Development of communication interface and GUI for Minec 3x

Authors:
Carine Andersson
Andreas Lång

Master Thesis, 20p, 2004
Department of Computing Science
Umeå University

Internal Supervisor: Jan-Erik Moström
External Supervisors: Johan Andersson, Joel Lindau

Abstract

This master thesis report contains a description of our development of a graphical user interface (GUI) and communication interface for a handheld computer named Minec 3x in the development phase. This computer has a very small display and limited input possibilities. Minec 3x is developed by Datalogic AB and is primarily used for warehouse management, goods shipping/receiving and inventory. This is the first handheld that the company has developed which uses Linux as operating system.

To be able to carry out this master thesis some background studies on human computer interaction, GUI guidelines, graphical toolkits and different data transfer protocols were made. From these studies the realization was planned and design decisions made. The first two chapters in this report contain the theoretical background used to develop the GUI and communication interface.

The handheld should be able to communicate over WLAN (TCP/IP), Bluetooth, IrDA, USB and RS232. From the theory studies Z-modem was chosen as transfer protocol for all serial communication over the media Bluetooth, IrDA, USB and RS232. FTP was selected as transfer protocol for all TCP/IP communication over the media Bluetooth and WLAN. The chosen graphical toolkit was GTK+ that runs against the X11R6 graphical server.

Table of contents

1 Introduction

The assignment of this master thesis is to design and develop a graphical user interface (GUI) and a communication interface for the handheld computer called Minec 3x. It was carried out at Datalogic AB in Upplands Väsby, Sweden. Datalogic AB develops handheld computers with built-in barcode scanners, normally used for warehouse management, goods shipping/receiving and inventory. Datalogic AB has in their earlier handheld computers been using Windows CE as OS. However they have decided that their new product should use Linux as OS. This new handheld goes under the work name of Minec 3x. Our part has been to develop a graphical user interface and communications interface for Minec 3x.

The GUI will mainly be used as a service interface, helping the administrator to configure the Minec 3x easily, such as performing time settings, different communication settings for WLAN, Bluetooth, IrDA, USB and RS232 and different kinds of task handling. The communication is foremost intended for data transfer from handheld to a stationary server, wireless or via cable connections. The most important aspects regarding the GUI are the limited input and output possibilities. The screen is only as big as mobile phone display (128 x144 pixels), the keyboard has a limited number of keys and no pointing device is available.

This report consists of three major blocks. The first block contains a study of different data transfer protocols suitable for different communication media. It also includes a comparison between these protocols as well as the final decisions on which protocols to use. The second block will discuss different toolkit and X-window systems for the GUI. This chapter also contains some information about HCI and some guidelines for creating user interfaces. The third and final block describes the realization of the GUI and the communications interface based on decisions made in the prior chapters.

1.1 Objective

Theoretical studies are essential to be able to solve the assignment in a satisfying way. This involves analysis of different user interface aspects and file transmission protocols before an implementation of decided design choices can take place.

The final software is intended to serve as a service-GUI, were the administrator is allowed to configure and maintain the handheld. It's important that the GUI is easy to use either for an inexperienced or a professional computer user. This demands both fast access to frequently used features, good feedback and useful error information from the software. Since the display is rather small, the displayed information must be held as short as possible but still be understandable.
The communication interface must be able to handle data transmissions over the following media: Bluetooth, WLAN, IrDA, USB and RS232.

The requirements specification for this master thesis can be found in Appendix D in this report.

To get a deeper understanding and to be able to decide upon design issues regarding the GUI, a pre-study have been done. It includes the areas human computer interaction, graphical engines and toolkits as well as design guidelines. To be able to design the communication interface a pre-study have also been done on different file transfer protocols that are suitable for the different transfer media.

These two studies together with discussions held with the company's head quarter in Bologna, Italy builds the foundation for the implementation of the final software.

2 Analysis of communication protocols

The final product should be able to transfer data in a number of ways, using different types of mediums. The different transfer mediums the Minec 3x should be able to use are WLAN (TCP/IP), IrDA, USB, RS232 and Bluetooth. To make data transfers as secure and error free as possible, computers use different protocols to communicate with each other. The protocol represents a formal description of the message format and the rules for how these messages should be sent. Different transfer methods require different protocols.

This theory chapter is a study of different transfer protocols and their pros and cons.

2.1 FTP

File Transfer Protocol was developed in the USA in 1969 by the Department of Defense's Defense Advanced Research Projects Agency. It is a standardized method for transferring files between computers over the Internet. FTP has been defined and redefined numerous times by the IETF in a series of standard documents known as RFCs. Today, RFC959 by Postel and Reynolds 1985 is the official FTP standard [1].

FTP uses the TCP/IP protocol to transfer data. FTP is usually used to download/upload files from a server to a workstation but can also be used to update, delete, rename or move files on a server.

The protocol is platform-independent which means that a user can transfer a file from one computer to another even if the computers have different operating systems.

Files can be sent in two different modes, ASCII or binary. ASCII mode is mainly used to transfer files containing text such as html files. Other non-text data such as images and applications are transferred binary.

To use FTP the user must log on to an FTP-server with username and password. The most common case is that users for security reasons only can download files. If upload is allowed it’s often restricted to a specific upload directory.

2.1.1 Transaction overview

To transfer data over FTP two channels are needed: one for control commands and one for the actual data. Both connections support two-way communication; the data connection does not need to exist all the time. The client starts by connecting to the FTP server's service port via TCP/IP. The service port is usually set to port 21. This connection will function as the control line (see figure 46). The FTP-client must identify itself with username and password. Some servers support the use of anonymous clients, this means that the user doesn't need to have a user account and can simply log on to the server using username "anonymous". After this the client sends some FTP commands to specify the parameters for the data transfer such as data port, transfer mode, structure, representation and the nature of file system operation (store, retrive, append, delete etc). The server responds to each of these commands over the controlline. If the connection was set up successfully the client starts to listen at the specified data port. The server then establishes the data connection and starts to transfer data according to the parameters that the client has specified. The byte size for transmission over the data connection is always set to 8 bit/byte independent of the systems internal data representation.

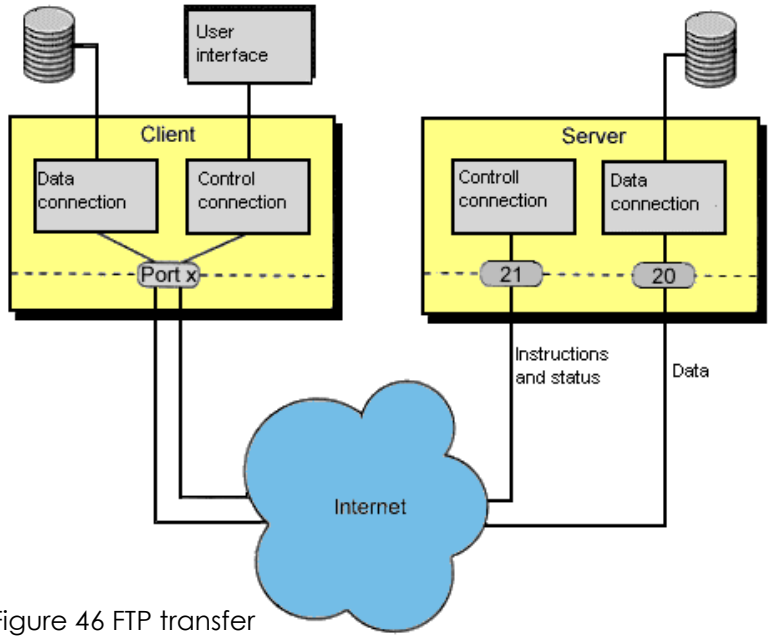

Figure 46 FTP transfer

The scenario described above is called an active transfer. An FTP transfer can also be passive; this is used when the client is behind a firewall and in other similar cases. In passive mode the client is not allowed to open ports for the server to connect to and can therefore not listen for the data connection from the server. To solve this problem the client tells the server to listen for a data connection on a specific port from the client instead. The server starts to listen on the port and the client can establish the data connection. This is thus called a passive data transfer.

It is also possible to use FTP to transfer files between two hosts neither of which is a local host. To do this the local host needs to set up two control lines to each of the servers and then arranges for a data connection to be set up between them. In this scenario the control commands are sent to the local host but all data is transferred between the two servers.

2.1.2 Control commands

When transferring a file between two computers, data is transferred from one storage device at the sending host to another storage device at the receiving host. Because that there are often differences in the data storage representation between the two systems, several commands are exchanged to define how the data should be transferred. More detailed information about these commands is found below.

2.1.2.1 Data types

The TYPE command is sent to specify the representations type. In other words the type defines the byte size for the data so that the receiving host knows how to interpret it. This byte size is referred to as “logical byte size” and is not to be confused with the byte size used for transmission of data over the data connection, called “transfer byte size”.

There are four different types: ASCII, EBCDIC, Image and Local.

ASCII Type

ASCII is the default type and because of this all FTP implementations must accept files with this type. It’s intended for transfer of text files whenever EBCDIC is not considered more suitable. It should be noted that ASCII type only transfers the text and does not take any notice of such things as how the file is formatted (newlines, tabs etc). It is thus not suitable for written documents that are not formatted like an HTML page.

EBCDIC Type

EBCDIC is IBM's eight-bit extension of BCD four-bit encoding of digits 0-9. This type is intended for efficient transfer between two hosts that use EBCDIC for their internal character representation. The only difference between the functional specifications of EBCDIC and ASCII is the character encoding.

Image type

This type is intended for transfer of binary data and for the efficient retrieval and storage of files. It is recommended that all FTP implementations shall accept the image type. The data is sent as contiguous bits that are packed into eight-bit transfer bytes. The bytes must always be eight bits. Because of this it might be necessary to pad the file size to an even number of bytes.

Local type

The data is transferred in logical bytes. The byte size is defined by the second parameter called "Byte size". This parameter is required and there is no default value. Note that the byte size need not be the same as the transmission byte size. If there is a difference between these two sizes the logical bytes are packed in eight-bit bytes, padding is used if needed. When the data reaches the receiver, the host will transform it in a manner depending on the logical byte size and the particular host. This transformation must of course be reversible.

Format control

The ASCII and EBCDIC types also take a second optional parameter. This is to indicate the vertical format control (if any) that is associated with the file. A text file may be sent to a host for one of these three reasons: for printing, for storage and later retrieval or for processing.

If a file is sent for printing the receiving host must know how the vertical format control is represented. It shall also be possible to store a file at a host and later retrieve it in precisely the same form. Finally it must also be possible to send a file to another host for processing without any trouble.

2.1.3 Data structures

When transferring a file, FTP allows the data structure to be set in addition to different representation types. The natural data structure of a file depends on the host that stores the file. A source code file can for example be stored as records in one system and as a stream of characters divided into lines in another system. When transferring a file between hosts with different file representation there must be some way to tell one host the other host's assumptions about the file. The three data structures defined by FTP are file-structure, record-structure and page-structure.

File-structure

File structure is assumed by default if the STRUC command is not set. In a file structure the file is considered to be a continuous sequence of data bytes. If a file is being sent to a record structured host, there is a question of what criteria the host should use to divide the file into records, which can be processed locally. If this division is necessary the FTP implementation should use the end-of-line sequence.

Record structure

Record structures must be accepted for files sent with ASCII or EBCDIC types (i.e. "text files"). In record structure the file is made up of sequential records with a fixed length.

Page structure

FTP defines a page structure when a file that is discontinuous is transferred. These files are sometimes known as "random access files" or "holey files". The special thing about these files is that they can contain other information associated with the file such as a file descriptor, page access controls or both. Sections of the file are called pages in FTP.

To handle various page sizes and associated information, every page is sent with a page header. Each page header has the following fields: *Header length, Page index, Data length, and Page type.* All the fields in the header are one logical byte long except for the field header length; this field is minimum four bytes long. The logical byte size is specified by the TYPE command.

2.1.4 Transmission modes

The final consideration in transferring data is choosing the appropriate transmission mode. There are three different transmission modes: stream mode, block mode and compressed mode.

All data transfers must be completed with an end-of –file, which can be stated explicitly or implied by closing the connection. A transfer can also end with an end-of –record or a “last-page” page depending of the data structure used.

When transferring a file the sending host will translate its internal end of line or end of record denotation into the representation given by the transmission mode and the file structure. The receiving host will perform the inverse translation to its internal denotation.

The following three modes are defined in FTP:

Stream mode

In this transmission mode the data is transferred as a stream of bytes, exactly as the name implies. There are no restrictions on which representation types that are allowed. This mode also allows files with record structure.

Block mode

The files are transferred as a series of blocks. Each of the blocks has a header that contains a count field and a descriptor code. The header is three bytes long; the high byte indicates the descriptor code and the two low bytes contain the count field. The count field accounts for the number of bytes in the block and thus indicates where the next block begins, which makes padding unnecessary.

The descriptor code defines: last block in the file or in the record, restart mark, or suspect data. This last code has nothing to do with error checking in the data transfer. The code is simply used by the sending host to indicate to the receiving host that the data may not be accurate. This code is set when the sending host has experienced errors such as “magnetic tape read error” and similar problems.

Record structures are allowed in this transmission mode and any representation type may be used.

Compressed mode

Compressed mode is useful for obtaining increased bandwidth on very large network transmissions with little extra CPU cost. It can be most efficiently used to reduce the size of printer files such as those generated by RJE hosts.

There are three types of data to be sent: regular data sent in a byte string; compressed data consisting of replications or filler; and control information, sent in a two-byte escape sequence. When a string of n replications of the data byte d is compressed, two bytes are sent. The first byte has its left most bit set to one and the bit after this set to zero. The rest of this byte contains the number n. The second byte is the data byte.

2.1.4.1 Error recovery and restart

There are no methods for detecting lost or scrambled bits in transferred data with FTP. This level of error control is left totally to the TCP protocol. However a restart procedure is provided for the control connection to protect users from big system failures. The restart procedure is only defined for the block and compressed transmission modes. It requires the sending host to insert a special marker code in the stream with some marker information.

2.2 TFTP

TFTP is an abbreviation of Trivial File Transfer Protocol and is a simpler form of FTP. This protocol uses UDP to transfer data on the Internet. Servers that need to boot diskless workstations, X-terminals and routers often use TFTP.

It is designed to be small and easy to implement. As a result of this TFTP lacks most of the features of regular FTP. TFTP can only read and write files from or to a remote server. It cannot list directories and in the current version of the protocol there are no provisions for user authentication. As in other Internet protocols data is sent in 8 bits bytes.

Since TFTP uses UDP and UDP is implemented in the Internet protocol all packets will have an Internet header, UDP header and a TFTP header. Additionally, the packets may have a header to allow them through the transfer medium.

TFTP supports three different transmission modes: netascii (ASCII with the modifications specified in the “Telnet Protocol Specification”); octet, raw 8 bit bytes; mail - netascii characters sent to a user rather than a file. Note that the last mode is obsolete and should not be implemented or used. In addition to these modes other modes can be defined as long as a pair of cooperating hosts supports them.

2.2.1 Transaction overview

To start a file transmission the host sends a request to read or write a file to a server. This request also functions as a connect request. If the server grants the request it sends an acknowledgement packet and a connection is opened. If the connection is refused an error packet is sent. In order to create the connection, each end of the connection chooses a TID (transfer identification) for itself. This ID is used for the duration of the connection. The transfer identification that is used during a connection should be randomly chosen so that the probability that both ends have chosen the same number is as low as possible.

The data that is sent over the connection is divided into blocks of 512 bytes and each data packet contains one block. Every packet that is sent is associated with the two hosts TID:s. If the source TID doesn't match, the receiver will discard the packet as erroneously sent by some one else. An error packet should be sent to the source of the incorrect packet if this doesn't disturb the transfer.

When a packet is received it must be acknowledged with an acknowledgement packet before the next packet can be sent. This procedure is called stop-and-wait. Because no data is sent without the previous packet being acknowledged the data packet also serves as an acknowledgement. If a packet gets lost the host waiting for the packet will timeout and may resend his last packet (data or acknowledgement packet). This procedure causes the sending host to retransmit the lost packet. The general acknowledgement packet contains the block number of the data packet being acknowledged. Each data packed is associated with a block number. The block numbers are consecutive and begin with one. This restriction allows TFTP to determine if the packet received is a duplicate of the last received packet and then discard it.

Since every packet must be acknowledged before the sender can transfer another one the sending host only has to keep one packet for retransmission, since the stop-and-wait procedure guarantees that all older packages has been received. Both machines in a transfer are considered senders and receivers. One sends data and receives acknowledgements and the other sends acknowledgements and receives data.

A packet containing less than 512 bytes of data signals termination of transfer. The host acknowledging the last data packet may terminate its side of the connection after sending the final ACK. Delaying is however encouraged; this means that the host sending the final ACK will wait for a while before terminating in order to retransmit the final ACK if it has been lost.

Most errors will cause termination of the connection. Sending an error packet signals an error. This packet is not acknowledged or retransmitted, so the receiver may not get it. Timeouts are used to detect a termination of the connection, in those cases when the error packet does not reach the other end of the connection. There are three types of events that can cause an error: not being able to satisfy the request (e.g. file not found, access, violation, or no such user), receiving an incorrectly formed packet and losing access to a necessary resource (e.g. disk full or access denied during transfer). TFTP only recognizes one error condition that does not cause termination. This is when the source port of a received packed is incorrect. If this happens an error packet will be sent to the originating host.

This protocol is very restrictive, in order to simplify implementation. For example, the fixed block lengths make allocation straightforward and the stop-and-wait feature provides flow control and eliminates the need to reorder incoming data packets.

2.3 Secure FTP

The Internet was not a public resource when FTP first was developed; security was therefore not a problem. As the Internet has become a public resource and more widely used the need for secure transfers has grown rapidly. Today it is almost impossible to run a business without having to move files with sensitive information between computers.

The most serious security problems with regular FTP concern authentication and file transfer. Authentication is based on password logins, which easily can be guessed. Since both username and password is sent as plain text to the server, anyone using a sniffer can see this data and use it to logon to the server. Files are also sent in plain text, which makes it possible for cybercriminals to access the information. This may be credit card information or other classified data. Data sent via FTP is also vulnerable to man-in-the-middle attacks. This is when a person intercepts the transmission, alters the data and then sends it back on their way to the recipient. Another scenario where problems may occur is when a website is transferred to the web server using regular FTP. A hacker can use a sniffer to get the webmasters authentication data and then later on logon to the server and edit the website with digital graffiti or fraudulent information. Insecure FTP transfers are one of the main reasons why websites get hacked [19].

The term “Secure FTP” includes two different types of secure file transfer protocols. First there is an enhancement to standard FTP (RFC959), which uses the same FTP commands (and protocol) over secure sockets i.e. over SSL/TLS. This protocol is variously known as FTPS, FTP-SSL or FTP-over-SSL. The second protocol is known as SFTP, which also provides secure file access, but which is not related to the standard FTP protocol. SFTP is implemented using SSH, a suite of secure connectivity tools (when used with SSH2 this is known as SFTP). The primary purpose of SSH is to enable users to connect to a remote machine over a secure connection. Note that FTPS and SFTP are completely different protocols and are not at all related, although they have very similar names.

SSL is a protocol designed and implemented by Netscape. Version 3.0 was used as the basis for the TLS standard, version 1.0 (RFC 2246). The differences between SSL 3.0 and TLS 1.0 are not substantial, but the two protocols do not interoperate. However, TLS 1.0 supports a mechanism to back down to SSL 3.0. The term “SSL” is generally used so that it is interchangeably with “TLS”. In this report the term “TLS” will be used from here on [3].

2.3.1 FTPS vs. SFTP

As already noted, SFTP is built on SSH2, while FTPS is standard FTP over a TLS connection. This is the greatest advantage with using FTPS. Since FTPS is an extension to an existing FTP infrastructure, most commercial servers and many open source servers support it. There is also no need to run additional servers when using an FTPS server since all FTPS servers support regular FTP. Nor does FTPS need to open any additional ports because it can use the same ports as standard FTP.
These are all advantages that come with using the standard FTP protocol as a base, which SFTP doesn’t. The rest of the significant differences between FTPS and SFTP come from differences between TLS and SSH. It should be pointed out that the goals and features of both TLS and SSH are similar, but both protocols have some pros and cons.

