TECHNOLOGY FOR LEAST-COST NETWORK ROUTING VIA BLUETOOTH AND ITS PRACTICAL APPLICATION

Replacing Internet Access Through Wireless Phone Networks by BT Data Links

Hans Weghorn[1,2]
[1)]*BA-University of Cooperative Education, Rotebuehlplatz 41, 70178 Stuttgart, Germany*
[2)]*University of Ulm, OMI, Albert-Einstein-Allee 43, 89081 Ulm, Germany*

Keywords: Wireless communication, Bluetooth Communication, Least-Cost Routing, Mobile Information Access, Location-based Services.

Abstract: Today, mobile devices are equipped with a variety of wireless communication interfaces. While initially small handheld devices only could use cellular telephony networks for establishing data communication such as Internet downloads, nowadays data contents can be retrieved additionally through communication standards like wireless LAN or Bluetooth. For the latter there exists a variety of technical and scientific papers that discuss how Bluetooth communication can be established in principle – especially between two mobile devices. On the other hand, a description of how data communication between a mobile device and a desktop computer can be implemented is not found in detail. Furthermore, the restrictions of Bluetooth communication like extended search times are not discussed in these qualitative articles. In a technical description here, it shall be displayed how to establish with a minimal effort a streaming data link between handheld devices and integral computer systems in terms of a software implementation recipe. On base of concrete application samples, it is shown that Bluetooth can be employed to construct location-based information services with least-cost Internet data routing, but also the constraints and efficiency of Bluetooth communication technology are investigated and discussed for the given applications.

1 INTRODUCTION

From the beginning of their market introduction, available communication standards on mobile computing systems, especially on small hand held devices like PDAs or cellular phones, evolved very quickly. In addition to the original data communication through wireless telephony networks, complementary communication standards like IRDA, and few years ago wireless LAN (Riezenmann, 2002) and Bluetooth (BT) were introduced. Due to much lower power consumption (Mahmoud, 2003), BT has clear advantages over WLAN for constraint devices, and it is available nowadays in many of these small handheld terminals.

Possible improvements by adding BT communication elements to mobile device software applications appear rather obvious: For instance, the BT link could be used for a direct access to e-mail contents and documents on office computers, or another desire could be saving high data telephony costs by remotely using the Intra- or Internet connection of desktop computers from handheld devices via BT. The latter represents an approach of least-cost routing for data accesses.

Concerning development of BT application software for handheld devices, a broad range of technical and scientific articles exists. Java technology appears of special interest, because currently it represents the only manufacturer-independent common implementation standard for constraint devices. With focus on Java development, there are various papers available that introduce the methodology of Bluetooth programming (e.g., Hopkins 2005; Kumar et al., 2004) on small handheld devices. In these descriptions, the basic outline is always the same: It starts with the discussion of discovery of BT devices, continues with how to search for BT services (e.g. such as exchanging data objects like business cards), and finally provide some hints about using discovered services.

There, in the displayed samples for programming of wireless handheld devices, which is obtained more precisely with a system called wireless JAVA (Knudsen and Li, 2005) that is equivalent to the term Java Micro Edition (J2ME), usually communication

Weghorn H. (2006).
TECHNOLOGY FOR LEAST-COST NETWORK ROUTING VIA BLUETOOTH AND ITS PRACTICAL APPLICATION - Replacing Internet Access Through Wireless Phone Networks by BT Data Links.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - ISAS*, pages 394-402
DOI: 10.5220/0002469303940402

between wireless devices is shown, but knowledge about communicating between a wireless terminal and a desktop computer running a standard Java application remains on a vague level.

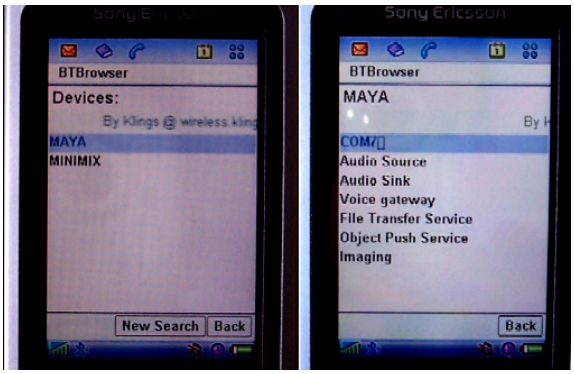

Figure 1: With a BT browsing tool the steps of BT device search (left part), and BT service search (right part) can be tested conveniently. The photographs show a J2ME PDA-like smart phone that was used for the experiments.

