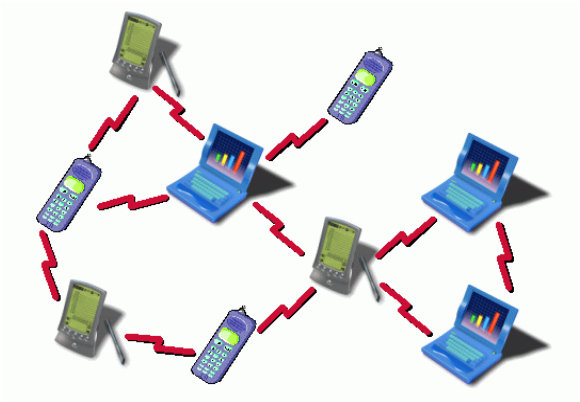
UNIVERSITY OF OSLO
Department of informatics

Final Report INF5261

Mobile Ad-Hoc Networking - Transparent Virtual Directory

Henning Berg
Jon Oldevik
Lars Kristian Snellingen Bye
Mansur Ali Abbasi
(Sven Jørgen Karlsen)

May 10th 2006



Contents

1 Introduction

Today, a lot of people carry around several portable devices, such as laptops, PDAs, mp3 players, and mobile phones. Mostly, the applications of these devices do not interact; they are used separately. Imagine that they instead could communicate intelligently and data could be exchanged seamlessly across these devices. Ad-hoc mobile networks may be a key for enabling this.

Ad-hoc networking is not a new thing. Early mobile ad-hoc networks (MANETS) were developed at Hawaii late 1960's and at DARPA in the early 1970's, then called packet radio networks [1]. Mobile ad-hoc networking is now used to denote self-configuring networks of mobile terminals connected by wireless links, which form arbitrary topologies. It requires no base stations, as devices communicate directly and can be routed to devices out of range by forwarding.

This report describes the work and findings in the development of a Mobile Ad-Hoc Networking concept, a Transparent Virtual Directory (TVD). TVD is a file sharing concept designed for ad hoc file sharing among mobile devices. The purpose is to be able to share files between arbitrary users and groups of users using the spontaneous connectivity of mobile ad-hoc networks.

2 Background

2.1 Main objectives of our project

The primary objective of our project has been to look at the possibilities and limitations of using mobile terminals for ad-hoc networking.

A secondary objective, which is of equal motivational value as the primary, has been to gain hands-on experience in using and creating mobile ad-hoc network applications.

To achieve these objectives within the time frame for the project, we decided early in the process to focus on finding an interesting example case / scenario on which a prototype application can be based. The process of designing and attempting to realize such a prototype application will illuminate challenges and possibilities at technical level, business level and social level, and enable us to gain insight into key concepts and technologies of mobile ad-hoc networking and mobile software development (without digging too deeply into low-level technical details such as various protocol details, etc).

2.2 Mobility

Mobility is the ability and willingness to move or change. It is a highly relative term, which may have many different meanings, depending on the context:

- In solid state physics, mobility generally refers to electron mobility or hole-mobility.
- In mobile computing, mobility refers to characteristics of device to handle information access, communication and business transactions while in state of motion.
- Academic mobility refers to the possibility for students and teachers to move between different institutions inside and outside their own country.
- Social mobility refers to the ability of individuals within a society to move between different social levels
- Mobility is also a computer game.
- In demography the mobility of a population measures migration within a population.

There are many additional contexts where the term mobility may apply. Our context, which is mobile ad-hoc networking, inherently adopts the mobility concept to describe the nature of these kinds of networks

and systems. A mobile system, in contrast to a stationary system, is changing its location (or has the ability to change its location), and is independent of a physical line for network connection. Instead, it allows for the terminal to be moved around while being connected. The provider of the actual mobility can be a person carrying the device (e.g. a mobile phone, PDA, or laptop), a car with a built-in GPS system, or similar.

In [2], it is argued that the traditional interpretation of mobility of being mobility of people is too limiting. The interaction among people is also increasingly mobilized thanks to the Internet, mobile phones, portable computers, etc. It is argued for an expansion of the mobility concept into *spatial, temporal,* and, *contextual* mobility. Spatial mobility is the mobility of objects, symbols, and space itself. Temporal mobility refers to the timeliness of interactions, i.e. if they are planned and sequences or more spontaneous and dynamic, and that given modern technology, there is a lot temporal mobility in interactions. Contextual mobility refers to the switching of context in interactions, which is highly influenced by modern technologies.

In this work, we focus on some enabling technologies for mobility, which allows people to use applications anywhere while on the move; Ad-hoc networking provides a means for being mobile and flexible and still being able to interact and exchange data with others. In relation to [2], we see that ad-hoc applications in general may impact all the mobility aspects addressed. Most interesting though, is the influence on temporal and contextual mobility. The spontaneous nature of ad-hoc applications will impact the user's interactions and increase the *polychronicity* of human interactions. In addition, it is likely to create new changes to interaction context, depending on the nature of the application.

2.3 Mobile Ad-hoc Networking

A mobile Ad Hoc Network (MANET) is a self-configuring network based on wireless links between mobile nodes [1]. These nodes are free to move randomly and organize themselves arbitrarily; thus, the network's wireless topology may change rapidly and unpredictably. Each node operates not only as an end-system, but also as a router to forward packets. The MANETs may operate in a standalone fashion, or may also be connected to another network, example the Internet.

The history of MANET's goes back to the early 1970's, when DARPA sponsored the development of packet radio networks. This was the world's first multihop network. In this context, multihopping (Figure 1) means that nodes cooperate to relay traffic on behalf of one another to reach distant nodes that would otherwise have been out of range. This is also one of the greatest advantages of today's MANETs. Why is this? First, you don't need more than the neighbor unit in range. This can dramatically reduce power usage. Secondly, if many nodes wish to communicate to a central unit or server, multihopping will help organizing and balance the load. Multihopping is, however, not required to call a network ad hoc. Ad hoc merely imply that an impulsive connection between units has been established. This communication might appear between just two nodes. The technology, however, has much more advanced possibilities.

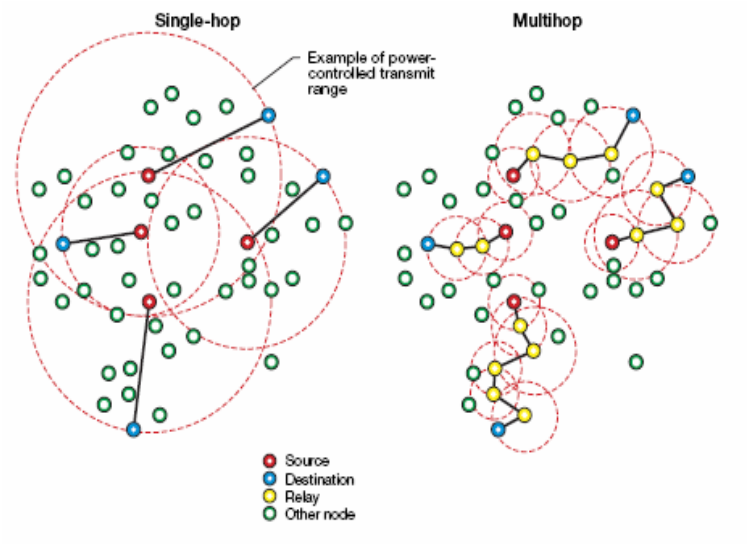


Figure 1 Single-hop versus multihop networks

So what are the benefits of MANETs? Well, combining the infrastructure-free environment and the self-configuring capabilities provides an extremely flexible and mobile platform. The technology gives you the opportunity to create a communication network almost anywhere; one only need some of the units being in range of each other. Earlier MANETs were mostly used by science groups and the military. Usage areas were for example exploration of other planets or areas at deep sea levels or military operations going on behind enemy lines. These were basically areas without any established communication infrastructure. Today, MANETs are consuming a bigger part of the daily life of everyday people. We see new products for home use that can communicate directly, for example audio/video equipment and alarm systems. We have mobile phones that can communicate directly with computers, PDA's, or other phones. The popular IEEE 802.11 ("Wi-Fi") wireless protocol also incorporates an ad-hoc networking system when no wireless access points are present, although it would be considered a very low-grade ad-hoc protocol by specialists in the field. The IEEE 802.11 system only handles traffic within a local "cloud" of wireless devices. Each node transmits and receives data, but does not route anything between the network's systems. However, higher-level protocols can be used to aggregate various IEEE 802.11 ad-hoc networks into MANETs.

