blueContext: using Bluetooth to provide context-aware information

David Ferrández Bell, Juan-Carlos Cano, Pietro Manzoni
Department of Computer Engineering, Polytechnic University of Valencia
Camino de Vera, s/n, 46071 Valencia, Spain.

Abstract

We present a “proof of concept” 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 Blue-tooth based context-aware application. We finally evaluate the overall performance of this technology using a small test-bed and simulations.

1 Introduction

Ubiquitous computing strongly depends on context-aware applications. These applications offer a flexible and adaptable service being totally conscious of the client, what surrounds him, its environment, or anything that might be useful in order to make a better experience for the end user. The application may change or adapt its functions, information and user interface depending on the context (by inferring or sensing it) and client’s preferences of that precise moment [1]. Context-aware applications require mobile or wireless network technology. Many possible wireless networking technologies are available, ranging from 3rd generation wireless networks to personal area network (PANs).

Unlike other interesting proposals, like [2], [3], [4] or [5] in this work we concentrated on a bounded environment and based our proposal on the use of the Bluetooth technology [6]. Bluetooth is a versatile and flexible short-range wireless network technology with low power consumption. It is designed to be small and to keep costs low to be included in practically any device. Above all, we were interested in its ability to locate neighbor devices and discover the type of services they could offer.

Bluetooth is based on a connection-oriented scheme. Nodes that are close-by can find their neighbors using the inquiry procedure. After discovering close-by devices, a node can decide to page to them and to connect to them. A dedicated protocol called SDP (Service Discovery Protocol) is then used to interchange information about all the available services at each node. Thanks to the inquiry procedure we are able to find devices in our neighborhood. Service discovery and browsing allows the user to find required services from those
devices and all the necessary configuration and connection parameters. Therefore the user has an easy and transparent way of interconnecting devices, making context-aware applications much easier. From the service provider view, these features allow identifying, locating and finding out what type of users are the ones nearby and giving them an according service.

Our experimental applications will provide 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 and 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 core network is based on the integration of a fixed Ethernet local area network and a wireless IEEE 802.11b LAN. The edge network integrates one or more close-by users’ devices. From the user viewpoint, information sources associated to any object are detected without intervention, obtaining new information about what they are viewing. Each piece of object will be provided with a Bluetooth device, that will be used as the information source point of access (POA).

We evaluated 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 stress the behavior of the application in a scenario where a museum information point will have to connect with a large number of low-power clients.

The rest of this paper is organized as follows: Sections 2 describes the overall system organization. Section 3 presents the details of the implementation prototype and Section 4 illustrates the evaluation of the proposal.

2 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 is not considered here since we assume that it 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 bridging points (BP). A visitor provided with a Blue-
tooth enabled PDA is the basic example of a MIC. There is a MIP associated to one or more pieces of art or objects. Finally, static computers will be used as BPs and might be provided 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.](image)

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 intervention, obtaining new information about what they are viewing.

The application is basically built around the Bluetooth’s Inquiry process and the Service Discovery Protocol (SDP). 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. If anything interesting is found the user will be notified to whether he prefers to see the new information. In that case the user device will connect to the POA and ask for the information. However, for the information source to know what type of information the user needs, it requires the context of the user. The user’s device will send to the information source the profile that was entered in the beginning configuration process.

After the client has sent the profile, the information point has all it needs to process the petition. The information point would then add to the user’s profile the object the user
is viewing and send it all off to the central server. There the petition would be processed, logged and returned to the information point, who would in turn send it to the client for viewing.

After viewing, the user can decide to search manually for more devices, or let the client application do it automatically. In any moment the user can change his profile if he finds that the information is too advanced or too basic, allowing futures views to be more in line with his own way of seeing things. This is the very idea of ubiquitous computing, as invisibly the system determines positioning, and without having to do anything, the user’s device gets in contact with the system for personalized information, in other words: the whole system, servers and own devices work together for the users needs.

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 [7]. The use of TCP/IP could help speed up the development as nearly every developer has experience with TCP/IP applications and the reliability that it gives. 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 high length of time Bluetooth can take in establishing a connection: up to half a minute between inquiry and connection (paging). If, on top of that, we set up the profile, the IP routes and gateways, and then do all the communication, too much time is lost. If it takes 20-30 seconds each time a client wants to view information, the user will most certainly want to stop using the system. Moreover the use of profiles disables inquiry and lower level Bluetooth operation. Therefore, if a user is connected and receiving information through an IP network by using one of the profiles it would be very difficult to find other servers. Trade off methods could be used but the profile would have to be modified to cope with SDP and inquiry. We therefore chose not to use profiles and to use directly the L2CAP layer.