When starting own BT programming, it turns out that in these articles many parameters of library function calls are only described on a very qualitative level, and hence, a working system can only be implemented with an invest of own considerable programming efforts. Furthermore, the described theoretical concept of sensing BT devices and services represents not an appropriate one for true applications, because too much waiting delay arises for the end user in such a construction. Performance issues, which are of big importance when BT applications are going to be used in practice, are not discussed at all in these articles.

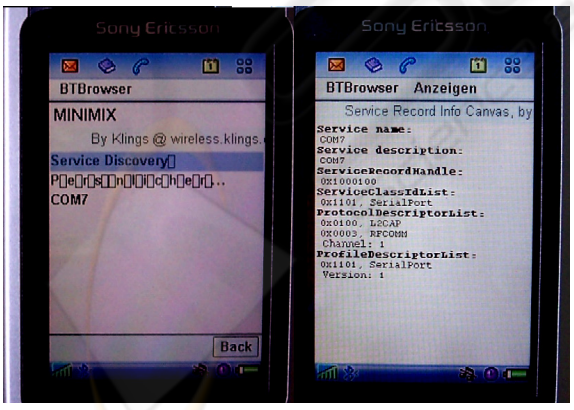

Figure 2: Furthermore, the BT browsing tool allows easily displaying information about the pseudo serial connection provided in this sample on host MINIMIX as COM7.

Since this multi-step procedure for establishing BT access appears rather complicate, even extended work is found that focuses only on developing a browsing tool for inspecting the environment of handheld J2ME devices (Klingsmann, 2004).

The scope of the description here is to provide a recipe how a streaming connection between a mobile application and a desktop computer can be implemented. All relevant technical details shall be shown here for achieving this aim with minimal effort. Due to the high number of alternative programming systems, the description only can be given for one dedicated technology, i.e. Java. Additionally, the performance and the restrictions of such BT connections operated on true-world devices are reported. On base of practical applications, it is discussed what the achievable benefit of adding BT communication can be, and what the actual constraints are in particular for mobile information systems.

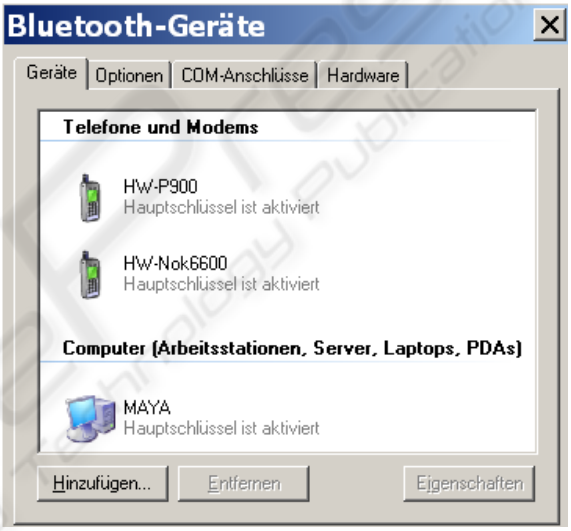

Figure 3: The state, whether close-by BT devices were already registered for a data exchange by the pairing procedure, is displayed in OS standard menus.

2 ESTABLISHING TECHNICALLY A BLUETOOTH STREAMING CONNECTION

If Bluetooth shall be employed for replacing a costly Internet link via the cellular telephony network, different components are required in the system: The mobile client has to establish a streaming connection via BT to a relaying host that is connected to the Internet. On the relaying host, service software is required, which forwards the download query from the mobile client to the Internet, and then transmits the response back via BT to the wireless terminal.