Since MANETs provide a very complex way of networking, they also bring some challenges. They push the limits especially regarding packet routing and security. Keeping track of nodes in a rapidly changing system and knowing who to trust are some important issues. Today, there are many different implementations that solve parts of this, each with some benefits and drawbacks. Which one to use depends on the situation and needs. There is also a challenge in battery life on the nodes in the network, since better range demands more power. But this is still rapidly developing both in batteries containing more power and equipment using less.

2.4 Limitation of scope

Investigating all possibilities and limitations of using mobile terminals for ad-hoc networking may prove to be very challenging, hence we will concentrate on possibilities and limitations discovered during our work with the prototype application.

Further, since various technologies for ad-hoc networking and mobile application development exist, we have limited our scope further by first making a choice of what technologies we want to use and learn more about.

2.5 Prototype application

2.5.1 Abandoned ideas for prototype application

The initial idea for a concept application was a Bluetooth-based application that could be utilized in a disaster management- or rescue operation scenario. We soon realized, however, that in order to satisfy the needs of such complex activities as disaster management and emergency rescue operations we would need to study more than just essential ad-hoc networking concepts and techniques for mobile software development. Hence, keeping the timeframe in mind, we switched to another idea which sounded more interesting and executable at that time.

The new idea now was to try to develop a software-based positioning mechanism which would allow a mobile Bluetooth-node inside a piconet or scatternet to be positioned relative to three static reference nodes (using triangular positioning). This idea was kept alive for two weeks before deeper analysis of the planned application concluded that the probability for the planned prototype to be successful not in favour of executing the idea.

2.5.2 New idea for prototype application

The prototype application should employ the basic ad-hoc principles. With this in mind we came up with a new idea of a file sharing application. The main concept of this application is to share files ad-hoc between Bluetooth-enabled devices. File sharing is a well-known application domain. The ad-hoc version of file sharing, however, still has several unanswered questions and provides a good fundament for technical reasoning.

Several restrictions have been made to our prototype application due to the time constraint and the nature of this project. However, we feel that most of these restrictions apply to other aspects of application development rather than the principles and methodology of ad-hoc networking. For instance, file replication and load balancing are issues strongly related to development and research in the domain of distributed systems.

The next chapter elaborates our idea of the ad-hoc file sharing application concept, which is called *Transparent Virtual Directory*.

3 The TVD Concept: Transparent Virtual Directory

3.1 Introduction

The main goal of a service like 'Transparent Virtual Directory' is to provide a simple and intuitive way of sharing files with other people ad-hoc using Bluetooth capable devices. The concept behind this service is rather simple to understand. However, there are several important issues related to implementation and technology restrictions which must be addressed in order to achieve a seamless, flexible and stable system. We will try to identify these issues in the rest of this chapter. We will use three main scenarios in our explanation of Transparent Virtual Directory, TVD in short. The studying and reasoning about these scenarios will also result in a deeper understanding of the principles and ideas behind the concept 'ad-hoc networking'.

3.2 Distributed file access and file sharing

In short; TVD is all about sharing a common virtual directory. This directory may contain any form of file content from plain text files to binary files like pictures and mp3-files. The term 'virtual' does in this context refer to a directory which is shared among all of the concurrent users of the TVD-service. This virtual directory does first exist when there are at least two Bluetooth devices connected together using the TVD-service.

At present there are hundreds of devices supporting Bluetooth where mobile phones, PDAs and laptop computers are the most common application areas. The term 'transparent' used in the TVD-context refers to a notion of transparency when using the TVD-service across multiple platforms and device types.

There are two different modes supported by TVD; passive mode and active mode. The operations supported by the two different modes are based on a simple File Awareness Model. This model presents the users with an updated, uniform file list representing the content of the virtual directory, and maintains the hosts and routing information. We will return to this model shortly.

The two different user modes are related to the different states of a TVD server/client and the actions performed by the end users. The active mode is characterised by an ongoing file transfer and the associated file requests. The passive mode indicates the user being idle or browsing the common TVD file list. The transitions between the different modes are transparent for the end user. However, the differentiation between the different states is necessary due to load balancing and replication issues. The only supported functionality in active mode is file downloading. One or more files can be downloaded from various nodes simultaneously. Uploading of files is not supported. This prevents misuse like spamming, flooding and simple forms of DoS.

3.3 How it works

Transparent Virtual Directory is based on a peer-to-peer ad-hoc network architecture. This means every node in the network has the same capabilities and works both as a client and a server. A node is therefore referred to as a 'server/client'. Please refer the section 'Distributed file access and file sharing'.

Each node in a TVD network may share one or more of its files with the rest of the network. The shared files will be visible for all the other nodes in the TVD network. However, the files are not copied to these nodes. A file is first copied to a node when this node requests the file. This way, all files presented in the TVD file list are distributed among the participating nodes in the network.

The basic operations of the TVD service can be categorised in three main scenarios:

1. The network consists of two nodes where node A requests one or more files from node B.

2. The network consists of three nodes where node A requests one or more files from node B and C. All nodes are within range of each other.
3. The network consists of three nodes where node A requests one or more files from node B and C. The nodes A and B are within range of each other. The same applies to node B and C. In other words, node A does not have a direct connection with node C and vice versa.

Other scenarios can be explained using a combination of the three aforementioned scenarios.

3.3.1 The File Awareness Model

Each node in a TVD network has its own unique File Awareness Model (FAM). A FAM consists of a table with information about every shared file in the TVD network and its associated node (owner). Each File Awareness Model represents a part of a complete TVD network. The union of all FAMs expresses the connection paths between every node in the network and works as a reactive routing protocol.

A given node does not store information about the addresses of nodes outside its own range. However, this node is still able to access all the shared files by the use of forwarding. The shared files are presented using a uniform file list. The exact same list is shared by all nodes in the TVD network and is extracted from a given node's File Awareness Model.

A given node in a TVD network may choose not share any files. However, this node is still useful for forwarding purposes. To be able to access this host, every reachable node has its own ghost entry in the File Awareness Model.

An example File Awareness Model is shown below:

File name	Host	Host address	Host alive
file A.txt	localhost	localhost	0:00:00
file B.txt	node B	btspp://102030405060A194C37429E4938:5	0:00:04
file C.txt	node C	btspp://1020304050604C277314562A3B0:5	0:00:10
file D.txt	node D	foreign	N/A
ghost	node B	btspp://102030405060A194C37429E4938:5	0:01:05
ghost	node C	btspp://1020304050604C277314562A3B0:5	0:03:10

3.3.2 Operation classes

The following figures use four operation classes which generalise four different types of operations. These operations are always executed in the following sequence; *requests*, *file existence tests*, *target destination tests* and *file transfers*. A *file existence test* is always preceded by a *request*, a *target destination test* is always preceded by a *file existence test* and so on. The following table shows an overview of the different operation classes. The number of hops indicates the number of forwards needed to get a requested file. In other words, how many nodes there are between a node requesting a file, and the target node on which this requested file is located. This number is equal to zero if the target node is within the range of the node requesting a file. The number of hops also indicates the distance between two nodes. A given *request* is closely related to a *file existence test*. The same applies for a *target destination test* and a *file transfer*. The figures are best read by following the order of the operation classes.

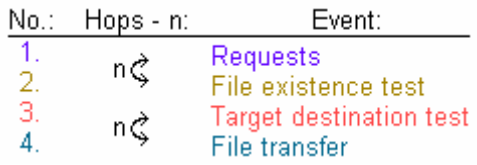


Figure 2: Operation classes in a TVD network

3.3.3 Initialisation phase

An initialisation phase starts ad-hoc as soon as two or more Bluetooth devices running TVD are within reach of each other. The following is performed:

- A detection of servers running TVD
- A detection of the TVD service
- Authorisation and authentication
- Selection of the files to share
- Building of a FAM for each node in the network
- Presenting a conform file list on each node

The events 4 to 6 are performed dynamically and keep the FAMs updated with the newest available file and routing information.

Please refer the sections 3.4.1 through 3.4.4 for details regarding file replication, routing, load balancing, and security.

