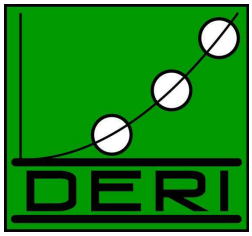
University of Innsbruck

Digital Enterprise Research Institute



Testbed for Mobile TSC

Bachelor Thesis

David Hellekalek

Rechengasse 7

A-6020 Innsbruck

Matrikelnummer: 0315772

Supervised by Reto Krummenacher

Innsbruck, June 2006

Abstract

The topic of this thesis is to investigate what is needed to make an inhouse navigation system for mobile devices, thereby demonstrating an application for mobile TSC.

This thesis mainly evaluates the technologies and possibilities and is to be seen as a preparation for a possible implementation.

Contents

1 Introduction **2**

2 Background **6**

3 Use Case **10**

4 Requirements Analysis **13**

4.1 From the Use Case to the Requirements 13

4.2 Additional Requirements "on site" 16

4.3 Conclusion 17

5 Technology Survey **22**

5.1 Mobile Phones 22

5.2 Personal Digital Assistants 26

5.3 RFID Readers 29

5.4 Discussion 30

6 Conclusion **33**

Bibliography **36**

Chapter 1

Introduction

The topic of this bachelor thesis is to evaluate the possibility of making an infrastructure for mobile devices, based on J2ME, TSC (Triple Space Computing) and YARS. The purpose of that infrastructure would be aiding people to navigate inside complex buildings. A detailed explanation of the terms J2ME, TSC and YARS can be found in Chapter 2 (Background).

The idea of making an inhouse navigation system for mobile devices is very suitable for demonstrating the possibilities of combining TSC with mobile computing. The triple space holds all necessary navigation information and additional information which other devices like sensors store there. The application combines and presents that information to the user on a mobile device. This aids the user to find his way through the building, and even allows giving the user additional relevant information. Such mobile assistance through networked computing can be counted towards the fields of ubiquitous or pervasive computing [8]. In the field of ubiquitous computing, devices usually communicate over a mobile ad-hoc network and share data between each other to an extent that they can be called a distributed system.

Part of the work is a use case definition which demonstrates the basic ideas of how the navigation system should work to the reader and serves as a red line for the requirements analysis. The very basic idea is that somebody visits an office building. He needs to find a person or a room and uses the mobile application to do that. In the use case it is shown what role TSC plays for the navigation system.

The next part of this thesis is the requirements analysis. The requirements concern both software and hardware. The software must be able to do what the use case requires, and the hardware must be able to run the software and all its functions like communication and storage.

This leads directly to the next part of the work, the technology survey. It has to be determined what hardware meets the requirements and how much it costs. It is also necessary to find out what additional possibilities some hardware can offer. For example devices with big displays sure have a better usability than those with small displays, but presumably for a higher price.

Goals

The goal is to find out the basic outline of how a possible implementation of the navigation system could look like and what is required for it. This can be separated into three parts.

- Use Case Definiton
  - Basic outline: Somebody visits an office building (the ICT building) and needs to find somebody. The mobile application assists him.
- Requirements Analysis
  - What must the software be able to accomplish?
  - What hardware does that require?
- Technology Survey
  - What hardware meets the requirements?
  - How much does it cost?
  - What additional possibilities does it offer?

Methodology

The main part of the work is gathering information for the requirements analysis and the hardware survey. Examples are reading about specifications of the software that is used, i.e. the J2ME runtime environment, and getting information about mobile devices to see what they have to offer.

The first step will be the use cases which needs to be specified clearly. A clear use case definition is absolutely necessary for a good requirements analysis. The use case must include all important parts of what user and application do. The use case is as well essential for explaining to the audience what user and application are expected to do.

When this step is done, it is time for the requirements analysis. This concerns requirements what the software must be able to do, as well as hardware requirements. The requirements will be deducted from the use case. This means that for each action that the user or the application performs, it will be decided what the requirements for doing this action are. The result should be a detailed and precise list of requirements.

Following this, the next step is finding out what hardware meets this requirement catalogue. The major part of that will be checking sites of manufacturers of mobile devices to see which devices meet the requirements. The second step will then be finding out how much they cost, what the trade-offs for cheap devices are and so on.

Some topics, like how to use RFID technology in a J2ME program, will be more difficult than others. Some of them will still be very interesting and play an important role.

Chapter 2

Background

To understand this thesis it is necessary to have basic knowledge about the technologies that are used. They are explained in this chapter.