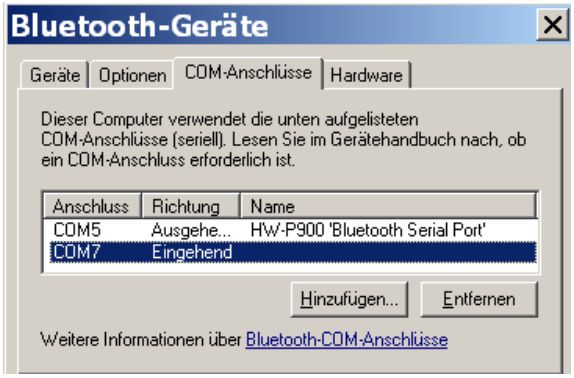

Figure 4: Another BT configuration menu of the OS allows configuring pseudo serial ports. In the sample here of the machine MINIMIX, COM7 is defined as server port.

As with the introduction of all recent communication standards into the desktop computer world, also BT data links were implemented together with simulated serial interfaces (similar to the RS232 standard), which are called pseudo serial connections. After binding two BT devices together (this is called "pairing"), which is handled automatically on BT partners by instances of the operating system that ask for manual input of a secret key on both hosts, pseudo serial ports can be configured for the communication between the two. Such pseudo serial ports are feasible for streaming connections in application layer software, and two operational modes are available for them: A server and a client port type (Fig. 4).

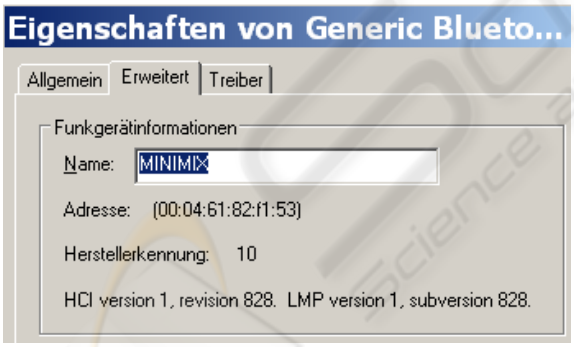

Figure 5: The hex value of the BT hardware address (MAC) can be read in the appropriate configuration menu.

The computer system, which should operate the BT server port in this scenario, obviously is the desktop computer, because only there the appropriate service and the behind-lying software (e.g. a proxy gateway) can run continuously without restrictions. For mobile devices, it would be almost impossible running continuously such a server tool, because the devices often shall be used for other purposes like making phone calls, and the running server software would discharge the battery of the device very quickly and by that reduce the so-called stand-by time to an unacceptable degree. Therefore, in the further sections only the communication mode is described of having the BT client on the mobile device (section 2.1), and the BT server on the desktop computer (section 2.2).

For completeness, it shall be mentioned that a server could as well be implemented on the mobile device. Due to the explained reasons, this should be started only when it is needed, but the consequence would be more uncomfortable handling of the communication system than with a server tool hosted on the desktop computer, which can be launched automatically and runs continuously as daemon task.

2.1 BT Serial Communication on a Mobile Device

In J2ME, there exist one unique class named *Connector* for the construction of any type of communication link. During initialization, three parameters are required, which are the protocol type, the target address, and optional parameters. For a BT pseudo serial link, these are defined like follows:

```
String proto = "btspp://";
String btmac = "00046182F153";
String params = ":1;authenticate=false;
encrypt=false;master=false";
```

It is important to observe that the only valid and working address is the BT hardware device MAC, which is coded as 12 digit hexadecimal number string that can be determined by standard operating system menus on the target computer (Fig. 5). The first parameter entry of "1" stands for the first BT pseudo serial server port on the host (e.g. COM7 like in Fig. 2, Fig. 3 and Fig. 4), and the client connection mode is commanded by the parameter entry for "master=". Obviously, there is no directly deductible relation between the name of the pseudo serial port on the host computer, and the opening parameter entry for the *Connector* class.

Whether the communication can be encrypted depends on the pairing relation between the host and the client, but there is no guarantee that this is served in actual J2ME environments. Despite what the authenticate parameter value connotes, the operational system layers ensure the proper authentification of the involved BT devices. With this all, a streaming connection can be initialized like follows:

```
StreamConnection con = null;
String conURL = proto+btmac+params;
String rdname = "unknown";
try {
```

```
  con = (StreamConnection)
    Connector.open(conURL);
  RemoteDevice rd =
    RemoteDevice.getRemoteDevice(con);
  rdname = rd.getFriendlyName(true);
} catch (IOException ioe) {…}
//System.out.println (rdname);
```

Through the appropriate static system function, a *RemoteDevice* object can be constructed, from which additional information like the textual name of the remote BT host can be retrieved.