3.3.4 File requests, file and node awareness and file transfers

This section explains the basic functionality of the TVD service using three scenarios. Several issues regarding load balancing and replication of files have been left out to make the explanation more clear. Please refer the sections regarding these topics for further details.

3.3.4.1 Scenario 1: Two nodes

The network consists of two nodes where node A requests one or more files from node B.

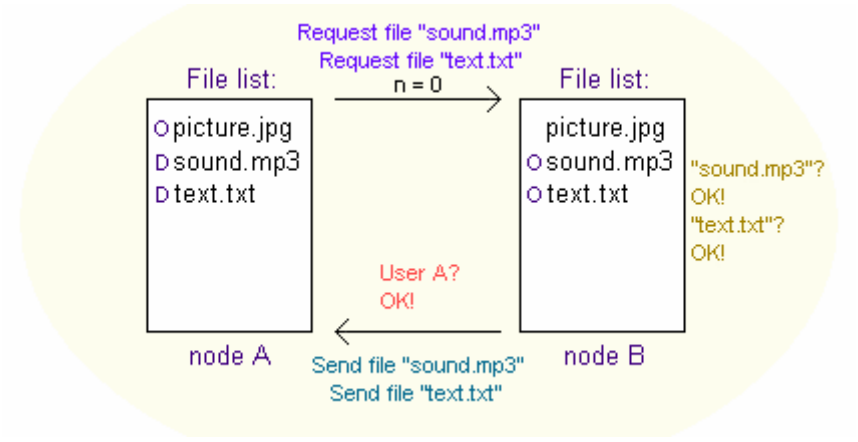


Figure 3: A TVD network consisting of two nodes

Figure 2 shows the basic application of the TVD service. There are two nodes/devices in the TVD network sharing three files. The figure shows the file lists as shown on the two Bluetooth enabled devices running TVD (some details are left out for the sake of clarity). The three shared files, as shown in the figure, are: *picture.jpg*, *sound.mp3 and text.txt*. *picture.jpg* is physically located at the device denoted by node A.

(This is indicated with a capital 'o' - own or ownership, beside the file name in the file list.) The other two files are located on the device named node B. The figure shows the situation after the two nodes have come within reach of each other and the initialisation phase of the TVD service has been performed. The user of node A has marked the files sound.mp3 and text.txt for download. (This is indicated with a capital 'd' beside the file names.) The next sequence of actions is as follows:

1. The user presses the download button on his/her device.
2. Two lookups for the files *sound.mp3* and *text.txt* is done in the File Awareness Model of node A. A simplified example of this model is shown below. (For instance, the ghost entries and host addresses are not shown):

File name	***Host***
picture.jpg	*localhost*
sound.mp3	*node B*
text.txt	*node B*

Figure 4 File Awareness Model - node A

Two file requests are sent to node B. In the following only the processing of the first file request – *sound.mp3* is explained. The processing of the other file request is similar to that of the first request.

Node B checks its File Awareness Model for the file *sound.mp3*. This file is marked with localhost which implies the local storage of this file in node B. A simplified version of the File Awareness Model on node B is shown below.

File name	***Host***
picture.jpg	node A
sound.mp3	localhost
text.txt	localhost

Figure 5 File Awareness Model – node B

Node B then checks its File Awareness Model again to see if node A exists within the range of node B. Node A is found in the model/table and a transfer of file *sound.mp3* starts. The file is then stored in the Record Store of node A.

Steps 4 and 5 are repeated for the file *text.txt*.

3.3.4.2 Scenario 2 – Three nodes within range of each other

The network consists of three nodes where node A requests one or more files from node B and C. All nodes are within range of each other.

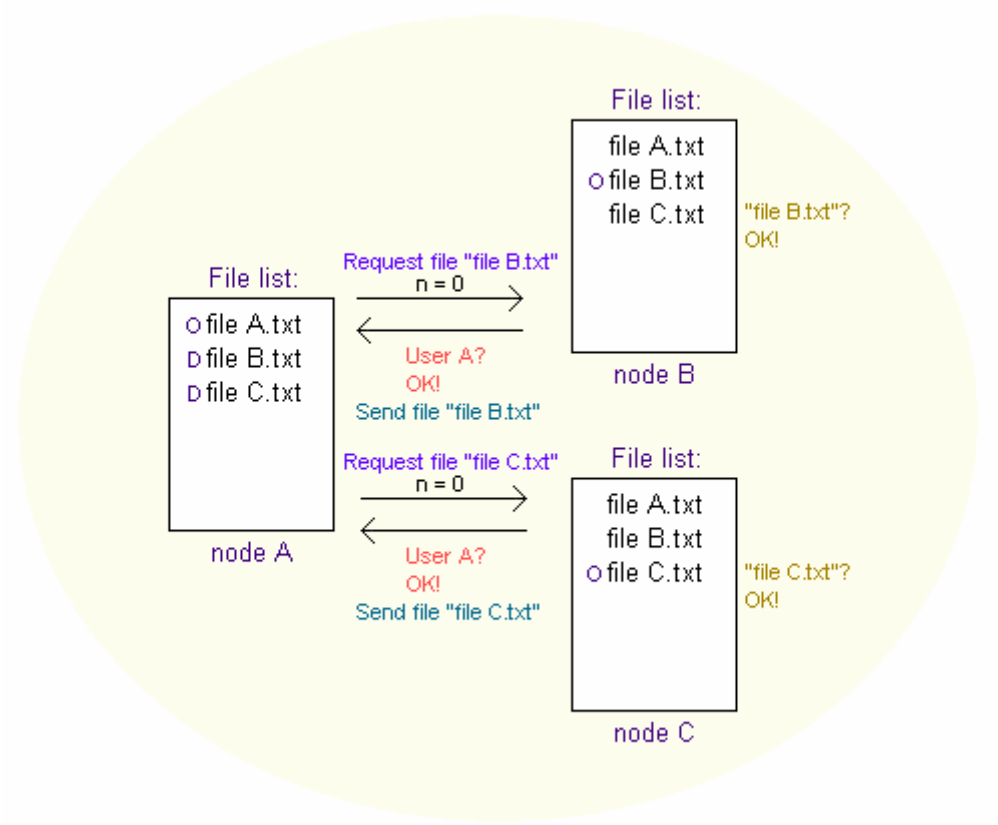


Figure 6 A TVD network consisting of three nodes/Bluetooth devices

Figure 3 shows a TVD network with three nodes. Every node is within reach of the other nodes. *file A.txt* is stored on the device represented by node A, *file B.txt* is stored in node B and *file C.txt* in node C. The figure shows the situation after all nodes have been initialised and the user of node A has selected *file B.txt* and *file C.txt* for download. Simplified versions of the File Awareness Models are shown below:

File name	***Host***
file A.txt	localhost
file B.txt	node B
file C.txt	node C

Figure 7 File Awareness Model - node A

File name	***Host***
file A.txt	node A
file B.txt	localhost
file C.txt	node C

Figure 8 File Awareness Model - node B

File name	***Host***
file A.txt	node A
file B.txt	node B

file C.txt	localhost

Figure 9 File Awareness Model - node C

The sequence of actions is as follows:

1. Node A requests the files *file B.txt* and *file C.txt* from the nodes B and C, respectively (after lookups in the FAM).
2. Node B checks its FAM and verifies the ownership of file *file B.txt*. It then finds node A in its FAM and sends the *file B.txt* to node A. The same operations are performed by node C regarding the file *file C.txt*.
3. The files are stored in node A.

3.3.4.3 Scenario 3 – Three nodes whereof two are not within range of each other

The network consists of three nodes where node A requests one or more files from node B and C. The nodes A and B are within range of each other. The same applies to node B and C. In other words, node A does not have a direct connection with node C and vice versa.

