of nodes: *Museum Information Clients* (MICs), *Museum Information Points* (MIPs), and the *Central Data Server* (CDS). A visitor provided with a Bluetooth enabled PDA is the basic example of a MIC. There is a MIP associated to one or more pieces of art or objects. Finally, the MIPs are connected to the CDS with an "adequate" combination of Bluetooth, Ethernet or IEEE 802.11b devices. The adequacy of the configuration depends on the physical structure of the facilities. Figure 1 shows a pictorial representation of a possible configuration.

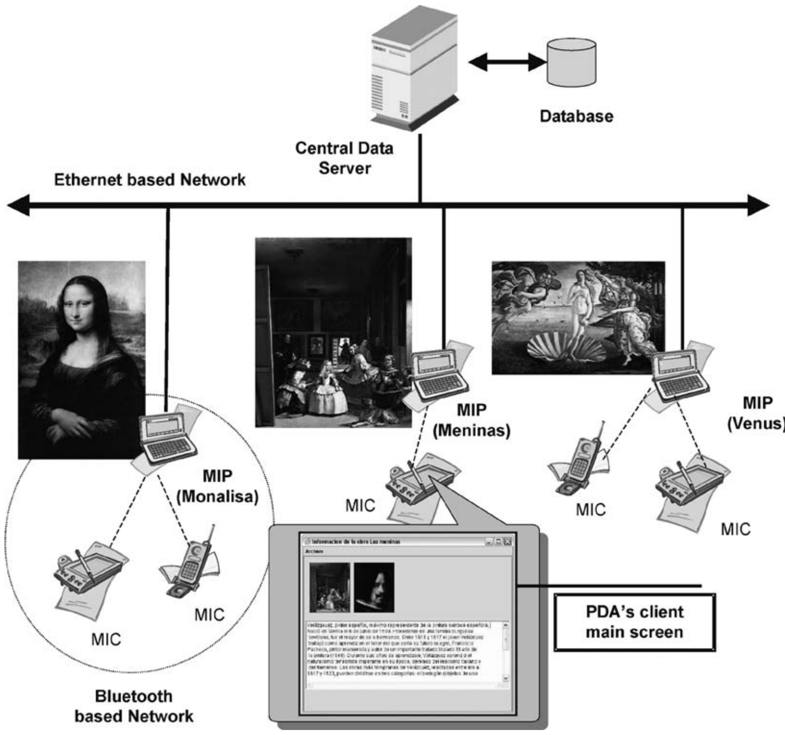


Figure 1. A pictorial representation of a possible configuration of the overall network.

The overall architecture is logically centered around the data server. A data caching scheme might anyway be adopted to reduce to load on the main server.

The edge network integrates one or more close-by MICs. Bluetooth allows to detect devices and obtain information about them relatively quickly. From the user viewpoint, MIPs are detected without the user intervention, obtaining new information about what they are viewing.

The application is basically built around the Bluetooths inquiry process and the Service Discovery Protocol (SDP). As the information point has one Bluetooth device per object represented, the object is determined by the interface where the connection is made; therefore location data is also determined. Other approaches like triangulation using the signal strength to reduce the area where the user is, might result in an excess of work and time spent (2 or three connections to do the triangulation) for something only valid in a very short period of time.

3. The implementation

Bluetooth has the capability of using TCP/IP through either the *Dial-Up Networking* (DUN) profile, the *LAN access* (LAN) profile or the *Personal Area Network* (PAN) profile [5]. We have however observed that there is an important bandwidth overhead in DUN due to the RFCOMM layer. Moreover, both the DUN and the PAN, introduce high latency. By latency we refer to the long time Bluetooth requires to establish a connection; up to half a minute between inquiry and connection. If on top of that we have to set up the profile, the IP routes and gateways, before performing the actual data communication, too much time would be lost. If it takes 20–30 seconds for a visitor to view the desired information, he will most certainly stop using the system. We therefore decided not not to use any profile but to use directly the "Logical Link Control and Adaptation Protocol" (L2CAP) layer.

The BlueZ distribution [Krasnyansky, 12] provides the socket API to L2CAP. L2CAP provides a reliable channel and uses segmentation and reassembly on "Asynchronous Connection-Less" (ACL) packets. L2CAP also multiplexes communications through the Protocol/Service Multiplexor (PSM) field abstraction that works basically as a TCP port.

3.1. The museum central data server

The operation of the museum central data server is quite straightforward: it starts and waits for a connection on the default server port. Incoming connection might be either configuration requests or information requests.