FTPS allows different levels of security. Client and server negotiates which security level should be used for the transfer. This means that in FTPS server authentication is optional.
In other words the protocol supports fully anonymous operations, in which neither side is authenticated. This makes the connection vulnerable to man-in-the-middle attacks.

In SFTP server authentication is mandatory, which protects against such attacks. Of course, it is always possible for a client to skip the step of verifying that the public key provided by the server actually belongs to the entity the client intended to contact. But SSH at least demands authentication. On the other hand SSH creates problems when a server wants to give access to a SFTP client but not SSH access. This is possible but very tricky. Great caution must be taken when the server is set up. If done incorrectly, users on machines with a SFTP client installed will be able to log on to the server with an SSH client and execute commands. This is not a problem with FTPS.

FTPS manages authentication with X.509 public-key certificates on both the server and the client side. This makes FTPS a bit more awkward since it requires a functioning public-key infrastructure to be in place. Certificates are also more complicated to generate and manage. The use of certificates, however, allows FTPS to take advantage of the “chain of trust” paradigm facilitated through Certificate Authorities. This paradigm makes it possible for two entities to establish a trust relationship without directly exchanging security information.
SSH uses keys rather than certificates to manage authentication. Keys are easier to create and manage but the SFTP client must install the keys on the server.

2.4 Xmodem

Xmodem was originally developed by Ward Christensen in 1977 to allow error free data transmission between CP/M computer systems. This protocol transmits data using standard serial telecommunications links. It is one of the most widely used protocols. Xmodem works over dedicated links, line drivers or modems and is a stop-and-wait protocol. Because of this Xmodem is not very fast. The data to be sent is placed in 128 byte packets (original version of Xmodem) and transmitted with a checksum. The first implementation of Xmodem used an eight-bit checksum, and is often referred to as “Checksum Xmodem”. Since it is possible for bad data to pass as error free with so few checksum bits the protocol incorporated the use of CRC16 to form the checksum. When using an eight-bit checksum it is expected that one out of 256 incorrect packets will have a matching checksum and thus erroneously be assumed to be correct. The same numbers when using a 16-bit checksum is that one out of 656536 incorrect packets will pass as correct even though it is not. The new implementation of Xmodem is called Xmodem CRC and the checksum method is very reliable. Due to the use of more checksum bits, the packet length used by Xmodem CRC is 133 bytes/packet instead of 128 bytes. Xmodem CRC has a slightly different initiation phase than Checksum Xmodem and is designed so that if the transmitting host does not support Xmodem CRC, the protocol falls back to using Checksum Xmodem.

Because of the fact that Xmodem was designed to transfer data between CP/M systems; its design does not always work well on other computer systems. The problem arises from the fact that Xmodem requires eight-bit communication links and it uses all data combinations between hex 00 and FF. This may work well on most point-to-point links, but can cause problems over networks or data concentrators that reserve some of the characters for their own use. The problem with this is to know where the actual data ends. To solve this Xmodem uses the Ctrl-Z flag to signal the end of text mode transfers.

This protocol has one packet type – data. Responses are single characters ACK, NAK and CAN. In everyday use this works quite well but under very noisy line conditions it is possible to experience transfer fails. Xmodem sends or receives one file each time it is run. When it encounters a name collision Xmodem can handle this either by aborting the transfer, replacing the file or appending the file.

Ymodem and Zmodem are enhancements of Xmodem. Ymodem is very similar to Xmodem; the only differences are that Ymodem sends 1024 byte packets and support batch file processing. Because of the similarities between Xmodem and Ymodem the latter protocol will not be described in this report. Zmodem is described below.

2.4.1 Transaction overview

When entering the protocol, both the transmitting and receiving computer must know what file is to be transmitted and where to put the data (file to store the data or buffer area). In this protocol, one side (local host) is always in charge, asking the other side (remote host) to transmit or accept a file. Commands to select a file name and to prepare to transmit/receive a file via Xmodem is sent to the remote host outside of the Xmodem protocol. Once this is completed the remote computer enters the protocol. Now the user must tell the local host which file to transmit/receive and then the local host enters the protocol.

When the transmitter enters the protocol it will wait a period of time (between 10 seconds and a minute) to receive a NAK or a CAN from the receiver. If a CAN is received the connection will be terminated and the local host (transmitter) will leave the protocol. After the receiving host has sent a NAK it will wait for 10 seconds for data to start to arrive. If no data has been received within this time the receiver will send another NAK and so on (will be repeated max 10 times).

If the receiver wishes to use Xmodem CRC it can request it by sending the letter C instead of a NAK in the initiation phase. If the transmitter supports Xmodem CRC, it will begin the transmission upon receiving the C. If the transmitter does not support Xmodem CRC it will simply ignore the C and in this way force the receiver to send another C after three seconds. If the receiver has sent three C:s and is still waiting for any data to arrive, it falls back to Checksum Xmodem and starts to send NAKs instead.

Upon receiving a NAK the transmitter starts to divide the data into packets (128 bytes) and calculate the checksum. When the remote host has received the whole packet it starts to calculate the checksum and compare it to the checksum received. If the checksums match each other the receiver sends an ACK and if not a NAK. When the transmitting host receives an ACK it sends the next packet and if a NAK is received the local host resends the last packet again.
When the transmitter has sent the last packet it sends an EOT to signal end of transfer and then waits for a final ACK. When this is received the local host leaves the protocol.

Error detection and recovery are the primary purposes of Xmodem as file transferring protocol and the transmitter and receiver continues to retry until ten errors in a row has occurred, the connection will then be terminated.

2.5 Zmodem

Advances in computing, modems and networking have spread the Xmodem protocol far beyond the micro-to-micro environment it was intended. This has exposed the weaknesses of the protocol. Among other problems the Xmodem protocol is for example slow, not very user friendly, the throughput suffers in timeshared networks due to the small packets and file attributes are lost. In 1985 Chuck Forsberg decided to try to fix these problems and developed Zmodem.

Zmodem has two significant features: it is extremely efficient and it provides crash recovery. Unlike Xmodem, Zmodem is a streaming protocol. This means that the protocol does not wait for acknowledgement but rather continues to send blocks until a NAK is received. When this happens the protocol resends the lost packet. If a Zmodem transmission is cancelled or interrupted for any reason, the transfer can be resurrected later and the previously transferred information need not be resent. The protocol uses a 16 or 32 bit CRC, which makes the transfer very reliable. Zmodem has the ability to change the packet length according to the quality of the transmission channel. The length of a Zmodem frame is only bound by the length of the file. This protocol divides the file into 1024 bytes subpackets; note that these subpackets are not the same as an Xmodem packet. Unlike Xmodem's packet these subpackets do not contain any information about synchronization or block number.

A data frame consists of a header containing a “frame-type”, four bytes of supervisory information and its own CRC; one or more subpackets follow this. Zmodem divides the data into subpackets for two reasons. First it allows the receiver to verify the accuracy of received data without tying up a lot of memory. Second, and more important, this permits Zmodem to check for errors without incurring the high overhead of regular packets. It is the length of the subpackets that is changed depending on the error frequency of the channel. The length of every subpacket is denoted by an escape sequence.

2.5.1 Transaction overview

Below follows a description of how a file transfer process may come about. Note that Zmodem uses a lot of different headers and frames. The headers used in the scenario below can also be used in other situations. For a full account of their meaning see the full Zmodem specification [2].

The receive program is started first and initialises the transmission by sending a ZRINIT frame or a ZCHALLENGE frame. The receiver continuous to resend either of these frames every 10 seconds (for max four times). If it still does not get any response it falls back to the Ymodem protocol.

If the sender receives an init frame the sender responds with a ZRQINIT frame, this causes the receiver to send another init frame. If a ZCHALLENGE is received the sender responds with a ZACK. This is done because the receiving program wants to make sure that it is connected to an operating program and not activated by a Trojan Horse or other suspicious data [2]. In case of bad or lost data the sender will repeat the invitation to receive a number of times until session starts. If the receiver gets the init header back, this indicates that there is no operational sending program.

The sending program on the local host awaits a command from the receiving program to start file transfer. If a “C”, “G” or NAK is received, an Xmodem or Ymodem transfer is indicated. In this case the Ymodem protocol will be used.

The sender may then send an optional frame to define the receiving program's attention sequence, or to specify complete control character escaping. Eventually the connection is set up correctly.

The sender then sends a ZFILE frame with Zmodem Conversion, Management and Transport options followed by a ZCRCW data subpacket containing the filename, file length, modification date and other information. The receiver examines the information received. The receiving program should insure that the pathname and options are compatible with its operating environment and local security requirements.

If the receiver already has a file with the same name and length it may respond with a ZSKIP frame. This indicates that the file already exists with the receiver and it will cause the sender to move on to the next file (if any) in the batch. If the management option ZMCRC is enabled by the sender. The receiver sends a ZCRC frame back to the sender instead. This will make the sender calculate the checksum for the file and send the CRC complement to the receiver. This will definitely tell the receiver if it really is the same file or not and therefore if it should be accepted or not.

A ZROPS frame from the receiver initiates the transmission of the data, starting at the offset in the file specified in the frame. Normally this offset is set to zero, indicating that the whole file should be transferred. If the offset is not set to zero the receiver may want to start the data transfer further down in the file, indicating that a part of the file already exists with the receiver. The transfer may have been interrupted earlier and will now be completed.

The sender sends a ZDATA binary header (with file position) followed by one or more data subpackets. The receiver compares the file position with the number of characters successfully received. If they do not match, a ZRPOS error response is generated to force the sender to the right position within the file. A data sub packet terminated by ZCRCG and CRC does not get a response from the receiver unless it contains errors, more sub packets follows immediately. There is another type of subpackets; these are terminated with ZCRQ instead. Such subpackets demand a response from the receiver (ZACK) with the receiver's file offset if no error, otherwise a ZRPOS with the last good file offset. These subpackets are used when a node buffer the packets between the sender and the receiver. The buffering will easily cause the transmitter to be ahead of the receiver. ZCRCW data sub packets also expect a response before the next frame is sent. The sender uses the ZCRCW to allow the receiver to write its buffer before sending more data.

In absence of fatal error, the sender sooner or later encounters an end-of-file. If the end of file is encountered within a frame, the frame is closed with a ZCRCE data subpacket (which does not need a response unless an error has occurred). The sender sends a ZEOF with the number of characters in the file. The receiver compares this number with the number of characters received. If the receiver has received the entire file, it closes the file. If the close was satisfactory the receiver sends a ZRINIT else a ZFERR is sent.

If the receiver has not received the entire file, the receiver ignores the ZEOF because a new ZDATA is coming.

2.6 Kermit

Kermit is a communication protocol that can be used for file transfers or for terminal emulation. It was developed in 1981 by the System Group of the Columbia University Centre for Computing Activities in New York City and named after Kermit the Frog, star of the TV series The Muppet Show. Kermit is not in the public domain but Columbia University allows people to use the protocol for free so almost all communication products support it. However, not all implementations support the full protocol. This has led to that the more advanced features of Kermit (like sliding window and long packets) are often referred to as SuperKermit although there is only one version of the protocol.

From the beginning the design of Kermit took into account the restrictions placed upon serial communications by mainframe computer systems. The resultant design is general enough to, unlike Xmodem, fit almost any system. The protocol is able to transferring eight-bit data over seven-bit communication lines. The only control characters used are SOH (Start Of Header) and CR (Carriage Return).

Kermit uses variable length packets characteristically ended with a CR. Kermit has ten different packet types compared to Xmodem, which only has three. Response packets are also complete packets with checksums. Kermit is a more robust implementation than Xmodem, with almost no chance of noise changing the meaning of a response. In addition the ACK and NAK packets include a reference to the packet number it confirmes. This feature allows protocol enhancements such as sliding window. There are several different implementations of sliding window but the main idea is the same. The sender has a set of sequence numbers corresponding to the packets that it is allowed to send. This set of numbers is called a sending window. The receiver also maintains a receiving window, corresponding to the packets it is allowed to accept. The numbers in the sending window represent the frames sent but not acknowledged. When a packet is acknowledged the “window” moves one step forward so that yet another packet can be sent and so on. The sliding window feature allows Kermit to have more than one packet unacknowledged, which makes it more similar to a streaming protocol.

This big amount of control and acknowledgement data that Kermit uses does however limit the efficiency of file transfer. Kermit Sliding Window (SuperKermit) improves throughput over networks a bit but at the cost of more complexity. SuperKermit requires full duplex and the ability to check for the presence of characters in the input queue.

For the full specification of the Kermit protocol please read the book "Kermit, A File Transfer Protocol"[4].

2.7 PPP

PPP is an abbreviation for point-to-point protocol and is designed for simple links, which transport packets between two peers. These links provide full duplex simultaneous bi-directional operation and are assumed to deliver packets in order. It is intended that PPP provide a common solution for simple connection of a wide variety of hosts, bridges and routers. PPP was designed much later than the original HDLC specifications. As a result the creators of the point-to-point protocol included many additional features that had not been seen in WAN data-link protocols up to that time.

The point-to-point protocol is comprised of three main components [5]:

1. A method for encapsulating multi-protocol datagrams.
2. A Link Control Protocol (LCP) for establishing, configuring and testing the data-link connection.
3. A family of Network Control Protocols (NPCs) for establishing and configuring different network-layer protocols.

2.7.1 Encapsulation

The PPP encapsulation supplies the means for multiplexing of different network-layer protocols simultaneously over the same line. The encapsulation has been carefully designed to retain compatibility with the most commonly used hardware.

The PPP encapsulation is used to disambiguate multiprotocol datagrams. This requires framing to indicate the beginning and the end of the encapsulations. The packet consists of three fields, protocol field, information field and padding.

The protocol field is one or two bytes big and its value identifies the datagram encapsulated in the information field. A value range defines which protocol (NPC, LCP network-layer protocol) the packet belongs to. The filed is transmitted and received most significant byte first. The information field is zero or more bytes. The information field contains the datagram for the protocol specified in the protocol field.

The maximum length for the information field including the padding but not including the protocol field is termed MRU, which is 1500 bytes default. On transmission the information filed may be passed with an arbitrary number of bytes up to the MRU. It is the responsibility of each protocol to distinguish padding bytes from real information.

2.7.2 Link Control Protocol

PPP provides a link control protocol to be sufficiently versatile and portable to a wide spectrum of environments. The LCP is used to automatically agree on the encapsulation format options, handle varying size limits on packets, detect looped-back link and other common configuration errors. Upon a configuration error the link is terminated. Other optional services that PPP provides are authentication of the peer on the link and determination of when a link is functioning and when it is not.

2.7.3 Network Control Protocol

Point-to-point links tend to intensify many problems within the current family of network protocols. For instance assignments and management of IP addresses, which is a problem even in LAN is especially difficult over circuit-switched PPP links. Such problems are handled by a family of network control protocols, which each manage the specific needs of their respective network-layer protocols.

The intention is that PPP links should be easy to configure. PPP is designed so that the standard defaults handle all common configurations; these are automatically communicated to the peer without user intervention. Finally, the operator may explicitly configure options for the link, which enables it to operate in environments where it would otherwise be impossible.

2.7.4 Establishment of a point-to-point link

The establishment of communication over a point-to-point link consists of several phases.
The first phase of all is called “link dead”. The link necessary begins and ends in this phase and it indicates that the physical layer in the OSI model is not ready to be used. When an external event (such as carrier detection or network administrator configuration) indicates that the layer is ready PPP will continue to the next phase, this is called the “link establishment phase”.

The Link Control protocol is used to establish the connection through an exchange of Configure packets. This exchange is complete and the LCP Opened state entered when a Configure-ACK packet has been both sent and received. All configuration options are assumed to be default values unless they are altered by the configuration exchange. It is important to point out that only configuration options that are network-layer protocol independent are configured by LCP. Configuration of individual network-layer protocols is handled by separate NCPs later on. Any non-LCP packet that is received during this phase must be discarded. The receipt of the LCP Configure-Request causes a return to the Link Establishment phase from the Network-Layer Protocol phase or Authentication phase.

On some links it may be desirable to require a peer to authenticate itself before allowing network-layer protocol packets to be exchanged. This is called the “authentication phase”. Authentication is not mandatory by default. If an implementation desires that the peer uses some specific authentication protocol for authentication, then it must request this during the link establishment phase. Authentication should take place as soon as possible after link establishment. However, link quality determination may occur at the same time. An implementation must not allow the exchange of these link quality determination packets to delay authentication indefinitely. PPP is not allowed to leave the authentication phase and enter the network-layer protocol phase until the authentication is completed. If the authentication should fail the authenticator should proceed to the link termination phase instead.
Only LCP, authentication protocol and link quality monitoring packets are allowed during the authentication phase. All other packets must be discarded.

As soon as PPP has finished the previous phases, each network-layer protocol (such as IP, IPX or Appletalk) must be separately configured by the appropriate network control protocol (NCP). Each NCP may be opened and closed at any time.
This is called the network-layer protocol phase. After a NCP has reached the Opened state, PPP will transmit the corresponding network-layer protocol packets. Any network-layer protocol packets received when the NCP is not opened must be discarded. During this phase, link traffic consists of any possible combination of LCP, NCP, and network-layer protocol packets.

PPP can terminate the link at any time. This might happen because of the loss of carrier, authentication failure, link quality failure, the expiration of an idle-period timer or the administrative closing of the link. This is called the “link termination phase”. LCP is used to close the link through an exchange of terminate packets. When the link is closed PPP informs the different network-layer protocols so that they may take the appropriate action.

After the exchanges of terminate packets, the implementation should signal the physical-layer to disconnect in order to enforce the termination of the link, especially if the termination is caused by authentication failure. The sender of the terminate request should disconnect after receiving a terminate-ACK or after the restart-counter has expired. The receiver of such a request should wait for the peer to disconnect until at least one restart time has passed after sending the terminate-ACK. Any non-LCP packets must be discarded during this phase. PPP then proceeds to the link dead phase.

2.8 SLIP

SLIP is an abbreviation of Serial Line Internet Protocol and merely a packet framing protocol. It defines a sequence of characters that frame IP packets on a serial line. SLIP has its origins in the 3COM UNET TCP/IP implementation from the early 1980's. Around 1984, Rick Adams implemented SLIP for 4.2 Berkeley Unix and Sun Microsystems workstations and released it to the world.

SLIP is commonly used on dedicated serial links and sometimes for dial-up purposes. It is useful for allowing mixes of hosts and routers to communicate with one another.

Because SLIP is such a simple protocol it is fairly simple to implement too, but it leaves a lot to be desired. The most commonly shortcomings perceived are addressing, packet type identification, error detection/correction or compression mechanisms. The problem with addressing is that both computers in a SLIP link need to know each other's IP addresses for routing purposes. Also when using SLIP for a host to dial-up a router the addressing scheme may be quite dynamic and the router may need to inform the dialling host of its IP address. SLIP does not provide any mechanism for a host to communicate addressing information over a SLIP link.

Since SLIP has no type field only one protocol can be run over a SLIP connection. So in a configuration of a pair of DEC computers running both TCP/IP and DECnet it is impossible to share one serial line between them while using SLIP. This is one of the SLIP protocol's shortcomings. If a serial line connects two multi-protocol computers, those computers should be able to use more than one protocol over the line. An error detection or correction mechanism would be desired because noisy lines can corrupt packets during transmission and low speed makes retransmissions very expensive. Error detection is not absolutely necessary since TCP/IP should detect damaged packets. Some packets do however slip through although they are damaged and since resending takes time it would be better if SLIP could do some sort of error correction itself. Compression should also be useful because dial-in lines are so slow. Packet compression would cause large improvements in packet throughput.

Usually, streams of packets in a single TCP connection have few changed fields in the IP and TCP headers, so a simple compression algorithm might just send the changed parts of the header instead. In all fairness it should be said that SLIP was designed a long time ago when these problems were not really important issues.

Today, SLIP has been extensively replaced with the point-to-point protocol though. This because PPP is faster, more reliable and supports a lot of features that SLIP does not.

2.8.1 Transaction overview

The SLIP protocol defines two special characters: END (decimal 192) and ESC (decimal 219). The ESC character is not to be confused with the ASCII ESCape character. To send a packet, a SLIP host simply starts sending the data in the packet. If a data byte happens to be the same as the END character, a two-byte sequence of ESC and decimal 220 is sent instead. If a byte is the same as an ESC character, a two-byte sequence of ESC and decimal 221 is sent instead. When the last byte in the packet has been sent an END character is then transmitted.