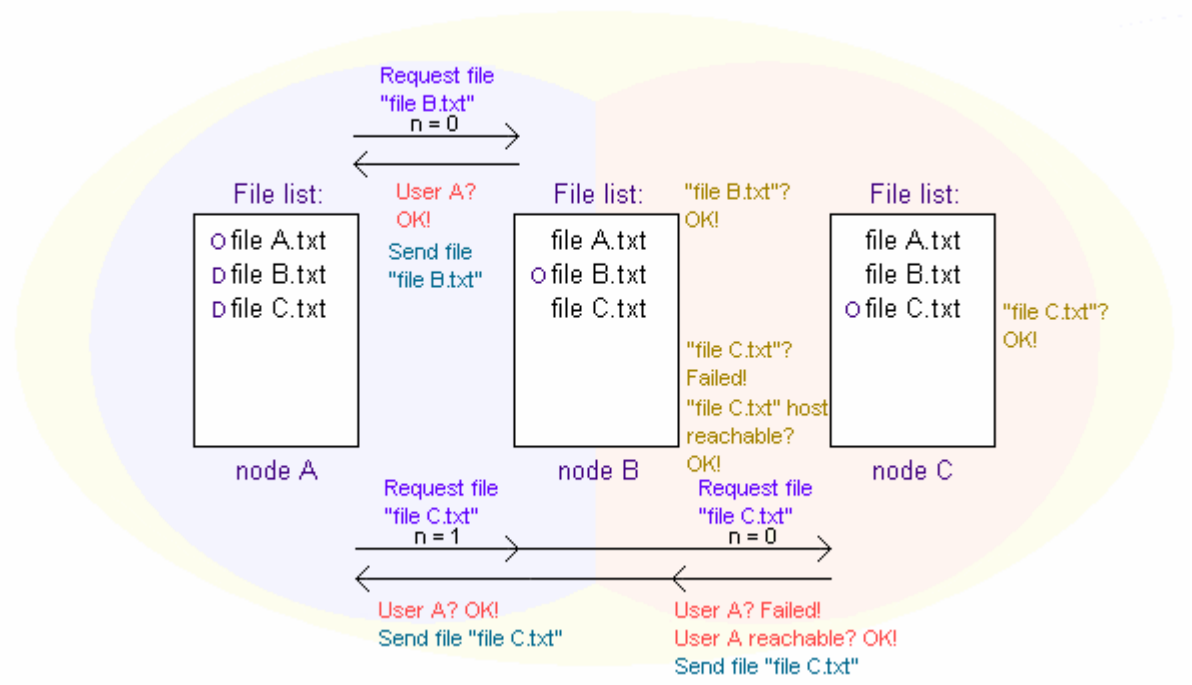


Figure 10 Three nodes in a TVD network where node A and node C is out of reach of each other

Figure 4 shows a TVD network consisting of three nodes. Node A is not within reach of node C. The figure shows the situation after all the nodes have been initialised and the user of node A has requested *file B.txt* and *file C.txt*. The simplified File Awareness Models are shown below:

File name	***Host***
file A.txt	localhost
file B.txt	node B

file C.txt	node C not reachable

Figure 11 File Awareness Model - node A

File name	***Host***
file A.txt	node A
file B.txt	localhost
file C.txt	node C

Figure 12 File Awareness Model - node B

File name	***Host***
file A.txt	node A not reachable
file B.txt	node B
file C.txt	Localhost

Figure 13 File Awareness Model - node C

The sequence of actions is as follows:

1. The user of node A requests the file *file B.txt* and *file C.txt*. *file B.txt* exists on node B and is transferred directly to node A. However, the FAM of node A does not contain the host address to node C which implies that node C is out of range of node A.
2. Node A sends a general request for *file C.txt* to all the reachable nodes in its FAM (here only node B). Node B does not own the file *file C.txt*, but checks its FAM to see if a registered, reachable node owns the file.
3. It finds the address to node C and forwards node A's request for *file C.txt*. The information about the path back to node A is carried along and presented for node C.
4. Node C checks whether it has the address to node A, the node which requested the file. Since it does not have this address it sends the requested file to node B which forwards it to node A.
5. Node A stores the file *file C.txt* in its Record Store.

3.4 Technical challenges

This section describes the main technical challenges which are expected during implementation of TVD.

3.4.1 File replication and file ownership

Many strategies for data replication in mobile computing have been proposed in research articles, such as (Barbara)[4] and (Hara)[5]. Most of these strategies assume an environment where mobile hosts access databases at sites in a fixed network, and create replicas of data on the mobile hosts. This because wireless communication is more expensive than wired communication. The strategies address issues of keeping consistency between original data and its replicas and propagating update operations to the replicas with low communication costs. Many of these strategies are considered to be similar to our approach; creating replicas on mobile hosts. However, these assume only one-hop wireless communication. The approach in the article 'Effective Replica Allocation in Ad Hoc Networks for Improving Data Accessibility' [5] addresses these issues for multi-hop communication in ad-hoc networks. It proposes three replica allocation methods which differ in emphasis put on access frequency and network topology:

- Static Access Frequency (SAF) method: Only the access frequency to each data item is taken into account.
- Dynamic Access Frequency and Neighbourhood (DAFN) method: The access frequency to each data item and the neighbourhood among mobile hosts are taken into account.

- Dynamic Connectivity based Grouping (DCG) method: The access frequency to each data item and the whole network topology are taken into account.

At this point we have decided to apply restrictions to our TVD-application to avoid the difficulties file replication may cause:

- Only one node in the TVD-network may share a file with a given name.
- The aforementioned node owns this file.
- Files downloaded by nodes in the TVD-network are not automatically shared by these nodes.

3.4.2 Routing

There are many different routing protocols and routing strategies available today. Many routing protocols have also been designed to be used in ad-hoc networks [11]. These ad-hoc compatible protocols are classified in three groups; proactive, reactive and hybrid routing protocols.

Proactive routing protocols work by keeping all nodes in the network updated with routing information to all the other nodes in the network. Reactive routing protocols, also known as on-demand routing protocols, maintain routing information for active routes only. Routes are determined on-demand when a given node needs to send data to another node in the network. Hybrid routing protocols use a combination of the principles of both proactive and reactive protocols.

At this point we think that a reactive point-to-point (hop-by-hop) protocol will be the most appropriate choice for TVD. With the use of this protocol each data packet only carries the destination address and the next node in the route. All intermediate nodes between the node issuing the request and the target node use the information in their respective routing tables and forward the data packets. This routing strategy is well adapted to ad-hoc networking and dynamic changes of the network. However, the nature of this routing strategy requires that each node stores information about the active surrounding nodes. This means that "hello messages" need to be sent to keep the stored routing information up-to-date.

JSR82 specifies the use of Bluetooth in J2ME-applications. This standard categorises the different nodes in the Bluetooth network as either a master or slave, where one master and seven slaves constitute a pico-network. Several pico-networks make a scatternet. TVD is thought to be a peer-to-peer service where each node in the TVD-network is equal with regard to the master/slave-philosophy. JBAN is a java.net project utilising JSR82. This library makes it possible to establish unlimited Bluetooth connections without the need for the end user of JBAN to explicitly think about the master/slave-issue. This library also contains functionality to deal with routing and load balancing.

We will be using the JBAN library in our prototype application. This library will work as the network interface to the rest of the application. This simplifies the implementation of the prototype application significantly. The JBAN project is released under the Lesser General Public Licence (LGPL).

3.4.3 Load balancing

An important issue in the context of file sharing is load balancing. A network may consist of several nodes sharing the exact same files. It is desirable that requests for these files are distributed among all the nodes sharing these files. Firstly, this reduces the load each node is exposed to. Secondly, it is more likely to avoid popular routes through the network which may cause traffic jams.

The article '*Dynamic Load-Aware Routing in Ad hoc Networks*' [6] presents a routing protocol named Dynamic Load-Aware Routing (DLAR), which supports load balancing. This protocol uses the load of intermediate nodes of a given route as the main route selection metric. The active routes in a given network implementing this routing protocol are monitored and alternative routes are constructed when a load threshold is exceeded. DLAR uses the number of data packets buffered on a given node as the main criteria for selecting routes. Load balancing is fulfilled by using the least-loaded routes.

Associativity-Based Routing (ABR) is the only major ad-hoc routing protocol available today that uses the load of nodes as a metric to decide which route to use. This protocol uses the number of routes a node is a part of as the indicator for routing load. However, this routing load is only used as a secondary metric.

We will not reason more about load balancing for TVD at this point. We will be using the JBAN library to deal with this issue. Construction and implementation of explicit load balancing would be somewhat out of scope of this prototype application.