J2ME is the Java 2 Micro Edition (http://java.sun.com/j2me/). It is an implementation of Java for devices like mobile phones and PDAs. Due to the limited resources of those devices, it has fewer capabilities than the J2SE (Java 2 Standard Edition, which is for desktop computers and is the kind of Java that is most commonly used for client systems). A J2ME environment is a combination of a J2ME Profile and a J2ME Configuration. The configuration defines which capabilities the Java Virtual Machine must have and defines some basic libraries. The profile defines APIs for certain tasks like for example the display of graphics. Those which are most common for modern mobile devices are MIDP (Mobile Information Device Profile) and CLDC (Connected Limited Device Configuration), either in the combination CLDC1.0/MIDP1 for devices that were released several years ago, or CLDC1.1/MIDP2 for recent devices.

Another abbreviation that will be mentioned later are JSRs. JSR stands for "Java Specification Request". JSRs are made to change specifications or add a new API. For example for allowing J2ME to use technologies like Bluetooth or RFID, JSRs were made.

TSC stands for Triple Space Computing. A triple space is a space, comparable to a memory space, which is shared between several hosts. The idea behind TSC is that it offers persistent access to information. Since the information is spread between many hosts, it does not get lost when one host goes offline. TSC is based on RDF-triples, therefore the name. Its basic operations are reading and writing triples. Therefore TSC can be used asynchronous. It is not necessary for two processes to exchange data directly, one process writes, the other process reads the data when it needs it. This also makes it easy to share data with multiple processes and reduces overhead. [4], [6]

Since all nodes that participate in the space are server and client at the same time, it has a peer-to-peer network architecture. A node of the network can be programmed to act as a proxy. This means that a computer outside the space can send a request for information which is in the space to the proxy. The proxy then retrieves the information from the space and sends it back to this computer.

Therefore the possible application that is discussed in this thesis can be either a Light or a Heavy Client. A Light Client gets data from the space via a proxy and does not participate in the space itself. A Heavy Client integrates itself into the Triple Space which means that it must be able to

accept incoming connections of other nodes and store data. The result is that the hardware requirements for a Heavy Client are higher than the hardware requirements for Light Clients.

YARS [5], "Yet Another RDF Storage", is an RDF database which allows storing and retrieving RDF triples and is implemented in Java (J2SE compatible). It uses Notation3 (N3QL) [3] for querying information. For storage it adds a context to the RDF triples (subject, predicate, object), therefore storing quads that consist of subject, predicate, object and context.

Connectivity is important as well as it is necessary to communicate with the Triple Space. Since the topic is "**Mobile** TSC", wired connections are not feasible. Mobile devices are usually connected to the Internet or an Intranet through wireless technologies like WLAN (802.11b/g), UMTS, EDGE, GPRS and HSCSD. In theory they can all fulfill the task of communicating with i.e. a server. It is necessary to distinguish between GSM-based technologies where the user is charged by his operator and WLAN which, in the case of this project, can be offered on site and does not cost the user any money. 802.11b/g is WLAN, the rest (UMTS, EDGE, GPRS, HSCSD and plain old GSM) all run via mobile phone operators who charge the users for using it. 802.11g WLAN supports up to 54Mbit bandwith and is compatible with 802.11b hardware, which only supports up to 11Mbits. GSM is the basic data transfer service that most mobile phones with WAP Browser have. GPRS and HSCSD are faster services that are based on it and in the case of GPRS, the user is able to be constantly online. This is due to GPRS being packet switched (user pays for transfered data), while GSM and HSCSD are circuit switched (user pays for the connection). EDGE is improved GPRS.

UMTS is the next generation of mobile communication, its data services can be described as "like EDGE but again faster".

Additional technologies: Use of technologies like for example RFID or Bluetooth for making the devices location aware will be considered. RFID (Radio Frequency Identification) is a technology that is based on chips which send out information like their ID. By comparing a received ID with a database (the triple space) which maps IDs to locations, the device can determine its own location.

Bluetooth is another wireless technology which usually has more range than RFID (5-100m while RFID usually has a range of only a few meters). Bluetooth is mostly used for data exchange and not for identification, nevertheless each Bluetooth device has an ID that can be used in a way similar to how RFID can be used.

It was decided to use J2ME since the code for YARS is written in Java and part of the idea of the Heavy Client is making a mobile counterpart to it. Besides that, J2ME is the only platform that offers portability between many kinds of mobile devices.