There is no real defined maximum packet size defined for SLIP but it is probably best to use the size of 1006 bytes including the IP and transport protocol headers (not including the framing characters) as the maximum packet size. Berkeley UNIX SLIP uses this size [6].

3 Analysis of user interfaces and graphical systems

The company Datalogic AB has on their earlier handheld computers been using Windows CE as operating system. However, in order to cut down on costs, they have decided to use Linux as the operating system on their new products. To give the product a higher grade of user-friendliness it has been decided that a graphical user interface (GUI) will be implemented and placed on top of the Linux kernel.

This chapter will describe the development of the interface. Some important aspects of user interface design will also be discussed. The parts, which are included here are the main areas of "Human computer interaction" (HCI), with a discussion on the differences between an interface towards a small screen and a regular PC-monitor. Based on this knowledge a couple of different "mock-ups" were designed and an interview was held with one end-user and one application-developing technician.

Finally, X-window systems are discussed and the one that fits this task the most will be chosen. Then some different implementation alternatives will be compared; PicoGUI, Microwindows, Qt and Gtk+.

3.1 HCI

3.1.1 What is HCI?

HCI stands for Human-Computer-Interaction. Interaction is what it is all about, the collaboration between a human (the user) and a computer. HCI is a research area that refers to development of systems with high acceptance of the user [7].

In the past, the focus was on creating an appealing technical solution, it would solve the task it was designed to do, but little or no consideration were given to system usability. This was no problem since the users were technicians and the functionality was rather limited. But today, when the user group has become more diverse, a well designed user interface has become an important part of the system. As a consequence, the focus is moved more towards the human.

HCI was introduced as an attempt to close the gap between the user and the computer. The computers should be adjusted after the human and not the other way around. Or as C.M. Allwood said: “This means that it is the computer-user and not the computer that is in control, and that should be in control of the interplay” [8].

Within the research-area of HCI there are scientists from a wide spectrum of sciences, for example computing science, behavioural science, and organization science [9]. But there are also other professions like photographers, graphical-designers, film-experts and so on [10]. Cognitive psychology gives a few thoughtful aspects on how the human thinks and works. The next chapter will briefly discuss that area.

We often assume that a product or system that is said to have “good HCI” also is user-friendly. The term user-friendly is too arbitrary to be used in this context. Something that someone thinks is user-friendly may for someone else be very abstract. So instead of saying that it ought to be user-friendly, the word useful or usability is more correct [10]. Booth Preece and Nielsen mean that there are five points or “quality attributes” that can be used to investigate if a product has good usability:

1. **Learnability:** How easy is it for users to accomplish basic tasks the first time they encounter the design?
2. **Efficiency:** Once users have learned the design, how quickly can they perform tasks?
3. **Memorability:** When users return to the design after a period of not using it. How easily can they re-establish proficiency?
4. **Errors:** How many errors do users make, how severe are these errors, and how easily users can recover from the errors.
5. **Satisfaction:** How pleasing is it to use the design?

These five quality elements can be evidence of a programs usability, but an equally important part in this discussion and that is utility (how good is the functionality)[11]. A bad system with low utility can hide behind a good interface but a good system can be ruined by a bad GUI.

3.1.2 Cognitive psychology

Cognitive psychology is a science that investigates our mental abilities [12]. Examples of cognitive capacities of humans are memory, learning, problem solving, perception, attention, and communication with others. The cognitive psychology's part in HCI is according to Ngaosuvan described like this: "cognitive psychology can be expressed as a type of database of theories and experiment results that a GUI designer can take advantage of" [12]. For example that the colours blue and red should not be used together, because they have a big gap between their frequencies, which will force the eye to refocus between these two.

Here are some cognitive abilities that are extra important within HCI [12]:

- Perception, attention, automation.
- Learning and memory
- Categorizations and conceptions
- Representation, problem solving
- Thinking, reasoning ability, decision-making

3.1.2.1 Perception, attention, automation

Perception means ability to receive information from the surrounding environment.

Attention research aims to explain what is preferential in our surroundings and why. Automation is about perceptual and attention processes that we cannot help doing.

3.1.2.2 Learning and memory

To get an understanding of what happens when a human is working with a computer, it is important to understand how a user functions.

Since the human memory and its processes have an important part in this, it will be reviewed more thoroughly.

In our memory there are constantly processes going on. Imaginations, understandings, attitudes to reality are activated, put together, changed and stored. Human thinking can, in other words, be seen as more or less aware/planned processes in the memory. When we observe and listen to the environment it will lead to great activity in our memory. We interpret and understand our surroundings depending on what has earlier been stored in our memory [8].

Allwood describes the memory as if it actually consists of several "sub memories" that in some way are independent of each other. He discusses three types of memories; sensor-memory (SM), short term-memory (STM) and long term-memory (LTM). The structure can be described with help from this sketch [8].

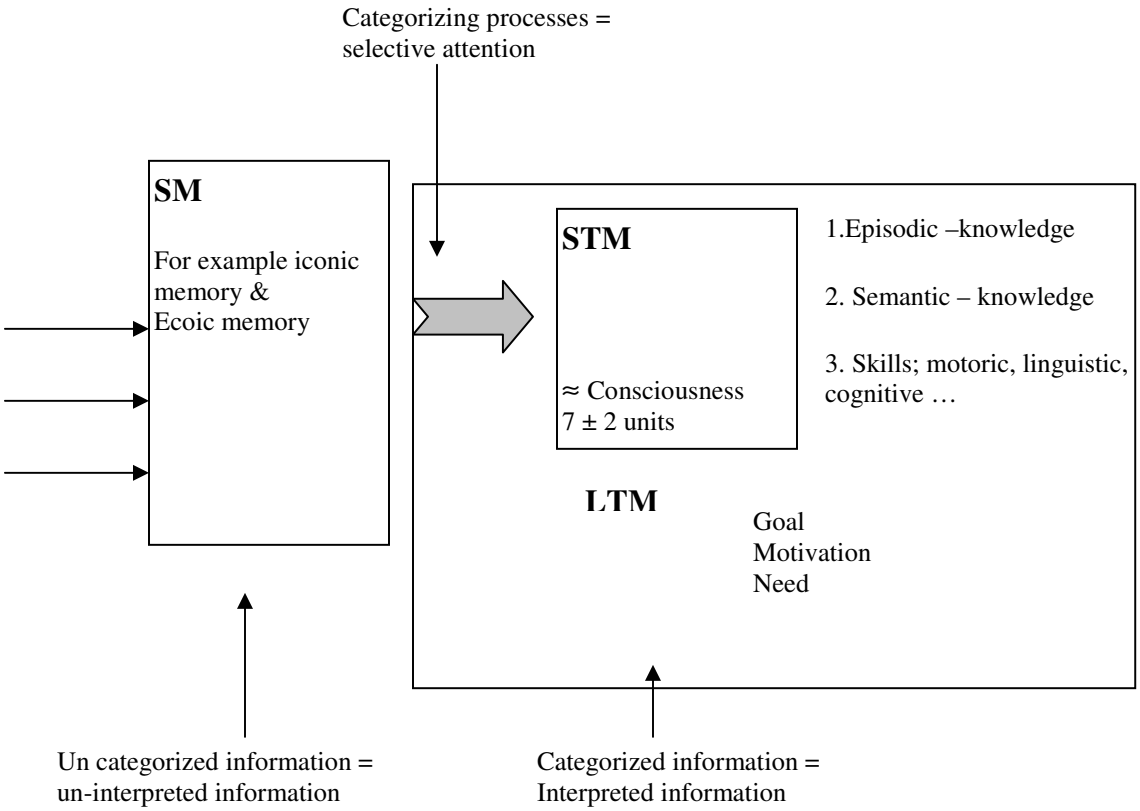

Figure 1. Memory model from C.M Allwood

Sensor memory (SM)

The sensoric memory of the eye is called the iconic-memory and the corresponding SM for the ear is called the ecoic memory. The other senses (smell, feeling and taste) are assumed to have their own SM as well. The time that the information is stored in the iconic-memory is calculated to approximately ¼ seconds. This however can differ depending on the background light- and contrast-condition. For the ecoic-memory the responding storing time is about two seconds.

The reading speed from the iconic-memory to the processes that categorize the information and pass it on to the short-term-memory is also time critical. The consequence of this is that only a limited amount of information can be presented to the user at a given time.

Short term memory (STM)

STM is defined in cognitive psychology as a memory where information of different kind is stored for a very short time, for example impressions and feelings. This part of the memory has a high degree of activation.

STM is not only used as a passage between the sensoric-memory and the long-term memory, it is also the place where the information is processed when it is about to be used. Because of this the short-term memory is often called the "working-memory" [8].

Short-term memory is used when we are focusing on a small part of the information that is important for just that moment, for example one line of text in a book. To store information from the STM into the long-term memory it will have to stay in the STM between 5 – 20 seconds. This can be one of the explanations why people seem to learn something better if they are typing down the things they hear or read. Because then they force the information to stay a little bit longer in the short-term memory and therefore it will be transferred to the long-term memory.

Inside the STM only a small amount of units can be stored at the same time, a commonly referred number is 7 ± 2 chunks [12]. These chunks can either be single characters/numbers or words. Experiments have shown that we can remember more items if we group the information in some way. For example it is harder to remember a phone-number that is typed in this form 0705555555, instead of 070-555 55 55. This grouping is called chunking.

Long-term memory (LTM)

The long-term memory contains representations of our surrounding world. When we for example picture us the cliff by the lake at our summerhouse without actually being there, it is made possible by our representation of that cliff. These representations that are stored in our long-term memory are called mental representations [8].

According to some cognitive psychologists, LTM really consists of several different types of memory, which are listed as 1 – 3 in the figure 1 above.

LTM can be seen as a storage place for our mental-representations of different kinds of contents, things like knowledge and skills. The knowledge can be separated into two parts, procedural and declarative. With declarative means competence like imaginations of objects, "knowing that". Procedural is knowledge according to skills, "knowing how".

3.1.2.3 Categorizations and Concepts

Concepts and categorization will focus on how we organize our knowledge. What this means is that it refers to the mental meaning of knowledge. If you have understood a concept as “cat”, then you know what is universal for all cats.

Our cognitive capacity will be less strained if we use some kind of cognitive economy. Cognitive economy means that information should be divided into a number of categories; the number should be adjusted after your surrounding. This must however be balanced against the information in the concepts. If there only were three concepts at hand, for example plants, animals and all other things there would be a very low cost but it wouldn’t be much of a system. It is also important to not put together categories that don’t have anything to do with each other, for example putting shoes and cats in the same category would only make things stranger. To sum it up, use an adequate number of concepts and make the classifications logical.

3.1.2.4 Representation and problem solving

Philosophers, linguistics and psychologists have for a long time tried to describe and explain how we represent the reality in our minds. A representation is an arbitrary notation, characters or a set of symbols that describes an item. A big part of problem solving is to be able to represent the problem in different ways and sometimes to represent parts in the problem mentally. The chess problems are an example of this [12].

3.2 To develop a GUI.

Designing a user interface can at first seem pretty easy and trivial, but that’s not at all the case. There are a lot of things to take into consideration.

- Which interaction style
- Which information shall be presented, and how shall it be grouped
- Which colours shall be used, and how shall they be mixed
- Deep or wide tree-hierarchy
- Selecting mechanisms
- How to order the windows to make it logical to the user
- How to make access fast for frequently used functions

This list can be made very long.

After analysing these points it is time to decide which interaction style that ought to be used. There are five different interaction styles to take into consideration [13].

1. **Menu selection**
   The user gets several alternatives from a list and chooses the one that he/she thinks is the most appropriate. If the terminology and the meaning of the items are understandable and distinct the users can solve their tasks with little learning.

   This interaction style is most appropriate for novice users because they don't have to learn the exact name of the command they can often figure it out if the terminology is good. Can also be appealing to frequent users if the display and selection mechanisms are rapid.

Pros	**Cons**
-Shortens learning. -Reduces keystrokes. -Structures decision making. -Permits use of dialog management tools. -Allows easy support of error handling.	-Danger of many menus. -May slow frequent users. -Consumes screen space. -Requires rapid display rate.

2. **Form fill-in**
   This can be used when data entry is required. The user sees a display of related fields, moves a cursor among the fields and enters data where desired. This interaction style is most appropriate for knowledgeable intermittent users since the user must understand what the labels at every data field means and what to type in.

Pros	**Cons**
-Simplifies data entry. -Require modest training. -Gives convenient assistance. -Permits use of form management tools.	-Consumes screen space.

3. **Command language**
   The user types commands at the prompt. The user must learn a lot of commands and syntax to make it useable. Often used by experts and frequent users who often derive great satisfaction from mastering a complex set of semantics and syntax.

Pros	**Cons**
-Flexible. -Appeals to "power" users. -Supports user initiative. -Allows convenient creation of user defined macros to save time for the user.	-Has poor error handling. -Requires substantial training and memorization.

4. **Natural language**
   The computer is supposed to respond properly to arbitrary natural-language sentences or phrases. Natural language interaction usually provides little context for issuing the next command, it will need frequently "clarification dialogues".

Pros	**Cons**
-Relieves burden of learning syntax.	-Requires clarification dialogues. -May require more keystrokes. -May not show context. -Is unpredictable.

5. **Direct manipulation**
   By pointing at visual representations of objects and actions, the user can carry out tasks and observe the result immediately. It is the typical "PC-way" of interaction, for example the desktop metaphor where the user clicks on icons representing different tasks and objects.

Pros	**Cons**
-Visually presents task concepts. -Allows easy learning. -Allows easy retention. -Allows errors to be avoided -Encourages exploration. -Affords high subjective satisfaction.	-May be hard to program. -May require graphics display and pointing device.

As can be seen here, there are pros and cons for all of these interaction methods. If taken into consideration what kind of product that is going to be constructed, there are two interaction methods that can be removed directly. Data entries will be very troublesome, because of the lack of a real keyboard. Neither the “command language” nor the “natural language” is a realistic alternative. Form fill-in can be of interest, for example when the user shall configure the network device. But the foundation of the system will be either a text menu based system or a direct manipulation based (icons) system.

3.2.1 Eight golden rules of interaction design

With the research in cognition-psychology the development in the HCI area has moved forward. A number of books treating this matter have been written; they describe different techniques and how a user friendly interface should be designed. The books, all in one way or another, describe some kind of rules or principles that the designer ought to think about when creating a user interface. Shneiderman lists what he describes as the eight golden rules of interaction design. These rules do not differ much from what other authors have written, but Shneiderman’s rules are described in a simple way [13].

These rules are put together to increase the system’s usability, but do take into consideration that they are basic principles/rules that you as developer will have to adjust to make them fit your special needs. It is alright to break these rules, but when you do, you should have a good reason for doing so.

1. **Strive for consistency**
   This is the most frequently violated one, because it is very hard to follow, since there are many forms of consistency. Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus and help screens. Consistent font, colours, layout and so on should be used.

2. **Enable frequent users to use shortcuts**
   As the frequency of use increases, so do the user’s desire to reduce the number of interactions. A frequent user appreciates abbreviations, special keys, hidden commands and macro facilities.

3. **Offer informative feedback**
   For every user action there should be some kind of feedback. For frequent and minor actions the feedback can be modest. But for more critical actions the feedback should be more substantial.

4. **Design dialogues to yield closure**
   Sequences of actions should be organized in groups with a beginning, middle and end. The completion of a group of actions gives the user a satisfaction of accomplishment. It will give him/her a sense of relief and an indication that it is okay to prepare for the next group of action.

5. **Offer error prevention and simple error handling**
   The system should be designed so that the user cannot make serious errors. For example: menu selection to enter text in a form fill-in and no acceptance of alphabetic characters in a numeric field. If the user makes an error the system should detect this and offer constructive help. The user should not have to retype the entire form only the fields containing errors. Erroneous actions should not change the state of the system.

6. **Permit easy reversal of actions**
   As much as possible actions should be reversible. This will relieve anxiety and encourage the user to explore unfamiliar options.

7. **Support internal locus of control**
   The users should sense that they are in charge of the system and that the system reacts to their actions. Surprising system actions, tedious sequences of data entries and inability to produce the desired action all build anxiety and dissatisfaction.

8. **Reduce short-term memory load**

   Since limitations in the human short-term memory is as low as 7 ± 2 units of information, it is important that the displays are kept neat and that the user will not have to remember information from earlier windows.

These underlying principles must, as earlier mentioned, be refined and/or extended to suit your environment. The principles can be used for all kind of interaction types, and they are used when deciding what should be displayed. Shneiderman also has five rules of thumb for data-entry alternatives. They will only be mentioned superficially since there is very little text entering using keystrokes in the product that is about to be designed [13].

3.2.2 Short description rules of thumb for data entry

1. **Consistency of data entry transactions**
   Similar sequences of actions should be used under all conditions, same delimiters, abbreviations and so on should be used.

2. **Minimal input actions by user**
   Less input mean better operator productivity and fewer chances for error. It's better to click a box or choose from a list than typing in a phrase.

3. **Minimal memory load on users**
   When data should be entered the user should not have to remember long complex commands or codes.

4. **Compatibility of data entry and data display**
   The format of data-entry should visually be displayed in a way that corresponds to the way it later will be displayed on the screen.

5. **Flexibility for user control of data entry**
   Experienced users may prefer to enter information in a sequence that they can control. Some users may for example like the address field at the top of the window but another one would like to put port number at the top.

   This kind of flexibility should however be used with caution since it will violate the rule of consistency.

3.2.3 Summary for GUI designing

The interaction styles that can be used for this product will either be a menu based or direct-manipulation based system. Perhaps there will be some kind of form fill-in style for handling file transfer and different kind of network settings (WLAN, bluetooth e.g.).

The user must have the possibility to reverse an action if a wrong decision was made. If a form fill-in is used for network and file transfer settings, there must be good error control and handling of the errors. The user should only have to change the field that contain incorrect data. And the network status should not be affected until the fields are correctly filled in.

To make frequent users happy some kind of shortcuts or short commands must be supported. The tree hierarchy should be held as shallow as possible, but it must be well thought through so that too much information is not presented at once.

3.3 Mockups

3.3.1 Suggestion No1:

This suggestion will use a direct manipulation interaction style, using icons in a size of 24 x 24 pixels that will represent the different tasks and objects in the system.

At the bottom of the screen there will be a bar that will be visible as long as no application is started. Different information about the computer, such as current time, battery status and connection status (Bluetooth and WLAN) will be displayed in the field marked *status*. The button named *start* will contain some shortcuts and also list the applications.

There are two options; the first is to put a "direct button" back to the desktop (main menu) and a help button that will display what options the user has in the window that is shown. Alternative two will be to skip these buttons and use that area for a label that displays what the pointing device is aimed at. Which alternative to choose will be decided when the implementation has started; if there is enough room to write the information under the icon.

Figure 2 shows the main menu that will be the first thing the user sees when the terminal is turned on. It is what we could call the "desktop" of the system. From here the user can start a terminal, initialize a file transfer, explore the file system, open the application directory or enter the control panel.

Figure 3 shows the control panel that is used as a control centre of the terminal. From here the user can check and change different settings (date/time, power, WLAN, Bluetooth, autostart and different services such as telnet servers). At the top left corner there will be an icon that will bring the user up one step in the window hierarchy.

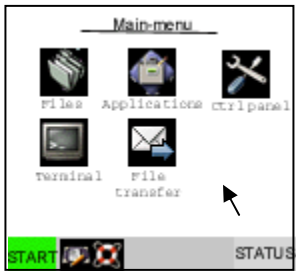

Figure 2 Main menu

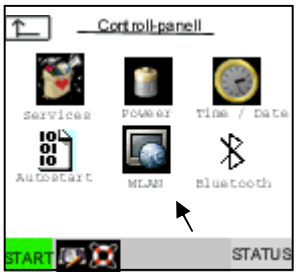

Figure 3 Control panel

3.3.2 Suggestion No2

This suggestion is based on a menu interaction model; the hierarchy will follow the one in suggestion 1.

The main menu (Figure 4) will contain the items start menu, applications, browse files, control panel, terminal, file transfer.

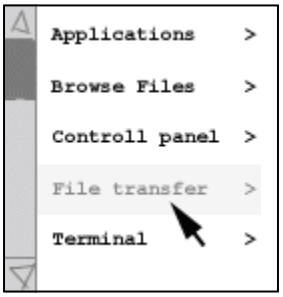

Figure 4

Start menu

Shortcuts associated to often visited objects will be available here, all of the applications will be arranged in this menu as well.