When receiving an information request, the server will spawn another process for it to receive the profile and the object name. For each profile option received the server will acknowledge it to make sure errors are controlled. Once all the profile are received, the server will log the petition for security and for statistical information. After logging the petition the data server will access the directory structure taking into account each profile option. This information will be sent to the information point who in turn will forward it to its client.

A client profile consists of a language of preference, the educational level, and the device type. The profile allows information to be formatted to take into account the graphical capabilities of the client and to determine the complexity of the explanations given. The information to be sent back to the information point is the contents of the object-record which in our prototype consist of two photos, one of the actual object being viewed and another of the author, and three text descriptions.

3.2. The museum information points

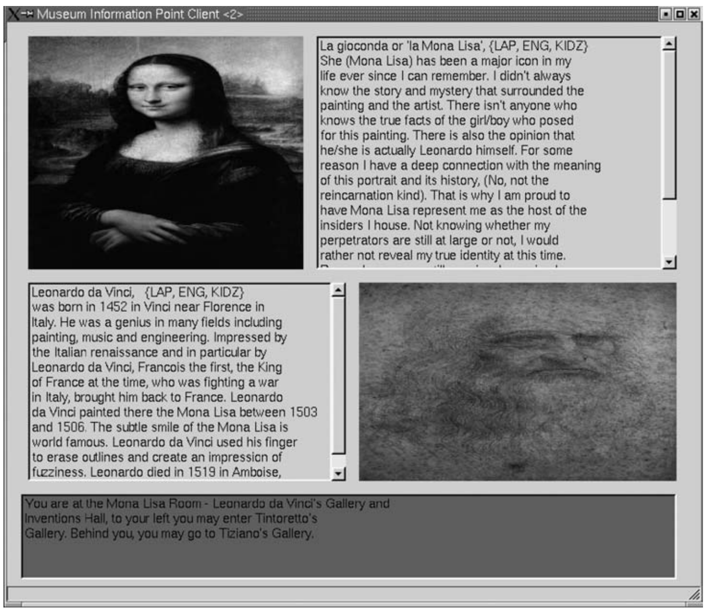
The museum information points first executes the SDP daemon to register its own services. The program will then proceed and connect to the museum server to retrieve the server's configuration file. This configuration file is presented to the information point technician who would choose an object for that information point to represent. Then, the program would start the SDP daemon and register its service along with the appropriate attributes like titles in various languages and PSM values for the clients to know where to connect to.

Once all the SDP registration has been done, the information point will create and bind the L2CAP layer sockets and wait for connections coming from the Bluetooth interface. When a connection comes in, the information point will spawn a child process to deal with the client's request. The child process first mission is to log the clients Bluetooth address and date of petition. Once the logging has finished the information point will receive the clients profile. The information point would append the local object's name and clients Bluetooth address and send it off to the central data server. There it will get processed and the required information will be sent back and passed on to the client. Once all the data has been sent to the client it will disconnect and the child process will end. The parent process will carry on waiting for further client petitions.

3.3. The museum information clients

The first step for each museum information clients is initialization: the user has to fill his profile with the corresponding preferences. Once all the data is filled in, the application will search for information on what he views. The client application will proceed to do a Bluetooth inquiry. If no devices with the correct class of device are found, it will show a message to the user to move closer to an object. If at least one device is found, it will proceed to do SDP searches on each of the devices found. If only one device was found the client will connect to that device, if not the application will create a list of found objects with the title of the object in the users preferred language. Once the information point has been chosen, either because it was the only one or because the user had chosen it from the list, the application will connect to the device.

Once connected, the client will send the stored profile to the information point and it will eventually receive the information. Once all the information is received the application will show it to the user. After the first retrieval of information, the entire



La gioconda or 'la Mona Lisa', {LAP, ENG, KIDZ}
She (Mona Lisa) has been a major icon in my
life ever since I can remember. I didn't always
know the story and mystery that surrounded the
painting and the artist. There isn't anyone who
knows the true facts of the girl/boy who posed
for this painting. There is also the opinion that
he/she is actually Leonardo himself. For some
reason I have a deep connection with the meaning
of this portrait and its history, (No, not the
reincarnation kind). That is why I am proud to
have Mona Lisa represent me as the host of the
insiders I house. Not knowing whether my
perpetrators are still at large or not, I would
rather not reveal my true identity at this time.

Leonardo da Vinci, {LAP, ENG, KIDZ}
was born in 1452 in Vinci near Florence in
Italy. He was a genius in many fields including
painting, music and engineering. Impressed by
the Italian renaissance and in particular by
Leonardo da Vinci, Francois the first, the King
of France at the time, who was fighting a war
in Italy, brought him back to France. Leonardo
da Vinci painted there the Mona Lisa between 1503
and 1506. The subtle smile of the Mona Lisa is
world famous. Leonardo da Vinci used his finger
to erase outlines and create an impression of
fuzziness. Leonardo died in 1519 in Amboise,

You are at the Mona Lisa Room - Leonardo da Vinci's Gallery and
Inventions Hall, to your left you may enter Tintoretto's
Gallery. Behind you, you may go to Tiziano's Gallery.