Chapter 3

Use Case

Alice, an informatics student at the University of Innsbruck, has an appointment with Bob who works at the University of Innsbruck. He is working in the ICT building, a big building with lots of corridors and a confusing room-numbering system. Alice is very confused by the complicated building plans and is a bit nervous already since she is running out of time. The meeting is scheduled for 13:00, it is that time already and she just cannot figure out where she has to go. Luckily Alice has a PDA with her. She sees a URL next to the building plan that is near the entrance on the ground floor. The URL is also printed there as a barcode so that it can be read by devices that are capable of this. She opens up the Web-browser, points it to that URL, and downloads the navigation utility via a wireless connection. This wireless connection is following the 802.11g standard and the access point for it, is run by the ICT administration. Alice starts the application and makes it aware of her position. Since her PDA is able to communicate with RFID chips that are placed everywhere in the building, this is done automatically: The application gets an ID from the closest RFID chip, and queries the according location from the Triple Space. In the background, the application does even

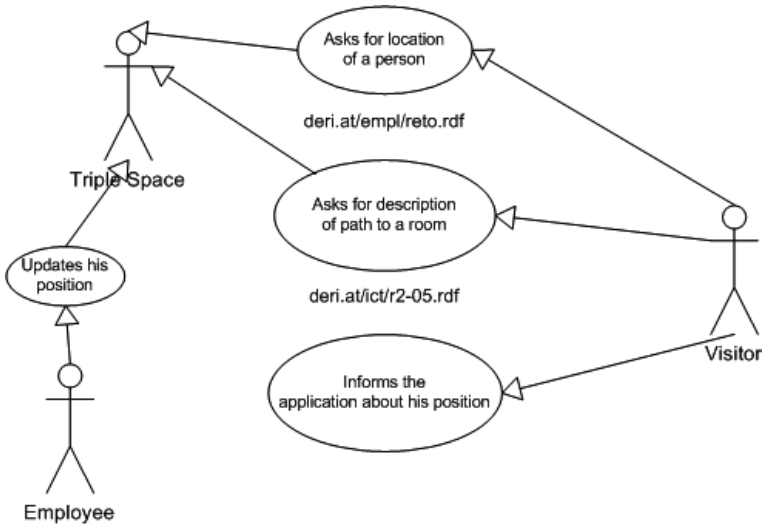


Figure 3.1: Use Case

more. Alice downloaded the heavy client which now integrates itself into the Triple Space. If somebody else would query the space now, it might be that it is Alice's PDA that answers. Next she wants to ask it for the location of Bob. She enters the name in the search and the application queries the Triple Space for information about Bob. Among this information is his location. The application answers the query by showing the room number and the floor where it is. It also shows when Bob's location was updated the last time so that she can estimate the reliability of the information. There even is a light sensor in Bob's room which writes its readings into the Triple Space. The application reads that there is light in the room, another good sign for Bob being there. Besides that, the application offers to show a description of the path to that room. The visitor chooses to have a look at the description. She gets a description at which level to exit the elevator, and which way she needs to walk to find that room, starting at the exit of the elevator. Alice follows the instruction and easily finds Bob.

A few minutes before what the plot above tells, Bob had lunch and then moved to his workplace to wait for Alice. When he logged on to his computer, a small program running on that machine automatically updated his position by storing his new position in the Triple Space. If he had not logged on to his computer this would not have been a problem though. In the ICT building, keycards are used to access rooms and when Bob used his to open the room, the existing server that controls the door-locks sent his new position to the space as well.

Michael has an appointment with Bob as well. He also has problems finding the correct room and its location. He does not have a PDA, but a high-end mobile phone instead, more precisely a Nokia 6230i. He is unable to access the WLAN that is accessible in the building since his phone has no WLAN hardware. Fortunately his phone has a WAP-Browser and can communicate with other servers via GPRS. He also sees the sign with the URL and the information about the navigation software. He visits the site and downloads the application to his phone via WAP. Since it is unable to communicate with RFID chips, he selects his position from a list. His position, ground floor, in front of the elevator, is the first in the list anyway so it is done quickly even on a device with a not very user-friendly input like his mobile phone is. The application tells him that Bob is in the room 1W03 on the first floor. He asks the application to display the description of how to find the room since the room number does not mean much for Michael. He walks the way that the instruction told him to, and gets to the right room.

Chapter 4

Requirements Analysis

As it was decided to program the clients in J2ME, the key to precise requirements is finding out what device configurations and profiles are required. For mobile devices, those are CLDC (Connected Limited Device Configuration) and MIDP (Mobile Information Device Profile). Of both specifications, several versions exist so it is necessary to find out what version is required. Although CLDC implies that the device has some kind of connectivity, it does not say anything about the type of it. Nor do MIDP/CLDC versions say anything about features like Bluetooth or RFID. Therefore it also has to be checked which requirements come up, which are not covered by certain versions of MIDP/CLDC.