File browsing

The file system will be displayed in shape of a list; directories and files will be displayed in different ways (Figure 5). Different colours or small pictures can indicate if the name represents a file or a directory (in this case indicated by [ ]).

Figure 5

Control panel
The same categories as in the control panel in suggestion one will be placed here. They are: autostart, Bluetooth, power, services, time/date and WLAN settings. They will be displayed in the same way as the main menu above (Figure 4). The user will use either buttons associated with the number in front of the menu item or use the pointing device.

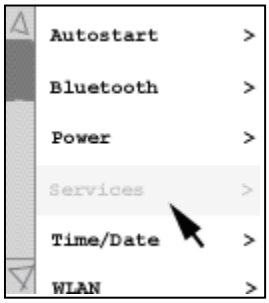

Figure 6

3.3.3 Suggestion 1 and 2
To handle data entry, there will be the same solutions for both suggestions. An easier type of form fill-in will be used. For example, the data entry for setting up the TCP/IP connections through WLAN will be done using data fields with error control so no invalid text will be entered. For other errors an informative error message will be displayed. For the Bluetooth settings it will be possible to get the information about the device, i.e. current name and hardware address.

3.3.4 Discussion with end user and developer
These two suggestions were presented to one end user (employee at a grocery store) and one application developer of similar handhelds. The attributes regarding the small display and the limited input possibilities were discussed as well. They were asked to specially consider these points.

- Which one was easiest to understand (the function of the icons/menus)
- Which one was fastest to use
- In which suggestion was it easiest to keep track of where you were in the hierarchy
- Which suggestion has the most professional appearance

They were also asked to give their own comments about modifications or changes in the mock-up suggestions.

End user

The end user thought that suggestion two (menu based) was easier to understand at first sight; in that alternative a text menu based interaction model was used. He though that it was hard to understand the meaning of some icons in suggestion one, especially since the text beneath them was hard to read. He, however, thought that the icon based alternative was a better choice in the long run. He based this opinion on the following facts: it is faster to navigate through the icon based system, easier to get an overview of available tasks and no need to scroll through menu lists.

Suggestion one was according to him the choice that had a more professional look. The only negative part of it was that the text under each icon was hard to interpret. This will however make no difference when the handheld has been used a few times, since the user will recognize the different task representations.

Application developer

The developer preferred suggestion one (icon based) in all aspects, although he had a few good suggestions for improvement. He mentioned how important it is to choose icons that are simple, clean and as classic as possible. In this way the user will recognise the icons from other situations and this makes it easier understand what task they represent.

He also mentioned that a level indicator could be a good idea if the window/menu hierarchy was deep, this would help the user to orient in the system. Finally we discussed the bar in the bottom of suggestion one. He thought it was a good idea to remove the start, help and the desktop buttons at the left of the bar and use that field only to display explanation texts about currently selected icon.

3.4 Different graphical systems

3.4.1 X windows systems

The X window system, often called just ‘X’ or X11 is the standard graphical engine for both UNIX and Linux operating systems. It provides the only common windowing environment bridging the heterogeneous platforms in today’s computing.

X provides for the display and management of graphical information, much as Microsoft’s Windows. The key difference is the structure of the X protocol, whereas Microsoft Windows simply displays graphical applications local on the PC, while the X protocol distributes the processing of applications by specifying a client-server relationship at the application level. The “what to do” part of the application is called the X-client, and the “how to do” part is called the X-server.

Definition of the X protocol

The X protocol defines a client-server relationship between an application and the display. The application is the client and the display is the server of the system. To solve this the application is separated from the display. The server side contains of two layers, one device dependent layer and one independent layer. The communication between the client and the independent layer of the server is handled asynchronous.

Summing up, the X protocol hides the peculiarities of the operating system and the underlying hardware. This masking makes it much easier for the developer to create applications and makes the X system very portable.

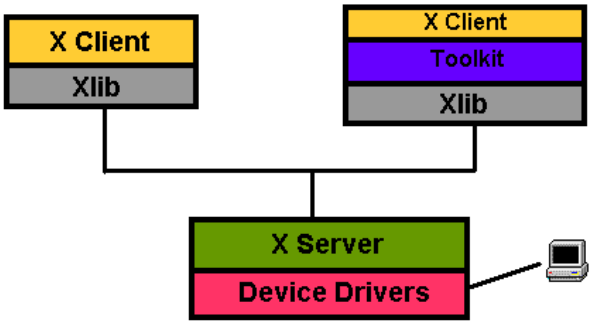

Figure 7 Graphic over how the X protocol works

This is what the X server does:

- Displays drawing requests on the screen
- Replies to information requests
- Reports errors in requests
- Mange's the keyboard, mouse and display devices
- Creates, maps and destroys windows

This is what the X client does:

- Sends requests to the server.
- Receives events from the server.
- Receives error messages from the server.

3.4.1.1 Xfree86

Xfree86 is a redistribution of the reference implementation of the X window system; originally modified to fit the wide variety of video hardware available for Intel x86 based machines (hence the "86" in "Xfree86"). The "Free" part of the name comes from the decision to release this X server under the same terms as the freely available X sources.
Because of this it has become very popular and now it does not confine the development to the x86 hardware.

3.4.1.2 TinyX

TinyX is a stripped version of Xfree86, with the purpose to be used on small embedded or handheld systems.

3.4.1.3 Microwindows

Microwindows is a graphical windowing system that doesn't require a large disk or RAM. Microwindows does not require any operating system or other graphic support, as it writes directly to the hardware, although it runs well under Linux framebuffer systems or under the X window system [14].

Microwindows has a distinct separation of three different layers, the driver-, the engine- and the API-level. This result in that complex functionality can be added without complicating the entire design.

The driver level is the lowest layer. This layer abstracts a data-structure for the display, pointing device and a keyboard device. The top-level of this structure will always look the same. This layer includes code to interface with different devices. Microwindows includes drivers for many different devices and operating systems. The drivers for the framebuffer supports 1, 2, 4, 8, 16, 32 bits/pixel and both greyscale and true colour displays.

The engine level, that is the drawing engine, is device independent, which rely on that the driver level is called to access the screen driver. The engine level presents a couple of standard entry points for the API-level to call for drawing functionality. The engine level also abstracts a RGB colour model for all colours; regardless if the physical display is greyscale or true colour. This level also handles all fonts.

The API level implements one of the two supported API:s, the win32 or the Nano-X. This layer implements the window-abstraction, which allows application programmers to contain their display data in full screen or overlapped windows. It also takes care of the event handling and passing received hardware events, like a mouse click, to the application.

3.4.1.4 PicoGUI

PicoGUI is not based on the X window system, and is not compatible with it either. PicoGUI is a new GUI architecture designed with embedded systems in mind. It includes low-level graphics and input, widgets, themeing, layout, fonts rendering, network transparency, and debugging features. To describe it in short it is similar to the following:

- Xfree86 + GTK + a window manager
- Microwindows + FLTK
- Qt/Embedded

PicoGUI is also client-server based, but the separation between the two halves is drastically different than in X.

In X the server maintains very little state information and only performance low-level drawing command. This means that the server is very flexible, but the client-server bandwidth is high and the client code has to be complicated.
PicoGUIs approach is to put everything possible on the server side. Widgets, themes and even custom graphics are implemented on the server side. The client library's only purpose is to give a convenient way to access the server's API.

Most GUIs make you specify widget properties in x and y coordinates and the look of the widget is hard coded in the toolkit. This means that there have to be assumptions about the hardware capabilities, the display for instance. With PicoGUI there is a layout engine and theme systems to solve this problem. The programmer types in a size and a position in relative terms, so the GUI makes no assumptions about the display units.

The theme system and the flexible video architecture allow it to run on a wide variety of displays, from large desktop monitors to text-only LCD.

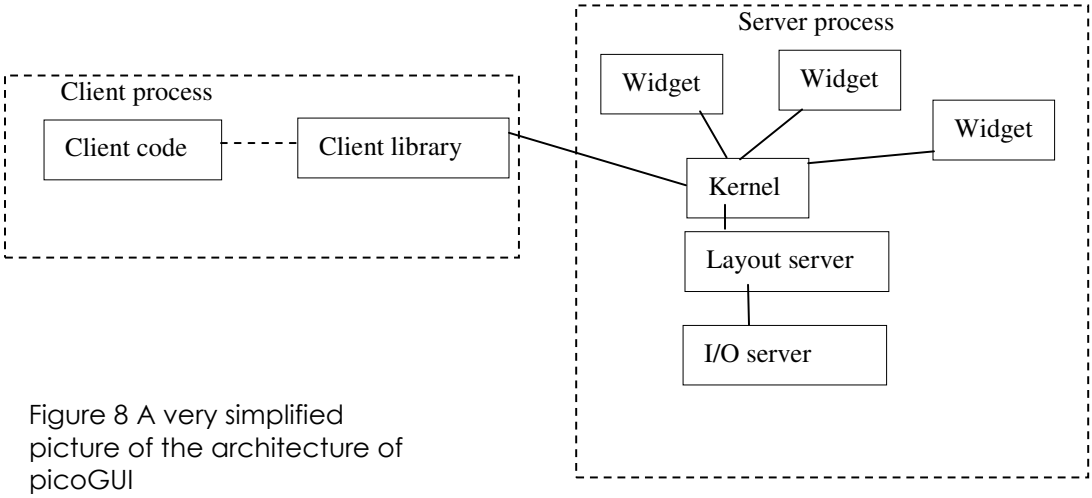

Figure 8 A very simplified picture of the architecture of picoGUI

3.4.1.5 Qt and Qt/Embedded

Qt is a C++ toolkit for multiplatform GUI development. It includes a rich set of widgets that provide for standard GUI functionality. Qt works on the X11 X window system for running on UNIX or UNIX like systems. But it also works on Microsoft Windows and Mac OS X, there are also a version called Qt/embedded that runs directly against the Linux framebuffer.

Qt is using the low-level API:s of the different platforms it supports (win32 and GDI for Windows, Xlib for X11 and carbon for Mac). This differs from traditional “layered” multiplatform toolkits that are thin wrappers over single-platform toolkits (MFC for Windows or Motif for X11). Since Qt uses a single source tree, a Qt application can easily be converted to fit another platform. This is accomplished by recompiling the source code on the other platform.

Qt provides a type-safe alternative to old fashioned callbacks; these are called signal and slots. Qt supplies a wide range of widgets that can be subclassed to create custom components.

Qt/Embedded is a port of the Qt C++ API for embedded devices. It has the same API and tools as the Qt/X11, Qt/Windows and Qt/Mac versions.

Qt/Embedded has much smaller system requirements than the original Qt, i.e. lower disk storage (flash) and memory (RAM) footprint. This depends on that it doesn’t require the X window system or Xlib library. It writes directly to the frame-buffer. Qt/Embedded use the Qt/Embedded-library to replace both the X server and the Xlib library.

The figure 9 picture can summarize the architecture and the differences in architecture between Qt/X11 and Qt/Embedded. It shows they share the same API and source code and that X11 version uses the X window system and Xlib and how the Embedded version only uses the Qt/Embedded library.

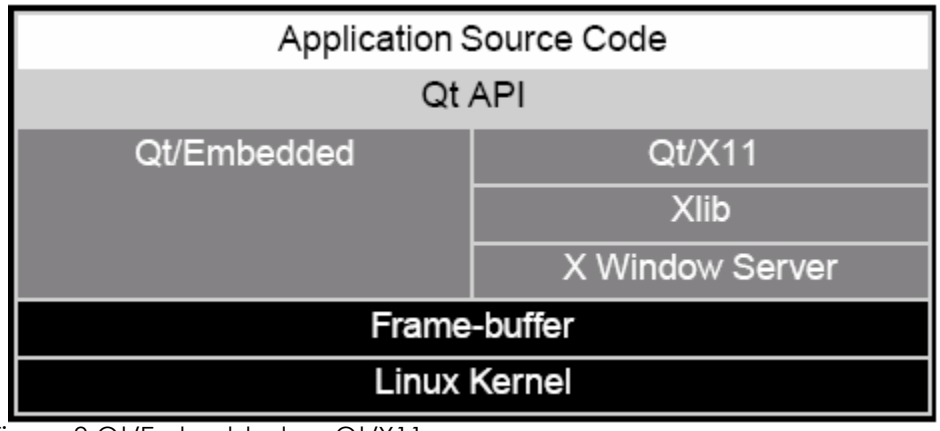

Figure 9 Qt/Embedded vs. Qt/X11

The windowing system of Qt/Embedded also follows some kind of client-server thinking. The window system consists of one or more processes, one of which act as a server. The server allocates regions to be displayed by clients, and generates mouse and keyboard events. Clients communicate with the server through shared memory, the communication is kept to a minimum. Clients perform all drawing operations directly to the framebuffer without passing through the server and are responsible for drawing their own title bars and other decorations. All of this is handled by the Qt/Embedded-library.

The Qt however has a negative aspect for the product that is going to be developed. Qt has two different editions, one for non-commercial, which is free of charge. But the one for commercial purposes will need to be licensed and costs money. Since the reason to change from WinCE to Linux was to avoid licence costs, Qt probably will not be the choice.

3.4.1.6 Gtk+

Gtk+ that stands for GIMP toolkit is a multi-platform toolkit for creating graphical user interfaces; it has a complete set of widgets. Gtk+ is used to create everything from small one-off projects to complete applications.

Gtk+ has been designed from the ground to support a range of different languages, not only C/C++. It can also be used with Perl or Python. It is free software and a part of the GNU project. However the licensing terms for the Gtk+ allow it to be used by all developers, including those developing proprietary software, without licence or royalties.

Gtk+ is mainly based on these three libraries, but as you can see in figure 10 it contains a few more parts.
-Glib
-Pango
-ATK

Glib is the low-level core library; it provides data structure handling for C, portability and interface for such runtime functionality as an event loop, threads, dynamic loading and an object system.

Pango is a library for layout and rendering of text with an emphasis on internationalisation. It handles the text and fonts for Gtk+.

The *ATK* library provides a set of interfaces for accessibility. If the application supports the ATK interface, it can be used with such tools as screen readers, magnifiers and alternative input devices.

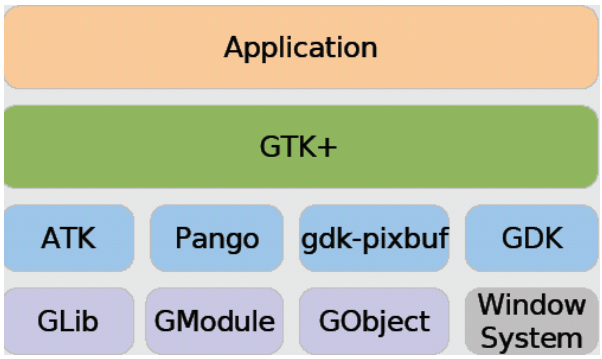

Figure 10. Architecture of Gtk (hellocity.net)

GDK is a drawing kit that works as a wrapper around the low-level functions, *gdk-pixbuf* handles image loading and image manipulation on the client-side. *GModule* is there to make it possible to add plug-ins dynamically and *GObject* is handling the signals and the notification mechanism.

Gtk+ API can also be used with GtkFB which do not require a X window system because it writes directly to the frame buffer. By using this alternative memory and disk resources can be saved. There also exists a specially developed environment for handheld computers called GPE, this is based on the X window system and gtk+ 2.2 widget toolkit. This environment is used together with the familiar kernel that is offered on www.handhelds.org.

4 Realization

This chapter will contain a detailed description of the actual implementation of the software.

4.1 Design

4.1.1 Protocol summary

As mentioned earlier in this report, the Minec 3x should be able to transfer data in many different ways. One of the tasks in this master thesis is to design and create the communication interface for the Minec 3x. To be able to do so, different data transfer protocols have been studied. Among all these protocols, the most suitable ones have been chosen for the communication interface. The choices made and the motivations for them are presented under this headline. More information about the studied protocols can be found in the chapter “Description of data transfer protocols”.

All serial transfers can be sent with the same protocol regardless of the transfer medium. The protocols that were studied for this purpose were Xmodem, Zmodem and Kermit.

Xmodem is a forerunner to Zmodem and is therefore simpler, slower and less reliable then Zmodem. The Zmodem protocol also have crash recovery and the ability to send more than one file when it is running, features that Xmodem lacks.

Kermit is also a very reliable protocol, but to achieve that reliability a lot of extra control-data is passed between the sender and receiver. Both Xmodem and Kermit are more or less stop-and-wait protocols. This means that the sender has to wait for an acknowledgement from the receiver before the next packet can be sent; this procedure slows down the transfer. Kermit does on the other hand have a sliding-window feature, which enables the sender to always send a specific number of unacknowledged packets before it has to wait for acknowledgements. Nevertheless the quantity of control data passed by the protocol makes the transfer unnecessary slow, even if the sliding-window feature is used. It should however be mentioned that dedicated Kermit supporters claim that Kermit is as fast as or even faster than Zmodem. More neutral observers argue that the circumstances during which these tests have been conducted are ideal for the Kermit protocol. The conclusions made from these tests are thus not considered to be correct. Zmodem is generally regarded as the faster protocol, since no impartial test has proved otherwise. The protocol that is most suitable for the Minec 3x communication interface is therefore Zmodem.

The ability to send files via FTP should be present in the Minec 3x. There are several different protocols that can manage this. The ones discussed for the communication interface are regular FTP, TFTP and Secure FTP (FTPS and SFTP).

Trivial FTP, also called, TFTP is a small and simple protocol as well as easy to implement. The simplicity does however come at a price; the protocol cannot list files and directories and does not provide any authentication possibilities. The biggest problem with TFTP is however that it is implemented on the UDP protocol. Since all Internet communication is wireless in the Minec 3x, this is not suitable. UDP combined with WLAN tends to create a great deal of lost and corrupt packets.

The software is most likely to be used without user authentication and today there are no plans to encrypt the data in the handheld. These facts make the use of Secure FTP a bit unnecessary. There is little point (if any) to encrypt the data when transferring it, when the data is not secured in any way the rest of the time. If an unauthorized person would like to access the data, it is probably easier to get access to the handheld itself than to try to intercept the traffic to/from it. If encryption of the data stored in the handheld becomes an alternative in the future, FTPS is the most suitable protocol. This due to the FTPS, which is based on the regular FTP protocol and is for that reason supported by most servers.
This gives that the most suitable protocol for the Minec 3x today, is regular FTP.

To be able to connect to different types of servers it can be useful with a protocol that supports different protocol datagrams. For this purpose SLIP and PPP has been studied and PPP is the chosen protocol for the task. PPP was chosen mainly because it has a lot of useful features that SLIP lacks. SLIP has no error detection/correction, compression or addressing. It is also impossible to let computers communicate with more than one protocol at the time when using SLIP. These are all shortcomings that PPP can handle. SLIP is the older protocol and PPP has a lot of features, which are useful today but that were not necessary when SLIP was designed.

To summarize, the protocols used in the communication interface will be Zmodem, FTP and PPP.

4.1.2 Graphics summary

Interaction model

From the experience of HCI and after comparing different X systems, a decision was made to use an icon-based interface implemented with help from the graphical toolkit GTK+. The X11R6 is used as X-window system, since the use of X11 as X-window system gives the possibility to run Java on the Minec 3x in the future.

The reasons for choosing an icon-based interface instead of the suggestion with menus are based on following details:

- The icon type has a more professional look than the other menu-based option.
- The icon-based suggestion makes it easier for the user to remember and learn the system, since the task will be represented both in text and with a guiding picture.
- Many of the administration tasks on the handheld are the same as the ones performed on a PC. Since the PC normally uses a lot of icons it will give the user an extra familiar feeling.

Two different handheld devices that are used in the everyday life were also compared. The first one is a Sony Ericsson mobile phone, which is icon menu based, the second one is an iPod mp3-player that is text menu based. The interface on the iPod works fine for scrolling through files, but when accessing the settings to change some of the settings it easily gets confusing, above all it is easy to loose orientation in the menu-tree. It is also, according to the HCI research, important with recognition. It gives the user better confidence about what to do next. Since most mobile phones and other handhelds such as iPAQ have icon menu based interaction style, it feels like taking a step back to use the menu-based style. There will however be some changes from the mock-ups in suggestion one. Most importantly there will not be any pointing device used. To navigate, the user must use the arrow-pad to step between the icons and buttons. In the bottom of the screen the start-, help- and desktop button will be removed. They will be replaced by an information label that tells the name of the marked icon.