After successfully opening the streaming connection, J2ME standard handling objects for input and output activities on this stream can be derived:

```
OutputStream os = null;
InputStream  is = null;
try {
  os = con.openOutputStream();
  is = con.openInputStream();

  String msg = "HELLO "+rdname+"\r\n";
  os.write(msg.getBytes());
  os.flush();
} catch ( IOException ioe ) {…}
```

On base of these I/O objects, all the standard functions for output (see above) and input can be used, for instance in an echoing loop that immediately sends all received characters back to the BT partner:

```
try {
  int nxt = 0;
  while (true) {
    if ( is.available() == 0 ) {
      // sleep a while
    }
    else {
      nxt = is.read ();
      if ( nxt == -1 ) break;
      switch (nxt) {
        // apply some translation
        // for dedicate control
        // chars such as line break
      }
      os.write (nxt);
      os.flush ();
    }
  }
} catch ( IOException ioe ) {…}
```

As seen with the above coding lines, opening a bidirectional streaming link in J2ME applications on mobile devices can be a very simple-styled process, if the proper parameters are known for all the required function calls. This sample works for low communication speed, but as discussed in section 3 for fast data exchange additional issues have to be regarded.

There is no requirement for applying searching mechanisms for the BT host or for the BT communication service before applying this code like it is suggestively described in all the articles about BT programming in J2ME (e.g., Hopkins, 2004). If the addressed host is not found in the BT micro cell, the *Connector.open()* call returns the appropriate error. This simplifies the software code considerably, but of course, the BT MAC address must be available in the mobile software.

2.2 Serving the BT Serial Communication on a Desktop Computer

A straightforward possibility of serving serial ports on desktop and workstation computers is programming in native development environment of the given operating system. For UNIX hosts, this would be C programming and for a desktop operating system like MS Windows, this also would be C/C++ development, but in both cases the appropriate system libraries would be used. BT pseudo serial ports are accessible in these environments in an identical manner like hardware RS232 interfaces with the exception that interface control parameters, e.g., for communication speed and handshaking modes, are disregarded. Since many projects in education and University environments are based on Java development, detail implementation hints shall be given again here for this programming environment (Bell and Parr, 2002).

Handling of hardware interfaces, and in particular serial communication ports, is not contained in the standard Java programming package. Extensions for supporting hardware like audio devices were added in additional standardized class libraries (javax.*), and these are available for use. Due to its inconvenient properties, serial communication still does not fit in to this unified frame, and a special software tools set has to be installed on the executing computer, because an interfacing is required between native device handling on the operating system of the host and the virtual computer that executes the Java application.

Serial support comes in terms of a bundle called *javacomm*, which contains a class library for the Java system, and for MS Windows environment an additional library (DLL) and a description file for the native interfacing, which have to be installed in the Java run-time directory tree. The *javacomm* bun-

dle contains documentation how the operational environment has to be set up, and source code of application samples for serial communication.

On base of this all, a communication application can be implemented, while only the opening of the device is a procedure specific to the serial interface communication. After opening a serial port, the Java standard streaming objects can be derived for programming input and output actions on the device:

```
import javax.comm.*;
…
String portName = "COM7:";
CommPortIdentifier portId = null;
try {
  portId =
  CommPortIdentifier.getPortIdentifier
  (portName);
} catch(NoSuchPortException nspe){…}

if ( portId.getPortType() != CommPor-
tIdentifier.PORT_SERIAL ) …

SerialPort sPort   = null;
String     myname  = "me";
int        timeout = 2000; //ms
try {
  sPort = (SerialPort)
    portId.open ( myname, timeout );
} catch (PortInUseException e) {…}

OutputStream sos = null;
InputStream  sis = null;
try {
  sis  = sPort.getInputStream ();
  sos  = sPort.getOutputStream ();
} catch ( IOException ioe ) {…}
…
```

Around this, e.g., an Internet relay for BT downloads can be developed straight forwardly.