4.1 From the Use Case to the Requirements

In the use case, the user downloads the application from a Web-server. The first requirement therefore is the ability to get the navigation application onto the device. Via Web is most comfortable but there are other possibilities as well, i.e. Bluetooth or Infrared. The possible solutions for this are

that the device has a Web or WAP-Browser or is able to accept and install applications that are sent via Bluetooth or Infrared.

The next thing that happens in the use case is that the user makes the application aware of his position. In the use case this is done automatically via RFID but it could as well be done via Bluetooth or manually. Therefore RFID- or Bluetooth-based location awareness is optional. The solution to this requirement would be devices with RFID or Bluetooth support and marking locations with RFID chips or Bluetooth devices.

When a location is being queried, this makes it necessary for the application to communicate with the Triple Space. The requirement that can be deducted from this is that the device needs at least one possibility of establishing a connection to a server or node of the space. MIDP1 would already serve as a solution for that since it can make HTTP connections which could be used for the queries. MIDP2 is better though since it also supports TCP/IP connections which are required by Heavy Clients for establishing connections to peers.

Internet access via technologies that are based on mobile phones, like GSM, GPRS, HSCSD, EDGE and UMTS, are all rather expensive since they cost money for each minute a device is online or for each byte that is transmitted. Therefore another requirement, especially for supporting Heavy Clients, is offering WLAN access. The most common standard for this is IEEE 802.11b. This of course means that the visitors who get access have to be trusted since they could potentially use that access to do damage to the site's network.

For the Light Client, the needs in processing power and memory are very low since it primarily displays data and only makes very basic calculations like i.e. after retrieving the information data set about a person, automatically retrieving the data about that person's current room. The Heavy Client needs to stay in touch with other nodes of the Triple Space so it needs more memory to be able to manage those connections. Besides that, it also needs to store data. The Heavy Client has to implement a memory and storage space management to make sure that the client does not accept more data from the triple space than it can handle. Otherwise the application would try to allocate more and more memory until the JavaVM would at some point deny the allocation of more memory and the program would crash. How this memory management will look would have to be decided in the process of the actual implementation of the Heavy Client application, which is not part of this thesis.

Both of the applications can be enhanced by making them location aware. This can be done via RFID or via Bluetooth. Besides requiring an RFID reader or Bluetooth in the mobile devices, this requires RFID chips or nodes with Bluetooth to be placed throughout the building. At least they need to be in the offices of the employees who want to publish their positions and next to important points in the building which a visitor passes by. In the case of Bluetooth, it would be enough to detect the Bluetooth devices and use their Bluetooth-ID to query a location database, so it would be possible to use existing stationary Bluetooth enabled devices that might exist in some offices.

Another point mentioned in the use case is a light sensor. There can be different sensors installed which write their data into the Triple Space, for example temperature or light sensors. They do not add any extra requirements since the application can simply get their data from the Triple Space and display it to the user. The sensors themselves are not part of this thesis, the reason why they are mentioned is that if data from sensors exists in the Triple Space, the application could make use of it.

4.2 Additional Requirements "on site"

The requirements listed above are all for the devices themselves. This part is about the infrastructure which is required at the place where the application is to be used.

Mandatory:

- For devices that run a Heavy Client offering WLAN access to them in the building is necessary. This means that there must be access points and the devices must be configured to use them.

Optional: retrieving the current location through:

- RFID chips in the building: RFID gives accurate positions due to the very low range of RFID chips.
- Bluetooth devices in the building: Bluetooth can give inaccurate positions because its range is usually 5-10m.

4.3 Conclusion

This section summarizes and explains the facts that were found out in the first part of the requirements analysis. Table 4.1 gives an overview about the mandatory and optional requirements of Light and Heavy Clients.

Remarks: MIDP1 does not allow accessing the TCP/IP Stack, it only allows opening HTTP Connections. Therefore it is necessary to distinguish between establishing a HTTP connection and accessing the TCP/IP socket directly.
WLAN access is listed separately since establishing HTTP or TCP/IP connections does not give any information about how this is done. For mobile phones this is usually via GSM services which is too expensive to efficiently run a Heavy Client [7]. An example for the costs can be found on the homepage of an austrian mobile services operator[1].
With "enough processing power for simple calculations" it is meant that the device must have at least half a megabyte RAM which is supposed to be sufficient for the Light Client since it only works with the data that it really needs, which is not much (see the use case). CPU is not a big issue since there are no realtime calculations, less MHz just means that the application is a little slower. The "ability to compute with bigger amounts of data" is difficult to determine exactly but it means that a device should have at least 32 megabytes of RAM and 100MHz of processing power to be sure that a Heavy Client can run at a speed where it is useful. This is because it needs to store and process a lot more data than the Light Client, namely storing and querying data of the Triple Space and managing its connections to the other nodes.