3.4.4 Security

As with almost all communication technologies, also TVD has its problem, or rather to say more difficult, areas. These areas are:

- File security
- File integrity
- Routing of data
- Unit security (phone, PDA)

Basically we have 3 places thing can go wrong; in the protocol layer, the application layer, and the user.

When it comes to the main transmission of data, the bits and bytes between units, we in this project relies on the built in security features of Bluetooth. This protocol is under constantly development, but offers all the basic security we need. Its communication and authentication is encrypted with an 128bit key, and it uses the SAFER+ algorithm. Consider the calculation power available in mobile devices; this is satisfying for most communication.

The application layer is the main base of the security we will add to the program. This section is not completely planned yet, but we have taken a couple of decisions. To start with, we will only let users be able to download data. There will be no way of pushing information onto a unit. Allowing pushing of data would make the security much weaker, and also it would be more challenging for the user to configure the software the best way. The other issue we have decided on is that we need some sort of group access. So you are given the choice between sharing to all users nearby, and only to those the information are meant to.

The last problem maker is maybe the hardest to protect the users from, themselves. Their ignorance and non thinking can lead to all possible situations, at this part there are currently much to come.

3.5 Why TVD?

3.5.1 Purpose and usefulness

Ad Hoc networking is basically a result of the demands modern human beings has to daily life technology. We want communication and information when and where we find ourselves, and we don't want to have to rely on an external part, which just offers a little service. Ad Hoc networking it selves has a variety of usage areas. Some examples are:

- Dating services
- Chatting
- Customer oriented sales / On demand
- Work situations out in the field:
- Emergency areas
- Military operations
- Hunters out in the wood
- File sharing
- Media streaming services
- Mobile game play

It's just the imagination that limits the service possibilities and usage areas of Ad Hoc networking. It's a very strong basis everywhere we want to communicate between mobile units.

Our application, TVD, will offer the users the possibility to easy share different type of files between multiple media devices, wherever and whenever they want to. It will be easy, intuitive and effective. Then who could be interested in using such a service? Well, here there are a lot of possibilities. You and your friends wanting to share pictures or music, colleges at work updating each other in front of a meeting, grandmothers meeting to share cookie recipes, or the lecturer that wish to share his presentations and maybe some background information to the audience. In all there are a lot of possible situations where we wish to share information as files.

One of the big advantages of the TVD is that you can communicate between different sort of equipment. The only demand is that it can communicate with the Bluetooth protocol. You can transferee between PC, PDA, cell phone, MP3 players and so on.

An important sales argument for TVD, is that it also will remove the need of one of all these electronic equipment we are carrying around today, the memory stick. Today we have one portable unit which integrates more and more functions, the cell phone. We can talk, listen to radio, take pictures, listen to music, play games, and so on. Today almost every cell phone is made with the possibility to contain a memory card, so why just don't use this space for file transport. This is of course also possible today, but it's not in common use reason to difficult GUI's and applications that desires much of the users. Our plan is to make this simple.

3.5.2 Usage scenarios

TVD has a usage area for as good as everyone wanting to share or transport data between multiple mobile units. Here are some examples:

3.5.2.1 Scenario 1

Ole is an important and busy businessman, which shuttles a significant distance to and from work everyday. He has much to do and is depending on being able to take some of the work home for the night, but since the travelling time to and from work takes up much time, he also wants to be able to some work at the train or bus. His work also contains sensitive information, so transfer over the Internet is not the best. Ole has earlier used both laptop and memory sticks to transfer data, but with the stick he can't work while he travels and the computer is big and much more than he needs for some reading and writing. The solution for Ole is TVD. At the office he easy transfer files to his PDA / cell phone lying in his jacket. No need for cables or installations. While travelling he has lots of space even is the train is full of people. And he can both read and write to his documents. At home Ole just runs TVD at his desktop computer and easy transferees the work to this, while the phone still leys in his jacket.

3.5.2.2 Scenario 2

Ole from scenario 1 also often has to go to different meetings in his workplace. In front of and during these meetings there is a lot of document trading going on. Today you have to make a print out in X number of copies or share a link to the documents. This is both expensive, time taking and either these links are forgotten or the paper thrown away by someone don't want anything more to carry around. Then they got TVD. Quick and easy Ole shares the documents current to the meeting, and those wanting them can grab what they want. Since this also can be documents containing sensitive information, Ole makes use of TVD's build in group access features, which only gives users with the right access rights, the possibility to download the files.

3.5.2.3 Scenario 3

Elisabeth has been on a class trip in for the whole weekend, and now they're all on the bus back home. Naturally there's been taken a lot of pictures during the weekend, and everybody wants to see what the eager photographer Elisabeth has gotten. In her cell phone she has a big memory card containing tons of pictures. Had this been the good old days, everyone would have had to copy each image one in a time from phone to phone, but now we have TVD. Easy Elisabeth shares all the pictures with access for everyone (well, she has nothing to hide) and the copying starts. The speed is good, since TVD each time

gets data from the nearest and fastest unit in the network. This way, Elisabeth's phone doesn't have to serve all the users downloading. The result is quick and easy sharing, and satisfied friends.

3.6 Prototype realisation

This chapter describes a realisation of the TVD Concept, in terms of high-level design and technologies. Chapters 3.6.1 through 3.6.3 outline the design; chapter 3.6.4 gives an overview of the technologies used; Java and Bluetooth.

3.6.1 TVD Context

Figure 14 depicts the system context of the TVD application. A user interacts with a local device, such as a mobile phone. The local device in turn interacts with a set of remote devices, which represents remote users.

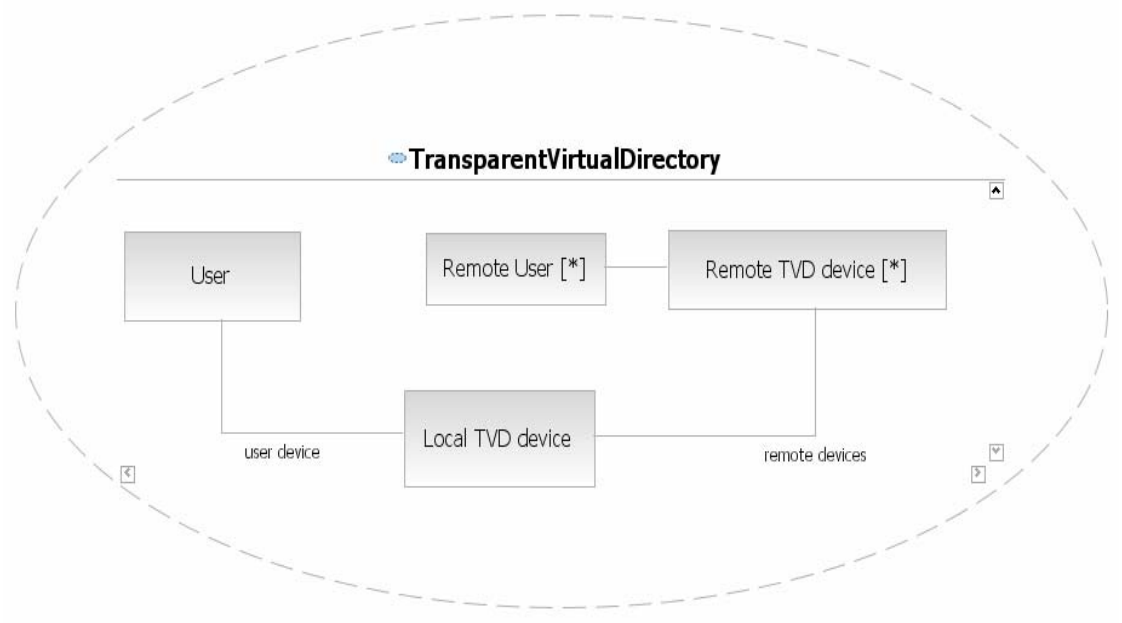


Figure 14 Context of VTD

This structure is the basis for managing the ad-hoc communication for exchanging files in the TVD.

3.6.2 High-level application structure

Figure 15 shows the high-level structure of a *TVDUnit*, i.e. which represents the TVD application that can be installed on mobile devices.