The BlueZ distribution provides the socket API to L2CAP. L2CAP provides a reliable channel and uses segmentation and reassembly on ACL packets. L2CAP also multiplexes communications through the PSM 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. If a connection request is received on that port a new process is spawned to attend it and will receive a code operation (codop). Depending on the value of the codop, the central server will do one of two things: if the codop corresponds
to a configuration request, the server will send to the information point the contents of the configuration file objects. This file has an object name per line. All the objects (pieces of art) names that reside in the museum and for what the service are given, will be listed in this file. Each line will correspond to a directory. In each directory there will be a structure of subdirectories that will correspond to information formatted for different devices, in various languages and at different levels of detail.

The structure is replicated for each object and information is adjusted for each profile option. Once the museum information point has obtained the configuration file, the data server will disconnect, and the spawned process will end. If the codop corresponds to an information request, the server will spawn another process for it to receive the profile and object name. For each profile option received the server will acknowledge it to make sure errors are controlled. Once all the profile is received, the server will log the petition for security and for statistical information. This can help the museum to know which are the preferred objects, which have had more visits and routes followed throughout the museum, helping the internal exposition and organization. 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 resend 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. On top of this, what the central server actually receives is the object name which is added to the profile by the information point. The information to be sent back to the information point is the contents of the directory which in our prototype based on a laptop are two photos, one of the actual object in view and another one of the author, and three text descriptions. The text descriptions correspond to a description or interesting information of the object, another text will go over the life of the author and the third will be related to the actual position of the object in the museum, giving the client an idea of where he is and of where he can go. Once finished sending the data to the information point the central server will disconnect and the spawned process that dealt with the communications will end.

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.

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 petition. 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 server. There it will get processed and send back the information that will be 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 Museum Information Clients

The first step for each Museum Information Clients is initialization. The user has to fill in his profile in the “Change Profile” menu, and if he prefers to change the language in the “Change Language” menu. Once all the data is filled in, the application will search for information on what he views.

To do that the user must press the “Go Get Information” button. In that moment 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 user’s 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 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 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 few screen capture of the final prototype related, clockwise, to the client set-up, the MIP initialization, and MIC visualizing a data-block. ¹

4 Scalability issues

This section presents the evaluation of our proposed application in a small test-bed and by using simulation. The goal of our 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

¹The prototype application is available at http://www.grc.upv.es/software.html.
Figure 2: A few screens captures of the prototype application.

Bluetooth suited for context aware networking. Simulation allowed us to stress the behavior of the application in a scenario where a museum information point will have to connect with a large number of low-power clients.

4.1 Application performance evaluation using the test-bed

In the test-bed the museum data server and the a museum information point run in an Intel Pentium IV 2400 Mhz standard PC running Suse Linux 8.1 with a 3Com CREB96 Bluetooth USB Dongle card. The museum information client runs in an Intel Pentium IV 2000 Mhz TOSHIBA Satellite 1900-303 running Suse Linux 8.1 with a 3Com CREB96 Bluetooth USB Dongle card. We used the BlueZ [8] 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 focus on the inquiry establishment delay and throughput performance of the Bluetooth channel. We basically evaluate how distance affects inquiry delay and throughput. Intuitively we might expect that the inquiry delay will increase while throughput should decrease with distance due to the relation between errors and distance.

Inquiry establishment delay

Inquiry is an asymmetric mechanism in which two devices must be in two complementary state, namely: inquiry and inquiry scan states. The inquiring device continuously sends out the inquiry message. A node in inquiry mode goes forward until a previous specified number of devices in inquiry scan mode have been discovered or until a predefined max-
Figure 3: Cumulative inquiry delay distribution for the 100 real test. Distance between the museum information point and the museum information client is 1 m.

imimum timeout expire. The Bluetooth standard recommends an inquiry timeout period of 10.24 s. From the energetic point of view, the inquiry procedure is extremely important because power consumption increases during the inquiry and inquiry scan mode.

Using a distance of 1 m, the average inquiry delay is of 2.313 s ± 0.361 with a standard deviation equal to 1.8122 ± 0.258 with 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 2.033 with the same confidence interval. It seem then that the inquiry procedure is not highly sensible to distance. Figure 3 shows the histogram distribution of the inquiry delay and the cumulative results as function of time over 100 tests. The results show that after 2.455 s the museum client discovers the information point in a 50% of the tests. We also realized that only after 5.264 s and 8.236 s this percentage increases to 95% and 99% of the total number of test.

Throughput analysis

We now evaluate the impact of both the ACL packet type and the distance on throughput. Bluetooth offers “Asynchronous ConexionLess” (ACL) links for data transmission. An ACL link is parameterized by packet size and data encoding. Each ACL link allows the use of 1, 3, and 5 slots data packet. Additionally it allows the optional use of Forward Error Correction (FEC). 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. As described before, after the inquiry process is completed, the information point will collect all the parameters it needs from the client. After that, it will send the actual data-block to the client. The application data-block size averages 150 Kbytes and includes images and text. 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 4 shows the obtained experimental results when the information point side is using $DH$ packets and $DM$ packets respectively.