[1] http://www.one.at/ - Tarife - Daten

Requirements Overview	Minimum Light Client	Optimum Light Client	Minimum Heavy Client	Optimum Heavy Client
Ability to establish a HTTP connection to a server	X	X	X	X
Ability to open TCP/IP Connections				X
WLAN access		X	X	X
Enough memory and processing power for simple calculations	X	X	X	X
Floating point support		X		X
Ability to store data			X	X
Ability to compute with bigger amounts of data			X	X

Table 4.1: Requirements Overview

J2ME Profile/ Configuration Overview	MIDP1	MIDP2	CLDC1	CLDC1.1
HTTP Connection	X	X		
TCP Connection		X		
Storing Data	X	X		
Floating Point Datatypes				X

Table 4.2: Requirements and Solutions

The solutions for the requirements are:

- J2ME/MIDP2 (A common J2ME-Profile which allows establishing a connection to a server, TCP/IP is available. Furthermore it allows to store data.)
- J2ME/CLDC1.1 (A common J2ME-Configuration which supports floating point operations.)
- Optional: Ways to communicate with RFID (JSR 257 [2]) or Bluetooth (JSR82 [1])
- For Heavy Client: Devices with WLAN support (802.11b compatible). For the Light Client WLAN is just one possibility among many like GPRS and GSM because it transfers far less data than a Heavy Client.

Table 4.2 shows a list of requirements and their solutions.

CLDC1.0 and MIDP1 are about 5 years old already but for the Light Client they are in theory still sufficient. In practice its better to say that CLDC1.1 and MIDP2 are required since most recent mobile phones and PDAs support it, and making the software CLDC1.0/MIDP1 compliant

would just be a lot of extra work and create additional possibilities for errors to occur, without having any noteworthy advantages. When programming for CLDC1.0/MIDP1, it would be necessary to program workarounds if floating point data would have to be used. It would also be necessary to run all communications via HTTP-requests.

With CLDC1.1 and MIDP2, it is possible to establish various types of connections, be it TCP/IP or HTTP1.1, which is all that is needed for both Light and Heavy Clients on the connectivity side. This is an important difference because MIDP1 does not support TCP/IP sockets and only offers communication on a higher layer via the HTTP protocol. The Light Client can do with HTTP alone since it only needs to get information from a server (a proxy to the Triple Space). For the Heavy Client, TCP/IP support is a must though, since it has to be possible for other nodes in the Triple Space to push information to the client which means that the Heavy Client has to be ready to accept incoming connections from them.

RFID chips can be placed pretty much anywhere due to their small size. Handheld devices with integrated RFID readers are by far not as common as devices with Bluetooth support though. Bluetooth based location awareness is easier to implement and can be used by a wider spectrum of devices since many devices support it through standardized APIs. RFID on the other side is more accurate, but not supported by as many devices and there is no standardized J2ME API yet. It is being worked on by an expert team consisting of the worlds leading mobile communication companies, see [2] for further information.

For updating employee's positions, it could theoretically also be possible to use the in-house keycard system. This would only be possible in co-operation with the operators of that system. Due to laws for the protection of personal data, it would most certainly be not possible to do that. Therefore this possibility will not be considered any further.

Chapter 5

Technology Survey

This chapter is about the evaluation of mobile devices which are candidates to be used.

Devices ranging from mobile phones to PDAs/Handheld computers, in different ranges of price and abilities will be investigated. For each device it will be checked whether it fulfills all mandatory requirements and which additional helpful features it offers beyond that. Finally it will be determined which advantages and drawbacks the devices have and what the most inexpensive options, that still guarantee to fulfill all requirements, are.

5.1 Mobile Phones

The first device that was tested is the **Nokia 6230** mobile phone (Figure 5.1). It offers MIDP2.0 with CLDC1.1 and it supports Bluetooth. It has 1MB RAM but only 700-800KB are available to applications[1].

[1] http://www.club-java.com/TastePhone/J2ME/MIDP_Java_telephone.jsp?m=22&brand=Nokia&model=6230

Figure 5.1: Nokia 6230

It lacks WLAN support so it is not a perfect device for a Light Client, and it is not suitable for a Heavy Client. It fulfills all "must have" requirements of the Light Client though. The ability to download and install a J2ME program via WAP has been tested and is confirmed to work. The ability to read IDs of Bluetooth devices is confirmed as well. It has a screen size of 128x128 pixels. The price seems to be between 100 and 200€ (according to list of offers on xonio.com, May 2006). The Nokia **6230i** is an improved version of that device with a screen size of 208x208 pixels. All other improvements are in fields that are not relevant for this kind of application. It costs about 100€ more than the 6230.

