File Synchronisation Software

Simon Grimshaw
Computer Science

Session (e.g., 2002/2003)

Summary

As the problem of the same important files (to the user) appearing on multiple machines the problem; how the file is kept up-to-date on all the machines arises. This is becoming an ever increasing problem as mobile computing is on the increase especially with the reduction in the price of laptops, PDA's, and to a certain extent certain mobile phones. Nowadays it is not uncommon for people to have more than one personal computer or computing device. Therefore the problem of file discrepancies between the different machines arises.

Therefore this project is about the synchronisation of file and folders between different computers. The aim of the project is to create a fast and efficient piece of software that is capable of synchronising files over a variety of different computers.

File synchronisation is incredibly important as it allows users to safely ensure the same copy of the files selected by the user is on multiple machines so that no discrepancies appear. Without this function the user would have to manually copy the files themselves. This could lead to errors, i.e. the wrong files copied, the more up-to-date file being overridden to name just a couple. These problems can lead to greater file discrepancy and hinder the user rather than helping them.

This project concerns itself with understanding the problem of file synchronisation, analysing software and consider what is lacking from the programs on offer and how they can improve. Finally creating an effective piece of software that adds something new to the file synchronisation market.

Acknowledgements

I would like to thank Dr Nick Efford (nde@comp.leeds.ac.uk) for all his valuable time and support throughout the duration of this project.

Contents

List of Figures

List of Tables

1. Introduction

1.1 Brief Outline of the Problem

As mobile computing becomes more popular as more and more people have more than one computing device, either a laptop, desktop, or a PDA. File synchronisation is becoming more important. Therefore the importance of synchronisation becomes more apparent. It is likely the same file(s) will be on both devices (be it a word document or other import files), therefore it is important that the files on both devices be as up-to-date as possible, especially if the files are used regular on both devices. Without a stable piece of software to synchronise the files, and check which file is the most up-to-date it will be up to the user. This maybe easy to do if there is only a handful of files however if there are large amounts then it becomes tedious and prone to error, this is where the importance of software dedicated to synchronisation becomes important.

1.2 Already Existing Solutions

There are already a number of file synchronisation software on the market (which will be looked at later in the report), therefore it is important to justify why another piece of software should be created in an already busy market.

Therefore this project will concentrate on creating a piece of software that is capable of file synchronisation, as well as justifying the need for any new piece of software. Also this project will look at other software on the market and to look at their advantages and disadvantages and draw an evaluation from the software available.

1.3 Minimum Requirements

- To produce a stable piece of software that can synchronise files between two different platforms. The software should be able detect when a file has changed and decide which copy to update on either computer.
- To produce a simple user interface to the software. The interface should be easy to use and intuitive for the user.
- To product basic documentation for the software. The documentation should be easy to follow, easy to understand and without too many technical terms for the novice user. The documentation should be in a text file.

1.4 Objectives

- To understand the concept behind communication between two or more computers. To understand the fundamentals on how and why it happens, and to look into more than one method.
- To understand the different styles of user interfaces available. Also to understand the different HCI techniques and to decide which is the most appropriate to use.
- Expand my knowledge of programming languages to include new libraries for either C++, Java or other language.
- To be able to synchronise files between at least two different computers.
- To create an easy to use user interface, either text base interface or a GUI interface.
- To produce documentation for the final piece of software.

1.5 Report Structure

This chapter briefly explains the problem of file synchronisation and list the minimum requirements and objective of the project. The second chapter researches the background reading to be able to tackle the problem effectively.

The third chapter analyses current file synchronisation software available in the market and discusses the advantages and disadvantages of each solution. A conclusion is then drawn on the information gathered in the analysis.

The fourth chapter is split into two parts, the first part of the design and explains the choice made in creating the software. The second part is the implementation which details the methods used and why they were used.

The final two chapters are the evaluation and conclusion, these chapters evaluate what has been done and any further enhancements that can be made.

2 Background Research

2.1 Overview

When developing a piece of software the designer must consider a number of different options, thus the research below covers the topics that must be considered. The first area a developer must consider before any programming is begun is; how to tackle the problem, i.e. what algorithms are going to be used and how are they going to function. This is why different file transfer protocols are going to be examined as this is going to be the backbone of the software. Once the method of how the program is going to be created is decided the next main topic is the choice of language. As the methods are now known the choice of language can vary on the methods chosen, therefore it is important to choose the right language as the wrong one may result in unnecessary work, thus the next topic of research is the choice of language on offer. In developing software consideration must be given to the user interface as it is one of the most important aspects of the system. Therefore the next area of research is the topic of human computer interaction (HCI). The final area to be considered is the topics of operating systems, as the software is suppose to be able to synchronise files over different systems it is important to look at the different systems available.

2.2 Different File Transfer Protocols

There are a number of different methods for the transferring of data between two devices, each have their own advantages and disadvantages. The protocols should be relatively simple to reduce overheads and thus hopefully speed us the synchronisation process. As there are lots of different solutions they should be considered as possible solutions to transfer data. It should be noted than these solutions may not be suitable for the creation of the software and a transfer protocol may need to be created. It is important that a suitable protocol is created as this will form the backbone of the software, if it is not correct if may mean that the software does not run correctly or it is inefficient. Therefore it is essential to research this area to find the best transfer protocol suitable for the software.

2.2.1 Sockets

A socket allows for peer-to-peer communication across a network. The socket has an advantage for the programmer as it ‘hides’ the details of the network, therefore the programmer does not need to worry about the TCP/IP layer all they need to know is that is there. There are typically three types of sockets available to the programmer; stream, datagram and raw. [22]

A datagram socket provides the programmer with an interface with the UDP (User Datagram Protocol). UDP handles the network transmission and does not detect duplicates, or maintain multipacket transmission. Using this method there is no built in retransmission for lost packet or duplicate packets it's up to the programmer to implement these. As little checking is performed then there is little overhead, which in turn speeds up the transfer of data over the network. [22]

Stream sockets are interface with the TCP transfer protocol, the advantage of this over the UDP is that the TCP controls the flow control, packet reassembly and connection maintenance, thus doing work for the programmer. The TCP checks for duplications, retransmission, and that they are received in the same order that they are sent. The downside using stream sockets over UDP is that there is greater overhead to be transmitted which slows the communication process down. [22]

Raw sockets interface with the IP network layer and ICMP (Internet Control Message Protocol). This socket does not provide any traditional peer-to-peer services, and tends to be used to test new protocols. [22]

2.2.2 FTP

File transfer protocol (FTP) is a type of socket, as it uses sockets to make the connection across the Internet. A client and server program is needed to communicate over the Internet; the communication is using TCP on port 21. The connection is made by the client sending one command and the waiting for the server to send one back. Using this command system the FTP gathers the information on the files that are going to be sent or received. FTP is the most popular method of transferring files over the Internet. [23]

2.3 Synchronisation techniques

When creating a synchronisation program two options are presented to the developer. The first is; does the computer automatically synchronise the data, or does the user do it manually. Both options are discussed in the following section.

2.3.1 Automatic

This option lets the user specify the files needed to be synchronised and every time they are changed or routinely (i.e. once a day/week) the files are synchronised by the computer. This option allows the user to specify the files needed just once, and the program will do the rest, this is a good option if the user does not want to spend time every day/week synchronising files. However this option does have a number of problems: the first if the file structure on the computer changes it is unlikely the synchronisation program will know this, and therefore will not be able to find the files. If this happens then the user will have to re-enter the files into the program again. Another problem which might occur with this method is that if the device being synchronised is not in contact with the host computer at the time of synchronisation. If this happens the errors may occur, however there is likely to be error checking to prevent this.

2.3.2 Manual

The second option is to let the user manually synchronise the files, i.e. the files are only every synchronised when the user specifies them to be. This option gives the user a greater freedom in what is synchronised than the automatic option as the user can change the file synchronised every time they run the program. If the file structure changes then the user can compensate for the occurrence, whereas in the automatic option it could not. The manual option offers a number of advantages over the automatic, however if has one main disadvantage. The main problem with this method is that if the user forgets to synchronise the files the discrepancies may occur on the different devices, which would hinder the user.

2.4 Choice of Computer Languages

The choice of which programming language to use is a crucial one, if the wrong language is chosen then the time taken to create the software could be increased dramatically. When choosing which language to use it’s a good idea to look at what libraries are available to use, for example they maybe a set of procedures already defined for communication between to computers therefore it is not necessary to create new one thus saving time. The main languages that I am considering to use are C++ or Java; there are other languages available, which will be looked at; however C++ and Java offer certain advantages, which are listed below. The choice of programming language is important as if the incorrect language is used then more work maybe done than is needed. The software is not limited to one language, for example Java might be more appropriate in one area, whereas C++ is more appropriate in another, and through the use of research it will become apparent which are the better options.