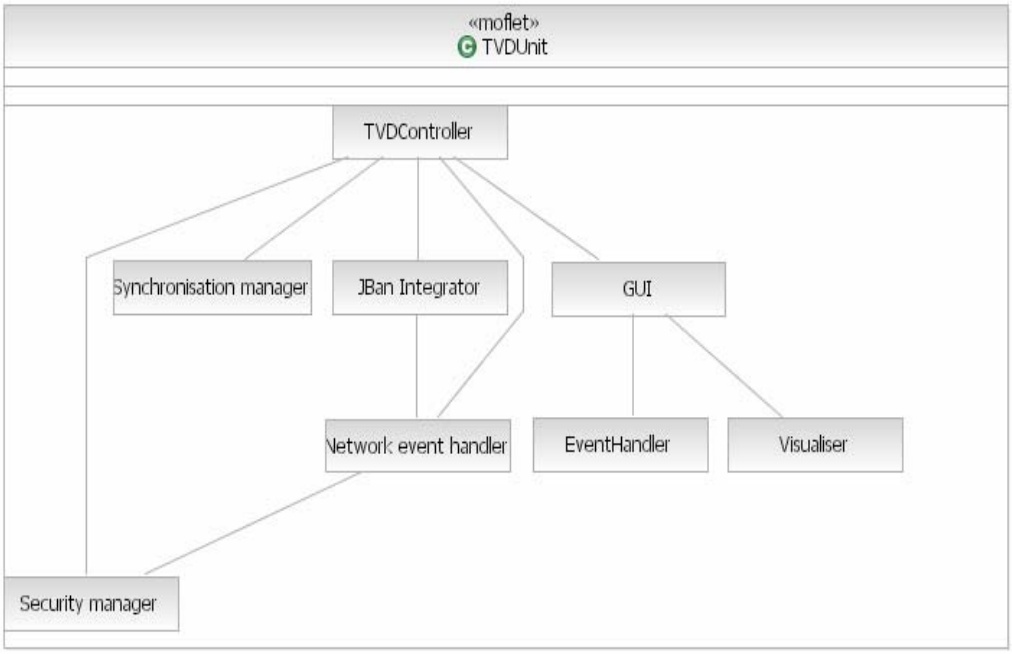


Figure 15 Structure of a TVD Unit

- The TVDUnit is a *Moflet*, which is an application type designed for mobile devices. The Moflet includes all the other system components.
- The *TVDController* handles the main logic of the *TVDUnit*. It initiates execution and manages the communication among the other components. It is responsible for managing the communication with other TVD Units using the JBan Integrator.
- The JBan Integrator handles the Bluetooth communication by integrating with the JBan components (3.6.4.3).
- The *GUI* part handles the user interface aspects of the application using two additional parts: The *Visualiser* provides the presentation layer for the application, such as menu choices and file list showing. The *EventHandler* handles user events, such as *'get file'*.
- The *Network Event Handler* manages network events from the Bluetooth network, such as probing for devices near by or devices going out of range.
- The *Security Manager* provides a placeholder component for security aspects of the TVD, as described in 3.4.4. No security aspects have been implemented at this point.

This structure is implemented using Bluetooth, Java (Java 2 Mico Edition – J2ME), and the JBan library. The details of the implementation are documented by the code.

3.6.3 Limitations

The survey of the TVD concept, presented in chapter 3, addresses several issues concerning routing, load balancing, replication of files and security. We have enforced limitations to our prototype application due to the time constraint and scope of this project. Therefore the TVD-prototype utilises a subset of these issues. The limitations are described in greater detail below.

We have chosen to use JBan as a third-party library for dealing with routing, and to avoid the server/client issue in the Bluetooth specification. This has resulted in simplifications in the implementation. An example is the File Awareness Model. In the TVD concept illustration the File Awareness Model has been presented as a combination of shared files and the corresponding addresses and other routing information to be able to acquire these files. However, since our prototype application utilises the routing functionality of the JBan library, the File Awareness Model is reduced to a file-address list. In other words; the File Awareness Model contains an overview of the files that are shared in the TVD network and the nodes

that have ownership of these files. This implies that all the devices in our prototype TVD network will share a common File Awareness Model.

For instance, the TVD concept usage scenarios describe TVD applications utilising different strategies like distribution of files to several devices in the network. This distribution of files, file replication and load balancing has not been implemented in the prototype. We have made a restriction that says that only one of the devices in a given TVD network can share a given file (with a given file name) at any given time. In other words; every file name presented in a TVD network is unique. This is coherent with how a regular recursive file system works; a given folder is only capable of linking one file for each file name (taking capital letters into consideration). This restriction has to be preserved by the users of the prototype application.

Security has not been implemented in the prototype. However, there is implemented some support, in the prototype, for introduction of user groups. At present, we do not use this functionality and consider every attending node in a TVD network as member of the same default group.

3.6.4 Technologies

This section summarises the technologies used for the TVD Concept Application; Bluetooth [12]; J2ME/Java, and JBan. Bluetooth was a natural choice as network technology for our project, since Bluetooth over the past 4-5 years has become the de facto standard for short-range, wireless, ad-hoc mobile networking and nearly every mobile phone being shipped these days has a built-in Bluetooth device.
Just as with Bluetooth, J2ME (Java Platform, Micro Edition) has more or less become the de facto standard for platform independent development of applications for mobile / embedded devices. It was therefore also natural to choose J2ME as the "target platform" for our example application's prototype.

3.6.4.1 Bluetooth

What is Bluetooth?

Bluetooth is a wireless standard for connecting devices, using short-range, low-power, inexpensive radios. The specification is managed by the Bluetooth Special Interest Group (SIG) founded by Ericsson in1998, together with IBM, Intel, Nokia and Toshiba, later joined by more that 2000 member companies. The latest version is 2.0 was released in 2004, and supports up to 3 Mbps speed. The radio operates in the licence-free 2.4 GHz ISM (industrial, scientific, medical) band, using 79 channels, each 1 MHz wide.

The original idea was just to get rid of the cables of the previous wired communication protocols, but the standard soon expanded in scope and, at version 1.1, at 1500 pages, had more than 13 profiles (applications / use cases in UP terminology), like file transfer and data synchronization, built on top of more than 9 protocols.

Why Bluetooth?

In this project, we want to experiment with ad hoc networks built on the Bluetooth radio technology. However, communication protocols built on radio of infrared waves is nothing new, and we have some alternative RF (radio frequency) technologies to consider. Below, we will briefly compare Bluetooth with the two main alternatives, infrared networks (IrDA) and 802.11* (Wi-Fi, WLAN), and point out the reasons why they were dismissed.

IrDA is similar to Bluetooth in cost and range characteristics, but requires line-of-sight (cannot penetrate solid objects), supports only point-to-point communication (the master of a Bluetooth Piconet can transmit to up to seven slaves simultaneously, half of the time) and has somewhat shorter range, usually only up to one meter.

802.11 is too expensive, both in money costs (three times as much) and energy consumption (uses five times more power), according to www.bluetooth.com. Neither has it gained widespread use on small,

handheld devices yet. This is probably changing in the next 6-12 months, when the first cell phones with built-in Wi-Fi arrive at the market.

There are also a lot of other RF technologies, like Ultra-WideBand, Certified Wireless USB, WiMAX, etc., that deserves further study, but most of these probably lacks widespread deployment on smaller handheld devices, or only works in specific geographical regions.

Drawbacks with Bluetooth

Bluetooth introduces some problems as well. It seems more difficult to use than 802.11, where one can build on a base of std. IP protocols. Bluetooth also has rather serious interference problem with WLAN, since they use the same radio frequencies. This was at least a known problem in version 1.1, where some user companies reacted with banning Bluetooth use in Wi-Fi zones (the forthcoming versions of Bluetooth might solve this, due to changing radio frequencies).

The present versions, 1.1 and 2.0, has a smaller data rate than both IrDA and 802.11, at 1 or 3 Mbps, but this also gets addressed in the two next generations of the std., which aims at 10 and 100 Mbps speeds. Compared to IrDA, Bluetooth is less secure from eavesdropping, since radio waves may penetrate building walls. And finally, Bluetooth lacks a single, open specification, managed by a neutral body (like the 802.* specs of IEEE). At present there exists one family of specifications at the SIG, and one overlapping set at IEEE (802.15.*), but these might converge in the future.