Sony Ericsson K750i Mobile Phone (Figure 5.2). Like the Nokia 6230, the Sony Ericsson K750i has support for GPRS connections, Bluetooth and J2ME/MIDP2. The display has a size of 176x220 pixels. It is another interesting choice for running the Light Client. This phone costs approximately 330€.

Figure 5.2: Sony Ericsson K750i

Figure 5.3: Motorola RAZR V3

Motorola RAZR V3 Mobile Phone (Figure 5.3). The RAZR V3 is quite similar to the other mobile phones that were investigated. It also has GPRS and Bluetooth and of course J2ME/MIDP2. The display has a size of 176x220 pixels. Although it is not suitable for a Heavy Client, it offers enough for a Light Client. The price of the RAZR V3 is about 180€ (xonio.com, May 2006).

Figure 5.4: Samsung SGH-D600

Samsung SGH-D600 Mobile Phone (Figure 5.4). The SGH-D600 has GPRS, J2ME/MIDP2 and Bluetooth. The display has a size of 320x240 pixels. This makes it very similar to Nokia 6230i, SE K750i and Motorola RAZR V3, and like them, it makes a good platform for a Light Client but is not enough to run a Heavy Client. It costs about 300€ (xonio.com, May 2006).

Figure 5.5: Nokia 9500 Communicator

Nokia 9500 Communicator Smartphone (Figure 5.5). The Nokia Communicator Smartphone is the most expensive device tested, it even costs more than many high-end PDAs. It supports WLAN, GPRS, Bluetooth and even has an IPv6 stack. The display has 65536 (2^{16}) colors and 640x200 pixels[2]. Of course it has MIDP 2.0 support. The memory size is about 16MB[3]. By being a smartphone it combines the abilities of a mobile phone like the Nokia 6230 with the abilities of the Pocket-PCs. Most Pocket-PCs however have bigger displays and are noticeably cheaper. It is the only mobile phone that was investigated, which is suitable for a Heavy Client. The Nokia 9500 Communicator uses the SymbianOS operating system and costs about 600€ according to xonio.com (May 2006).

[2]http://europe.nokia.com/nokia/0,8764,54108,00.html

[3]http://www.club-java.com/TastePhone/J2ME/MIDP_Java_telephone.jsp?m=188&brand=Nokia&model=9500

5.2 Personal Digital Assistants

Figure 5.6: HP iPAQ rx1950 Pocket-PC

HP iPAQ rx1950 Pocket-PC (Figure 5.6). With 300 MHz Processor, 32MB SDRAM and WLAN (802.11b standard), it offers everything that is required for the Heavy Client. Unfortunately it does not support Bluetooth. Its lack of Bluetooth makes it rather uninteresting for purchase. Due to the sandbox-way of how the J2ME JavaVM works, it is unlikely that an adapter would be able to add Bluetooth-Support. Besides that, it would of course be easier and most certainly not any less cost-efficient to find a device that comes with built-in Bluetooth. It uses the Windows Mobile Operating System, the price is approximately 300€ (HP store, May 2006).

A better iPAQ is the **HP iPAQ hx2410 Pocket-PC**. It comes with 520 MHz, 64MB SDRAM, WLAN (802.11b) and Bluetooth support. So it has pretty much everything that one could wish in the scope of this work. The only thing on the wish list that it does not have is an RFID reader. It does have a CF slot though, through which an RFID reader could be added (see section 5.3). The Operating System is Microsoft Windows Mobile 2003. This PDA costs 449€ (HP store, January 2006)

Figure 5.7: Dell Axim X51

Another considered PDA is the **Dell Axim X51 520MHz Pocket PC** (Figure 5.7). With an Intel XScale PXA270 520MHz processor, 64MB of SDRAM and WLAN(802.11b), it has everything that is required to run a Heavy Client. In addition it has Bluetooth support. It does not have RFID but like many of the other PDAs, it does have a CF slot which can be used for a RFID reader addon. It uses the "Microsoft Windows Mobile" Operating System. The price is about 360€ (expansys.de, May 2006).

From Acer, an interesting candidate is the **Acer n311 Pocket-PC**. It has a Samsung 400MHz processor, 64MB RAM, 802.11b WLAN and Bluetooth. It uses the Microsoft Windows Mobile Operating System. Like all previously tested Pocket-PCs, the n311 does not have a RFID reader. It has everything that the heavy client needs. The recommended price is 400€, this matches with the prices listed on the internet-sites of vendors (May 2006).