Figure 2. A screen capture of the prototype application.

process of inquiring, SDP searching, profile sending and information receiving is done automatically by a separate thread. To not bother the user with continuous updates, the thread will only modify the user viewed data when new servers are found or the current information is no longer available.

Figure 2 presents a screen capture of the final prototype related to a MIC visualizing a data-block.[1]

3.4. Scatternet support

Bluetooth standard allows up to seven slaves devices and a master one to form a centralized *ad hoc* network called a piconet. In our system, each MIP acts as the master of its own piconet which will allocate slots for up to seven active clients. In a crowded museum it is expected that more than seven visitors (i.e., a piconet) can be at the same time looking at the same piece of art. To deal with those situations we increase the scalability of the overall systems using a scatternet formation algorithm to interconnect multiple close-by piconets.

The Bluetooth standard does not specify a specific scatternet formation algorithm. Achieving an optimal scatternet topology is currently the subject of intensive research [Law and Siu, 13; Salonidis et al., 18; Basagni and Petrioli, 1]. However, most of the studies have not directly addressed the issues related to the implementation. The different approaches try to obtain a scatternet topology where two piconets can communicate by sharing one or more "bridge" devices. These bridges may either act as a master

[1] The prototype application is available at http://www.grc.upv.es/software.html.

in one and as slave in the other (master/slave) or as a slave in both piconets (slave/slave), but not as a master in both.

In [Siegemund and Rohs, 21] the authors showed that master/slave bridges could result in reduced throughput, while slave/slave bridges require more complex negotiation protocols between masters sharing the slave devices. Moreover, the inter piconets scheduling protocols in scatternets with slave/slave bridges devices require complex coordination mechanism to enable such bridges in an appropriate manner [Johansson et al., 10].

Since in our application nodes do not need excessive bandwidth we will use bridges devices operating only in the master/slave scheme, passing all the inter-piconet communication via the masters. This approach allows us to simplify the inter-piconets scheduling protocols.

We proposed a scatternet algorithm based on using the *hold* mode [4] to allow a device to leave a piconet and to join another one without any modifications to the Bluetooth specifications. We limited a piconet to a master and a maximum of five slave devices. According to [Seth and Kashyap, 19] using five slave devices allows a tradeoff between path length and piconet congestion.

We, thus, reserve two connection per piconet that will be used for bridge connections. The MIP of each object in the museum will create the first piconet of the scatternet. When more than five clients get within reach of the same MIP, they will create successive piconets. Basically, when a client device cannot join the MIP's piconet, it will try to discover any other master that is acting as a bridge to the MIP's piconet. If no bridge is found the device creates its own piconet acting as the master and as a bridge to the MIP's piconet. For this new master to be discovered it will register a new service called Bridge_to_the_MIP. The bridge device periodically goes into the *hold* mode to relay packets from the MIP's piconet to its own piconet members. The master device will periodically POLL its slaves.

When a client requires the associated object information, its piconet master will relay the requested information to the MIP. To join the MIP's piconet, the bridge enters the *hold* mode in its piconet and then enables the INQUIRY SCAN status in the MIP piconet. The MIP master device will discover it using periodic INQUIRY messages. When the *hold* time expires, the bridge will leave the MIP piconet, and relay the received information to the slave, that is the client. The minimum interval that the bridge will spend "outside" its piconet is calculated according to the following parameters: the overhead incurred by entering the *hold* mode, the time a bridge requires to join a piconet and the time to get the information from the MIP. Anyway, this period should be smaller than the maximum *hold* time specified in the standard (40.9 seconds or 65440 slots). The criteria to set a reasonable minimum *hold* time should be based on the results of section 4 where we evaluate the inquiry delay and the throughput performance of the application in our laboratory test-bed. Figure 3 shows the overall sequence diagram for a bridge device.