2.3 Chat Tool as Implementation Sample for BT Communication

The above described client software code lines are an excerpt out of a chat tool, which was developed for execution on J2ME-enabled handheld BT devices. In a first approach, there is no own application required on the host side, because a standard terminal application can be configured for using the appropriate pseudo serial port (Fig. 6).

Figure 6: Standard terminal emulations can serve a BT serial connection (here the pseudo port is COM7): The first three lines are output sent from the wireless terminal, while the below two lines were entered manually on the desktop computer's keyboard. These are only visible because of the echoing function of the wireless chat tool.

After the connection between the handheld chat tool and the terminal application on the host is established, all entered input on the host terminal is displayed on the mobile device, and echoed back to the host (Fig. 7). Only by that, the user can read his/her input on the desktop computer. For the proper display of line breaks, all special control characters are translated in to the proper control sequence or actions for the terminal display, and for the J2ME display respectively.

Figure 7: The BT chat client outputs some debug information, before it prints out the input entered on the keyboard of the remotely-connected host MINIMX.

The J2ME chat tool uses as output UI element a StringItem. The StringItem contents are updated with each input from the terminal on the remote host. If the enter key on the host computer is pressed, in the J2ME StringItem a new line character is added, while a CR-LF combination is sent back for the proper display on the host. If the FF (form feed) code is entered on the host terminal, the con-

tents of the J2ME StringItem are purged completely, and the erase code is sent back.

In additional experiments, the terminal application executed on the host was replaced by automated Java software that serves the pseudo serial BT interface. With this, response and turn-around times were measured, and the BT chat tool was used as predevelopment state for the Internet proxy used in the application of section 4.

3 TECHNICAL RESTRICTIONS IN J2ME BLUETOOTH PROGRAMMING

3.1 Time Consumption of Data Communication

Generally, setting up a Bluetooth (BT) communication link consumes time, which is in an order that is well noticed by the user. If experienced often, this delay may even be encountered as disturbing. In particular, the search for close-by BT devices and services may take up to minutes.

If the target BT host is known, directly opening an above described serial connection requires approximately one second only, but until the querying system decides that the desired communication partner is absent, several seconds of waiting delay will elapse. Within this time span, in many applications the required information packet could have been transmitted already through the phone network, and the question arises whether BT overall is the best data link for the application.

When such an information query often is repeatedly used, the described pseudo serial communication makes only sense, if the direct communication partner is know and does not need to be searched for each time. Hence, an information tool on the client has to be structured into two parts: One branch should offer entering configuration, through which a BT communication partner is defined, and this branch is used rarely. The second branch automatically uses the configured BT communication partner for an information access.

During the implementation of the BT communication link, it turned out that when sending from the BT host to the mobile device the connection easily was overran, and transmitted information bytes were lost completely. Neither in Java environment, nor in C programming environment a working handshaking was available for the pseudo serial communication. Hence, the transfer had to be secured on the application layer. For this, the simplest approach is sending always one information byte and waiting for an echo response from the wireless device before sending the next byte. This concept was working reliably, but it slowed down the communication extremely.

For chat applications with relatively slow manual user inputs, this does not represent any issue, and for an also investigated automated information system, which downloads data from the open Internet, the overall access time was in comparable order like for GPRS access.

The exchange data rate can be increased by transferring not only single but a set of bytes on each transmission, but with bigger packets than 4 bytes, unavoidable data loss was observed again. A negotiation of the best transfer size is presumed producing an overall increase in transfer time, because the exchanged packets are small. Single byte transmission is working, but effectively it is unsatisfactory, and a reliable faster communication has to be developed, because the BT communication would be useless for other applications than plain textual communication, e.g. transferring pictures for a weather forecast, or checking student IDs for examinations by displaying their photographs on the handheld device.

3.2 Location Sensing

Opening a BT link from a J2ME application on a wireless device theoretically can be performed for different communication partners simultaneously. When this kind of method is used in practice, it has to be observed that only the last initiated connection will be preserved by the operational software. Again, the finding is that what is working in Sun's simulating environment for J2ME-enabled devices does not work on real devices. Hence at the moment, BT location detection makes only sense with one fixed communication spot, because searching for many BT partners would require a sequential query process that would consequence too long a waiting time for the end user. Already this possibility introduces a true benefit, because if the defined communication partner for a PDA is, e.g., the user's office PC, this scenario can detect the location of the user, and it can use least cost information access through the office computer for retrieving contents from the Internet or Intranet, such as e-mail, news, traffic messages, and other.