Figure 5.8: ASUS MyPal A636

The **ASUS MyPal A636 PocketPC** (Figure 5.8) is a quite interesting Pocket PC since it comes with GPS. This is not of any use for the application that is being discussed in that thesis since GPS is not suitable for inhouse positioning. It would be interesting though for extended applications which do positioning inhouse the way it is described in this thesis and on the outside with GPS. The CPU which the

MyPal A636 uses is a Intel 416MHz processor. It has 64MB RAM, WLAN (802.11b), Bluetooth and Windows Mobile 5. Therefore it fulfills all requirements of a Heavy Client. The price is about 400€ (May 2006).

One important thing about PocketPCs is that they often have no Java Virtual Machine that supports CLDC/MIDP. In the lists of technical details of all the PDAs that were investigated, not a single one mentioned J2ME support. According to vikdavid.com[4], iPAQ PocketPCs used to come with J2ME support but this is not the case any more. An easy and cheap solution is IBMs J9 VM which is also often referred to as "IBM WebSphere Everyplace Micro Environment". It costs about 6$.

[4]http://www.vikdavid.com/mobile/

Figure 5.9: RFID Reader Addon

5.3 RFID Readers

Many PDAs can be enhanced by an RFID reader (Figure 5.9) like:

- PCR125 / MFR135 RFID Reader for PDA [5]
- Socket RFID Reader Card 6E (Compact Flash) [6]

They all cost several hundred Euro and it needs to be checked whether they can be accessed through J2ME. It was not possible to find good information about this issue on the Web. A mail with that question to the producer of the Socket RFID Reader Card was not answered yet. From the authors understanding, it cannot work due to the fact that J2ME-applications run in a sandbox environment, therefore being unaware about add-on hardware where the JVM offers no APIs to access it.

[5] http://www.globalsources.com/gsol/I/Proximity-card-reader/p/2000000003844/3000000149681/sm/1000336870.htm

[6] http://www.expansys.de/product.asp?code=121337&asource=126057

5.4 Discussion

The following section illustrates the results of the technology survey. Table 5.1 gives an overview about the devices that were investigated.

The table shows pretty clearly that the prices of PDAs with WLAN and Bluetooth are all very similar, being about 400€. It also shows that all mobile phones are only able to run a Light Client while all Pocket-PCs can run a Heavy Client. The only exception is the Nokia 9500 Smartphone which is as good as the Pocket-PCs (and twice as expensive) and additionally has EDGE/GPRS/HSCSD/GSM connectivity.

There are dozens if not hundreds of other mobile phones which have MIDP2.0/CLDC1.1 which is, in most cases, sufficient to run a Light Client, without any possibilities of location awareness though. Many of those phones are already very cheap since they are relatively old.

Since using existing things is cheaper than buying something new, and since many people own such phones, they can and should be used. Due to how J2ME works, a Light Client should work on all of those with MIDP2/CLDC1.1. It can only be confirmed by testing it on an individual basis though, since nothing is perfect and it is known that in some cases, faulty implementations exist. Then even the firmware version on the device can make a big difference. An example for that is the Nokia 6230, with firmware version 04.28, it does not allow establishing parallel TCP connections and fails without error message. A listing of MIDP and CLDC version for a lot of mobile phones can be found at club-java.com[7].

[7] http://www.club-java.com/TastePhone/J2ME/MIDP_mobile.jsp

Device	MIDP2 + CLDC1.1	Connectivity	Bluetooth / RFID	Suitable for Light or Heavy Client	Price (€)
Nokia 6230i	YES	EDGE	BT	Light	200
Sony Ericsson K750i	YES	GPRS	BT	Light	330
Motorola RAZR V3 Mobile Phone	YES	GPRS	BT	Light	180
Samsung SGH-D600	YES	EDGE	BT	Light	300
Nokia 9500 Communicator	YES	EDGE / WLAN	BT	Heavy	600
HP iPAQ rx1950 Pocket-PC	YES	WLAN	-	Heavy	300
HP iPAQ hx2410 Pocket-PC	YES	WLAN	BT	Heavy	450
Dell Axim X51 Pocket-PC	YES	WLAN	BT	Heavy	360
Acer n311 Pocket-PC	YES	WLAN	BT	Heavy	400
Asus MyPal A636 Pocket-PC	YES	WLAN	BT	Heavy	400

Table 5.1: Devices Overview

Remark: For Cell-Phone connectivity, only the best type is listed in the table. For example when it says "EDGE" support, it means it also supports GPRS and GSM, which are the technologies that EDGE is built on.

Best Choice

Mobile Phones:

- Nokia 6230(i): It has everything that is needed for the Light Client and it has Bluetooth. It comes at a reasonable price and besides being suitable for the application that is being discussed in this thesis, it is a nice phone which is important as well. The Nokia 6230i does not bring any extra possibilities, but since it is an improved follow-up of the 6230 it is as interesting, the only reason to prefer the 6230 over the 6230i is the price. The 6230 costs about 100€, the 6230i about 200€.