Graphic

The choice of using GTK+ as graphical toolkit, running against the X11R6 X-system was pretty easily made. The picoGUI was rejected quite early, because there is currently no development on this graphical system. This, together with the fact that there has been a discussion of running Java on the Minec 3x in the future, makes picoGUI a bad option.

Microwindows has the advantage that it can run directly on the hardware and therefore does not require any other X-windowing system. It makes its footprint very small, which is a big advantage when using a small handheld.

But even this option is rejected because of the future thoughts about running Java, which needs the handheld's X-system. Microwindows also has a bit of a bad "look and feel" that makes it look cheap.

Qt and GTK+ are pretty similar systems; they are both widget toolkits that work against the X11-protocol, they are both used to create GUI:s on Linux computers (Qt is used to implement the KDE-desktop and GTK+ is used for gnome). Both of them also have good documentation and well-described API:s, which makes it easy to get started. They also by far have the most modern appearance. Since Qt has its strange licence agreement the final choice fell on GTK+.

4.2 Environment

The final software is supposed to run on a handheld with Linux as operating system. The new handheld, Minec 3x, is still under construction due to some unexpected delays. Since this GUI is meant to run on this machine, the program has never been tested on the actual target. Instead the program has been run and tested on an iPAQ, which contains an Intel X-Scale processor that is similar to the one in the Minec 3x. The iPAQ also has the same familiar distribution and window manager as the Minec 3x will have. The only difference between them, besides physical appearance, is that the Minec 3x probably will be equipped with a newer version of the Linux kernel than the iPAQ. The kernel that has been used for testing on the iPAQ is 2.4.19, the kernel intended for the Minec 3x is 2.6.X. The only problem this has caused the project is the possibility to choose and install drivers, in particular the USB device driver.

The development of the GUI has taken place on a laptop running the Linux flavour Gentoo 2.6.5 [15]. The software has then been compiled for the laptop using gcc 3.3.2 and cross-compiled for the handheld using a tool-chain downloaded from Handhelds home page [16]. This tool-chain contains the compiler arm-linux-gcc version 3.3.2. To get access to GTK+ graphical widgets, the GTK+2.2 libraries from the familiar installation were copied into the tool-chain on the PC and linked to in the makefile [17]. The GTK+2.2 libraries are already present in Gentoo 2.6.5 so nothing needed to be installed for the PC libraries.

During the development of the GUI, several drivers had to be installed on the iPAQ to make the different transfer media work. These drivers will be described in this text. Please take notice that the drivers are not the same as those who will run on the Minec 3x.

The Minec 3x's transfer modules are all built on the printed circuit board, these devices are therefore not the same as the ones used for this development. The handheld's modules may thus need different drivers to work. To make IrDA work on the iPAQ two modules were installed, thereafter the interface and discovery were activated. Discovery is a feature that allows IrDA to look for other IrDA units. USB on the iPAQ needs an USB gadget driver to be able to act as a peripheral device. Unfortunately this driver is not available for such an old kernel as the one running on the iPAQ and has therefore not been tested. Since there was no access to a WLAN card, an ordinary LAN card was used for file transfer over FTP.

4.3 Hardware architecture

The service-GUI is especially designed for the Minec 3x. Its hardware architecture has therefore played a big part when designing and developing the software. The hardware parts that have affected the design and development strategies the most are: the processor, RAM and flash memory, keyboard and finally the display. The different transferring modules have of course also played a big part during the development. In fact some modules are so specific that its functionality could not be completely implemented, since the hardware is not yet available.

Processor: The processor selected for Minec 3x is an Intel PXA255 processor with X-Scale technology. The processor runs on a frequency of 400 MHz. It is especially designed for handheld computers and is one of the most used processors for PDAs.

RAM: 32 MB SDRAM is available on the handheld. The memory is divided between two chips with 16 MB memory in each chip.

Flash: The flash card used in the Minec 3x can store 16MB data. All memory is present in one card. No memory expansions are possible.

Keyboard: The keyboard has a total of 27 buttons, 21 of these have multiple functionality. Due to the multifunctionality the Minec 3x's keyboard can produce 59 different scan codes. This solution makes it unsuitable to use some standard key combinations such as <ALT+TAB>, since the ALT and TAB functionalities can share the same button (see Figure 11).

Display: The display used is a colour display with backlight but it is not a touch-screen. The display has 144 x 128 pixels divided on a 51,5 x 41 mm viewing area.

4.4 Implementation performance

The software is completely implemented in C, since standard GTK+ without any wrappers were used. There is, for example, a possibility to use a wrapper called gtkmm that enables programming in C++. There exist interface builders for GTK+, Glade is an example of such a builder. The Minec 3x GUI is however not implemented using any interface builders. Glade makes it difficult to control appearance as well as size of text and windows. Since the Minec 3x has such limited screen size it is very important to be able to have full control over the object's appearance.

In GTK+ there are many pre-built widgets, such as file selectors and dialog windows with buttons, but they are made for a large screen, such as the screen for a PC. It is unfortunately not possible to change size on the widgets, which makes them impossible to use for our purpose. The only option left was to implement all the widgets ourselves.

Although the GUI is implemented in C, it is made as objectified as possible to make it easy to add or remove features or views (windows). Because of the need to upgrade the software when the hardware is updated, this has been an important aspect from the beginning. The Minec 3x is also a multi purpose handheld where the customers can choose which hardware modules they need. For example, some may choose to exclude Bluetooth and then the Bluetooth views/settings will be easy to inactivate or remove. To make inactivation as easy as possible, every view is implemented as a module in a separate file, without unnecessary dependencies. As simply as one module can be removed, one can be added.

If the end-user wishes, the service-GUI can be started after the user applications, this will not affect the GUI's possibility to handle the already started applications or settings. The interface can however not be stopped by a user without executing a command from an X-terminal, this is a design request from Datalogic AB. The interface design also serves as a security barrier for inexperienced users, since you can never unintentionally close down the GUI without knowing how to activate it again. If it would crash, any started user application will not be affected by it.

Minec 3x uses a flash card instead of a hard disk as PCs do. A significant difference between these two, that affects this implementation, is that the flash memory can only be rewritten a fixed number of times (ca. 100000 cycles). Although this number is very high, the GUI should not use the flash more than necessary. An exception from this rule in our implementation is two scripts that generate a file each. The motivation for this extra writing is that the GUI uses these files to be able to control already started applications. For all other inter-process communication, pipes or named-pipes (FIFOs) are used.

The data that is written to the flash consists only of configuration files, which are required for the settings to be saved when the handheld is shut down. All configuration files have a standard layout; it is therefore possible for an administrator to configure several handhelds at once, simply by uploading the same configuration file to all computers.

4.5 System description

GTK+ is an event-triggered toolkit, which means that the program spends most of its time in a main loop waiting for an event to occur. E.g. an event can occur when a button is clicked or a timeout interrupt is triggered. The callback functions can only take one data parameter, so to be able to send several parameters, structs have been widely used. Even here the structs are as objectified as possible, which in this case means that every view has its own struct. Some additional structs to handle settings and configuration files are also included in the solution.

When an event occurs, the system calls for a specified callback function and executes the code. This particular system contains three timeout events; all other events are triggered by user activities. The first two timeout events handle the events in the status window and the third timeout event handles the information update in the power view. All of these status indicators are shown in the status window (the status bar shown at the bottom of the window). The mentioned events are triggered every seventh respectively every second.

As mentioned earlier, the program is made of several different views. The views are created as separate windows and are placed on top of each other. This goes for all windows except for the status window that is always visible to the user; the status window is situated at the bottom of the display.

The program is initiated to show the main menu as the first view, since this view is the big navigation wheel in this solution. From the main view the user can navigate through the whole system. You can compare this view with the desktop in Microsoft Windows on a PC. Exactly like the PC's desktop, the user can never unintentionally close down the main menu. The only ways to close down the main menu is to logout or kill the GUI from a terminal window. Further more the logout option is only activated if the user uses the version of the software which contains user authentication. In the original version of the GUI, no user authentication is required, which is a design request from Datalogic AB. Datalogic AB's experience is that most customers do not use user authentication, so the original GUI version will probably be the most frequently used.

This application is dependent of the existence of a certain directory structure to be able to work and appear properly. The file tree must look like this:

- **/GUI/apps/** contains all applications.
- **/GUI/auto/** contains links to all applications that should be started automatically
- **/GUI/icons/** contains all images in the program
- **/GUI/script/** contains all help scripts
- **/GUI/settings/** contains all configuration files.
- **/GUI/tmp/** contains all temporary files and named pipes

4.5.1 Navigation

Since there is no touch-screen or other pointing device, all navigation is handled by the arrow-pad and the <TAB> button. To move focus from one widget group to another (i.e. move focus from buttons to text entries) the <TAB> button is used. For moving the focus between widgets in the same group the arrow pad is used. To select a focused object (i.e. click a button or select a file) the <ENTER> button is used. Entering letters in a text field works like writing text on a mobile phone (i.e. same key for at least three letters) with one exception -on the Minec 3x the user must press <ALPHA> before entering letters. To see the keyboard layout, se Figure 11.

Figure 11 Keyboard

4.5.2 View functionality

Under this headline the different view functionalities are described in detail.

As shown in the theory study it is important that the user easily can orient him/herself in a menu system. To help the user with this, certain measures have been taken to provide as much feedback to the user as possible, regarding arrow movements, menu selections and location in the menu hierarchy. Therefore all focused icons and list items in the program will be displayed with a dark blue background. This is handled by two different events: focus-in and focus-out. When an icon gets the focus-in event, the icon's background is switched to dark blue. The focus-out event turns the background back to normal. These events occur when the user navigates to the icon using the arrow pad. All kinds of buttons indicate that they are in focus with a dotted frame around them which is automatically triggered by the button widget. All views have a headline at the top of the window, explaining to the user which view is opened.

The views that handle settings that need to be saved, use configuration files, all the configuration files have the same design. The configuration file names starts with a dot '.' to separate them from ordinary files and they have similar design as the configuration file shown in figure 12. The text before the colon is always written with capital letters and its only function is to show which setting the saved data represent. The file design was chosen so that the user easily can edit the configuration files without starting the GUI, for example when using remote login via telnet. An example of the file notation is shown in Figure 12.

```
ESSID:minec
ENCRYPT:1
KEY1:khog76erty23fd
KEY2:asdytlkqwertoi4
KEY3:
HEX:0
USEDKEY#:1
```

Figure 12

4.5.2.1 Main menu

Because of the decision to use an icon-based menu system, different images represent different menu options. The images have been carefully chosen so that each image in an obvious way represents the menu option itself. For example an image of a folder represents the file browser.

This GUI has ha lot of different functionalities that require different menu options. To gather them all in one menu is impossible, so many menu options in one view would be very cluttered and difficult to grasp. To solve this, the most frequently used functionalities were placed in the main menu, which makes them easy to access. All configuration functionalities were placed in a sub menu called *Control menu*.

As a precaution the main menu can never unintentionally be closed down by the user.

4.5.2.2 File browser

The file browser displays the file tree. To help the user navigate in the file tree the file browser always starts in the user's home directory. The user can browse through the file tree and perform certain editing possibilities (*new folder, cut, copy, paste, delete, rename, transfer file* and *exit*). To implement these features different Unix commands and functions are used to execute the tasks, for example, "mkdir" to make a new folder and "cp" to copy. A child process executes all of these commands.

To ensure that no data is lost, a file that is cut will not be removed from its original location until the user has performed a successful paste. The file will be pasted even if the directory already contains another file with the same name as the one being pasted. The user must check this him/herself, no warnings will be displayed. This will result in that the old file will be overwritten by the one pasted into the directory. *Paste* only fails if the user doesn't have writing permissions in "paste directory". The command *rename,* like *paste,* do not care if there already exist a file with the same name as the renamed file, which means that the old file will be overwritten if the user is not careful. If the user tries to rename a file without having permission for it, nothing will happen to the file. Copied and cut link files are made absolute instead of relative to avoid broken links. This means that the path is given from "/" instead of from the links position in the file tree.

It is possible to send a file from the file browser, the selected file will be sent with the transfer settings in the configuration file for file transfers.

4.5.2.3 Applications

This function displays all executable programs in the application directory. From here the user has the possibility to start and stop programs. The programs that are started from here are started as independent processes that will not be affected by the GUI. If the user should choose to start an application several times there will be several applications with the same name running in the system. These applications will then only be listed as one running application. When the user chooses to stop one of the applications, the GUI will stop all applications with the same name.

It is therefore impossible to start two applications named X and stop only one of them.

To keep track of the running applications a script is used to save all, by the user, started applications to a file. The names of the applications in the application directory are compared to those in the file. If the same name appears in both places the application is assumed to be running. When the user wishes to stop an application, another script is run to save all pid numbers and the corresponding application name to a file. The main process then kills all pid numbers with an application name that matches the one the user wants to stop.

4.5.2.4 X-terminal

The X-terminal is started as a background process to allow the GUI to function as usual. The X-terminal used is rxvt, which is a slimmed-down version of xterm. The terminal starts with the working directory set to the user's home directory.

4.5.2.5 Transfer File

The user can choose between two different protocols: Z-modem or FTP. If Z-modem is chosen, the transfer medium used is the one that is saved in the configuration file. This may be either one of the following: Bluetooth, IrDA, RS232 or USB. If FTP is chosen, the TCP/IP protocol makes a decision between using Bluetooth or WLAN for the transfer, given that both media are present. TCP/IP makes this choice based on which medium that can offer the cheapest way. The path with the fewest routers and the fastest connections is in general called the cheapest way.

The selected file is sent over the chosen medium. The user then gets feedback if the transfer was successful or not. Only one file can be chosen at a time and the user cannot choose to send a whole directory. Figure 13 illustrates a flow scheme of a file transfer.

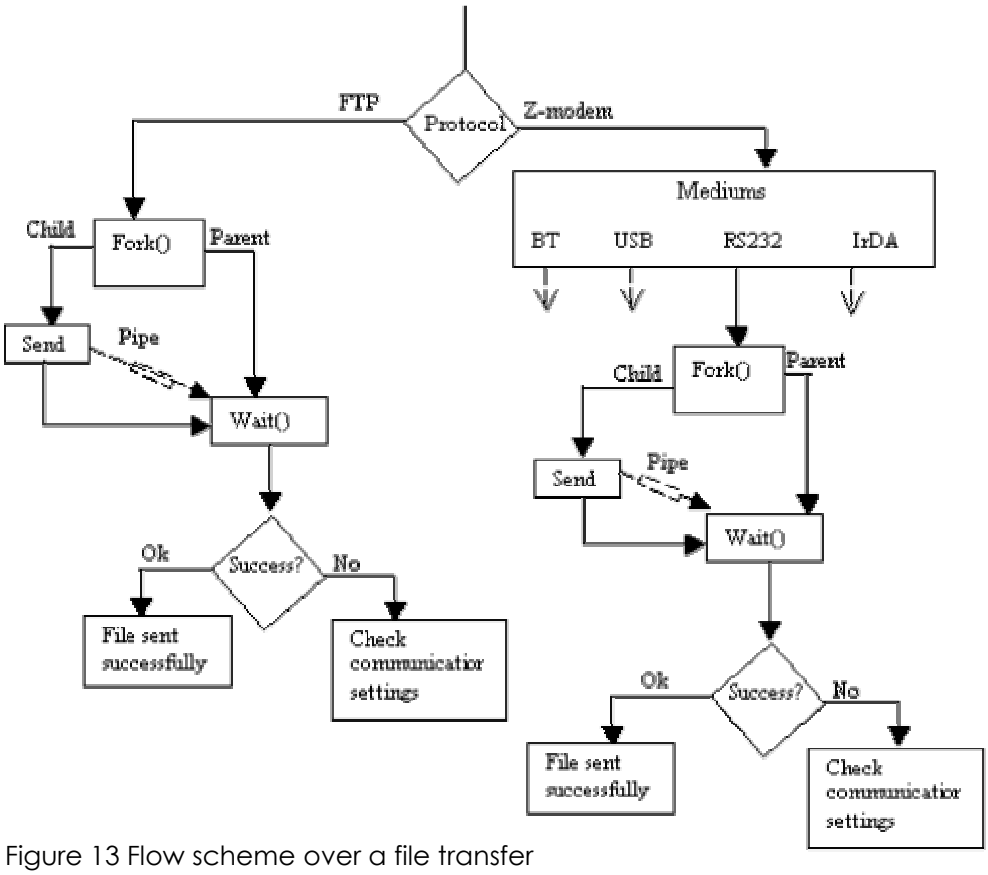

Figure 13 Flow scheme over a file transfer

A separate help script manages the FTP connection. The script uses the settings saved in the configuration file, which are: server address, username and password. The password is thus saved in a file in plain writing. It would however make little difference if the files were encrypted, since its content will be decrypted and read up to communication settings. Thereby anybody having access to the handheld can access the server without knowing the password. This design decision was made to make the transfer as simple as possible. Another option is to never save the password to a file and thus make the user write it every time. In the present solution the FTP server must handle all security issues. An example of this can be to let the server check that the handheld's IP address matches the one assigned to the account. The server should have restricted directory access; a user specific upload directory would prevent the user from seeing somebody else's files. Only some examples are mentioned here; there are many different approaches to handle this issue.

4.5.2.6 Control menu

All configuration functionality has been gathered in a separate menu, called the control menu. This menu can be seen as the second navigation centre in the system. The control menu handles the focus events exactly as the main menu.

4.5.2.7 Services

The menu option services is intended for administration of different services on the handheld. At present, the only service on the handheld is a telnet server but it is possible to add web server and/or FTP server controls in the future. To administrate these services the user must have root privileges, feedback about this is given to users who are not allowed to manipulate these services.

Regular users (not root) are not allowed to manipulate and bind ports between 1 and 1024, because the OS reserves them for common processes (mail, FTP, HTTP, etc.). If the user has root permissions, the user can start the server. When starting the telnet server, a link to the telnet program is also added to the autostart directory, so that the settings do not have to be remade on every start-up. If a version of the telnet server is already running, nothing will be done. Thus, two telnet servers can never be started from this view.

The telnet server started is the telnet demon utelnetd, which is derived from Axis tools [18]. This telnet demon was chosen because of its ability to handle several logins without having to restart. The default telnet server that followed with the Linux distribution, used for development, needs to be restarted after every login.

4.5.2.8 Power

This function's only purpose is to display the status of the handhelds power source. To be able to always provide the user with accurate information, the information is updated every second. At first this was handled by using multiple threads, but it resulted in somewhat mysterious graphic disturbances. Instead the update is triggered by a timeout interrupt. The reason for the mysterious behaviour is probably that the X-window system batches up commands and sends them to the X-server in batches instead of immediately. It is not a problem when running non-multithreaded programs, because the batches are sent as soon as the process returns to the main-loop. In multithreading programs this must in someway be handled separately. When the timer times out an event occurs and information from the event handler about the battery is fetched, calculated and displayed.

The information about the battery is obtained with the apm command, which translates the information in the kernel “file” /proc/apm to human-readable format. The files under the directory /proc are actually not a part of the file system and should be considered as a virtual file system containing kernel information. At first only the sub function apm_read was used, but it proved to be a bad idea. The problem that occurs is that the GUI interferes with the kernel when the kernel updates the proc “files”. Therefore a decision was made to use the complete apm command, which solved the problem since apm checks when it is ok to access the “file”.

4.5.2.9 Time and date

This function can only be used by users with root privileges and feedback is given to users who are not allowed to change the time settings. The reason why root privileges are needed to change the time is that when the user saves the changes the BIOS time is changed as well. For setting the time the stime function is used, it calculates the time and date based on the number of seconds past since 00:00:00 on January first, 1970 (GMT).

Autostart

The autostart option is as the name suggests a set of programs that will be started automatically when the handheld is turned on. The set is simply symbolic links to different programs in the file hierarchy. The autostart set can be edited by the user by adding and removing programs. The changes will of course take affect at the next start-up.

When a program is added to autostart, a link is created to the chosen program. The link is then moved to the autostart directory. If the selected application is not executable, it will not be added to the autostart list. Since the autostart directory only contains links, no programs can be permanently removed from this view. It is a design decision to prevent the user from unintentionally remove the actual program.