Both C++ and Java are portable languages, therefore in theory if it works on Windows then it should work on Linux with little or no change to the code. However both languages are portable for different reasons. C++ is portable as it is based on C, however there can be problems if different compilers are used to create the executables. For example from past experience using the g++ compiler on Linux to compile a OpenGl program, and then using the Microsoft Visual Studio to compile the same program, the same code is used but different executable are created the g++ compiler works as expected. However as different ‘rules’ are used in the Visual Studio compiler the program does not work as expected. Java is portable for different reason; the main one is that it has been developed by one company (Sun Microsystems) who have designed it with the same APIs to work for all the main operating systems; however less popular operating systems may find errors and such like in the programs. Also the same compiler (Javac) is used on both Windows and Linux the same code works. There are a number of different languages that do not support portability between operating systems; Visual Basic and C# are examples of these. Therefore the likelihood of using these to implement the software is unlikely; however the different languages available will be looked at in order to asses which will be the best option for developing the software. Other possible languages that could be used are: Visual Basic, C#, FORTRAN and Delphi to name a few.

It should be noted that the same language may not be used on both systems, for example C++ maybe used on the Windows machine, while Java maybe used on a Linux machine. The analysis of different languages below will decide which language will be most efficient in developing the software.

2.4.1 High/Low Level Languages

Computer languages tend to fall into two different categories ‘high level’ and ‘low level’, there are no set rules for the categories as some people may consider a language ‘high level’ while others consider it to be ‘low level’. It is generally considered that languages that are more natural, like SQL, are considered ‘high level’, and languages like BASIC is considered ‘low level’, languages like Java can be both depending on people’s views. It is also considered that ‘high level’ languages are generally quicker to develop for as the amount of code is reduced. However ‘low level’ programs tend to allows great manipulation of bytes and memory allocation and such like. Both ‘high level’ and ‘low level’ languages are worth considering and are discussed below.

2.4.2 C/C++

The first language to be considered is C and C++, the language has both advantages and disadvantages, which are looked at below. One of the main advantages of C/C++ is that it is a very popular language and there are lots of libraries that have already been created to do a number of functions, for example create a graphical interface, to creating sockets and communicating over networks. The C language also offers portability over different systems which can speed up the creation of software. As the language is popular there is an abundance of help on the subject either through books or the Internet. Another main fact of C/C++ is that its architecture neutral meaning that if an executable has been created for a Linux machine then the executable will run on all Linux machines regardless of the hardware (i.e. AMD, Intel, Seagate, etc).

However there are downsides to the language, for example some GUI's created for Windows will not work on Linux as the GUI uses features that are in Windows but not Linux.

2.4.3 Java

The Java language offers a lot of the advantages that C/C++ offers in that its: object oriented, architecture neutral, portable, dynamic to name just a few.

With the majority of programming languages a program is either compiled or interpreted so it can run on a computer; however java is different as it is both compiled and interpreted at the same time. The Java compiler takes the code and converts it into an intermediary language called Java bytecode (a platform independent code interpreted by the Java language). The compilation occurs just once, and the interpreter is used every time the program is executed (Figure 1). [3] This offers a number of advantages as it means only one program is needed that is capable of running over multiple platforms, where as some languages may need code to be heavily modified to run over different systems.

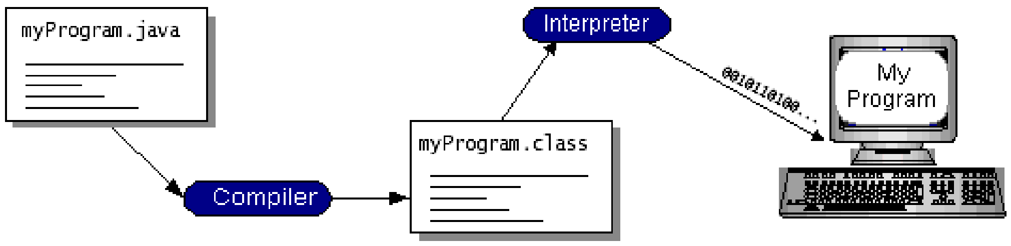


Figure 1: Diagram to show how the Java language is compiled and executed. [3]

Another advantage of the Java language is that all the libraries available are from the same source (Java Sun), and have been thoroughly tested and will run over the different platforms. It should be noted that other libraries are available through third parties but there is no guarantee that they are platform independent, and the library has to be on each system it is run on in order for the program to work. The Java language also has a library for the creation of graphical interfaces (Java swing); this library allows the creation of interfaces quickly and will little code compared to certain libraries in C++ for graphical interfaces.

2.4.4 Python

The Python language is object-oriented, interactive and interpreted. The language is often compared with other high-level languages such as Perl, Scheme, Java or TCL. Python has a very clear syntax which helps make the language very powerful; it is based upon classes, modules, exception, dynamic typing, and high level dynamic data types. Python is also capable of making system calls, and can also create windowing systems, such as X11, Motif and Mac to name just a few. The language can also be used as an extension language where modules written in Java or C/C++ can be written. [16]

Python is a ‘higher-level’ of language than Java and as such programs can generally be written quicker, typically 3-5 times faster. The speed up in development time can be attributed to Python’s built-in high-level data types and its dynamic typing. However as Python is the ‘higher-level’ of language programs tend to run slightly slower than if Java was used. [17]

Python and Perl originate from the same background, UNIX scripting, and therefore have many similar features. Even though both languages originated from the same background they have both developed different viewpoints. A couple of examples in favour of the Perl language is that it has a greater emphasise for supporting common application-oriented tasks, such as report generating features and file scanning. However a couple of examples in favour of the Python language are it has more emphasises for supporting programming methodologies like object-oriented programming, and to write readable code, to name just a few. Perl is the closest language to Python; however Perl tends to be the more dominate language in the original application domain. Saying this however Python’s applicability is beyond of Perl’s, and Python is the more flexible language overall. [17]

Most of the difference said about Python and Java can be applied to C/C++, this is due to C/C++ being ‘lower-level’ languages. As C/C++ is a ‘lower-level’ language the Java all the arguments made above are greater. For example where generating Python code it tends to be between three and five times quicker than Java, and for C/C++ it is generally five to ten times quicker. Anecdotal evidence has suggested that the amount of work one Python programmer can do in two months, two C++ programmers cannot complete in a year. [17]

2.4.5 Perl

The Perl language is often compared to Python and vice versa, the language is supported on the Linux, Microsoft Windows and Macintosh operating systems. The Perl language takes the best features from other languages such as C, C++ and Basic, Perl can also work with third party database like Oracle. Perl is also compatible with mark-up languages such as HTML and XML; this makes Perl extremely useful when designing web-pages especially with its ability to work with databases. The language is also flexible in that it can cater for procedural and object-oriented programming. Also much like the Python language Perl can interface with other languages such as C/C++. [18]

Perl is an extremely flexible language as it can interact with other languages (as mentioned above) as well as being flexible over the Internet, this results in Perl being a powerful language that is worth considering for this project. Perl has many of the advantages of Python (see Python section).

2.4.6 Conclusion

There are a large number of different languages on offer and each one offers certain advantages and disadvantages. The language chosen to create the software is Java. The main reason for this choice was the portability of the language and the wide range of libraries from the creation of GUI's to the library for the creation of sockets and communication over networks. The Java language offers all the functions needed to create the software. Python was also closely considered as the ease of use of sockets, being already built in, and also the portability of the language. The main downside of Python was learning a new language from scratch, especially since Java was already known, the time taken to learn a new language and syntax it was decided to opt for Java since same functions are on offer in both languages.

2.5 Human Computer Interface (HCI)

The user interface is one of the most important aspects in any design for software as this is what the user 'sees'. Most of the time the user does not concern themselves with how the program does something, they do not care about the algorithms used and how efficient they are. So long as the program works and is easy to use that's what most users want from a piece of software, therefore it is an important part of the software.

When designing a user interface a large number of different factors need to be taken into account, and examples of these can range from the type of interface (command or graphical), the colours used and how the user manipulates objects in the interface. An entire project could be done on the subject of HCI as this subject is that vast, therefore the information presented below is what I consider to be the most important to be presented. It will discus the advantages and disadvantages of both command and graphical interfaces as well as other important aspects in the creation of the interface.

There are two main types of user interface to be considered in creating this software, they are a command based interface, like MS-DOS, or a graphical user interface, like Windows, both the advantages and disadvantages will be discussed below. There are other types of interfaces available to use, like speech recognition, but with the requirements of the project, i.e. can be used over multiple devices, it is best to concentrate on the two main types of interface which are available on most systems.