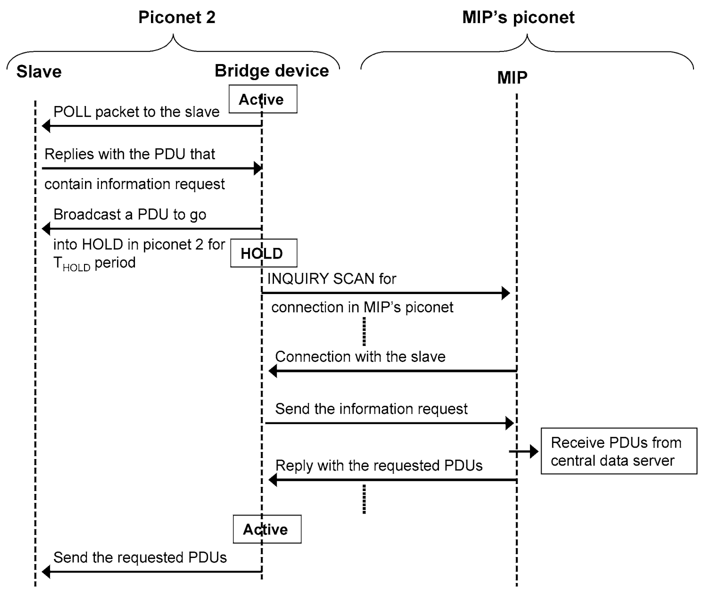

Piconet 2
MIP's piconet
Slave
Bridge device
MIP
Active
POLL packet to the slave
Replies with the PDU that contain information request
Broadcast a PDU to go into HOLD in piconet 2 for T_HOLD period
HOLD
INQUIRY SCAN for connection in MIP's piconet
Connection with the slave
Send the information request
Receive PDUs from central data server
Reply with the requested PDUs
Active
Send the requested PDUs

Figure 3. Sequence diagram for the bridge operation.

4. Design evaluation

This section presents the evaluation of our proposed application in a small test-bed and by using simulation. The goal of the test-bed experiment is to acquire experimental data about how well is Bluetooth suited for context aware networking. Simulation allowed us to adjust the behavior of the application.

4.1. The application performance evaluation using the test-bed

We built a test-bed where the museum data server and the museum information point run on an Intel Pentium IV 2400 Mhz standard PC based on Suse Linux 8.1 with a 3Com CREB96 Bluetooth USB Dongle card. The museum information client runs on an Intel Pentium IV 2000 Mhz TOSHIBA Satellite 1900-303 based on Suse Linux 8.1 with a 3Com CREB96 Bluetooth USB Dongle card. We used the BlueZ [Krasnyansky, 12] protocol stack. BlueZ is the official Linux Bluetooth stack and it has strong support for USB dongles devices. BlueZ is part of the 2.4.20 version of the Linux kernel.

The performance experiments focused on evaluating how distance affects inquiry delay and throughput performance of the Bluetooth channel. The inquiry delay offers an estimation of the applicability of the system, a high inquiry period will discourage the user from using the system. The throughput performance gives us some clues related to the application design. From the energetic point of view, the inquiry procedure is also extremely important. According to [15], power consumption highly increases during the inquiry and inquiry scan mode.

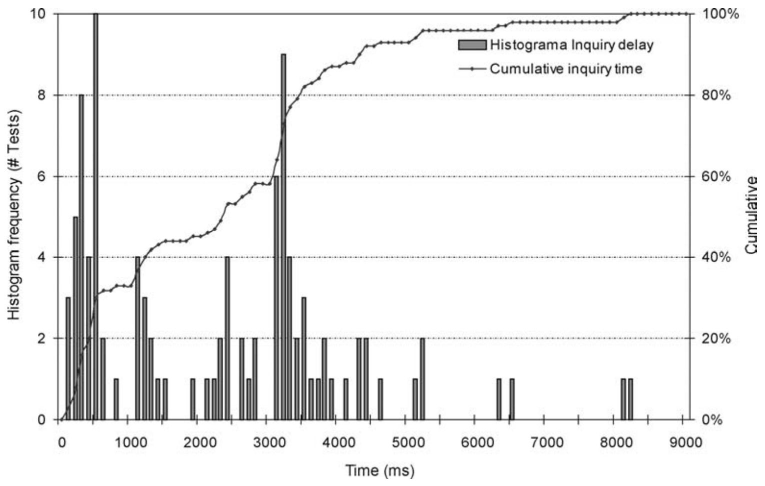


Figure 4. The cumulative inquiry delay distribution for the 100 tests. Distance between the museum information point and the museum information client is 1 m.

4.1.1. Inquiry establishment delay

We measured the duration of the inquiry procedure in two different cases: with the distance from the client to the master device fixed to 1 m, and with a distance of 10 m. To avoid consuming the whole inquiry timeout period of 10.24 s we fixed the maximum number of discovered devices equal to 1. When using a distance of 1 m the experiments showed an average inquiry delay of 2.313 s $\pm$ 0.361 and a standard deviation equal to 1.8122 $\pm$ 0.258 with a confidence interval of 95%. As we increase the distance among nodes to 10 m we obtain an average inquiry delay of 2.364 s and a standard deviation of 0.203 with the same confidence interval. Therefore, it seems then that the inquiry procedure is not highly sensitive to distance.