3.3 Uniqueness of BT Identities

When opening a BT connection, the primary access address that is used is the MAC value. This hardware address is unique, since it is bound physically to the hardware device. If removable BT communication dongles (e.g. with an USB interface) are used,

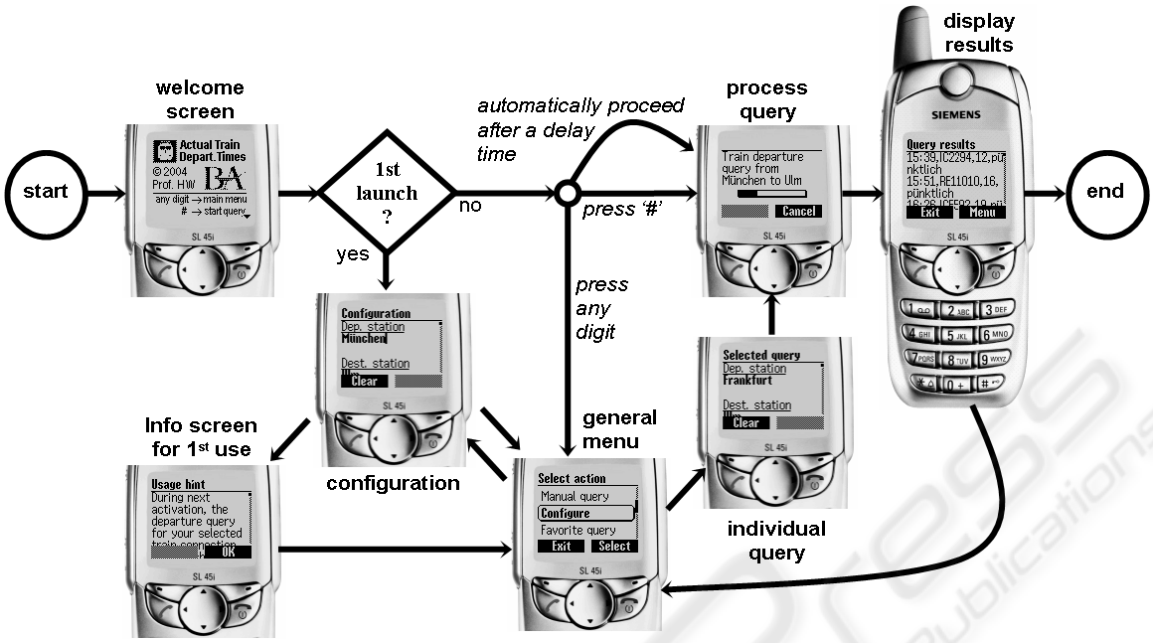

Figure 8: UI chart of the information tool on actual train departures for commuting people. Without user input, the query process automatically proceeds along the line from start to end.

the BT interface hardware easily can be attached to different computer systems. In this case, the presence of a BT communication partner could be misinterpreted in the before discussed location-based approaches.

In the SW samples of section 2, it is seen that after establishing a BT connection, the retrieval of a name identifier string can be commanded. Unfortunately, this name string is not unique, because it can be set manually (Fig. 5). If a unique identification is required in a BT application, the host identity has to be checked and ensured on application layer. Due to the mechanisms discussed before, this method would manifest the only truly reliable method feasible for the exchange of security relevant or confidential information. For simplicity of implementation of other mobile accessible information systems, the rule could be not to exchange BT transceiver hardware between different host systems, because else the information system could end up in a confused state.

3.4 Summary on Restrictions

Overall, it has to be stated that the practical BT software implementations face a list of severe restrictions, which prevent the implementation of an optimal concept for wireless BT access. On the other hand, with the working features it is already possible to improve existing information systems considerably, because technology concepts like location sensing for location-based services (Schiller and Voisard, 2004) can be brought to life without fulfilling additional requirements, e.g. having a GPS receiver hardware installed. Special care has to be addressed to the question how many locations can be detected, and how a reliable detection is ensured. The achievable improvement range with these new possibilities shall be demonstrated with the sample application in the next section.