3.6.4.2 J2ME

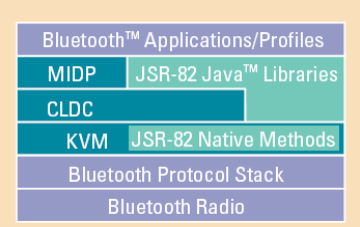
What is J2ME?

J2ME, the Java Platform Micro Edition, is the mobile and embedded devices' Java platform [15]. J2ME is basically made up by a set of technologies, specifications and APIs targeted at application development on consumer and embedded devices, and employs a smaller virtual machine tailored for resource constrained devices. Most of the components in J2ME are small-footprint subsets of Java SE (Standard Edition) components.

Why is J2ME suitable for us?

J2ME was found to be suitable for our needs for the following three reasons:

Provides a Java Bluetooth API that hides much of the complexity of Bluetooth-protocols and the Bluetooth stack and hence allows us to focus on the application being developed [13].



The Java Bluetooth API in J2ME is an open, non-proprietary standard for Bluetooth application development and hence relieves us from fear of vendor lock-in in our choice of technology (as long as the devices support Bluetooth and J2ME, they can run on our application).

J2ME is well documented and in wide-spread use, and can almost be viewed as a de facto standard for mobile application development.

3.6.4.3 JBAN – A possible pain reliever

Although the J2ME and the Java Bluetooth API hides much of the complexity with Bluetooth, there are still some headaches left for the application developer. Firstly the process of forming a scatternet out of piconets is usually not dynamic and practical, and the scatternets formed are not robust. Secondly, the application developer has to give significant consideration to routing and network management details during application development.

JBAN is a newly "born" open source Java library for Bluetooth networking which aims to remove a lot of these headaches from the developer and allow him/her to focus on the actual application being developed. JBAN provides a new way to form an ad-hoc network at the Java application level so that routing and network management is performed at high level. The result is more robust networks (where

there is no distinction between master and slave nodes in contrast to standard piconets). JBAN also allows unlimited devices to form a network dynamically.

In our TVD concept application development, we have used JBAN for providing the basic bluetooth networking functionality.

3.7 Proof-of-concept implementation

In this section we describe our proof-of-concept application that was implemented in order show that the prototype realisation is feasible.

3.7.1 Introduction

The proof-of-concept application, hereby referred to as the POC, was implemented using Eclipse IDE together with the EclipseME-plugin (www.eclipseme.org), which simplifies development of J2ME applications, and Sun Wireless Toolkit (used for emulating mobile devices for testing the POC). The JBan JAR-file was linked as a library into the Eclipse-project. Three packages were created:

- no.uio.ifi.inf5261.tvd - containing the main application classes and business logic
- no.uio.ifi.inf5261.tvd.network - containing the network
- no.uio.ifi.inf5261.tvd.ui - containing the user-interface classes

The implementation of the POC consists of several Java classes, namely `TVDMain`, `TVDAppLogic`, `FileAwarenessModel`, `NetworkInterface`, `UserInferface`, `PersistentStorage` and `Visualiser`. In the further explanation we will use some of these classes as a natural abstraction for the different functionality, and explain how/when the different interactions/events in a TVD network are performed.

3.7.2 Establishing the Bluetooth Ad-hoc Network

The first step in the implementation of the POC was to utilize JBan and establish the Bluetooth Ad-Hoc Network which TVD will run upon. This as done via the NetworkInterface-class by accessing a static class in JBan that forks out a new thread from the application in order to handle discovery of other devices and the register the JBan network service on the Bluetooth nodes discovered.

JBan, as expected, did make the detection of other devices and formation of the high-level network easier for us, but we ran into the problem that nodes that left the network still appeared as available to our application. We learned from the developer of JBan that, at the time of writing, network management has not yet been fully implemented in JBan, so changes in the network are not detected. Without this, mobile ad-hoc networking (the way we want it) is not fully possible to implement using JBan. But as soon as the JBan library is expanded to include network management (i.e. becomes able to detect that nodes come and go), we will have what we desire. This is purely a matter of maturity of the JBan library.

3.7.3 The Application Logic

Thought chain of events

A thought chain of events when using the POC application (brief explanation)

- A user specifies files to be shared in the TVD network (and eventually the corresponding group, if not the default group is used).

- The user starts the POC application and finds other TVD devices present (if any).
- A 'File Awareness Request' is issued to all the other devices.
- Several 'File Awareness Response OK' are returned, and perhaps some 'File Awareness Response Failed' (see description further down why this may be sent).
- The POC application updates its File Awareness Model based on the received responses.
- The user can now see what files that are shared in the network.
- The user may select one of these files for download. This causes a 'File Request' to be sent to the corresponding address of the specified file.
- If there is no immediate change in the network, a 'File OK' and the file will be returned by the remote device. The file transferring is abstracted by the JBan library. The file will be stored in the persistent storage.
- On the other hand, if there has been an immediate change in the network the device owning the specified file may have left the network (or lost it connection), or does not share the file any more. If the remote device is no longer available an error occurs (covered by JBan, exact error not known at this point). If the file is not shared any more (in the given group) a 'File Not Found' is returned from the remote device.

The Business/Application Logic

The core TVD classes of the POC comprise: `TVDAppLogic` and `FileAwarenessModel`, together with several classes in the JBan library. We choose not to present the JBan specific classes in this section even though device discovery and the likes are central principles in the TVD concept.

- `TVDAppLogic`

`TVDAppLogic` controls the course of all interactions between TVD nodes. Six essential interactions/events have been addressed. These are:

Event:	Initiated/happens when:	Explanation:
File Awareness Request	A device is connected to the network for the first time, or need an update of the shared files available.	A file awareness request is issued to the devices in the network which send a copy of their own FAM to the requesting device. (A future implementation of the POC or an application fully conform with the TVD concept would only send this request to the nearest nodes. The information about hops could be stored in an extended FAM.) The FileAwarenessModel class contains different methods for updating and extracting the given node's FAM-information. When a device get a 'File Awareness Request' from another device, it calls its `extractFAM()` – method and sends the returned information to the requesting node if the FAM is nonempty.
File Awareness Response OK	A device has issued a 'File Awareness Request' and gets a response from a node with a FAM representation whose	The device calls its `updateFAM(…)` method in the `FileAwarenessModel` class and updates it with the new

	size is different from zero.	information.
File Awareness Response Failed	The device receiving the 'File Awareness Request' has an empty FAM or the requesting device can not be authorised with regard to group attendance.	The POC application has implemented support for groups and if the requesting node does not provide the correct group identification, a 'File Awareness Response Failed' may be issued from the responding device.
File Request	A device in the network wants to transfer (download) a given file.	File name and group identification are used to address the correct file. (A device is able to attend several groups.)
File OK	The device that owns a given file has granted downloading of the file.	The file is transferred to the device that requested the file and stored in the persistent storage.
File Not Found	A specified file is not found on the given device.	This event occurs when the requested file does not appear on the specified device. It also happens when the requesting device is not a member of the group in which the owner of the file belongs.

- `FileAwarenessModel`

The `FileAwarenessModel` class keeps and updates a set of <file name, address> pairs. This is implemented using a hashing structure. A given file name associates to an address. The `FileAwarenessModel` contains methods for easily updating and extracting the information stored in the FAM and verification of file awareness, in other words; if a file is available in the network (and the address to the device sharing this file).

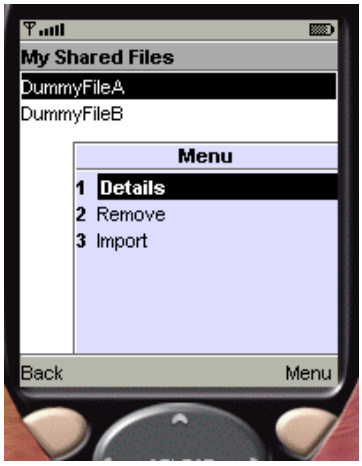
- `PersistentStorage`

`PersistentStorage` stores the shared and transferred files using the J2ME `RecordStore`-feature. This feature creates a sand box for all the files associated with the TVD prototype. The implementation of `PersistentStorage` differentiates between shared files and files stored in the `RecordStore`, but not being shared.

3.7.4 User Interface

A simple user interface was created for the POC in order to demonstrate the user interaction. (Note that this is just a proof-of-concept; better approaches to the user interface probably do exist).