When the GUI is started, a separate process for each program is created so that the programs will remain unaffected if the GUI should go down. If a link to the telnet daemon is found in autostart, a check is performed which verifies that the user has root access. If it turns out that the user does not have root access, the telnet server is ignored in the upstart list. However, it will remain in the autostart set. A user without root access still has the possibility to add or remove the telnet server from the autostart set, but can never actually turn it on. The reason for this is simply that Unix and Linux do not allow a user to bind a socket to a port between 1 and 1024 when the user does not have super-user permissions.

The summary is that it is impossible for an administrator to configure the telnet server to be started at all times for all users.

4.5.2.10 WLAN

WLAN has a lot of configuration possibilities, for example different WEP keys, encrypted transfer or not, network name etc. Such settings as these require a lot of space to be easy to grasp. At the same time it is difficult for the user to configure the WLAN settings if they reside in different menus. The solution to this problem was to use tabs.

When the user saves the settings a new configuration file is created. A new script file will also be created when saving the settings that the user has chosen to use. Executing this script-file starts the WLAN-card, the script file uses the iwconfig commands to set the parameters of the network interface. The difference between these two files is that the configuration file stores all settings, even the inactivated. The script file only contains the settings that are about to be activated. For example, encryption keys may be stored but the user has chosen not to use encryption. This means that the keys will be saved in the configuration file but not in the script file. The saved settings are used as default/initializing values the next time the user enters the WLAN settings.

```
ESSID:minec
ENCRYPT:1
KEY1:asdjkrewu32
KEY2:hjas876adfd
KEY3:dalsdf4353er
KEY4:
KEY5:
HEX:0
USEDKEY#:2
```

Figure 14 Configuration file

```
#!/bin/sh
iwconfig eth0 essid minec
iwconfig eth0 key s: hjas876adfd
dhcpcd
```

Figure 15 Script file

An ordinary user, i.e. a user without root access, cannot start the WLAN-card. If the user does not have root access, feedback will be given to the user about why the WLAN card cannot be started. A user with root access can of course start the WLAN-card whenever he/she wishes. This setting can however not be saved, the user can therefore not start the WLAN-card once and then expect it to be started every time after this. This is most consciously done since starting the WLAN-card tends to take quite some time. If it were to be done on every startup the user might get the feeling that something went wrong and that the program has failed to start. Because of this delay the user is warned when he/she is starting the card. The delay arises when the WLAN-card tries to locate the access points and communicate with the dhcp-server. This delay may vary from time to time.

4.5.2.11 Bluetooth

The functionality for the Bluetooth menu option could not be fully implemented. The Bluetooth module for the Minec 3x is integrated on the main PCB and since the hardware was still not functioning there was no Bluetooth module available. Datalogic AB has chosen to use their own Bluetooth stack but like the hardware it was not yet ready to use.

The Bluetooth functions available to the user on the handheld are possibilities to see the address to the Bluetooth device, check/change device alias and investigate the surrounding area for other Bluetooth modules.

Everything is implemented for this functionality, except for the execution of the Bluetooth commands.

Parts of the code that is affected by this are commented so that the developers at Datalogic AB know where to replace the comments with their own commands against the Bluetooth module.

The Bluetooth settings are saved in a configuration file, which is used when initiating the Bluetooth module at start-up. The file contains alias to the handheld's Bluetooth device and the address to the server's Bluetooth device.

4.5.2.12 Wedge

The wedge is a software program which uses a barcode scanner for input and sends data directly into an application by emulating a keyboard stroke. When the wedge is activated the barcode scanner is redirected to stdin, e.g. the same as the keyboard, which makes the scanned code appear on the screen. The Wedge supports the use of end characters to separate one barcode from another; this end character can be selected by the user.

These settings are also saved to a special configuration file. The file makes it possible to keep the settings when the handheld has been restarted.

4.5.2.13 Communication settings

The communication settings were created to make the administrator's work a little bit easier. With help from these functions the user can configure both FTP and Z-modem settings. The thought is that all transfer settings that demand a little bit more computer knowledge can be set in advance, by someone that knows what he/she is doing. The settings are saved and restored at every start-up. In reality this means that an inexperienced user never should have to edit something in here. All transfers will be handled automatically if communication settings are configured properly.

A user can select a default transfer protocol. The settings also give the user the possibility to set the default protocol to "none". This results in that the user will be prompted to select a protocol every time a file is about to be sent. This design choice was made to give the expert user a possibility to easily change protocol, when perhaps sending files over different protocols might be more suitable. A less experienced user may prefer to move around in the GUI as little as possible. When using Z-modem the user can choose between a number of different media (i.e. IrDA, Bluetooth, RS232 or USB), while FTP transfer only can offer WLAN or Bluetooth. On the other hand, FTP offers user authentication and in general a faster file transfer. The inexperienced user might find it confusing to choose between different protocols. If the user does not know what Z-modem and FTP mean it is difficult to make a choice between them. It is therefore better if these settings are set by default. Then the user does not have to worry about anything else than selecting a file and press send.

To set up a FTP connection a server address, port number and authentication data are needed. The Minec 3x will most likely be used in such a way that it will connect to the same FTP server every time. Things that can change from time to time are authentication data, such as username or password.

Every character in the password is displayed as an asterix ('*'). Since the configuration file is in plain text there are no big obstacles to access the FTP password. However the asterix makes it impossible for any unauthorized person to peek over someone's shoulder and see the password. If someone wants to get authentication information they at least have to get their hands on a handheld. To encrypt the configuration file would probably take care of this problem, but as mentioned earlier the security needs for this product are very low.

To use the Z-modem as transferring protocol, a medium to transfer the data over must be selected. If Bluetooth is selected, a possibility to change the default Bluetooth server address is given. It is the same address as the one that can be set in the Bluetooth view. The only reason for this address setting possibility is that the administrator should not have to move through several views in the GUI to configure the transfer settings. If the server's Bluetooth address is changed in this view it will automatically be changed in the Bluetooth view as well and vice versa. A Z-modem transfer is managed by the command sz. This command demands that stdin and stdout is redirected to the serial device used for the transfer (for example /dev/ttyS0 or /dev/ircomm0). This is handled by a script and the user never has to think about this.

4.5.2.14 Status window

The status window functions as an information bar. Here the user can see the menu option that is in focus, Bluetooth status, WLAN status and battery status. Current time is also presented. The status window is always visible to make this information available to the user all the time.

To always keep the text string that explains the focused menu option up to date, it is updated with each icon's name in the focus-in event. The rest of the status icons are updated continuously by timeout interrupts.

The Bluetooth functionality is not implemented due to the problems with the hardware, as mentioned earlier. The purpose of the Bluetooth status icon is however to indicate if Bluetooth is active or not.

Unlike Bluetooth the status icons for WLAN and Power does not only show if the device is active or not, they do also display the link quality for WLAN and the battery's power level. The values for the WLAN and the power icons are fetched from the kernel's "proc files". These values are divided into several intervals representing stages from low to high quality.

The Clock is presented with 24-hour notation and adjusted according to the computers time zone. The clock's value is calculated from the number of seconds past since 00:00:00 on January first, 1970 (GMT).

5 Results

After 20 weeks the service-GUI is ready. The result is an interface that is both easy to learn and to handle according to the usability test. The service-GUI has been implemented so that it should be easy to add extra features and setting possibilities. All the different views have been as isolated from each other as much as possible, making it easy to add or remove views in the future. The design has been somewhat altered during the project, mostly because of Datalogic AB requirements and wishes. Though the central idea and main design have remained the same since the first proposition was presented to Datalogic AB.

The major goal with this GUI was to present a graphic interface to the user, where it is simple to configure and maintain all necessary settings without Linux knowledge. Some aspects that had to be considered during the design phase are the lack of pointing device, limited input possibilities and the small display. The fact that no pointing device exists isolates the navigation possibilities to the keypad. The keyboard on the Minec 3x is built so that the same key is used for several different characters or functions. This makes it time consuming for the user to write a lot of text. To solve this, the GUI offers the user, as often as possible, to choose input data from lists. A small display is also challenging, because information to the user must be kept short enough to fit the display but still be understandable.

To make the user feel at ease, the GUI design is built on a combination of every day environments, for example menu systems found in mobile phones and the ordinary confirmation popups, used in operating systems for PC:s.

To verify that the GUI really is as intuitive as hoped, three independent test persons were asked to perform certain tasks with help of the GUI. The result from this test came out well and is described under the headline *Usability test.*

If the user wants to benefit from the entire GUI he/she must have root privileges. Linux as operating system does not permit certain commands to be executed by anyone else but root. This GUI is running on top of Linux and cannot override Linux security rules, so this is not a GUI design choice. Examples of these tasks are starting telnet-server, changing time/date or starting the WLAN card. If a user enters a view that needs root privileges without having those privileges nothing can be changed in that view and the system informs the user why.

As mentioned earlier in this report, GTK:s complete API cannot be used. This is because some parts of the API can only be displayed when PC sized monitors are used. Therefore these items have been created from scratch to suit this implementation. Different problems have occurred due to this, however all problems but one has been evaded without affecting the GUI:s appearance and functionality. The one problem that could not be solved without changing the original design was the edit possibilities in the file browser. Originally the intention was to represent the edit possibilities with a drop-down menu. The drop-down menu did however cause focus problems in the feedback popups. After testing different approaches to solve this problem, the developers of GTK+ were consulted. Even today no one seems to know what is causing this problem or how it should be solved. A decision was taken to alter the design and now the edit possibilities are shown in a separate view, for more information about this please read the chapter *Presentation of service-GUI for Minec 3x*.

The hardware for the Minec 3x is not yet completed. This has foremost affected the communication interface. The communication interface today handles data transfers over the media WLAN, RS232 and IrDA. The Bluetooth module is integrated with the hardware on the printed circuit board and is therefore not available. The USB device needs a certain gadget driver on the peripheral side to be able to work as a slave. This driver cannot be used with the kernel available in the familiar distribution (2.4.19). Some time was spent on trying to patch this kernel to 2.5.2, but since not all patches were found this turned out to be too time demanding. A decision was then taken together with Datalogic AB that no effort was to be made to make USB work. The Minec 3x will have a 2.6.X kernel, which solves the problem.

Even though the data transfer using USB and Bluetooth is not implemented, everything surrounding USB and Bluetooth transfers is implemented. For example, all graphics to change the Bluetooth settings are ready and it is indicated in the code where all executions of Bluetooth and USB commands shall take place.

Due to the delay of the hardware, it has not been possible to test the software on the actual target. The GUI has however been continuously tested on a similar hardware architecture (iPAQ H5550) and seems to work just fine. The delay of the hardware also made it impossible to implement the entire GUI. Since this is out of our control, the project overall must still be considered as successful.

5.1 Usability test

A usability test was conducted with three different persons with different backgrounds to get some indication on how user friendly the final product really is.

The three persons carried out the test individually. To make the test as realistic as possible, the test persons received some information about the product they were meant to test and the general idea behind the GUI. All three persons also received the same icon description as the one that can be found under the heading *Presentation of the service-GUI for Minec 3x*. The description contains a picture of the menu icons and a brief description of the view the user will see when pressing this icon. Five minutes before the tasks were presented to the test persons, they were handed the icon description and given the possibility to get acquainted with the navigation functionality of the service-GUI. The complete assignment, evaluation criteria's and the individual results can be found in appendix C. In short the tasks were to create a folder, copy a specific file to this folder, start a specific application and finally transfer a specific file to a specific server using FTP.

5.1.1 Result summary

As expected the result of the test reflects the test person's computer knowledge well. The most experienced computer user, test person B did not have any big problems solving the tasks. He is also familiar with Linux/Unix and therefore recognized the environment such as path notation and icons. The other two test persons have only been in contact with computers running Windows and found the environment a bit unfamiliar. Their problems with task one has to do with their non-existing experience of Linux/Unix. This conclusion is drawn because the copy/paste part of task one went well and this part is common for both Windows and Linux/Unix.

Task number two went fine for all test persons. If test person reads the icon description he/she is almost told how to solve this task. When entering the right view there are few options and they are very intuitive.

The task where the user should transfer a file proved, as expected, to be difficult for the less experienced users. If the default communication settings were correct, all users would have succeeded with sending the file at once. The problems occurred when the less experienced users had to configure the FTP settings. As mentioned earlier in this report the communication settings view is intended for a person with a bit more computer experience. Considering that is was the settings themselves that caused the problem and not where to find them the decision to create the communication setting view seem to be correct.

This would of course not be suitable if the settings had to be changed often, but the Minec 3x will most likely be connected to only one server. This makes it unnecessary to display the FTP settings every time a file is to be transferred.

The overall impression is that once the users have become familiar with the Linux environment and the lack of pointing device, the GUI will function very well. The problem with no pointing device will probably solve itself when the GUI is run on the correct target. For example this is no problem when handling a regular mobile phone.

5.2 Presentation of the service-GUI for Minec 3x

The Minec 3x can in some aspects be custom made, e.g. which transfer medium devices to include. The customer may not wish to include all transfer media, simply because they have no use for them. In that case, the parts of the GUI handling those devices will be greyed out.

This presentation shows the GUI when all transfer media are present and the user has root privileges. Therefore the GUI is shown on a fully equipped Minec 3x. To be able to show the complete main menu the presentation is of the version using user authentication.

Under the following headings all different views that the GUI contains will be described. For each view a screenshot is displayed, accompanied by a detailed description of the view's functionality.

5.2.1 Login window

The GUI developed for the Minec 3x comes in two versions. The only difference between the two is that one version uses user authentication and the other one does not. A login window is only present in the version that supports user authentication. A logout option is also activated in the main menu. These handhelds does not often require user authentication so the version without login window will probably be the standard. In the version with user authentication, the user must login with username and password to be able to proceed to the main menu. The password will be shown with asterix characters to prevent other users from seeing the password in plain text. When this is done the user simply presses "Login" and a verification of username and password is done. If the authentication data is correct the user will proceed to the main menu. Otherwise a red warning text will appear in the login view and the user will get a second chance to login. If the user chooses to press "Cancel", the text fields are emptied.

Figure 16 Login view

5.2.2 Main menu

The main menu is the first thing the user will see after login or directly if no user authentication is used. It functions as the main navigation wheel. Every icon represents a shortcut to different tasks in the GUI. When the user clicks on an icon the corresponding view will appear.

Figure 17 Main Menu

These six tasks can be accessed from the main menu:

File browser
When the file browser icon is clicked the file browser view will appear. From this view the user can browse through the file system and also edit the file tree as he/she may see fit. This view will be further described under the headline *File browser*.

Application
When the user clicks the application icon the application view will appear. Here the user is given the possibility to view all applications in the handheld and choose to start or stop them. The application view will be further described under the headline *Applications*.

Control menu
When the control menu is accessed the user will enter the second navigation wheel in this implementation. From the control menu the user can choose to edit and/or check several different settings. This view will be described in further detail under the headline *Control menu*.

X-terminal
In contrast to the other icons the X-terminal shortcut does not open another view in the service GUI, instead the X-terminal application is started. This terminal is a regular command prompt window, which the user can use to execute other Linux commands.

File transfer
When the file transfer icon is clicked, a view that gives the user possibilities to choose a file and then send it to a server will appear. The functionality will be further described under the heading *File transfer*.

Logout
This option is only activated in the version of the GUI which supports user authentication, otherwise it is faded out. When the user clicks this icon all programs started by the user are shut down and the login window will become visible.

5.2.3 File browser

The file browser view lists all files, directories and links in the file tree. The three different types of items that are shown in this view are all marked with different images. That way the user can easily see which item is a file, directory or a link. An image of a folder is placed in front of a directory; an image of an ASCII page indicates a regular file and an image of an arrow in front of the name indicates a link. To browse the tree the user simply clicks on a directory and then the directory's content will be displayed in the list. Note that the file browser in this GUI cannot be used to open files and execute programs.

If the user wishes to edit the file tree he/she can simply press the *Edit* button, the edit view will then be visible. The *Exit* button closes down this view and the main menu will be presented again.

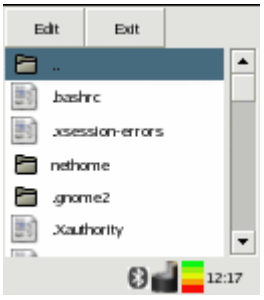

Figure 18 File browser view

The edit view contains a number of buttons, one for each of the standard browser features, such as:

- New folder
- Cut
- Copy
- Paste
- Delete
- Rename
- Transfer file
- Cancel

Figure 19 Edit view

When the user presses an option that requires some input information, a popup will appear, prompting the user for the necessary information. For example, if the user presses *New folder* a popup appears, asking for the new folder name. If no file is copied or cut the *Paste* button is inactivated. This button will of course be activated as soon as there is an item to paste.
If any drastic changes are made in the file tree, such as deleting an item, a popup will appear asking the user for confirmation to this action.

As soon as any changes have been made to the file tree the list displaying the tree will be updated, so the user can see the changes at once.

The only unusual feature in this file browser is the option *Transfer*. When the user chooses this alternative, the file marked is automatically sent using the transfer file view. If no default settings are defined the user gets an error message, where he/she is told to check the communication settings. Because of the information exchange between these two views, the file browser's full functionality requires that the transfer file view is included in the software. When the file is sent the user will get feedback on how the transfer went in form of a popup.

The *Cancel* button simply closes the edit view and the file browser becomes visible again, no changes to the file tree are done.

5.2.4 Applications

In the applications view the user can see which applications are installed on the handheld. All applications are listed with a red or a green indicator in front of the application name. The red indicator shows that the program is not currently running. The green indicator shows that the program is started and is running. The user can start and stop applications from this view as well. The user simply marks the application he/she wishes to start/stop and then clicks the appropriate button. This way it also functions a bit like a task manager.

The text below the list informs the user which file is currently marked. When an application is started or stopped the indicator in front of the application name will be changed. When the user presses the *Exit* button the view is closed down and the user returns to the main menu. No programs are affected.

Figure 20 Applications view

5.2.5 Transfer file

This file transfer view appears when the transfer file icon is pressed. The start view varies depending on the communication settings. If no default transfer protocol is chosen in communication settings, the user is prompted to choose between FTP and Z-modem. The user must make a choice between these protocols. To select a file to send the user simply presses the *Next* → button and a file browser appears. If a default protocol is defined in the communication settings view, the file browser is the first and only view visible to the user. If the user presses *Exit* instead, the view is closed and the main menu reappears.

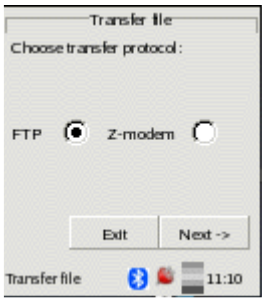

Figure 21 File transfer view

In the file browser the user can browse his/her way to the correct file and then click on it. The file name will be displayed in the text entry. When the user presses *Send* the file is sent to a server with the transfer settings defined in the communication settings view. When the file is sent successfully the user gets feedback in form of a popup. Feedback is also given to the user if an error occurs during the transfer. The error message states what went wrong; no extra information is however given to the user if the transfer was successful.

If the user presses *Close* in the browser view the result is somewhat different. If no default protocol is set in communication settings and the user has been prompted to choose one, the browser view is closed down and the user may select transfer protocol again. If a default protocol is set, the user never has to select any protocol and will therefore return to the main menu instead. Common for the two alternatives is that no file is sent.

When the file has been sent the browser window appears again, giving the user a chance to send more files.

Figure 22 Transfer browser view

5.2.6 Control menu

The control menu view appears when the corresponding icon in the main menu is pressed. This is the second navigation centre in the GUI. Precisely as in the main menu this view contains icons with access possibilities to different views. The tasks positioned in the control menu are control and setting possibilities. When an icon is pressed the corresponding view is opened.

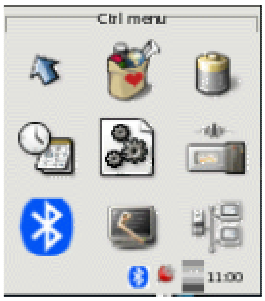

Figure 23 Control menu

From here the user can navigate to:

Go back
When this icon is pressed the control menu view is closed down and the user returns to the main menu.

Services
This icon opens the services view. Here the user can start and stop a telnet server. More information about the services view can be found under the heading *Services*.