4 LEAST-COST BLUETOOTH INTERNET ROUTING FOR A MOBILE RAILWAY DEPARTURE INFORMATION SYSTEM

In the recent years a mobile information system was developed for people commuting between their living home and their working place (office, school, University, …) by train (Weghorn, 2004). The railway company in our country provides on the open Web departure and arrival tables for each main station, but for a commuting person it does not make sense querying these directly from a mobile terminal, because these tables contain much more information than is required.

The system concept is like that the user configures a certain connection link by defining the train stations close to his living home and the working place. During a departure query, the mobile device's software contacts an intermediate agent with the parameters of the traveling link and this agent mines the desired contents from the general tables on the Web. Finally, the agent transmits only relevant information about time departures (for instance delays) or additional information (e.g., change of platform) to the wireless device, where the result is displayed (Fig. 8).

It was reported already that this query via a data link through wireless telephony networks takes between 3 and 40 seconds, and that the costs for one query range in the order of 10 to 40 Eurocents. These absolute values depend on the network operator and on the tariff of the customer, while the experiments showed that best performance is achieved with GRPS data links: Setting up a first GPRS connection including access to the departure information usually doesn't require more than 10 seconds, while each further access needs only around 3 seconds.

For the users' best convenience, the query automatically is processed without any necessity for user interactions once the tool is launched (Fig. 8). After launching the SW tool, a welcome screen is displayed, during which the user can interrupt the regular query and enter a main menu. In this main menu the commuting link can be defined, and individual queries can be placed.

The application suffers from one free parameter, which is the travelling direction. This cannot easily be determined automatically without further requirements. There would be a possibility of using network cell broadcast messages for obtaining the location of the user and by that the entrance station, but although the access to cell broadcast is defined since many years in J2ME specifications it is not working on true devices (Weghorn, 2005). In the application, the problem was overcome by defining a switching time (e.g. 1:30 pm), when the system automatically assumes that the user has changed between going-in and going-home. This is a feasible but not always accurate measure.

BT technology can now help improving this system in two ways: It provides a limited location sensing, and it allows replacing the costly download of Internet data through the telephony network link by an almost costless BT data communication. For achieving both, the office computer of the user shall be defined as sense key: When a departure table query is launched, the information tool tries to directly open a BT pseudo serial link to the office computer. If this succeeds, the information system assumes that the user is on the way home and sets the appropriate travelling direction, and furthermore it downloads the information directly through the BT link. If opening the BT link fails, the query tool continues with the original procedure, i.e. using the wireless phone network for the data communication. The BT query can already be performed in parallel to the welcome screen (Fig. 9).

As convenient time for displaying the welcome screen, a value between 1.5 and 2 seconds was experienced. The BT query – in case it fails – will extend this screen display to a duration of approx. three seconds. The benefit of automatic location detection and saving of costs for phone network use has to be paid back by this extended delay. If desired, this small inconvenience can be overcome by adopting the function of the input buttons during the welcome screen: In case the '#' button is pressed, the system can proceed automatically with the original method without further searching for a possible BT connection. Pressing any of the numeric digits will still bring the user to the individual handling menu of the information tool.

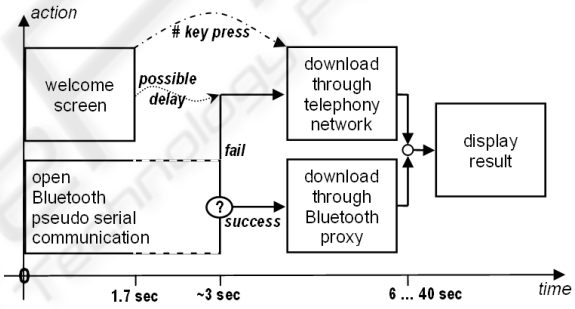

Figure 9: Process flow for automatic Bluetooth routing.