2.5.1 Theories

There are many different theories about different HCI techniques; each theory adds something new to the field, either in graphical interface design to how colours should be used in an interface. Some of the theories are worth considering, but not all the theories are considered 'correct' and there is some debate about certain theories. That's why it is more important to look at the theories but pay more attention to the guidelines that are available; the reason for this is that the guidelines are generally more agreed upon.

2.5.2 Command Based User Interface

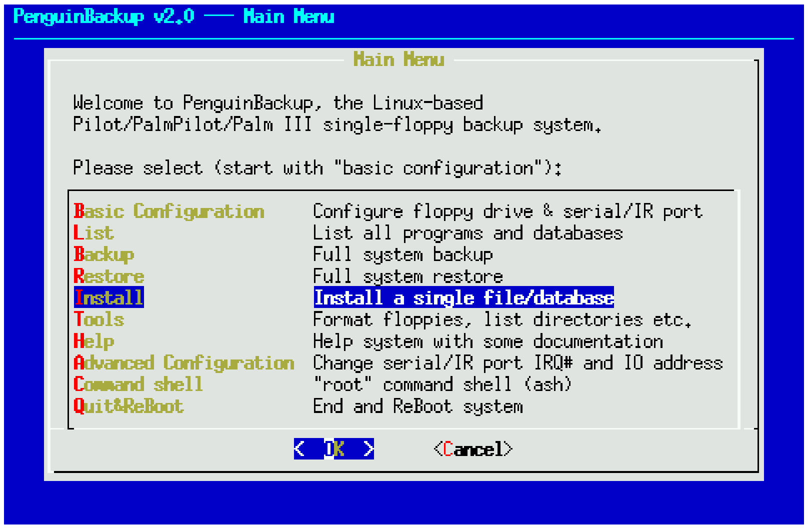
Command based user interfaces have in numbers in recent years due to the release of such operating systems like Microsoft Windows, which are designed to run GUI. Even though the market has more GUI interface it is still worth considering a command based interface. A command based user interface has both advantages and disadvantages and it is important to look at these in deciding which type of interface to make.

When designing the user interface there are a number of different options to be considered in how the commands can be entered. The first is just a single command, e.g. 'run' the command which will carry out just a single task. If only a small system is in use then this approach maybe suitable as there is only a handful of command needs to be remember and is simple to learn. However for larger systems such as UNIX and some text based editors where the number of commands is large these type of commands become difficult to remember and not as easy for the user to learn. The next step up from using just a single command word is to use a command word and an argument, e.g. 'print stuff'. This type technique can help larger systems as the number of commands needed to remember can be reduced and it can speed up the execution of the programs. However it does have some disadvantages, the system becomes harder to create because checks have to be made to make sure every argument is compatible with the argument. Another option is a command, option, and then an argument, for example 'print 3 stuff'. Like the command-argument option this choice creates more option for the final user and can speed up the execution of larger systems, however for smaller systems this choice could be difficult to implement efficiently as the number of commands is small. The final choice for command based input is the hierarchical command structure; this creates a tree structure, example shown below:

Action	Object	Destination
CREATE	File	File
DISPLAY	Process	Local printer
REMOVE	Directory	Screen
COPY		Laser printer
MOVE		

This type of hierarchical structure can offer a meaningful structure to a large number of commands. However for large systems the number of commands can be large and it could be difficult for each user to remember every command, therefore for smaller systems it may not be a suitable option. [14]

Command based interfaces offer a number of advantages such as speedy execution of multiple tasks at the same time using arguments and option, this is an advantage over graphical interfaces to a certain extent. However a command based interface also offers a number of disadvantages; the main disadvantage is initially remembering the different commands on offer and also in remembering the options and arguments that are available to the user. Another disadvantage is than novice users maybe put off with a command based with the command and syntax needed to remember, the novice user will likely find the systems harder to use initial than the intermediate or expert user.



```
PenguinBackup v2.0 ── Main Menu

                         Main Menu

  Welcome to PenguinBackup, the Linux-based
  Pilot/PalmPilot/Palm III single-floppy backup system.

  Please select (start with "basic configuration"):

  Basic Configuration     Configure floppy drive & serial/IR port
  List                    List all programs and databases
  Backup                  Full system backup
  Restore                 Full system restore
  Install                 Install a single file/database
  Tools                   Format floppies, list directories etc.
  Help                    Help system with some documentation
  Advanced Configuration  Change serial/IR port IRQ# and IO address
  Command shell           "root" command shell (ash)
  Quit&ReBoot             End and ReBoot system

                  <  OK  >        <Cancel>
```

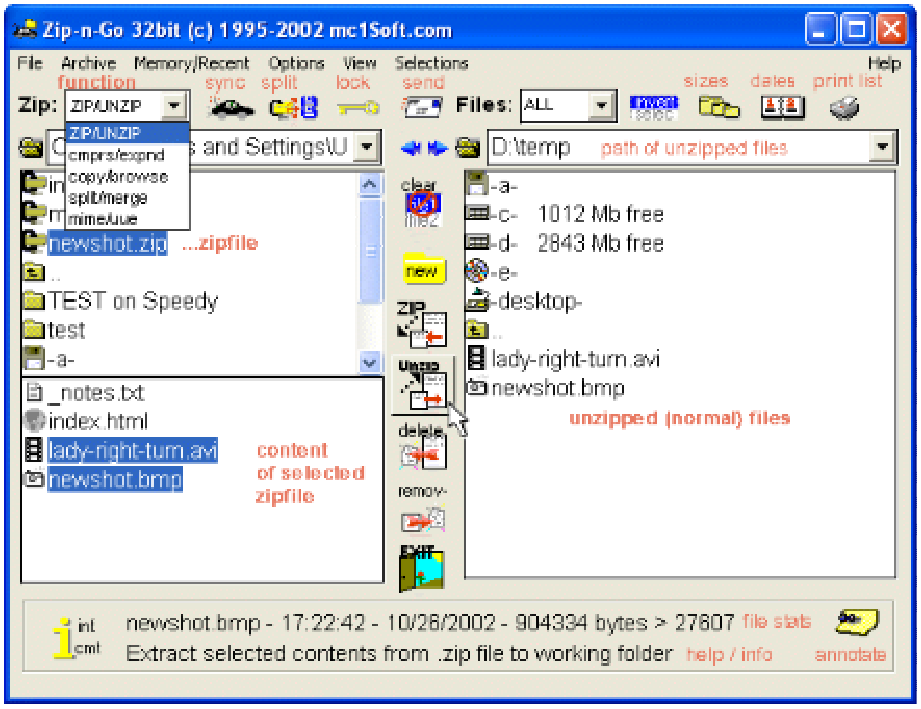
Figure 2: An example of a text based used interface (example shown is PanguinBackup v2.0). [5]

2.5.3 Graphical User Interface (GUI)

"The primary goal for menu designers is to create a sensible, comprehensible, memorable, and convenient semantic organisation relevant to the user's tasks." [14]

A graphical user interface has a lot of different aspects that need to be considered before it is created. The main area for consideration is the ease of use for the user. However other areas need to be considered such as the layout and the HCI part. Therefore it is important to research and consider other different user interfaces, in order to make the software user friendly. [6]

A GUI interface can have a lot of different features; menus, buttons, dialog boxes, radio buttons, check buttons to name just a few. All these features need to be implemented correctly in order to create a suitable interface. For example if the buttons are too small then the user may have difficulty clicking the button, however if the buttons are too big then they may take too much space on the screen and restrict the user in their tasks.



C:\Documents and Settings\U

newshot.zip

TEST on Speedy

test

-a-

_notes.txt

index.html

lady-right-turn.avi

newshot.bmp

D:\temp

-a-

-c- 1012 Mb free

-d- 2843 Mb free

-e-

-desktop-

lady-right-turn.avi

newshot.bmp

newshot.bmp - 17:22:42 - 10/26/2002 - 904334 bytes > 27607

Extract selected contents from .zip file to working folder

Figure 3: An example of a graphical user interface (example shown is Zip-n-Go). [7]

2.5.4 Colour Usage

The colour usage is important in both the design of a command based and graphical interface. Even though there are no set rules for the use of colour there are some guidelines which should be considered [14]:

- Use colour conservatively.
- Limit the number of colours.
- Recognise the power of colour as a coding technique.
- Ensure that colour coding supports the task.
- Have colour coding appear with minimal user effort.
- Place colour coding under user control.
- Design for monochrome first.
- Use colour to help formatting.
- Be consistent in colour coding.
- Be alert to common expectations about colour coding.
- Be alert to problems with colour pairings.
- Use colour changes to indicate status changes.
- Use colour in graphic displays for greater information density.
- Beware the loss of resolution with colour displays."