Power
This icon leads to the power view, were the user can see various kinds of information about the power source. The power view is further described under the headline *Power*.

Time/date
When this icon is pressed the user arrives to the time/date view. Here the user can check and configure the current time and date settings. More information about this view can be found under the headline *Time/date*.

Autostart
When this icon is pressed the autostart view is opened. When this view is opened all applications that are autostarted is listed. The autostart view is further described under the heading *Autostart*.

WLAN
This icon opens the WLAN view. In the WLAN view the user can configure the WLAN interface card. More information about this view can be found under the heading *WLAN* in the report.

Bluetooth
This icon opens the Bluetooth view. Here the user can communicate with the Bluetooth module. This view is further described under the headline *Bluetooth*.

Wedge
The wedge view is opened when the wedge icon is pressed. The user can configure how the barcode scanner should behave here. More information about this view can be found under the headline *Wedge*.

Communication settings
This view is opened when the communication settings icon is pressed. Here the user can configure different transfer settings. More information about this view can be found under the headline *Communication settings*.

5.2.7 Services

The services view is opened when the services icon is pressed. The user can only edit this view when he/she has root privileges. If the user does not have root access a red warning text is displayed in the bottom of the view, telling the user that nothing can be activated. All selection possibilities are also inactivated so that the user cannot mark any alternatives.

If the user has root privileges the telnet server is activated by marking the radio button *On* and pressing *Apply*. The radio buttons are set to always follow the telnet server's status. So if the telnet server is running, *On* is checked. This provides the user with an easy way to check the status of the server.

If *Exit* is clicked the services view is closed down and the user returns to the control menu. The telnet server is not affected.

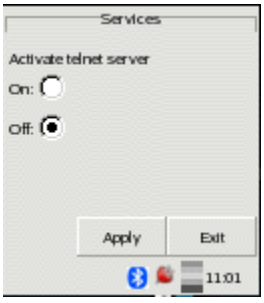

Figure 24 Services view

5.2.8 Power

The power view contains information about the power source. The progress bar at the top of the view shows how much power is left in the battery. The status label displays in which mode the battery is. The different modes are charging, charged, full, half, low, critical. The user can also see how long the battery will last, by checking the lifetime label. This time is given in hh:mm notation. The time may be somewhat inaccurate because some actions can be more power demanding than others and it's impossible to anticipate which action the user will choose to perform. The final information that can be fetched from this view is the current voltage level of the power source. As can be observed this view only offers information, no settings can be performed.

The *Close* button will of course close this view and the control menu appears again.

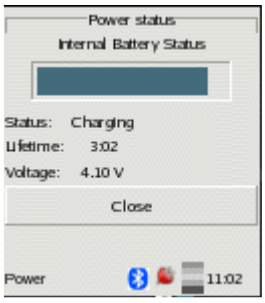

Figure 25 Power view

5.2.9 Time and date

In the time and date view the user can check and set time and date. To be able to set the time or date the user must have root privileges. If the user does not have these rights, a red information label, telling the user the reason is displayed. The *Ok* button is deactivated as well as all entries, so that they are not editable.The input data in both date and time entries, must be in correct format. The correct date format is “yyyy-mm-dd” and the format for editing the time entry is “hh:mm”. If the fields are not given in the correct format or if the user tries to set date/time to a non-existing value, for example “2004-09-32”, a red warning text will be displayed. The text appears at the bottom of the view, telling the user which field(s) is incorrect. The field that needs to be corrected is also focused. When the user presses *Ok*, the time/date data is changed and saved. If *Cancel* is pressed the view is closed and the control menu appears again and no data is saved.

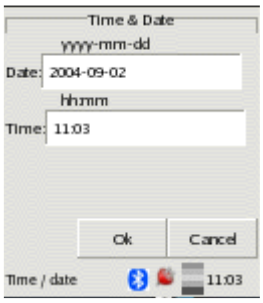

Figure 26 Time & Date view

5.2.10 Autostart

The autostart view displays a list of all programs that are automatically started at the same time as the GUI. The user can choose to add or remove programs from this list. When the user clicks on the *Add* button, a file browser appears. Through this browser, the user can navigate to the program he/she wishes to add to the autostart list. The user must first mark the program and then click the *Ok* button. If the chosen file is not executable, it is not added. The browser view is then closed down and the user returns to the autostart list, which now also contains the added program. If the browser's *Cancel* button is clicked, the browser is closed down and the user returns to the autostart list. No program is added.

The program added is started automatically when the GUI is restarted. If *Remove* is clicked, the program focused in the list will be removed. Before the program is removed, the user is prompted about this action and asked to confirm it. The program removed from the list is not stopped, nor is it removed from the handheld. However, it is not automatically started on next start-up.
When the *Exit* button is clicked the autostart view is closed down and the control menu appears again.

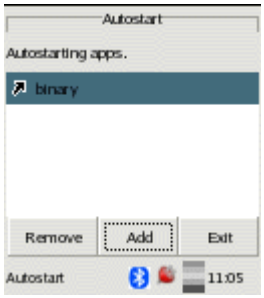

Figure 27 Autostart

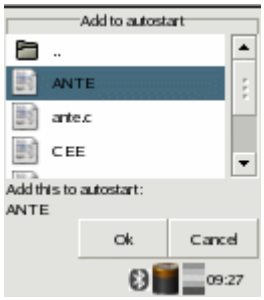

Figure 28 Add to autostart

5.2.11 WLAN

In the WLAN view all different settings regarding the WLAN card are gathered. These settings are divided into three different tabs. Under the first tab the user can fill in the network name (which is also known as the ESSID) and enable or disable the encryption of WLAN transfer data. Under tab number two all encryption settings can be found. There are five fields for encryption keys and a possibility to choose between using hex or string notation for the encryption keys. In this tab the user can also choose which one of these five keys he/she wants to use. The last tab is for starting the WLAN interface card, using the configuration file. This can only be done if the user has root privileges, otherwise no changes are possible here. Without root privileges a red warning text is displayed, telling the user that he/she does not have authority to start the WLAN-card.

All three tabs have an *Exit* button. Regardless of which one of these the user presses the same actions will take place. When the *Exit* button is pressed the user is prompted by a popup asking the user if he/she wants to save the changes. If the user clicks on *Ok* an error check of the encryption keys is performed and if all data are correct the settings are saved. If *Cancel* is clicked instead the WLAN view is closed down and the user returns to the control menu. No changes are saved. This confirmation popup always appears when *Exit* is pressed, even if the user has not made any changes. The error check performed on the encryption keys just checks that these keys are written in the correct format. The user determines the format by choosing hex or string. If it turns out that the key is written incorrectly, when *Exit* is pressed, the view is not shut down and the changes are not saved. Instead the second tab is focused and the incorrect entry is marked with a red exclamation mark.

Figure 29 – 31 WLAN settings

5.2.12 Bluetooth

In the Bluetooth view the user can configure and check the Bluetooth settings for the handheld. Uppermost in this view the Minec 3x's own Bluetooth address is displayed. The address is hardware specific and can therefore not be changed. In the text entry the user can check or edit the alias name of the Bluetooth device. By clicking on the *Inquiry and settings* button the inquiry view becomes visible to the user. If the user clicks on *Save* in the Bluetooth view, a popup is displayed asking the user to confirm the save. If the user clicks *Ok* in this popup, the device alias is changed and saved in the configuration file. Then both popup and the Bluetooth view are closed down. If *Cancel* in the popup is clicked instead, only the popup is closed down and no changes are saved. If *Exit* in the Bluetooth view is pressed, the view is simply closed down, no changes are made.

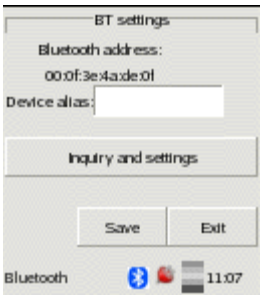

Figure 32 Bluetooth view

As mentioned before, the user is given a possibility to change the default Bluetooth server and make a Bluetooth inquiry when the *Inquiry and settings* button is pressed. In the inquiry view the user can make a Bluetooth inquiry simply by pressing *Inquiry*. The result from the inquiry is displayed with name and Bluetooth address in the list.
The uppermost entry displays the default server address. The user cannot edit this entry freely and it is thus write-protected. This is the same address that is displayed in the communication settings view. The only way to change the default server address is to choose an address in the inquiry list. To change the address the user clicks on the new server address in the list below and the new address will then appear in the default server address entry.

When the user presses *Save*, a popup is displayed, demanding a user confirmation before saving. Then the popup and inquiry view is closed down and the Bluetooth view appears again. If the user presses *Cancel* in the popup instead, no changes are saved, the popup is closed and the inquiry view becomes visible again.

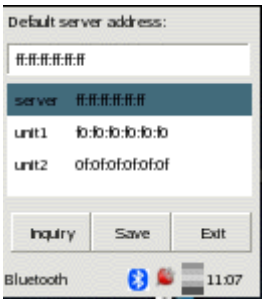

Figure 33 Inquiry view

As mentioned earlier in this report, the Bluetooth functionality has not yet been implemented. The hardware is not ready for use so there is no Bluetooth device available. Because of this, all values displayed in the two pictures above, concerning Bluetooth, are fictional.

5.2.13 Wedge

The wedge view is displayed when the wedge icon is clicked. In this view the user has the opportunity to activate or deactivate the wedge function. The function is used when the barcode scanner is redirected to stdin. This view is initialised to follow last saved settings. This means that if the wedge is deactivated, the radio button *Off* will be checked on next start-up. The user can also choose which end character to use after every scan. The available choices are *TAB*, *NEWLINE* and *NONE*. The user selects which one to use simply by focusing the combo box and then scroll in the box by using the arrow pad. The option that is visible to the user is the one chosen.

When the user clicks on *Ok* the settings are saved to a configuration file and the view is closed down. Then the user returns to the control menu.

Figure 34 Wedge view

5.2.14 Communication settings

The communication settings view is divided into three different tabs. Using this view is the fastest way to edit the transfer settings all at once.

The first tab is named *Protocol* and here the user can choose which default protocol he/she wishes to use. The selected protocol is the one that will be used for all file transfers. The user can choose between *Z-modem*, *FTP* or *None*. If *None* is set as default protocol, the user is asked to select a protocol at each transfer occasion. When no default protocol is set, no file transfers can be made from the file browser view, since that view requires a default protocol to be able to transfer files. The *None* alternative is not suitable for the daily work with the handheld and should rather be seen as a good option when performing maintenance on the Minec3X.

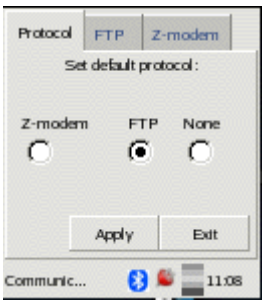

Figure 35
Communications view

The second tab is called *FTP* and as the name suggests, the tab contains all settings needed by the FTP protocol. In the first entry the user can write the address to the FTP server. The address can be written as an IP number or as a regular hostname. In the second entry the user must set the FTP server's port number. It is usually 21, but can be set to any number between 0 and 65535 that corresponds to the server's settings. The last two fields in this tab are probably the fields that need the most editing by the user. These fields contain username and password for the user account on the FTP server. If all users have personal, different login accounts on the FTP server, all users must check the username. If wrong username is entered in the username entry, the user must change username and password so that these fields contain the right data before using FTP. However, it will probably be more common for all users to share the same FTP account. In that case no changes need to be made. Every character in the password is displayed as an asterix ('*') to prevent the password from being displayed in plain text.

Figure 36
Communications view

The third tab contains the Z-modem settings. Here the user can choose between a number of different transfer media. The medium selected here is used for all serial transfers. The user can choose between *Bluetooth*, *USB*, *RS232* and *IrDA*. The *Edit* button only concerns the Bluetooth settings and is deactivated if any other transfer medium is chosen. When *Edit* is pressed a popup appears and the default Bluetooth server address becomes visible. This address is the same address that can be seen in the Bluetooth view. The user can simply change the Bluetooth server address by erasing the current address and replace it with another. When the user clicks on the *Ok* button in the popup, an error-check is performed, the address saved and the popup closed. The error check simply controls that the address is given in the correct Bluetooth address format.

If it is not, the popup is not closed and a red exclamation mark is shown to indicate that something is wrong with the address. If the user clicks on the popup's *Cancel* button instead, the popup is closed down and nothing is saved. Note that if the user changes the default Bluetooth server address here, it is changed in the Bluetooth view as well.

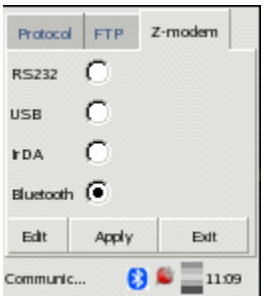

Figure 37
Communications view

Every tab in the communication settings view is equipped with an *Apply* button and an *Exit* button. These buttons do the same thing, regardless of in which tab the user presses them. The *Apply* button is deactivated until a change is made in any of the views. When the *Apply* button is pressed an error check will be performed. All settings are then saved and the button is deactivated. The error check performed makes sure that the port in the FTP settings is within a correct interval (i.e. between 0 and 65535). Observe that when *Apply* is clicked all settings in all the tabs are saved. If the *Exit* button is clicked, the view is closed down and no settings are saved.

5.2.15 Status window

The status window is always visible to the user, even if applications are running on top of the GUI. This window is situated at the bottom of the display and contains several status icons. Here the user can see the icons focused, Bluetooth status, battery status, power level, WLAN status, link quality and finally the clock. Below are some examples of the different icons that can be displayed in the status window and what they represent.

This icon represents that external power is connected to the Minec3X and that the battery is now charging.

This icon is displayed when the battery is charged and when the battery is full. The difference between full and charged is that the power is still connected to the handheld even though it is charged.

This is what the battery looks like when it is half full. The user will see the battery level shrink correspondingly to how much power is left in the battery.

The Bluetooth icon is highlighted (e.g. its blue colour is showing) when the Bluetooth device is active.

The Bluetooth icon is made grey when Bluetooth is deactivated.

This indicates the WLAN status. The icon is highlighted (e.g. its colours are showing) when a WLAN-card is present and initiated. As shown in the icon the link quality is very good.

This icon will appear to the user when the link quality is somewhat worse. The link quality is indicated by the height of the icon and it will shrink/grow correspondingly to the link quality.

This WLAN icon will be shown either when the link quality is very bad or if there is a WLAN-card present and it has not been activated by the user.

The WLAN icon is made grey when no WLAN-card is present.

This icon shows the current time.

This text string display which view is currently opened or which menu icon that is in focus. The field is left blank when no icon is marked or if no GUI view is opened.

Below are two examples on how the status window can look when the GUI is running (For icon explanation please look in the table above).

Figure 38 - 39 Different status icons

5.2.16 Feedback views

Here follows some examples on how different feedback views can look. Some views contain only information and other popups need the user's confirmation before an event can take place.

Some examples of when a popup with only information is used for feedback are when a file has been sent or when the transfer has failed. The picture on the left is a confirmation to the user that everything went well during the file transfer. On the right is an example of an information popup that tells the user that something went wrong. In these kinds of popups some information on what went wrong and how the user might solve the problem is also displayed.

Figure 40 – 41 File transfer feedback

Another example when the user gets only informational feedback from the system is when he/she does not have the privileges required to make certain changes. This will then be displayed as a red information text in the specific view. As shown in the picture below, all setting possibilities are inactivated, so that the user cannot be tricked to believe that he/she has changed the settings anyway.

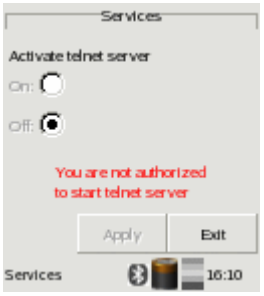

Figure 42 Authorization failure

The second kind of feedback views, are those that require confirmation and interaction from the user, before the event takes place. Below are some examples on popups that need the user's verification before the actual event is executed. If the user should choose to press *Cancel* everything will remain unchanged. These kind of popups also serve as a "wakeup-call" for the user. He/she must acknowledge critical events. Therefore the number of unintentional actions, such as saving the wrong settings or lose important data, decreases.

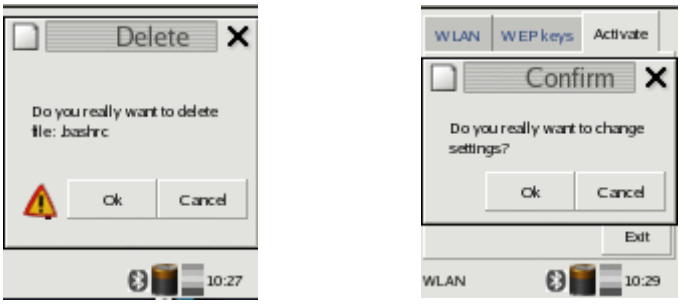

Figure 43 – 44 Confirmation popups

Another type of pre-event popup needs a little more interaction from the user. In the example below, the user must write a name in the popup before the object can be renamed. The event will not be executed until the user presses *Ok*. If *Cancel* is pressed the event will not be executed. Just like in an ordinary file browser nothing happens to the marked file/directory if the entry is left empty and the user presses *Ok*.

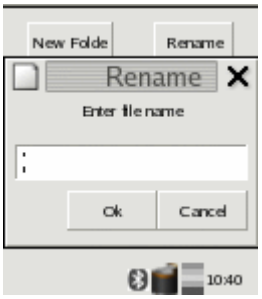

Figure 45 Interaction popup

6 Discussion

This master thesis has offered many interesting design issues to consider. The software must satisfy both the experienced and the inexperienced user, as well as provide an informative and well-presented interface even on a small display. Some measurements that have been taken to achieve this are discussed below.

6.1 Master thesis requirements

The requirements specification presented in Appendix D is as you can see not specified in detail. Because of this some requirements modifications have taken place during the development. These modifications have by Datalogic not been regarded as demands but rather as requests and desires, these requests have therefore not been documented and included in the original requirements specification. An example of such a request is the possibility to transfer a file directly from the file browser.

With this implementation all parts of the requirements specification must be considered as fulfilled, with two exceptions. In the early stages of the design phase Datalogic decided to remove the web and FTP servers from the solution, these have therefore not been implemented. The second deviation from the requirements specification is the functionality in the Bluetooth settings. These are as mentioned before in this report not yet implemented due to delays in the hardware development. Both the deviations from the requirements specification has been out of our control. This combined with the fact that Datalogic was very pleased with the final product, makes us come to the conclusion that the result must be considered a success.

6.2 Theory decisions

As mentioned earlier in this report the actual realization of the service GUI was preceded by a theory study that resulted in certain design choices. These decisions and how well they fell out in practise are discussed here.

The protocols that were chosen are Zmodem, FTP and PPP. The point-to-point protocol (PPP) was discarded early in the implementation. Together with Datalogic AB a decision was made that it would be unnecessary for the handheld to connect to the Internet through a serial connection (for example USB or RS232). This decision eliminates the purpose of PPP and the protocol was therefore removed from the solution. Regarding FTP, it was determined in the theory study that regular FTP should be the protocol for file transfers over TCP/IP. The main reason given for this decision was that no user authentication would be used on the handheld, which made Secure FTP unnecessary. A version with user authentication was however implemented later on. In this version Secure FTP should perhaps have been considered.

To motivate the use for Secure FTP the whole system needs a security upgrade. Today all users share the same unencrypted setting files and scripts, which means that authentication data for FTP is public to every one that can access the Minec 3x. There is no point in making the transfer secure if the FTP authentication data is not kept securely. The Minec 3x is not expected to run this version of the GUI, therefore these security aspects have been down prioritized. The only use for user authentication on the Minec 3x today is that every user gets his/her own home directory and no root privileges in the GUI.

The choice to use GTK+ as widget tool kit has proved to work well even if some strange phenomenons have occurred during the development. For example focus problems have arisen in different situations; the effects of them have nevertheless been solved. The positive sides with GTK+ have dominated the implementation experience though with this GUI. The things that have been valued the most are the structured and well-documented API, good mail support and the great possibilities to place as well as build own widgets.

The GTK+ libraries should perhaps have been upgraded to 2.4 instead of 2.2, to implement the GUI with the latest version of GTK+, but after examining the API for GTK+ 2.4, no important changes that would affect this implementation were found. Further more the fact that the familiar distribution comes with GTK+ 2.2 led to that no upgrade was made.