Figure 4 shows the histogram distribution of the inquiry delay and the cumulative results as a function of time over a total of 100 tests. The results show that after 2.455 s the museum client discovers the information point in a 50% of the tests. Most importantly we observe that this percentage increases to 95% only after 5.264 s. In [Salonidis et al., 17] the authors reports that the expected inquiry delay is around 1 s between two devices in inquiry and inquiry scan modes. The reason might stand in the environment where the experiment were conducted. Our laboratory has a IEEE 802.11 based access point that might interfere with Bluetooth devices. As described in section 2.1 many IEEE 802.11 connections might be present in the museum environment. We are thus representing a more realistic situation. Therefore, we can approximately assume 5 s has the highest waiting time for the inquiry process.

4.1.2. Throughput analysis

We now evaluate the impact on throughput of both the ACL packet type and the distance. An ACL link is parameterized by packet size and data encoding. According to this parameters, at the baseband layer, Bluetooth offers 6 different data packets that can be classified in two main groups. Data Medium Rate packets, DM packets, which provide a 2/3 FEC Hamming code and Data High Rate packets, DH, which provide no FEC coding at all.

We configured the application point to continuously send a data-block to the museum client during 100 s. The throughput for each data-block received is calculated at the client side. Figure 5 shows the obtained experimental results when the information point side is using DH packets.

We first observe that Bluetooth offers a quite steady throughput below 10 m, independently of the data packet type selected. Passing the 10 m limit the application still works without a sharp performance reduction. When we select the more efficient DH data packets, we improve the observed throughput with respect to the DM packets. At 10 m the DH packets outperform the DM ones by 57%, 46% and 32% respectively for 1, 3 and 5 slots packets. The results confirm that although the use of DM packets increases efficiency while reducing the probability of successful transmission, the DH packets can be a candidate for more efficient transmissions. Moreover, after 10 m, as distance increases, longer packets (DH5 and DH3) suffer a higher degradation. Finally, all the obtained results are below the maximum throughput provided in the specification. Below 10 m the application gets a 74%, 85% and 91% of the maximum throughput provided in the standard for the DH(5, 3, 1) packets, respectively. When using the more conservative DM(5, 3, 1) packets this percentage gets to 85%, 87%, and 93%.

In conclusion, even if intuitively we might expect that the inquiry delay will increase while throughput should decrease with distance due to the relation between errors and distance, the experiments showed that the inquiry procedure is not highly sensitive to distance. We can approximately assume 5 s as the highest waiting time for the inquiry process. We observe that Bluetooth offers a quite steady throughput below 10 m,

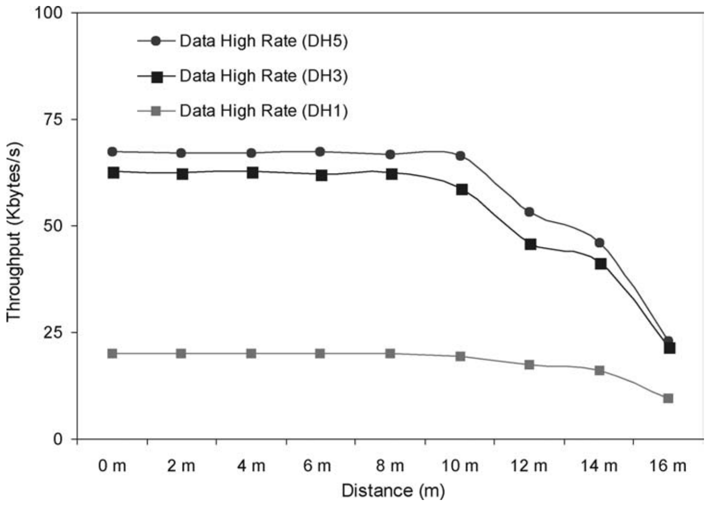


Figure 5. Obtained application throughput comparison of an ACL link using different DH packets as a function of distance.

independently of the data packet type selected. The results also confirm that although the use of DM packets increases efficiency while reducing the probability of successful transmission, the DH packets can be good candidates for more efficient transmissions.

4.2. Application performance evaluation using simulation

We now take advantage of the flexibility of a simulation tool to evaluate the performance of the context aware application. We used the network simulator version 2 (ns-2) [Fall and Varadhan, 9] and the BlueHoc [3] tool. BlueHoc is an open source code that extends the ns-2 functionality with an implementation of Bluetooth. It includes the baseband layer and the link manager of the Bluetooth technology. As with the test-bed experiments we analyze the inquiry establishment delay and throughput performance. We test how the distance between MICs and MIPs devices can affect performance.