The extension of the train departure query system shows, how BT communication can be used in parallel for two improvements, in particular by introducing:

- Geographical location-sensing by proximity detection
- Least-cost Internet information routing

It has also to be stated clearly that an optimum system cannot be constructed, because the absence detection delay is a fundamental BT property and cannot be reduced. Furthermore, parallel sensing of alternative BT hosts like other computers close-by the office area, and a home computer cannot be obtained without introducing additional unacceptable waiting delays for the customer. Hence, a notable improvement of this information channel for mobile users can be achieved, but not to the full theoretical extend.

These benefits would also be applicable to most of the other personalized wireless information sys-

tems (Weghorn, 2004), which we developed in the recent years, like traffic and weather channels.

5 CONCLUSIONS

In the work reported here, a technical programming concept was developed. This describes how information access from handheld devices, which usually use for this a wireless phone network connection to the Internet, can be replaced by Bluetooth communication.

Doubtlessly, the best solution would be, if an automatic rerouting is implemented in the operational software layers of the wireless device instead of the application software, since BT provides a dedicate service for using the Internet connection of other hosts. Unfortunately – although this would be technically achievable – other samples of missing possible features, e.g., making direct calls between phone terminals via BT that is also within the standard since years (Miller, 2001), show us that this is not very likely to happen soon. Obvious economic interests of wireless telephony network operators prevent this advance for the end customer, and since these operators are sponsoring to a large extend the purchase of wireless devices, there exist good and valid reasons for this strategy.

Fortunately, if an information tool shall be optimized in terms of cost efficiency by using BT communication, a direct programming of the rerouting is possible, and works like the samples of the chat client and the railway departure information system show. Technical restrictions in the implementation of BT in true-world wireless devices also prevent an optimal application of all potential benefits of BT data links. Especially the train information system demonstrates that already at the current state of technology clear advantages arise for the end user, because other systemic drawbacks like the well-defined but non-functional location sensing (Weghorn, 2005) can be replaced by BT sensing to a helpful extend.

As first advance, information access costs can be reduced, and secondly the query tool on the wireless terminal can be used everywhere seamlessly, so there is no need for changing between the wireless terminal and a desktop computer depending on the location, where the user wants to run the information retrieval. As seen from the above photographs of true-world devices, this all was experimentally verified with end devices from different manufacturers. Concluding, it should be noted that the current implementation of the pseudo serial communication via BT does not easily allow a fast data stream, but in the end this fits well to the before developed customized information system that transmits only few relevant contents through the wireless link to the mobile terminal (Weghorn, 2004).

REFERENCES

Bell, D., and Parr, M., 2002. *JAVA for Students*, Prentice Hall. Dorchester, 3rd edition.

Weghorn, H., 2004, Personalized Information Retrieval System for Improving the Use of Data Services through Digital Wireless Phone Networks, IADIS International Journal on WWW/Internet, ed. P. Isaías, Volume II, No. 2, 43-56

Weghorn, H., 2005, Multi-Sourcing Web Services for an Improved Access to Web Contents, in *WWW/Internet 2005*, IADIS Proceedings, Vol. II, 226-231, Lisbon

Hopkins, B., 2004. Getting Started w. Java and Bluetooth. http://today.java.net/pub/a/today/2004/07/27/bluetooth.html, last access Oct. '05.

Klingsmann, A. N., 2004. *J2ME Bluetooth Programming*, Master's thesis, Department of Informatics, University of Bergen. July 2004.

Knudsen, J., and Li, S., 2005. *Beginning J2ME: From Novice to Professional*. Apress, Berkeley.

Kumar, C., Kline, P., and Thompson, T., 2004, *Bluetooth Application Programming with JAVA APIs,* Morgan Kaufmann. San Francisco, 1st edition.

Mahmoud, Q. H., 2003. Wireless Application Programming with J2ME and Bluetooth. http://developers.sun.com/techtopics/mobility/midp/articles/bluetooth1/, last access Nov. '05.

Miller, B., 2001. Future Applications for Bluetooth Wireless Technology, in *Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communications*, Prentice Hall. 2nd Edition.

Riezenman, M. J., 2002. The ABCs of IEEE 802.11. IEEE Spectrum Online, September 2002. http://www.spectrum.ieee.org/WEBONLY/resource/sep02/802ABCs.html, last access April '05.

Schiller, J., and Voisard, A. (eds), 2004, *Location-Based Services.* Morgan Kaufmann Publishers, San Francisco, USA.