One might speculate on why this GUI is not written in Java since Datalogic AB prepares to equip the Minec 3x with a Java virtual machine. The answer is fairly simple, all end-user applications are previously written for MS-DOS and thus implemented in C. It is unsuitable to have one part of the system written in Java and another in C. Besides, the service GUI functions as an X-shell and should therefore be as fast as possible, which makes C a better choice than Java.

6.3 Implementation

The most difficult design issue has, without a doubt been to display information on such a small display. The line between displaying too little information to make the GUI understandable and displaying too much information so it becomes unreadable, is very thin. Everything from error messages to entry descriptions has been considered as information. For example the word “Host” is used even though the words “Server address” would be much more explanatory. The reason for this is simply the length of the word; the longer more describing phrase makes it impossible to fit the rest of the view within the display. Another example of where a decision has been made on what is most important to display is in the file browser. When opening this view no information is given to the user concerning current position in the file tree. Instead a decision was made that it is more important to see as much as possible of the current directory content. The content of the directory helps the users to orient themselves in the file tree, especially after browsing through it a few times. The time spent locating the current position in the file tree grows rapidly if the user can see less of the directory content. These facts led to the conclusion that the user may find it useful to see the path name the first times he/she opens the file browser. The user will value the possibility to see more items directly without having to scroll, as he/she grows more used to the file tree.

In the early parts of the design stage a decision to use shortcuts was made. The judgement comes in handy for the everyday-user, because it makes the work go faster. Later in the design phase it became clear that the menu structure would be so flat that shortcuts were unnecessary and they were removed from the solution. There are, however, two things in the design that could be regarded as shortcuts. They are not shortcuts in the aspect that the user can press two different buttons and end up in the same view. They are rather shortcuts in the aspect that the user can perform the same task from two different places. These “shortcuts” gives the possibility to send files and change the Bluetooth server address from several places. The opportunity to send a file from more than one place was created for two major reasons. If a user wants to edit the file’s properties he/she should not have to move from the file browser view to send the edited file. The second aspect is that the less experienced users may find it more logical to first locate the file and then send it, than to press “transfer file”, then locate the file and press send again. The “shortcut” concerning the Bluetooth server address, was created to let the user edit all necessary transfer settings for all media in the same view.

A thorough discussion was held before these two “shortcuts” were added to the final solution, since it can be seen from two different angels. One user might find it unnecessary and confusing to be able to perform the same task from different places, he/she may feel insecure that the changes made will actually take effect. Another user can see it as an asset that saves time and makes the work easier. The “shortcuts” were finally added based on the following conclusion; this product will probably be used in everyday work and the users will get acquainted with the software fast. An experienced user appreciates features that make the work go faster more than an inexperienced user. If someone, even after some time finds the “shortcuts” confusing, there is no need for he/she to use them. The ordinary ways to perform a task work just fine.

The complete design and implementation of this service GUI, as described in this report, are made with the thought that everybody using the GUI will have the same login (or no login at all). This is obvious when looking at the file tree needed to execute the GUI. Scripts and setting files are now placed in the root directory, which results in that there is no possibility for user specific configuration files. To make further development of the user authentication version of the GUI easier, some of these should have been placed in the home directory instead. It would have made the software configuration more personal. In the market today there is no request for user authentication on products like this. If the market changes, no big modification to make the configuration files user-specific are needed.

7 Acknowledgement

We would like to send our deepest gratitude to these people, in no particular order,
Johan Andersson, Sven-Erik Arenbalk, Joel Lindau, Viveka Lång Norberg, Helene Meyer, Jan-Erik Moström, Magnus Norberg, for their valuable help and support during our work with this master thesis.

8 References

[1] RFC, Protocol standard page, webpage, N/A,
http://rfc.net/std9.html
Last visited 2005-02-26

[2] Techfest, Zmodem standard, -,
http://www.techfest.com/hardware/modem/zmodem.htm
Last visited 2005-02-26

[3] Enterprisedt, Secure FTP information page, webpage, N/A,
http://www.enterprisedt.com/products/edtftpjssl/faq-answers.html#2
Last visited 2005-02-26

[4] Columbia University, Kermit information page, webpage, 2004,
http://www.columbia.edu/kermit/manuals.html
Last visited 2005-02-26

[5] RFC, Protocol standard page, webpage, N/A,
http://rfc.net/rfc1661.html
Last visited 2005-02-26

[6] RFC, Protocol standard page, webpage, N/A,
http://rfc.net/rfc1055.html
Last visited 2005-02-26

[7] STIMDI, HCI information page, webpage, 1994,
http://www.stimdi.se/old_web/web/public/mdi.html
Last visited 2005-02-27

[8] Allwood Carl Martin, 1991, *Människa - datorinteraktion : ett psykologiskt perspektiv*, Lund, Studentlitteratur, book.

[9] Eriksson, M. *Människa-dator interaktion*, Computer Sweden nr 46 (1997), article.

[10] Preece Jennifer, Rogers Yvonne, Sharp Helen, cop. 2002, *Interaction design : beyond human-computer interaction,* New York, Weily, book

[11] Useit, Technology information page, webpage, 2003,
http://www.useit.com/alertbox/20030825.html
Last visited 2005-02-27

[12] Umeå University, HCI information page, webpage, 2002,
http://www.cs.umu.se/~leo/HCI/Tobbe.html
Last visited 2005-02-27

[13] Shneriderman Ben, cop. 1998 (3:d ed.), *The user interface : strategies for effective human-computer interaction,* Massachusetts, Addison Wesley Longman, book

[14] Microwindows, Graphics information page, webpage, 2005
http://www.microwindows.org/
Last visited 2005-02-27

[15] Gentoo, Gentoo Linux mirrors, webpage, 2001
http://www.gentoo.org/main/en/mirrors.xml,
Last visited 2004-09-23.

[16] Handhelds, Handhelds download page, webpage, 2003
http://www.handhelds.org/downloads.html,
Last visited 2004-09-23

[17] Handhelds, Familiar download page, webpage, 2003
http://familiar.handhelds.org/familiar/releases/v0.7.2/install/download.html
Last visited 2004-09-23

[18] Pengutronix, utelnetd download page, webpage, 2003
http://www.pengutronix.de/software/utelnetd_en.html ,
Last visited 2004-09-24

[19] Intranet journal, Technical newsletter page, webpage, 2005
http://www.intranetjournal.com/articles/200208/se_08_14_02a.html
Last visited 2004-09-23

9 Bibliography

Webpages

Searchnetworking, Definitionpage, 2003,
http://searchnetworking.techtarget.com/sDefinition/
0,,sid7_gci213976,00.html
Last visited 2005-02-26

Webopedia, Xmodem definition page, N/A,
http://www.webopedia.com/TERM/X/Xmodem.html
Last visited 2005-02-26
Techfest, Zmodem standard, N/A,
http://www.techfest.com/hardware/modem/zmodem.htm
Last visited 2005-02-26

RFC, Protocol standard page, webpage, N/A,
http://rfc.net/rfc783.html
Last visited 2005-02-26

Columbia University, Kermit information page, 2004,
http://www.columbia.edu/kermit/ck80.html
Last visited 2005-02-26

Umeå University, Cognition psychology page, 2002
http://www.cs.umu.se/~leo/HCI/Tobbe.html
Last visited 2005-02-27

X Foundation, X system information page, N/A,
http://www.x.org/
Last visited 2005-02-27

X Foundation, X protocol information page, N/A,
http://www.x.org/X11_protocol.html
Last visited 2005-02-27

Microwindows, Graphical information page, 2005,
http://www.microwindows.org/
Last visited 2005-02-27

Linux devices, device information page, 2004,
http://www.linuxdevices.com/
Last visited 2005-02-27

Pico GUI, Graphical information page, 2003,
http://www.picogui.org/
Last visited 2005-02-27

Trolltech, Graphical information page, 2005,
http://www.trolltech.com/
Last visited 2005-02-27

GTK, Graphical inforamtion page, 2005,
http://www.gtk.org/
Last visited 2005-02-27

Handhelds, Handhelds information page, 2005,
http://www.handhelds.org/
Last visited 2005-02-27

iPAQ, iPAQ information page, N/A,
http://www.ipaq.com/
Last visited 2005-02-27

Linux, Linux information page, 2005,
http://www.linux.org/
Last visited 2005-02-27

W3, FTP Standard page, 1994.
http://www.w3.org/Protocols/rfc959/Overview.html
Last visited 2005-02-27

Cisco Systems, Protocol information page, 2002,
http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ppp.htm
Last visited 2005-02-27

Programmersheaven, protocol download page, N/A,
http://www.programmersheaven.com/zone22/cat198/2614.htm
Last visited 2005-02-27

Books

Allwood Carl Martin, 1991, *Människa - datorinteraktion : ett psykologiskt perspektiv*
Lund, Studentlitteratur

Preece Jennifer, Rogers Yvonne, Sharp Helen, cop. 2002, *Interaction design : beyond human-computer interaction,* New York, Weily

Shneriderman Ben, cop. 1998 (3:d ed.), *The user interface : strategies for effective human-computer interaction,* Massachusetts, Addison Wesley Longman

Backman Jarl, 1998, *Rapporter och uppsatser*, Lund, Studentlitteratur

Walla Erik, 1990, *Så skriver du bättre tekniska rapporter,* Lund, Studentlitteratur

Appendix A - Abbreviation dictionary

ACK - ACKnowledgement

ASCII – American Standard Code for Information Interchange

BCD – Binary Coded Decimal

CAN - CANcel

CP/M – Control Program/Microprocessors

CPU – Central Processing Unit

CR – Carriage Return

CRC – Cyclic Redundancy Check

EAN – European Article Numbering

EBCDID – Extended Binary Coded Decimal Interchange Code

EOF – End Of File

EOR – End Of Record

EOT – End Of Transfer

FTP – File Transfer Protocol

FTPS – File Transfer Protocol SSL

GMT – Greenwich Mean Time

GUI – Graphical User Interface

HCI – Human Computer Interaction

HDLC - High-level Data Link Control

HTTP – HyperText Transfer Protocol

IETF – Internet Engineering Force

IPX - Internetwork Packet Exchange

IrDA – Infrared Data Association

LAN – Local Area Network

LCP – Link Control Protocol

NAK – Negative AcKnowledgement

NCP – Network Control Protocol

PC – Personal Computer

PDA – Personal Digital Assistant

PPP – Point-to-Point Protocol

OSI - Open Systems Interconnection

RFC – Request For Comments

RJE – Remote Job Entry

SFTP – Secure File Transfer Protocol

SLIP – Serial Line Internet Protocol

SOH – Start Of Header

SSH – Secure SHell

SSL – Secured Sockets Layer

TCP/IP – Transmission Control Protocol/Internet Protocol

TFTP – Trivial File Transfer Protocol

TID – Transfer IDentification

TLS – Transport Layer Security

UDP – User Datagram Protocol

USB – Universal Serial Bus

WAN – Wide Area Network

WLAN – Wireless Local Area Network

Appendix B - Terminology

Access points:	Receiver for the WLAN traffic, it sends the data through to LAN.
Bluetooth:	A chip technology that enables data connection between different devices on a short-range two-way radio.
C/C++:	Programming languages
Callback function:	A function that is called when a specific event occurs and can not be called otherwise
Cross-compiled:	Compiled on one architecture (for example a PC) with libraries and compiler for another architecture (for example a handheld).
Daemon:	A process that runs in the background, handling commands delivered for remote command execution.
DHCP-server:	Dynamic Host Configuration Protocol, it distributes IP addresses from its internal pool.
Driver:	A program that determines how a computer will communicate with a peripheral device.
Flash-card:	A small memory chip that can be rewritten and hold its content without power.
Gentoo:	A flavour of Linux.
GTK+:	Multi platform toolkit for creating graphical user interfaces.
Handheld:	Any computing device that is small enough to fit in the user's hand. It includes palm-sized PCs, PDAs, Smartphones, smart pagers, and larger mini-laptops.
Icon:	A small graphical symbol representing a program, file or folder.

IPAQ:	A handheld computer (PDA) from Hewlett Packard.
Linux:	Open source operating system derived from the Unix operating system
Makefile:	A makefile details the files, dependencies, and rules by which an executable application is built.
Minec 3x:	A new handheld developed by Datalogic AB and it is for this computer the software is written.
Pipes:	Creates a communication channel between two processes
Root:	Administrator account with special privileges.
RS232:	Standard for serial data transmission using a cable
rxvt	rxvt is a terminal emulator that runs in the X Window System. It is a slimmed-down replacement for xterm.
Script:	A file containing commands that can be executed without being compiled.
Sniffer:	A utility program that captures packets going over a network
Socket:	A TCP/IP interface that allows a two-way link between two systems.
Stdin:	The UNIX standard input device, by default the keyboard.
Stdout:	The UNIX standard output device, by default the monitor.
Struct:	Groups of elements defined by programmer into a new data type.

Telnet:	Telnet allows you to connect to another computer, write input commands and execute programs.
Tool-chain:	A tree structure containing compilators, linkers and libraries for a certain platform.
Widget:	A widget is actually the name for small device, (a substantive). When we are using the word here it is an assembly word for buttons windows text entries i.e.
Wrapper:	An object that encapsulates and delegates to another object to alter its interface or behaviour in some way.
X11R6:	X11 is a windowing environment used in Unix and Linux and R6 stands for the version number.
X-server:	A software that allows X-Windows applications to be drawn on a display. The X server converts X-Windows protocol commands to local machine language commands, and the local GUI messages to X-Windows protocol commands for the X client.
X-terminal:	A window that enables a user to write commands to the computer.
X-window system:	A graphical windowing environment for UNIX. The underlying programming required by many user interfaces. Also known as X.

Appendix C - Usability test

Assignment

The assignment was divided into three tasks as shown bellow:

- Create a folder called “Test” under the path /home/candersson/ and copy the file gnome-applets.png from the path /home/candersson/src/demo and paste it in newly created folder “Test”.

- Start the application “ANTE”.

- Transfer the file gnome-applets.png from the directory created in task one with FTP. The transfer shall be made to the server: **213.113.222.57** and port **21** with these authentication data user: **test** password: **usability**.

Evaluation Criterias

The test persons’ results have been evaluated with respect to the following four criterias and the test persons’ own comments:

1. Total time
2. Number of errors (opening the wrong views, creating folders in the wrong directory, entering the wrong communication settings etc.).
3. Did the test person succeed within reasonable time?
4. General impression of the test person (Did they seem confused, nervous, surprised, reaction to feedback).

Individual test results

Test person A

Sex: Male
Age: 32
Profession: Buyer at a major energy company
Computer experience: Ms Office user and web surfing.

Result:

The first task was completed in two minutes and 41 seconds. The most time consuming task was to learn how to browse in the file tree and to understand the path that was given in the task. Since this person is not familiar with Linux/Unix, some time was spent on identifying the path notation. When the test person understood this, creating the folder and copying the file did not cause any significant problems. The task took a bit longer time than what it would have taken for a person used to Linux/Unix but this was expected. The actual assignment (create, copy and paste) did not cause any problems.

The second task was solved without any problems. It was performed in ten seconds and went pretty straightforward. This was not to surprising since the task is very easy and the icon description describes were to solve this task.

The third task was a bit difficult for the test person since he never has used FTP before. It took a bit longer than expected and the total time was three minutes and 12 seconds. The test person made a couple of mistakes. The first mistake was to try to send the file without changing the communication settings. Thereafter he tried to change the Zmodem settings. The test person was very doubtful when he read the assignment and asked how we could expect him to solve the problem.

After comforting him that most information could be found in the icon description and in the views, he made an effort to solve the problem. A bit later he was told that nothing could be broken and that it was ok to use the trial-and-error method, after this he finally solved the problem.

The test persons own comments were that he did not recognize the file structure and was searching for some kind of partition for example “c:\”. The application task was however no problem at all. He never thought he would succeed with the third task, since this was his first time transferring a file with FTP. When entering the FTP tab he understood that he was in the right place. Since he was told that it was ok to test he gave it a shot and it worked.

Test person B

Sex: Male
Age: 25
Profession: Software developer for mobile phones
Computer experience: Very good. Have used and is acquainted with most computer systems.

Result:

The first task was completed in one minute and 17 seconds. The most time consuming task was to locate the path for creating the folder and then finding the file to copy. The test person did not make any significant error such as opening the wrong view or editing the wrong directory. The person did not use more time than expected. The general impression was that the person recognised the surroundings and did not hesitate on how to use the commands.

The second task was completed in ten seconds. Which was as fast as expected. This is a very easy task when having access to the icon presentation. The test person seemed to know exactly what to do and did not make any mistakes.

The third task is most difficult one and it was also the most time consuming task for the test person, it took one minute and 36 seconds. At first the person tried to send the file without changing the communication settings, which did of course fail. When realizing this the test person glanced at the icon description to locate the communication settings icon and after that he made no more mistakes. It was easy to notice that this person is familiar with ftp and knew exactly which data to fill in were.

The test persons own comments are that it was somewhat unfamiliar with a totally new view for edit but after a couple of times it makes no difference from a usual drop-down menu. The person's comments about task number three were that he missed some information on the current settings. Upon entering the transfer file view he would have liked to see which protocol and medium that were selected.

Test person C

Sex: Female
Age: 27
Profession: Kindergarden teacher
Computer experience: Limited. Mainly uses the computer for chatting, surfing and mailing.

Result:

This test person found the tasks somewhat more difficult than the rest. The test took very long time, mainly caused by the fact that she is a very inexperienced computer user. The first task took five minutes and 43 seconds, which is too much. The user had trouble to understand that the search path was defined from the root directory and not from where she started. The ".." notation for stepping up was also something that she had never seen before. We described how the file system worked and after that things went better. After some help from us, she managed to complete assignment one.

After assignment one, we told the test person to take some time and look through the icon description once more. The second assignment went a lot better and she had completed the assignment after 20 seconds.

The third assignment did not go well at all. The test person has in similarity with test person A never used FTP. She managed to enter the file transfer view and as the other users the tried to send a file with current settings, which were set to be inaccurate. The send did of course fail and instead of reading the error message the test person clicked "Ok" and asked us what to do. After some instructions she understood that she had to go to the communications settings. When the person reached that view she managed to choose "FTP" in the first tab, the second tab turned out to be hard to understand. She never associated the label "Host" with the server address. After some guidance, she begun to enter text in the fields that she recognized from the question and after that she entered the sever address in the host field.

Then she returned to the transfer view and managed to send the file. Without help she would probably never have completed the assignment.

The third test persons own comments were mainly about that the file transferring was something she never had done before and therefore considered it difficult. But she thought that if she had to do it again she would succeed. She also thought that navigating up in the file tree was confusing and she missed the “back arrow” in Windows.

Appendix D – Requirements specification

Assignment description

A user interface shall be implemented and it will function as a base for this handheld. The interface will be the first thing that the users see if no applications are running. It is important that it is designed so that booth users with advanced and basic computer knowledge can operate easily within this interface.

The handheld is developed against an Intel X-scale processor and a display with 144 x 128 pixels. The user interface shall be implemented with a suitable graphical engine. It is up to the participating students to decide exactly which graphical engine that is the most suitable for this assignment.

An implementation of a communication interface is also included in this Master Thesis. The handheld should be able to transfer data over the following transfer media: Bluetooth, IrDA, USB, RS232 and WLAN. The participating students must decide how the transfer of data shall take place for each medium (i.e. which protocols is most suitable for the task).

The list below defines necessary features of the GUI and a brief suggestion of an interface layout:

GUI

- Shell (The ”start page” shown when no applications are running)
  - Power status (Visible from an application if the user wishes)
  - Link quality WLAN (Visible from an application if the user wishes)
  - Start file transmission
  - Start different functionality (Start menu?)
    - Start applications
    - Terminal program
    - Control panel
      - Autostart
      - Power options
      - Time adjustments
      - Bluetooth configuration
      - WLAN configuration
      - Wedge configuration
      - Services
        - Web server
        - Telnet server
        - FTP
  - File handling (Browsing possibilities similar to Windows Explorer)

Theory studies

This Master Thesis includes an analysis of the most suitable design for the user interface as well as the most suitable graphical engine. Participating students shall also investigate which transfer protocols to use for file transmission in the handheld.

All conclusions and decisions must be presented and acknowledged by Datalogic AB, before the design phase may take place.

Time estimation

This Master Thesis including theory studies is suitable for two students and should approximately be finished within 20 weeks.