2.5.5 Use of Error Messages

As with the use of colour in the creation of an interface the use of error messages apply to both the command based and graphical interfaces. An error message needs to be useful and informative so the user knows what is wrong and what needs to be done to correct the error. For example if the error message 'error' appeared then the user will not know what is wrong however if the message 'error line 145 syntax error unmatched left parenthesis' then the user knows exactly where the problem is and how to correct it. As with the use of colour there are certain guidelines available which are below [14]:

- Have a positive tone.
- Be specific and address the problem in the user's terms.
- Place the user in control of the situation.
- Have a neat and consistent, and comprehensible format.

2.6 Different Desktop Operating Systems

The operating system that the software is going to be designed on is an important decision. Depending on the operating system chosen will also depend on the user interface, Linux is more text base where as Windows is more graphical. Also depending on which programming language is chosen might depend on the operating system. For example if C++ or Java is chosen then the code should be portable between the operating systems. However if Microsoft Visual Studio is used some libraries (especially the .dll's) will not be able to be transferred to Linux. Therefore it is important to consider which operating system to use and its advantages and disadvantages for the software. It has to be noted that Windows and Linux are not the only operating systems available; some of the other operating systems will be looked at briefly below.

2.6.1 Microsoft Window 95/98/ME/XP

Microsoft Windows is the most popular operating system in use (due it being shipped with most PC). Taking this into account it is important to look into the development of software on a Windows based system.

2.6.2 Unix/Linux

Linux is the other possible operating system to develop the software on, like Windows it has its advantages and disadvantages. Both these will be looked at in choosing the right operating system for developing the software. [8]

2.6.3 Other Operating Systems

It has to be noted that there are more operating systems available than just Microsoft Windows and Linux such as; DOS, Solaris, etc. The number of operating systems available is huge and the possible combination of synchronisation is equally as huge. Therefore it is important to look at other operating systems for further enhancements in the future. However the only operating systems available for use are Microsoft Windows and Linux, therefore they are the likely choice, but it is important to realise that there are other option available and the advantages and disadvantages of them. [12]

2.7 Different Operating Systems for PDA Devices

There are a number of different operating systems that can be used on a PDA device; the main operating systems are as follows, Palm OS, Windows CE, and EPOC operating systems. Each of these both have their positive and negative attributes. It has to be noted that these are not the only PDA operating systems, there are others available however the three listed above are the main ones in used in PDA's. As the software that is going to be developed is going to go over more than one operating system it is important to consider the PDA operating systems as a viable alternative to another desktop operating system. The PDA market is increasing, as newer PDA's are being released with greater memory and faster processor speeds more people are being attracted to buy them. With this in mind file synchronisation software that is capable of synchronising with a PDA may be more beneficial than software that can synchronise between different operating systems. Therefore it is important in considering implementing a PDA operating system for my project, below are the three main PDA operating systems being considered. [1]

2.7.1 Palm OS

Of all the different PDA operating systems available, Palm OS is the most popular in the number of PDA's it is installed on. Like all operating systems it has both its advantages and disadvantages. This is one of the main operating systems being considered to be implemented into the software, as it is one of the most widespread. Through the use of the Palm website there is a large amount of information on implementing software on the Palm OS, and software in a windows environment. This will help in the implementations as it is well documented and should help in the creation of the software. There are also emulators for the various Palm devices this will help in the testing of the software in development. [2]

2.7.2 Pocket PC (Windows CE)

Windows CE is another main contender to be considered in the implementation of the software. It has a number of advantages worth considering. The main reason is that it is based on Microsoft Windows and thus is relatively easy to program in as it's based on many of the Windows commands. This would probably speed up the creation of the software. Windows CE makes use of MSDN (Microsoft Development Network), which means that there are documentation and development libraries for creating software in Windows CE. The downside of using the MSDN library is that Windows 2000/XP Professional is needed and the latest Microsoft Visual Studio in order to develop software for Windows CE.

2.7.3 EPOC OS

The EPOC operating system is the one used for Psion device. Recently more sales in the PDA market have had either Palm OS or Windows CE on them; however the EPOC operating system is still worth considering.

2.7.4 Linux

The Linux operating system for PDAs is one of the newer operating systems that have become available on the market. It is being considered as the third alternative to the Palm OS and Windows CE, now that EPOC is in decline. It is worth considering the Linux operating system as it is based on Linux and that it is well documented which should help in the creation of the software. [13]

3 Analysis of Existing Solutions

3.1 Overview

There are many different existing solutions available to the general public, either shareware/freeware or commercial packages. Some of these packages have been looked at (see references for all), each of these packages offers something that the others do not, and they all have their advantages and disadvantages. These already existing solutions are what the software being designed will be compared to. It is important that the software designed has features that are not available to these other pieces of software, as well as some if not all the feature they do offer.

It is important to analyse already existing solutions as it shows what is already available and what has been done so far in the market. This analysis will show how problems have been tackled by other developers, and therefore can be looked at and if possible better solution's created.

The following section compares a number of different software synchronisation tools available on the market. It is important to understand that the software below are not only products available to purchase, there are many more and it is not possible to examine every one.

In analysing the different software a 'marking scheme' needs to be created in order to judge each of the different software fairly. The scheme will look at the same qualities in each program, and will be compared to the other software available. It should be noted that the final software created will also be subject to this 'mark scheme' and compared to the other programs tested. The scheme is as follows:

- How well presented is the software?
- Is the interface sufficient?
- Is the software interface easy to use, or is there a learning curve?
- How portable is the program, is it capable of synchronisation over different systems?
- How efficient is the synchronisation?

3.2 Rsync

Rsync is a Linux designed program, however it is possible for some versions to run on Windows 9x\NT systems. Rsync is a replacement for the rcp function that has many more features. Rsync uses the "Rsync algorithm", this algorithm gives the user a stable and fast way of synchronising multiple files. This is achieved by sending only the difference in the files across the communication channel, after checking that both files are present. [9]

The following section lists some of the features of Rsync [9]:

- The algorithm used can update both directory trees and file systems.
- There are many options available which can preserve file ownership, permissions, symbolic and hard links to name just a few.
- Any user can install the software as no special privileges are needed i.e. super user.
- If multiple files are being transferred then internal piping is used to reduce latency.
- More than one method can be used to communicate, i.e. sockets, ssh, and rsh.

The Rsync code is not fully compatible as if a Windows machine is being used then it can only send and not receive, however if Rsync is run under Linux it is capable of sending and receiving.

The Rsync program is a fast and efficient way to synchronise files between computers, however there are some disadvantages, the main disadvantage is mentioned above in that Rsync is not fully compatible with machines running Windows. However another disadvantage, especially for novice users, is the interface. The interface is a command based interface and with the large number of commands and options available to run Rsync it can difficult to learn and remember the different options.

3.3 Unison

Unison is a file synchronisation tool for UNIX and Widows. The software allows two identical groups of directories and files to be stored on different computers, or different drives on the same computer. These files and directories can be altered separately, and then synchronised by updating the files structure in each of the groups. [10]

There are several areas where Unison differs from these programs and the following section discusses these [10]: Unison is capable of running on multiple platforms (Windows/UNIX), and is also capable of synchronisation across the different platforms, i.e. Windows to UNIX and vice versa. Also unlike other UNIX synchronisation software it is a user-level program, i.e. there is no need to own the kernel or have super user privileges. Unison can synchronise over a variety of different communication methods such as the internet, networks, sockets, and ssh to name just a few.

Unison builds on the features the Rsync has as well as extending them which makes Unison a powerful tool in synchronisation. However there are a number of disadvantages to using Unison, the main disadvantage is that it is a command based program, this may not be a bad thing for some programs. However with Unison there are a lot of different commands and to remember all the commands can be difficult, especially for novice users.

3.4 Novell iFolder

The latest version of Novell IFolder (2.1) "seamlessly synchronizes the contents of all of your iFolders, no matter which of your computers you may be using" [20]. Novell iFolder is part of the Novell Nterprise networking and storage solutions that is available. The software is supported on the following server platforms: NetWare 6 Support Pack 2, NetWare 5.1 Support Pack 5, Windows 2000 Service Pack 3, and Redhat Linux 8; and the following web servers Apache 1.3.27 and above on NetWare, Apache 2 on Linux (included with the iFolder 2.1 installation program), and Internet Information Server (IIS) 5 on Windows 2000. The iFolder program is designed for large networks of computers with a dedicated server rather than a home with only one computer and PDA for example. Novell iFolder works by when a file is change on a computer a copy of this file is copied to the server and saved under a dedicate iFolder directory, from here the file is distribute over the network and updates the file where necessary.