![Figure 4: Obtained application throughput comparison of an ACL link using different $DH$ (top) and $DM$ (bottom) packets as a function of distance.](image)

We first observe that Bluetooth offers a quite steady throughput below 10 m, independently of the data packet selected. Passing the 10 m limit the application can go on for further distances without a sharp performance reduction. When we select the more efficient $DH_x$ data packets, we improve the observed throughput with respect to the $DM_x$ packets. In the range of 10 m the $DH$ packets improve the $DM$ ones by 57%, 46% and 32% respectively for 1, 3, and 5 slots packets. The results confirm that although $DM_x$ packets tradeoff efficiency and probability of successful transmission, the $DH_x$ packets can be a candidate for more efficient transmissions. Moreover, after 10m, as distance increases, the more efficient longer packets ($DH_5$ and $DH_3$) suffer from a higher degradation. Finally, all the obtained results are below the maximum throughput provided in the specification. The Bluetooth standard provides the following reference values: 90.40kB/s for $DH_5$, 73.20kB/s, for $DH_3$ and 21.60kB/s for $DH_1$; 59.72kB/s for $DM_5$, 48.40kB/s, for $DM_3$ and 13.60kB/s for $DM_1$. Below 10m 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%.

### 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) [9] and the
BlueHoc [10] 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. Likewise with the test-bed experiment we test inquiry establishment delay and throughput performance. We test how the distance between MICs and MIPs devices can affect performance.

**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 m to 10 m. Figure 5 shows the inquiry delay variation with distance values of 1m and 10m respectively.

![Inquiry delay graph](image)

**Figure 5:** Inquiry delay for the 100 simulated test. Distance between the museum information point and the museum information client is 1 m (left) and 10 m (right). The Inquiry timeout value is fixed for all the test.

When we consider the 1 m scenario, the results show an average inquiry delay of 1.440 ± 0.0357 s. 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 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.
- The simulation results obtain a quite more smooth behavior that the real experiment. The standard deviation differs by far 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 now check how frequency offset can affect inquiry delay. We repeated all the simulation using a randomly selected frequency offset between the two devices. Instead of using a fixed offset, we modified the BlueHoc configuration file to randomly assign the InqScanOffset parameter. When the frequency offset is randomly selected and the distance between the two devices is 1 m, the results show an average inquiry delay of 1.516 s ± 0.147 and a standard deviation value of 0.738 ± 0.105 with a confidence interval of 95%. The results exhibit a higher variability than in the previous case. However, there are still some differences with respect to those obtained in the experimental cases. Figure 6 shows the histogram distribution and the cumulative results as function of time. The results show that after 1.506 s 50% of the simulated tests the inquiry device discover the inquiry scan device. We also realized that only after 2.456 s and 5.828 s the percentage increases to 95% and 99%.

The obtained results differs from those obtained by the real test, basically because:

- The Bluetooth layers model implemented could fail to produce realistic results.
- The data we obtain previously in the real experiment were affected because of the IEEE 802.11 AP presence.

Finally, when we increase the distance to 10 m, we obtained similar results. The inquiry procedure does not seem to be sensible with distance.

**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 7 shows the obtained
simulation results when using $DH$ packets and $DM$ packets as a function of distance. For each run, we average the throughput over the 100 s from the output.

![Graph showing throughput comparison](image)

Figure 7: Obtained simulated throughput comparison of an ACL link using different $DH_x$ (top) and $DM_x$ (bottom) packets as a function of distance.

The main difference with respect experimental results stands in the distance from where performance starts to decrease. We can observe how the average throughput starts to get worse at around 4m. 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 provide a quite high penalty from distances higher that 4 or 5 m specially for $DH$ packets.
4.3 Final comments

For those scenarios where distance stands lower that 4 m, the more efficient \( DH_x \) packets achieve a better performance than its \( DM_x \) 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 packets 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 m 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. The application also aims to help the maintainers of the museum, reducing costs in guiding their visitors, and keeping track of what are the preferred pieces of art.

We organized the network 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 edge network integrates one or more close-by users’ devices. From the user viewpoint, information points associated to any object are detected without user intervention, obtaining new information about what they are viewing.

We evaluated our proposed application in a small test-bed and by using simulation. The goal of the test-bed experiment was 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 stress the behavior of the application in a scenario where a museum information point will have to connect with a large number of low-power clients. We do think of Bluetooth as a outstanding contender among providers of network support system for context-aware applications.
Acknowledgments

We would like to thank Carlos Calafate for helping us setting up tests and solving some implementation problems. This work was partially supported by the Oficina de Ciencia y Tecnologia de la Generalitat Valenciana, Spain, under grant CTIDIB/2002/29.

References