4.2.1. Inquiry establishment delay

We first set up a BlueHoc scenario where two devices are in inquiry mode and inquiry scan mode, respectively. We increase the distance from 1 to 10 m. Figure 6 shows the inquiry delay variation with distance values of 1 m and 10 m, respectively.

When we consider the 1m scenario, the results show an average inquiry delay of 1.440 s ± 0.0357. The standard deviation is 0.179 ± 0.025 with a confidence interval of 95%. As we increase the distance to 10 m, we obtain similar results. The main difference stands in that in several simulated test where the inquiry procedure timeout of 10.24 s expires with no device discovered. With respect to the test-bed experiments results we observe that as distance increases the obtained results are quite similar. The only differences stands in the timeout expiration of selected test. Moreover, the simu-

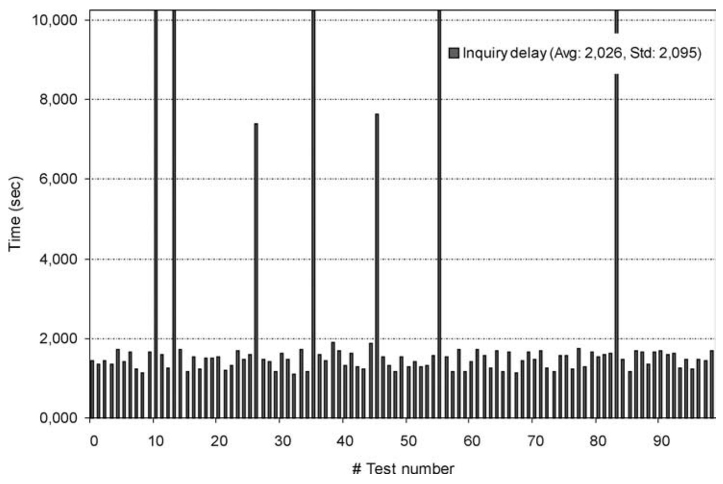


Figure 6. Inquiry delay for the 100 simulated tests. Distance between the museum information point and the museum information client is 10 m. The inquiry timeout value is fixed for all the test.

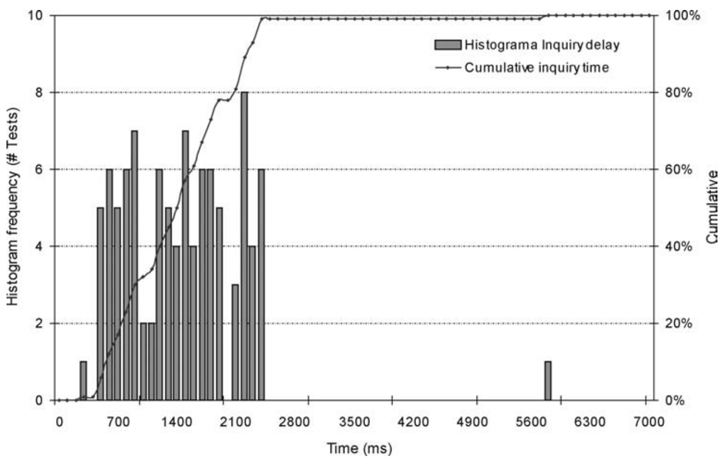


Figure 7. Inquiry delay distribution for the 100 simulated tests. Distance between the museum information point and the museum information client is 1 m. The frequency offset is randomly selected for each test.

lation results obtain a quite smoother behavior that the real experiment. The standard deviation notably differs from the one obtained with the real tests.

While the former observation could depend on the BlueHoc implementation of packet error model, the latter seems to depend on the frequency clock offset between the scan and scan mode devices. We therefore checked how frequency offset can affect inquiry delay. We repeated all the simulation using a randomly selected frequency offset between the two devices. The simulated inquiry delay when the frequency offset is randomly selected and the distance between the two devices is 1 m show an average value of 1.516 s $\pm$ 0.147 and a standard deviation value of 0.738 $\pm$ 0.105 with a confidence interval of 95%. Figure 7 shows the histogram distribution and the cumulative results as a function of time. The results show that after 1.506 s 50% of the simulated tests the inquiring device discovers the other device. The 95% of cases presents a value of 2.456 s. Finally, when we increase the distance to 10 m, we obtained similar results. The inquiry procedure does not seem to be sensitive with distance. The results exhibit a higher variability than in the previous case. We think that the differences with respect those obtained in our test bed are basically due to the fact that the Bluetooth layer's model implemented fails to produce realistic results and because the data we obtained in the real experiment were affected by the presence of the IEEE 802.11 access point.