Novell iFolder has the following benefits for the end user; some of these benefits are as follows: with the automatic backup facility it can help against disasters and help in the recovery. Using iFolder offers enhanced file management and can reduce administrative costs. Also through the use of the internet the ability to synchronise files throughout the world, this is helped with reliable safe guards that are in place (using 128-bit encryption). [20]

According to [20] Novell I Folder has the following features:

- Transparent and intelligent file updates.
- Automatic back up of local files.
- Increased bandwidth efficiency.
- Convenient file access from anywhere through a standard Web browser.
- Secure file encryption and protection via pass-phrase identification.
- Support for Windows XP/2000/NT/Me/98/95 client platforms.
- Support for Microsoft Active Directory.
- Scales to support thousands/millions of users.
- Browser-based server administration from any location.

The Novell iFolder software has a lot of features that many other synchronisation software does not offer, like the ability to synchronise over the internet. The software offers increased security (128-bit encryption) where most do not offer such a high level of encryption. However iFolder does have some disadvantages and the main downside of the software is that a server is needed to run the software and only large companies are likely to have a dedicated server large enough to be able to run the software, and home users are unlikely to benefit from the software.

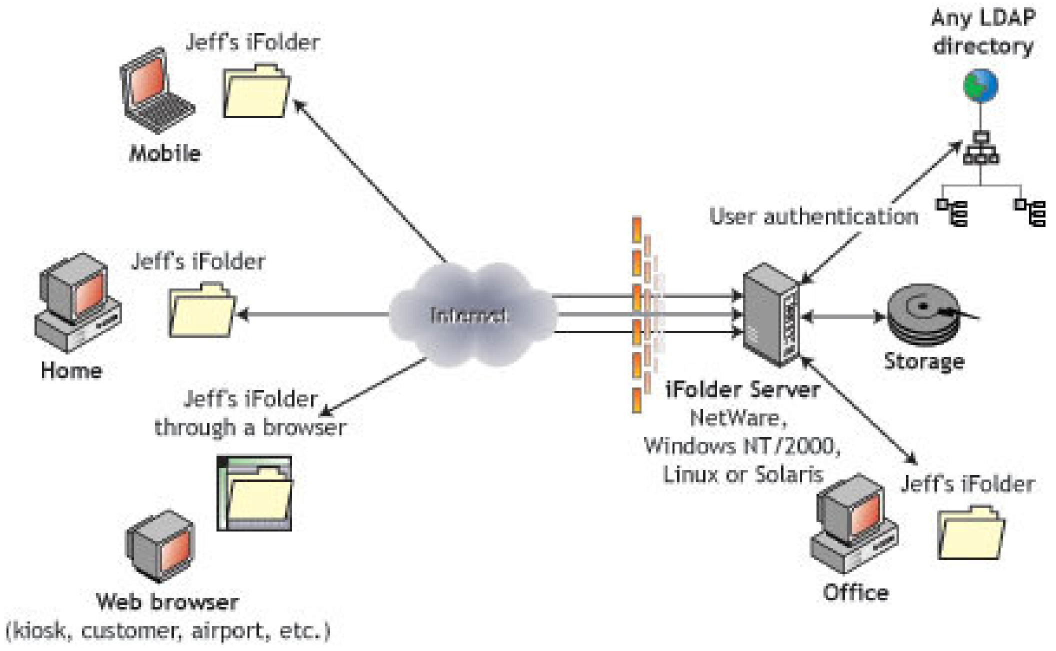


Figure 4: iFolder provides universal file access, anywhere and anytime. [20]

3.5 Xfiles

The Xfiles synchronisation software is a Linux only tool and cannot be synchronised with other operating systems (i.e. Windows). Xfiles is designed using the Java language and uses the GUI libraries (Figure 5).

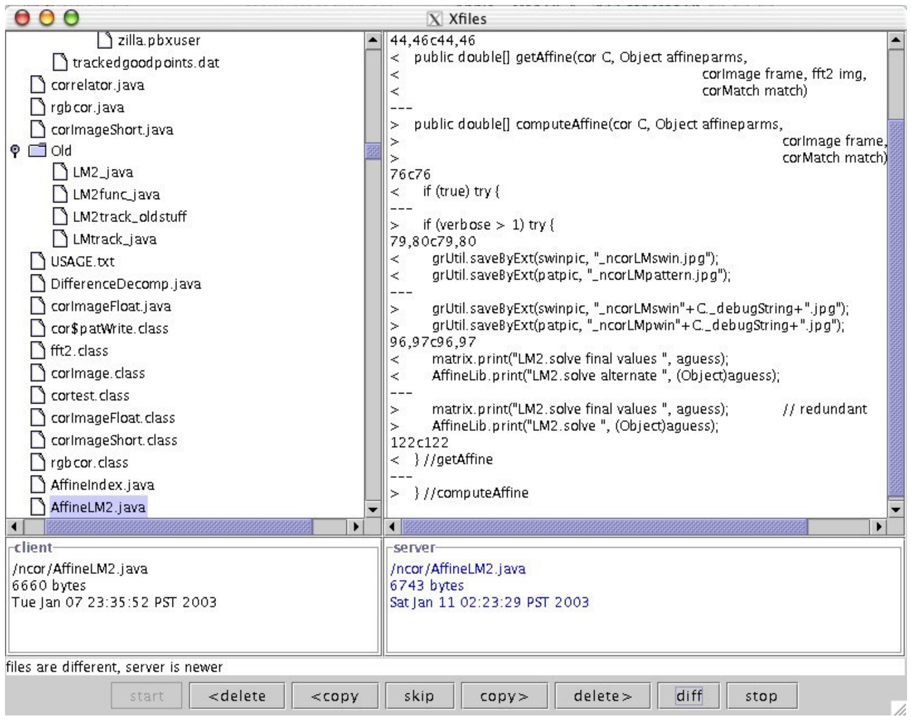
The Xfiles software is capable of checking and merging one file tree with another file tree over the network. There is no need to keep track of the changes being made on the separate machines on the networks as it supports free form work.

The software is presented in two programs, a client which is GUI based, and a server, command-line based. The programs traverses the file trees and makes note of files that are not present on either the client or server side, or any files that are different. A number of checks are performed when two files are different, the first two are checking if the file sizes are different and the modification date, and the last is a comparison using the UNIX diff function. [19]

The Xfiles program is not a simple as ‘pick-up and run’ program as it first appears. Even though a GUI interface is used it cannot be used to select the directories that are going to be synchronised, this is done through the use of a command line argument when the program is first run. The use of the command line argument means that if the directory needs to be changed then the program needs to be closed and restarted with a different command line argument. Through the use of the command line a number of synchronisation options are available (Table 1), these options give the program more flexibility in its operations, an ability that many other synchronisation programs do not have.

Table 1: Shows option available for Xfiles 1.4

-fast	only compare sizes and dates
-clientonly	focus on client, ignore missing server files
-serveronly	focus on server, ignore missing client files
-exclude str1 str2...	exclude files that contain these substrings



```
44,46c44,46
<   public double[] getAffine(cor C, Object affineparms,
<                                        corImage frame, fft2 img,
<                                        corMatch match)
---
>   public double[] computeAffine(cor C, Object affineparms,
>                                                corImage frame,
>                                                corMatch match)
76c76
<     if (true) try {
---
>     if (verbose > 1) try {
79,80c79,80
<       grUtil.saveByExt(swinpic, "_ncorLMswin.jpg");
<       grUtil.saveByExt(patpic, "_ncorLMpattern.jpg");
---
>       grUtil.saveByExt(swinpic, "_ncorLMswin"+C._debugString+".jpg");
>       grUtil.saveByExt(patpic, "_ncorLMpwin"+C._debugString+".jpg");
96,97c96,97
<       matrix.print("LM2.solve final values ", aguess);
<       AffineLib.print("LM2.solve alternate ", (Object)aguess);
---
>       matrix.print("LM2.solve final values ", aguess);          // redundant
>       AffineLib.print("LM2.solve ", (Object)aguess);
122c122
<   } //getAffine
---
>   } //computeAffine
```

```
/ncor/AffineLM2.java
6660 bytes
Tue Jan 07 23:35:52 PST 2003
```

```
/ncor/AffineLM2.java
6743 bytes
Sat Jan 11 02:23:29 PST 2003
```

```
files are different, server is newer
```

Figure 5: Screenshot for Xfiles 1.4 synchronisation program. [19]

In conclusion the Xfiles software is an efficient synchronisation tool, with many different synchronisation features. However it does have some big disadvantages, the first is that the novice user may find it difficult to use, with the mix of command line and graphical interface; if the command line option could be done by the GUI it may make the program easier to use, especially for novices. The graphical interface is also a disadvantage, if the button has been better labelled (i.e. clear descriptions of what they do) then the program would be understood. The last disadvantage is that the program is not compatible with other operating system other than Linux, this limit the capabilities of the programs compare to Rsync for example.