When the application launches the user is presented with a main menu with options. Upon selecting "authorize devices" the user can initiate a search for devices, and among the devices discovered by JBan, authorize one or several devices. Mutual authentication among devices will result in the devices being able to share files over TVD. This concept can be expanded to formation of groups so that a node can be a member of multiple TVD-groups.



The other menu choices lead the user to screens for viewing their local shared files, remote files that are available via TVD and a overview of active, paused and broken file transfers. (Some screen-shots are missing).

3.7.5 Results from the proof-of-concept implementation

- JBan is still a bit immature as it lacks complete network management. According to the developer this is just "around the corner", so in the near future JBan will be able to do what we desire from it.
- JBan does ease a lot of the pain in Bluetooth network programming (discovery, service registration, routing, etc).
- TVD is fully realizable on the technology platform we have chosen (J2ME+Bluetooth+JBan).
- A smarter UI, sercurity model and other advanced network and usability issues still need some consideration and planning.
- Potential users should be included in further development (especially of the user interface).

4 Related work

4.1 TVD and collaboration

The nature of mobile ad-hoc networks is to spontaneous establish connection between units that are within range of each other. They have frequently been used for creating networks to support some collaborative task, such as a rescue operations or military tactical operations. As such, it is a well suited technology for building dynamic collaborative systems. We have also seen ad-hoc networking being used for collaboration for entertainment purposes, as described in chapter 4.2.

The TVD concept application developed as part of this project also provides a collaboration metaphor for its users. The collaboration of ad-hoc bluetooth units (e.g. mobile phones) requires proximity of the people owning the units; thus, it may also be physical social collaboration involved when using the TVD.

In [7], Harrison and Dourish describe place and space as important collaborative concepts. A space represents a physical location, such as a meeting room. They define the concept place, which represent an awareness of the meaning of a place, i.e. that a place is defined by how it is used. They identify different kinds of places, such as space-less places (places with no physical location, but a virtual one) and hybrid places (a mixture of physical and virtual spaces). The TVD concept application defines a *place* where its users meet to share content. Since the users share a physical proximity when meeting at the TVD place, it can be viewed a place with multiple dynamic spaces (i.e. it is not space-less).

In [9], Schilit et al describe context awareness issues and application areas and characterise how different kinds of technologies influence awareness and behaviour with respect to social context and communication. They do not address the field of mobile ad-hoc networks, which brings a different aspect to context awareness. Social awareness can be established implicitly triggered by the technology awareness of established ad-hoc networks.

In [10], Kortuem et al address these aspects in what they call impromptu collaboration, and exemplify this with an ad-hoc MP3 file sharing scenarios. With respect to the Virtual Transparent Directory application, proximity awareness can be triggered by established ad-hoc connections and the visibility of specific files, or types of files. A particular file (e.g. a picture, music file, interesting text), may influence the ad-hoc participants to engage in more specific interaction, either using the mobile ad-hoc network or by engaging communication in a physical space (with the physical person).

In [14], Agre argues that the concept of place is changing due to the emerging of technology for mobile and ubiquitous computing. Bluetooth is used as an example of an important technology that is important because it initiates communication based on physical proximity, which invites a style of thinking grounded on geographic locality as well as embodied interaction. These technologies enable an 'always-on' world that will change the expectations of interaction and thus the very concept of place. In this regard, TVD is a provider of place changing technology.

4.2 Related technologies

BEDD [8] is a technology company which develops social networking software including wireless community applications and services. It provides a mobile ad-hoc platform for socializing and sharing. It is built on platform-independent technology and currently supported on Symbian Series 60. It uses Bluetooth, but can be integrated into any wireless platform, such as WiFi or WiMax, according to BEDD. It is supported on a wide range of phones, and requires 1MB of memory, as well as Bluetooth support. BEDD is free to use.

Compared to the approach taken by our project, BEDD provides

more mature an advanced functionality. They seem to have solved the technical challenges to allow for ad-hoc file sharing (in their BEDDpic tool). The major disadvantage with BEDD is the relatively big footprint.

5 Summary and conclusion

We have described the Transparent Virtual Directory (TVD), a concept application to test mobile ad-hoc networking scenarios using Bluetooth technology. The TVD has been designed and implemented in the course of this project.

The TVD provides ad-hoc file sharing capabilities to mobile terminals using Bluetooth connectivity. It allows for mobile devices to connect by proximity, creating a virtual directory in which files are shared between the TVD users.

Our main goal of expanding our knowledge on mobile ad-hoc networking concepts and technologies has been met in the process. We have seen that Bluetooth technology successfully can be used to realise ad-hoc applications. Further, we have seen that ad-hoc networking applications can provide new dimensions to collaboration, by spontaneously creating virtual places where information can be exchanged. We have seen that mobile ad-hoc networking opens for a range of applications using collaborating units with physical proximity. The physical proximity may certain cases represent a limitation.

Several aspects of mobility and ad-hoc networking have not been covered by this project, both technological and social aspects. Possible future work could include a deeper analysis of security and trust in the context of ad-hoc networking in general, and the TVD specifically. Further studies on implications on collaboration and social aspects of this kind of applications could be done. Additionally, work could be done on analysing actual use of the TVD, e.g. through user surveys.

References and bibliography

[1] Magnus Frodigh, Per Johansson and Peter Larsson, Wireless ad hoc networking—The art of networking without a network, Ericsson Review No 4, 2000

[2] Kakihara, M. and Sørensen, C. 2001. Expanding the 'mobility' concept. SIGGROUP Bull. 22, 3 (Dec. 2001), 33-37.

[3] Andrew S. Tanenbaum: "Computer Networks", section 4.6, 4. ed., Prentice Hall PTR, 2003.

[4] Barbará, D. and Imieliński, T. 1994. Sleepers and workaholics: caching strategies in mobile environments. In Proceedings of the 1994 ACM SIGMOD international Conference on Management of Data (Minneapolis, Minnesota, United States, May 24 - 27, 1994). R. T. Snodgrass and M. Winslett, Eds. SIGMOD '94. ACM Press, New York, NY, 1-12.

[5] Hara, T., "Effective replica allocation in ad hoc networks for improving data accessibility," INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , vol.3, no.pp.1568-1576 vol.3, 2001

[6] Lee, S.-J.; Gerla, M., "Dynamic load-aware routing in ad hoc networks," Communications, 2001. ICC 2001. IEEE International Conference on , vol.10, no.pp.3206-3210 vol.10, 2001

[7] Harrison S and Dourish P: Re-Place-ing Space: The Roles of Place and Space in Collaborative Systems, 1996. CSCW/ACM.

[8] BEDD, http://www.bedd.com

[9] Schilit, B.N.; Hilbert, D.M.; Trevor, J., "Context-aware communication," Wireless Communications, IEEE [see also IEEE Personal Communications] , vol.9, no.5pp. 46- 54, Oct. 2002

[10] Gerd Kortuem, Jay Schneider, Dustin Preuitt, Thaddeus G. C. Thompson, Stephen Fickas, Zary Segall, "When Peer-to-Peer comes Face-to-Face: Collaborative Peer-to-Peer Computing in Mobile Ad hoc Networks," p2p, p. 0075, First International Conference on Peer-to-Peer Computing (P2P'01), 2001

[11] Mehran Abolhasan, Tadeusz Wysocki, Eryk Dutkiewicz, "A review of routing protocols for mobile ad hoc networks", Ad Hoc Networks Journal 2, 2004, Elsevier

[12] Chatschik Bisdikian: "An Overview of the Bluetooth Wireless Technology", IBM Corporation, IEEE Communications Magazine. December 2001 WWW: http://mia.ece.uic.edu/~papers/Wireless/pdf00004.pdf

[13] Qusay H. Mahmoud: "Wireless Application Programming with J2ME and Bluetooth", SUN Developer Network, 2003 WWW: http://developers.sun.com/techtopics/mobility/midp/articles/bluetooth1/

[14] Philip E. Agre, Changing Places: Contexts of Awareness in Computing, Human-Computer Interaction 2001, Vol. 16, No. 2, 3 & 4, Pages 177-192

[15] R. Riggs et al, "Programming for Wireless Devices with the Java 2 Platform, Micro Edition", Second Edition, Addison Wesley 2003, ISBN 0-321-19798-4 (chap 7,9,10,14).