4.2.2. Throughput analysis

We now perform a throughput simulation study by using the previously generated scenarios. We used an FTP application as data traffic source model. We repeated each simulation run by using all the possible DH and DM packet types. Figure 8 shows the obtained simulation results when using DH packets as a function of distance. For each run, we average the throughput over the 100 s from the output.

The main difference with respect to experimental results stands in the distance from where performance starts to decrease. We can observe how the average throughput starts

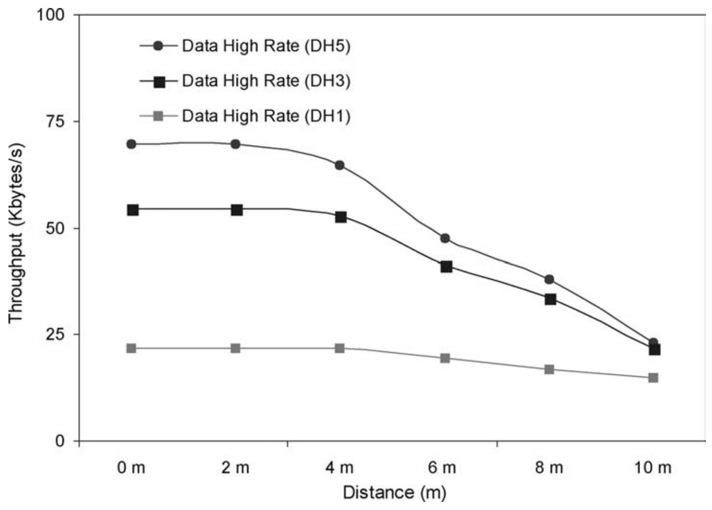


Figure 8. Obtained simulated throughput comparison of an ACL link using different DH packets as a function of distance.

Table 1
Theoretical maximum throughput specified in the standard for different Bluetooth packet type. For each packet type, column 2 and 3 show the test-bed and simulated obtained throughput percentage with respect the maximum in the standard.

Packet type/throughput (KB/s)	Test-bed (%) 2 m/16 m	Simulations (%) 2 m/10 m
DH5/90.40	74.22%/25.33%	76.90%/25.44%
DH3/73.20	85.57%/29.77%	74.67%/29.78%
DH1/21.60	92.71%/44.23%	99.97%/68.70%
DM5/59.72	83.42%/39.71%	52.85%/24.27%
DM3/48.40	87.84%/42.02%	54.97%/28.41%
DM1/13.60	93.53%/52.30%	70.37%/62.78%

to get worse at around 4 m. Moreover, we could not reach distances longer than 10 m because the BlueHoc simulator failed to work. The obtained results are quite more strict that those obtained in the experimental test-bed. It seems that BlueHoc propagation model provides a quite high penalty from distances higher that 4 or 5 m specially for DH packets.

Table 1 compares the throughput percentages obtained via test-bed and via simulations with respect the theoretical maximum in the standard. We have selected the 2 m and 16 m scenario for the test-bed experiments. We also selected the 2 m and 10 m scenarios obtained via simulation.

For those scenarios where distance stands lower that 4 m, the more efficient DH packets achieve a better performance than its DM counterpart. With respect to the theoretical maximum throughput DH(5, 3, 1) achieve a 72%, 76% and near 100% while DM(5, 3, 1) only obtain 52%, 54% and 70%. Contrary to the real test, it seem that the

1 slot packet are quite more robust that the multi slots ones. For some specific simulated tests the DH1 packets improve the maximum theoretical value in the specification by 1% or 2%.

Finally, from distances over 4 m, the DM_x packets exhibit less negative impact that the DH_x ones. Therefore, as distance range from 4 to 10 m the DH_x packets obtain a similar throughput that the DM_x.

5. Conclusions

This paper presented a proof of concept of how Bluetooth technology can be used to design, develop and deploy context-aware applications. The forefront Linux tools, BlueZ, is still under development and many features are missing (like scatternets or advanced hold-park administration). It is however mature enough to be the underlying technology for these types of applications, with a very optimistic future.

We presented an application to provide context depended information to the visitors of a museum. The system was designed to give to visitors precise information about what they are viewing at their level of knowledge in their natural language of preference and giving a possibility for the user to have a GUI adapted to their device, enhancing their experience.

We evaluated our proposed application in a small test-bed and by using simulation. The experiments showed that the inquiry procedure is not highly sensitive to distance. We can approximately assume 5 s as the highest waiting time for the inquiry process. We observe that Bluetooth offers a quite steady throughput below 10 m, independently of the data packet type selected. The results also confirm that although the use of DM packets increases efficiency while reducing the probability of successful transmission, the DH packets can be good candidates for more efficient transmissions. We do think of Bluetooth as a outstanding contender among providers of network support system for context-aware applications.