3.6 Backer (version 6.1)

Backer 6.1 is a synchronisation program designed for Windows 9x/Me/NT/XP; it is not compatible with Linux or PDA operating systems; i.e. can only be synchronised from a Windows machine to another Windows machine. The software has a graphical interface (Figure 6) the design of the interface is self explanatory, unlike the Xfiles software. The synchronisation works by the user creating a profile of the files needed to be synchronised, once this profile is created the software checks to make sure the files are synchronised. The software is extremely simple to use, less complex than a lot of the other synchronisation software that has been looked at in this chapter.

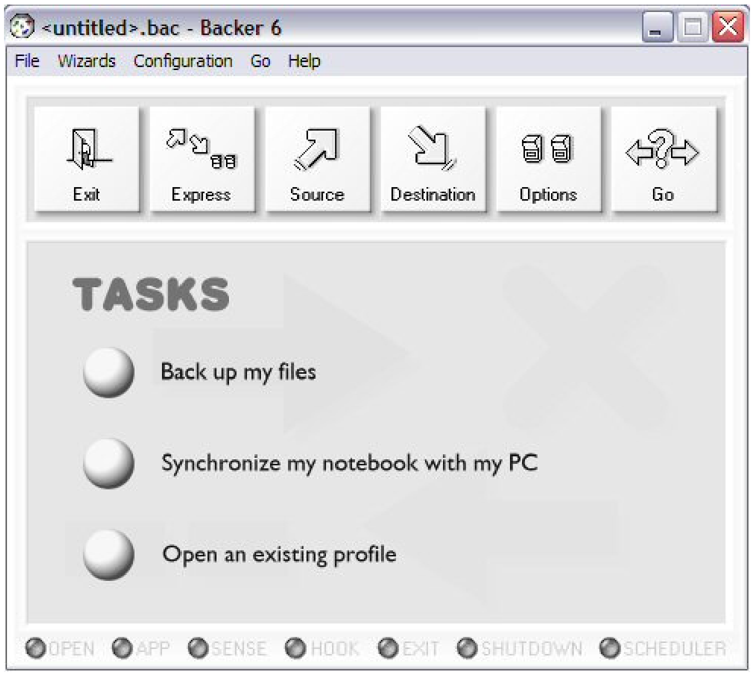


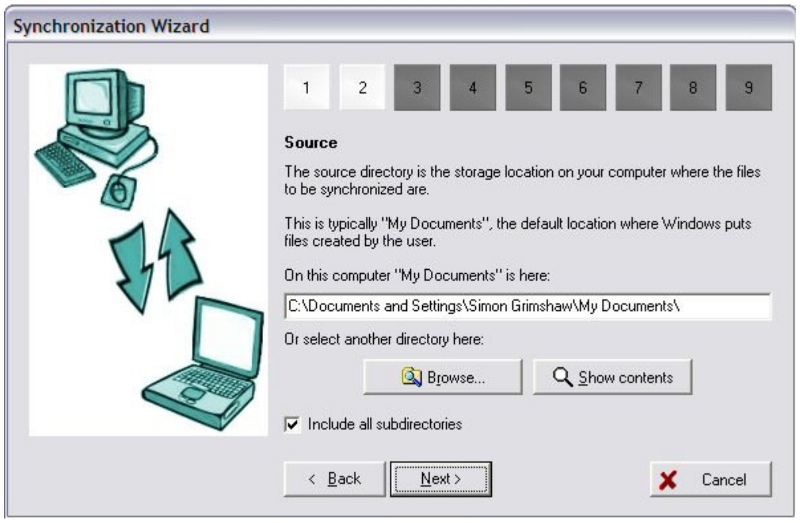
Figure 6: Screenshot of Backer version 6.1, synchronisation software. [21]

For the user to create a profile two methods can be used; the first is to manually specify the source and destination directories, and the second is to use a wizard (Figure 7), it should be noted that a user can have more than one profile. In order to use the wizard there are nine steps in order to create a profile; they are as follows:

1. An introduction to the software and explanation about the wizard.
2. The user selects the source directory i.e. the 'My Documents' folder.

3. This step allows the user to select files within the directory, i.e. all of them or for example all the Word and Excel documents.
4. A destination directory is chosen on the next step, this can either be a destination on the host computer or it can be another computer on the network or a computer via direct cable.
5. The next step gives the user synchronisation options, these are:
   'Delete the files in the source or destination that have no more opposite'
   'Expressly report double modifications'
   'Delete empty directories'

The last four steps simple explain to the user where the profile is saved to, and explains the feature that the profile contains.



Source

The source directory is the storage location on your computer where the files to be synchronized are.

This is typically "My Documents", the default location where Windows puts files created by the user.

On this computer "My Documents" is here:

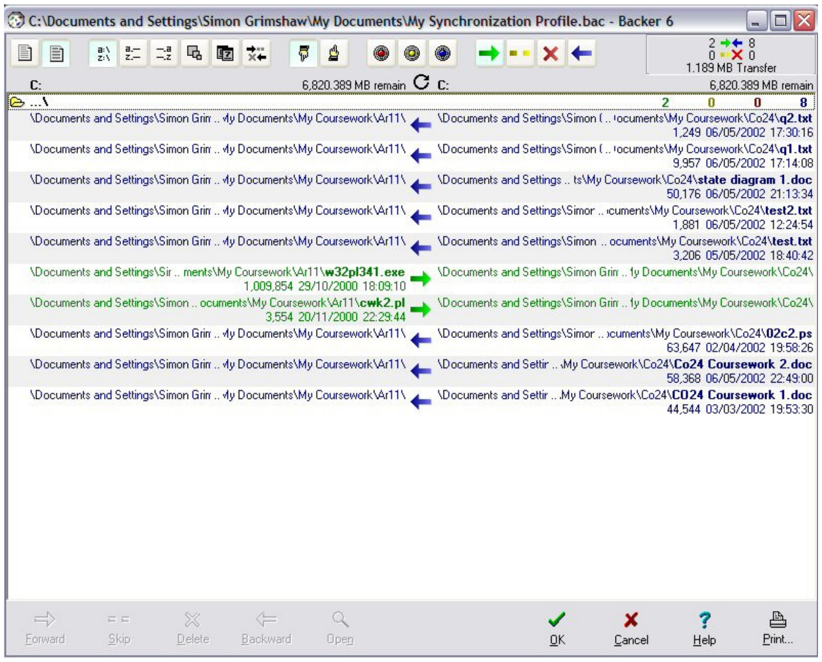
C:\Documents and Settings\Simon Grimshaw\My Documents\

Or select another directory here:

☑ Include all subdirectories

Figure 7: An example of Backer 6.1 profile wizard. [21]

Once a profile has been created that contains both a source and destination directory the profile can be executed and synchronisation can occur. The synchronisation process (Figure 8) shows the user what files are not synchronised. It shows the files that need to be transferred from the source to the destination and vice versa. The software also has the ability for the user to cancel any transfer that the software suggests.



```
C:\Documents and Settings\Simon Grimshaw\My Documents\My Synchronization Profile.bac - Backer 6

                                                        2  8
                                                        0  0
                                                        1.189 MB Transfer
C:                                 6,820.389 MB remain   C:                                 6,820.389 MB remain
...\                                                                                          2     0     0     8
\Documents and Settings\Simon Grin .. Ay Documents\My Coursework\Ar11\  ←  \Documents and Settings\Simon ( .. ıocuments\My Coursework\Co24\q2.txt
                                                                                                  1,249  06/05/2002  17:30:16
\Documents and Settings\Simon Grin .. Ay Documents\My Coursework\Ar11\  ←  \Documents and Settings\Simon ( .. ıocuments\My Coursework\Co24\q1.txt
                                                                                                  9,957  06/05/2002  17:14:08
\Documents and Settings\Simon Grin .. Ay Documents\My Coursework\Ar11\  ←  \Documents and Settings .. ts\My Coursework\Co24\state diagram 1.doc
                                                                                                 50,176  06/05/2002  21:13:34
\Documents and Settings\Simon Grin .. Ay Documents\My Coursework\Ar11\  ←  \Documents and Settings\Simor .. ıcuments\My Coursework\Co24\test2.txt
                                                                                                  1,881  06/05/2002  12:24:54
\Documents and Settings\Simon Grin .. Ay Documents\My Coursework\Ar11\  ←  \Documents and Settings\Simon .. ocuments\My Coursework\Co24\test.txt
                                                                                                  3,206  05/05/2002  18:40:42
\Documents and Settings\Sir .. ments\My Coursework\Ar11\w32pl341.exe  →  \Documents and Settings\Simon Grin .. 1y Documents\My Coursework\Co24\
                          1,009,854  29/10/2000  18:09:10
\Documents and Settings\Simon .. ocuments\My Coursework\Ar11\cwk2.pl  →  \Documents and Settings\Simon Grin .. 1y Documents\My Coursework\Co24\
                              3,554  20/11/2000  22:29:44
\Documents and Settings\Simon Grin .. Ay Documents\My Coursework\Ar11\  ←  \Documents and Settings\Simor .. ıcuments\My Coursework\Co24\02c2.ps
                                                                                                 63,647  02/04/2002  19:58:26
\Documents and Settings\Simon Grin .. Ay Documents\My Coursework\Ar11\  ←  \Documents and Settir .. My Coursework\Co24\Co24 Coursework 2.doc
                                                                                                 58,368  06/05/2002  22:49:00
\Documents and Settings\Simon Grin .. Ay Documents\My Coursework\Ar11\  ←  \Documents and Settir .. My Coursework\Co24\CO24 Coursework 1.doc
                                                                                                 44,544  03/03/2002  19:53:30
```