PDAs:

- Dell Axim X51: This PDA has everything that is needed for the Heavy Client as well as Bluetooth. Most of the PDAs tested have those features but at a price of only 360€ this PDA is a cost-efficient choice and cheaper than the others. Like many PDAs, it uses the "Windows Mobile" operating system. The Acer n311 and the Asus MyPal A636 are pretty similar concerning features and price and are an option as well.

Chapter 6

Conclusion

The most important goal of the work was to find out what is required for making an inhouse navigation system based on Triple Space Computing, and how these requirements can be fulfilled.

The requirements analysis ended with the result that Java2 Micro Edition (J2ME) with the MIDP2 profile and the CLDC1.1 configuration fulfills all must-have requirements. It is available on most modern mobile phones and offers everything that is needed, including networking and data storage. For location-awareness, Bluetooth-Support (following the JSR82 specification) is required, together with Bluetooth-nodes at various locations. Other options like location-awareness based on RFID or compatibility with older J2ME profiles and configurations were considered, but turned out not to be very promising.

While there is a lot of work in progress, for example in the field of RFID, where the Java Specification Request (JSR) is not finished yet, most relevant technologies are standardized already. JSR82 which defines a standard-

ized API for Bluetooth was completed in 2002 for example. When looking at recent high-end mobile phones, its clear that most of them support CLDC1.1/MIDP2 and JSR82 (Bluetooth). Therefore, RFID was not considered as a possible solution for location-awareness.

The technology survey was positively surprising. Since almost all average to high-end mobile phones have CLDC1.1/MIDP2 and Bluetooth support, they are likely to fulfill all requirements for a Light Client. All of the phones that were closer investigated fulfilled these expectation. In the field of Personal Digital Assistants, there was a wide range of interesting devices as well. Unfortunately it was impossible to find any interesting offers of PDAs with built-in RFID readers. A surprising point considering PDAs was that many do not get shipped with J2ME support. This is available from third parties like IBM for a pretty inexpensive price, so it is no problem. Its noteworthy since a few years ago, all or at least most Palm and iPAQ PDAs came with J2ME. For mobile phones this is totally different, there J2ME is standard and if a device does not have it, it is usually impossible to change that.

Another interesting result of the technology survey is that while all the PDAs fulfill all requirements to run a Heavy Client, the mobile phones are all just capable of running a Light Client. The exception are smartphones which combine the best of both sides, at a pretty high price.

The inhouse navigation application is a nice way to demonstrate the ideas of ubiquitous computing using Triple Space Computing. To make it really useful it needs location awareness. While this seems easy from the technical point of view, it would require a lot of work to for example place bluetooth

dongles on the PCs in the various rooms and make sure that they are powered up all the time. In a year or two, when the specification for a standardized API for using RFID through J2ME is ready, there will be better possibilities. It can be expected, that there will be a lot less mobile phones and PDAs which support RFID, compared to the number of such devices with Bluetooth-support.

Apart from making an actual implementation, interesting ideas are for example to combine the navigation tool with a messaging program. When making an appointment, one could set the meeting room as destination for everybody else or easily tell somebody about his current location.

Bibliography

[1] JSR 82 Expert Group: Java APIs for Bluetooth Wireless Technology, JSR Specification, 2002, available at http://www.jcp.org/en/jsr/detail?id=82

[2] JSR 257 Expert Group: Contactless Communication API, JSR Public Review Paper, October 2005, available at http://www.jcp.org/en/jsr/detail?id=257

[3] T. Berners-Lee: Notation 3: An RDF language for the Semantic Web, W3C Design Issues, November 2001, available at http://www.w3.org/DesignIssues/Notation3.html .

[4] D. Fensel: Triple-space computing: Semantic Web Services based on persistent publication of information. Proc. of the IFIP Int'l Conf. on Intelligence in Communication Systems, Bangkok, Thailand, 2004: 43-53.

[5] A. Harth, M. Magni, and St. Decker: Scalable Distributed RDF Storage Infrastructure. DERI Lion Deliverable 1.02, June 2005.

[6] R. Krummenacher, M. Hepp, A. Polleres, Ch. Bussler, and D. Fensel: WWW or What is Wrong with Web services, Proc. 3rd European Conf. on Web Services (ECOWS), Nov 2005: 235-243.

[7] J. Rabin and C. McCathieNevile: Mobile Web Best Practices 1.0, W3C Working Draft, January 2006 (Section 2.3)

[8] M. Weiser: The Computer for the Twenty-First Century, Scientific American, pp. 94-10, September 1991