Acknowledgements

This work was partially supported by the Spanish CICYT under Grant TIC2003-00339 and by the *Junta de Comunidades de Castilla la Mancha*, Spain, under Grant PBC-03-001.

References

[1] S. Basagni and C. Petrioli, A scatternet formation protocol for ad hoc networks of bluetooth devices, in: *Proc. of IEEE VTC Spring* (2002).

[2] J. Beutel and O. Kasten, A minimal Bluetooth-based computing and communication platform, Technical Report, Computer Engineering and Networks Lab, Swiss Federal Institute of Technology (ETH) Zurich (2001).

[3] BlueHoc, IBM Bluetooth simulator, `http://www-124.ibm.com/developerworks/opensource/bluehoc/`.

[4] Bluetooth SIG, Specification of the Bluetooth system – core. Version 1.1 (February 2001).

[5] Bluetooth SIG, Specification of the Bluetooth system – profiles. Version 1.1 (February 2001).

[6] B. Chatschik, An overview of the Bluetooth wireless technology, IEEE Communications Magazine 39(12) (2001) 86–94.

[7] G. Chen and D. Kotz, A survey of context-aware mobile computing research, Dartmouth College, Technical Report TR2000-381, Department of Computer Science (November 2000) available at: `ftp://ftp.cs.dartmouth.edu/TR/TR2000-381.ps.Z`.

[8] C. de Morais Cordeiro, D. Sadok and D.P. Agrawal, Piconet interference modeling and performance evaluation of Bluetooth MAC protocol, in: *Proc. of IEEE GLOBECOM*, San Antonio, TX (25– 29 November 2001).

[9] K. Fall and K. Varadhan, ns Notes and documents, The VINT project, UC Berkeley, LBL, USC/ISI and Xerox PARC, `http://www.isi.edu/nsnam/ns/ns-documentation.html` (2000).

[10] P. Johansson, M. Kazantzidis, R. Kapoor and M. Gerla, Bluetooth: An enabler for personal area networking, IEEE Network 15(5) (2001) 28–37.

[11] F. Kargl, S. Ribhegge, S. Schlott and M. Weber, Bluetooth-based ad-hoc networks for voice transmission, in: *Proc. of the IEEE Hawaiian Internat. Conf. on System Sciences* 36, Hawaii, USA (2003).

[12] M. Krasnyansky, BlueZ: Official Linux Bluetooth protocol stack, `http://bluez.sourceforge.net/` (2003).

[13] C. Law and K.Y. Siu, A Bluetooth scatternet formation algorithm, in: *Proc. of the IEEE Symposium on Ad Hoc Wireless Networks* (2001).

[14] J. Mäntyjärvi, P. Huuskonen and J. Himberg, Collaborative context determination to support mobile terminal applications, IEEE Wireless Communications 9(5) (2002) 39–45.

[15] E. Microelectronics, ROK 101 007 Bluetooth module datasheet rev. PA5 (April 2001).

[16] A.F. Nigel Davies, A.F. Keith Cheverst and K. Mitchell, Future wireless applications for a networked city: Services for visitors and residents, IEEE Wireless Communications 9(1) (2002) 8–16.

[17] T. Salonidis, P. Bhagwat and L. Tassiulas, Proximity awareness and fast connection establishment in Bluetooth, in: *Proc. of the 1st Annual ACM Mobile and Ad Hoc Networking and Computing Workshop* (2000).

[18] T. Salonidis, P. Bhagwat, L. Tassiulas and R. LaMaire, Distributed topology construction of Bluetooth personal area networks, in: *Proc. of IEEE INFOCOMM*, Anchorage, Alaska, USA (2001).

[19] A. Seth and A. Kashyap, Capacity of Bluetooth scatternets, Master's thesis, Computer Science and Engineering Department, Indian Institute of Technology, Kanpur, India (2002).

[20] F. Siegemund, Spontaneous interaction in ubiquitous computing settings using mobile phones and short text messages, in: *Proc. of the IEEE Workshop on Supporting Spontaneous Interaction in Ubiquitous Computing Settings, UBICOMP*, Göteborg, Sweden (2002).

[21] F. Siegemund and M. Rohs, Rendezvous layer protocols for Bluetooth-enables smart devices, in: *Proc. of Internat. Conf. on Architecture of Computing Systems* (2002).

[22] A.J. Weatherall, Ubiquitous networks and their applications, IEEE Wireless Communications 9(1) (2002) 18–29.

[23] M. Weiser, Some computer science problems in ubiquitous computing, Communications of the ACM 7 (1993) 74–83.