Figure 8: Screenshot Backer 6.1 shows the synchronisation progress. [21]

This software has a lot of features that other products in this chapter do not. For example an easy to follows user interface, ideal for novices, and more sophisticated options for the expert user. The software also has a wizard that takes the user through synchronising their files; programs such as Rsync and Unison have no such abilities. However the software does have some disadvantages, the main one is that the software is only available on Windows and is not capable of synchronising with Linux and other operating systems. Another disadvantage of this software is that not as many synchronisation options available as compared to Rsync for example.

3.7 Conclusion

In this chapter a number of different file synchronisation programs have been examined, each offers there own advantages and disadvantages, and each offers different ways of doing the same task. What has been realised from the analysis of these programs is that the user interface is an essential part in the program, the interface needs to be easy to use, self-explanatory and to be able to synchronise files with the minimum of effort from the user. Another factor is that the software being developed must be portable, and the software must be able to successfully send and receive files no matter what operating system is in use. Something that Rsync and Backer can not do. It is also important that the program is flexible and is capable of a multiple of different types are checks; these features should be present but should not overwhelm the novice user.

4 Design and Implementation

4.1 Overview

The following chapter is split into two parts, the first part deals with the technique that is going to be used in the implementation of the software and is going to discuss when these techniques are chosen above the others which are available. The second part of this chapter discusses how the techniques are implemented and why the methods are used.

4.2 Design

4.2.1 Choice of language

To implement the software the Java programming language will be used because Java offers a number of advantages over other languages for this particular program. The main advantage that is offered is the portability of the language, as discussed in the research chapter once the Java bytecode is generated the program is capable of running on any machine with a Java interpreter. This not only speeds up the creation of the software, as only one program needs to be written, but it also allows the designer to created the 'look and feel' of the program without needing to worry about how it is going to look on a different operating systems. Another reason why Java has been chosen is that the libraries enclosed in the JDK allow the creation of graphical interfaces (thought Java swing library) and sockets (through the net library); this will help the creation of software as separate libraries will not need to be created.

4.2.2 Communication Methods

There are a number of different ways in which to communicate over a network, as mentioned in the background research chapter. The final choice of communication was to use a socket; this was chosen for a number of reasons. The main reason is that the Java language is that it offers a class for creating a socket, the class is relatively easy to implement (as shown below). The Java socket class offers a great deal of flexibility in the creation of the socket and what can be sent and received (through the use of input and output streams).

4.2.3 Choice of Interface

Through the analysis of different synchronisation software in the previous chapter it was decided to opt for a graphical interface. There are two main reason for this choice, the first and most important one is that in examining the other software it was concluded that the graphical interface was easier to use for the more novice user, and this software will appeal to a wide selection of people, and some of which will not have any experience of a command based interface. It is also worth noting that the software is suppose to run over multiple systems, and with the introduction of a GUI with Windows a lot of users will never have used a command based systems, and therefore the program would not appeal to the general user. The second reason for this choice of interface is that Java offers a very powerful yet short and easy way on implementing a GUI (though the use of Java swing).

4.2.4 Choice of Operating System

As the Java language is being used, and in that it is portable over multiple operating systems it has been decided to concentrate on these operating systems to synchronise with. As Java is compatible with Windows, Linux, Solaris and MacOS it is important to create a stable program that is capable of successfully synchronising files. The choice to implement software for PDA's was available; but it was decided better time would be spent in having the software being able to synchronise over the more mainstream operating systems (such as Windows and Linux).

4.2.5 User Interface Design

The creation of the user interface is one of the most important aspects of the software, as it is what the user 'sees' and is what the judgement of the software will mainly be on for the user. As mentioned in the background research chapter there are two types of interface to consider command based and graphical interfaces. The type of interface that is going to be developed is a graphical interface; this has been chosen for a number of reasons. The main reason that a graphical interface has been chosen is that most mainstream operating systems nowadays (i.e. Windows, MacOS) are graphical. This means that all the novice users the software is aimed at are use to graphical interfaces, this could result in some difficult for the users if command based interface were to be used. Another factor in choosing a graphical interface is the use of the Java swing library this allows quick creation of GUI products, at not much extra time than command based. The command based interface is still popular to a certain extent with Linux users however this software is meant to appeal to a number of different operating systems, and in be a GUI it will appeal to more people.

In creating a GUI a number of important decisions need to be made, these are; the choice of components, the choice of colour, and the layout of the components. Each one is important in order to create an easy to use and efficient user interface. The first aspect to consider is the choice of colour, the Java Swing library allows the designer the ability to specify the colour choice, however every user has different tastes so it is best to let the Java program use the default colour of the operating system it is running on. This will allow the user to have 'free reign' over the colour choices. The next to areas to be considered are the choice of components and the layout, both areas overlap each other so both are going to be discussed together. The layout is an important part of the program as a lot of different aspects need to be considered, there are a lot of guidelines available for help people design GUI's and these have been looked at in a previous chapter. The design that has been decided is a 'standard' Windows layout, i.e. a menu bar along the top and a work area in the centre, with a tool bar on the bottom. The reason this was chosen is that the software is meant to appeal to a wide audience, and therefore if it has a Windows 'look' more people will be able to learn how to use it quicker as they are already familiar with the layout.

4.3 Implementation

In creating synchronisation software, or most software for that matter, a number of different methods need to be implemented. For this particular piece of software two separate programs are to be created a sever and a client. The server will have a number of methods; a method for creating the graphical interface, creating the socket, receiving files, and a checking method for the files (i.e. stop duplication; always keep the most recent version etc.). The client programs will have some of the same methods as the server (with slight alterations) these are creating the socket, checking the files where the client will differ is that it will have a sending method instead of the receiving method. The client will also contain a method for creating the GUI but it will differ from the server as the interfaces differ.

4.3.1 Creating the Socket

The Java language has a library available that provides the classes that are needed for networking applications (the java.net library). Within this library there is a socket class that can be used to create a socket over the network.

To create a socket a host name has to be declared, the host name is the name of the computer that the socket is going to connect to i.e. 'localhost'. A port number also has to be declared, this is the port that the server is communicating on, it should be noted that not all port numbers are available to communicate on. Also when creating a socket in Java exceptions need to be thrown and caught, there are two possible exceptions the first is and unknown host exception, this is where the name of the host is not known, and the second is and input/output exception, this is where the socket cannot communicate over the given host name and port number. An example of socket code can be seen in Table 2.

Table 2: Java code for creating a client side socket.

```
Socket client = null;
try {
        client = new Socket(host Name, host Port);
} catch (UnknownHostException e) {
        System.eer.println("Don't know about host: " + hostName + ":" + hostPort);
} catch (IOException e) {
        System.err.println("Couldn't get I/O for the connection to: " + hostName);
}
```

Creating a socket on the server side in Java is slightly different from creating one on the client side. The main difference is that two classes are used, the first is on called ServerSocket, and the second is as in the client side version is called Socket. The first part in create the socket is that ServerSocket is used to creates a new socket on the requested port number; an exception is thrown if the port is unavailable. The next stage involves waiting for the client to connect to the server; an exception is thrown here if a connection between the server and client cannot be made. See Table 3 for the code for the socket.

Table 3: Java code for creating a server side socket.

```
ServerSocket server = null;
Socket incomingConnection = null;
try {
        server = new ServerSocket(listenPort);
} catch (IOException e) {
        System.err.println("Could not listen on port: 4444.");
}

try {
        incomingConnection = server.accept();
} catch (IOException e) {
        System.err.println("Accept failed.");
}
```

4.3.2 Sending Data

Unlike creating a socket in Java there are not any classes already available that are dedicated to transferring data, this means that a method needs to be created. It should be noted that there are third person classes available that are capable of doing this, however the class will need to be on every machine that the program is run on otherwise it will not work. Taking this into account it was decided to create a method, in creating a method and not using third party classes it allows greater customisation in how the class operates.

To transfer a string over a socket is relatively simple in Java, however to transfer files is more difficult. The file need to be read into a buffer and then send over the socket, the end of a file is indicated by -1 so a flag has to be set, this basically means when the buffer reads in -1 stop sending as the file complete. A buffer is used and as space is limited on the computer a buffer limit is used, once this limit is reached the socket stop transferring the file over the socket. To read a file into a buffer a BufferedInputStream is used, this simply reads in the predefined number of bytes from the file. To send the file over the socket an OutputStream is used, this stream uses the socket and writes the data to the socket, which in turn gets sent over the socket. The code for the send method can be seen in Table 4.

Table 4: Java code for sending files over a socket.

```
public void send(File file) {
        int BUFFER_SIZE = 8192;
        try {

                byte[] buffer = new byte[BUFFER_SIZE];
                BufferedInputStream din = new BufferedInputStream(
                        new FileInputStream(file));
                OutputStream out = client.getOutputStream();
                while (true) {
                        int amountRead = din.read(buffer);
                                        out.write(buffer, 0, amountRead);
                        if ((amountRead == -1) || (amountRead < BUFFER_SIZE)) {
                                break;
                        } // if
                } // while
        } // try
        catch (Exception e) {e.printStackTrace();}
}
```

4.3.3 Receiving Data

To receive data the same problem occurs as with sending data, in that there is not dedicated class to receive data in Java, except for third party classes. The receive method is very similar to the send, in that a buffer is used and that a -1 represents the end of the file. The methods differ slightly in that in the send method the output stream was used to send the data, and the input stream was used to read in the data to be sent. The receive method does it by using the output stream to write to the file and the input stream to read in the file from the socket.

Table 5: Java Code for receiving files.

```
private void receive(String fileName) {
        try {
                int BUFFER_SIZE = 8192;
                File file = new File(fileName);
                BufferedInputStream din = new BufferedInputStream(
                        client.getInputStream());
                FileOutputStream fout = new FileOutputStream(
                        new File(fileName));
                int i = 0;
                byte[] buffer = new byte[BUFFER_SIZE];

                while (i != -1) {
                        i = din.read(buffer);
                        if(i > 0)
                        fout.write(buffer, 0, i);
                } // while
        } // try
        catch (Exception e) {e.printStackTrace();}
}
```

4.3.4 File Checking

File checking is a crucial part of the program as it will decide which file on either the server or client is the correct one to update, if this is wrong then the wrong file might be modified and discrepancies may occur. The Java language has many built in file checks in the input/output library; this can range from simple modification dates to hash code. The first task that needs to be completed when checking files is to make sure the file exists on both the server and client, if it does not exist on one or the other then no further checking need to be done, the file just needs to be sent to the relevant source (server or client). However if the file exist on both the server and client further checking needs to occur, the checking method that is being used to decide which file to update is the date the file was last modified. The assumption made using the date last modified function is that the user always wants the newest file to be kept. There are several possibilities when updating files which can be seen in Table 6.

Table 6: Possible combination for checking.

Problem	Action
File on client is newer than file on server.	The file on the client is sent to the server.
File on server is newer than file on client	The file on the serve r is sent to the client
Both files have been modified at the same time.	No data transfer is necessary.

4.3.5 Creating the GUI

To create a GUI in Java there are many different ways this can be achieved. The Java language includes a library called Java Swing this allows the creation of many of the standard Windows/Linux GUI functions i.e. tool bars, menus, buttons, etc. It should be noted that before the Swing library was included in Java there were other methods available to create GUI's but these were often long and difficult code to create, this is why the Swing library has been chosen. To create a GUI using swing the first thing that needs to be done is to create a frame, from this all the components such as menus, buttons, etc. will be placed on this frame. There are a number of different ways in which to create a frame in Java, the technique used was to have a separate method for all the frame components, this method extends the JFrame class, another option would have been to create the frame within the programs ‘main’ method. The code for the ‘method’ is shown in Table 7, and the code for creating the frame in Table 8.

Table 7: Java code for where the frame is called in the main function.

```
public static void main(String[] args) {
        JFrame frame = new Client();
        frame.addWindowListener(new WindowAdapter() {
                public void windowClosing(WindowEvent e) {
                        System.exit(0);
                }
        });
        frame.pack();
        frame.setVisible(true);
} // main
```

Table 8: Java code for creating the frame.

```
public Client() {
        super("Client");
        Java swing component are added here.

        JPanel pane = new JPanel();
        BoxLayout leftBox = new BoxLayout(pane, BoxLayout.Y_AXIS);
        pane.setLayout(leftBox);
        Java components added here.

        JPanel contentPane = new JPanel();
        BoxLayout box = new BoxLayout(contentPane, BoxLayout.X_AXIS);
        contentPane.setLayout(box);
        contentPane.add(pane);
        setContentPane(contentPane);

} // client
```

Once the frame has been created then all the other components can be added. For these particular programs the menu, button, text field, text label and dialog boxes will be used, the client program will also have a text area in order to display information to the user and a file chooser so the user can select which files to synchronise. Each of these components is discussed in detail below.

The first component to be included into the frame is a menu bar; this is relatively simply to do using swing. The first task is to create a JMenuBar, this is a blank menu bar with nothing in, and the next stage is to add a JMenu. The menu is named whatever the programmer wants for example ‘file’, ‘edit’, etc. The final stage is to add a JMenuItem such as ‘open’, ‘save’, etc. once this has been done a menu system has been created such as ‘File’ -> ‘Open’ or whatever the programmer designed. Example code for menu systems are in Table 9. It should be noted that when creating the menu bar shortcuts for the keys, e.g. ALT-1, can be created.

Table 9: Example Java code for menu bars.

```
JMenuBar menuBar;
JMenu menu;
JMenuItem menuItem;

menuBar = new JMenuBar();
setJMenuBar(menuBar);

menu = new JMenu("Menu");
menu.setMnemonic(KeyEvent.VK_M);
menu.getAccessibleContext().setAccessibleDescription(
        "The only menu in this program that has menu items");
menuBar.add(menu);

// create the choices
menuItem = new JMenuItem("Select Files", KeyEvent.VK_S);
        menuItem.setAccelerator(KeyStroke.getKeyStroke(
         KeyEvent.VK_1, ActionEvent.ALT_MASK));
menuItem.getAccessibleContext().setAccessibleDescription(
        "This doesn't really do anything");
menuItem.addActionListener(this);
menu.add(menuItem);
```

The next stage in creating the GUI is including a button; this will be used to create the socket connection. As with much of Java swing creating a button is relatively simple, JButton item is created with the designer specifying the name and the name/button that is going to be displayed on the button itself. Example code in Table 10.When the user ‘clicks’ the button it activates an Action Listener, this basically say when the button is pressed do the predefined task, this also happens when the user clicks on an item it the menu bar (see above for details).

Table 10: Java code for creating a button.

```
JButton button = new JButton("Button name");
createConnection.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
        Do something
}
```

In order to allow a user to input information a JTextField is used this object allows the user to enter data, in order for the program to use the data an Action Listener is used like the ones in the button and menu. Generally when a text field is used it has a text label ask a question, this is done by using JLabel object. Both these part of the GUI are easily implemented as shown in Table 11.

Table 11: Example code for text and label fields.

```
JTextField:
JTextField portField = new JTextField(20);
portField.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                Do Something
        }
})

JLabel:
final JLabel prompt = new JLabel("Some text to display.");
```

The above code for the GUI explains the common components in both the client and server programs, the following two components are only in the client program. The first component is the text area; example code can be seen in Table 12, which uses the JTextArea component. This is relatively simple to implement as the designer only needs to specify the size of the text area, the size of the margins, if the user can edit text in the area, and if the text area has a scroll bar. The final component to be created for the client software is a file chooser, this with allow the user to select which files they want to synchronise. To create this component a JFileChooser object needs to be created, once the object is created an action listener is used to chosen when it appears. The code can be seen in Table 13.

Table 12: Example code for a text area.

```
JTextArea log;


log = new JTextArea(15,30);
log.setMargin(new Insets(5,5,5,5));
log.setEditable(false);
JScrollPane logScrollPane = new JScrollPane(log);
```

Table 13: Sample code for creating a file chooser.

```
JFileChooser fc = new JFileChooser();


int returnVal = fc.showOpenDialog(Client.this);
```

4.3.6 Overview of the Code

It should be noted that there is more code than what is listed above, see appendix; there are a number of reason why they are missing from the design parts. The main reason why they have been omitted is that most of it is trivial code, for example displaying the names in an array, or adding/subtracting from an array. Other parts of the code that have left out because the Java documents explain how it is done and the code is not that dissimilar, for example the use of dialog boxes to report error. The code that has been excluded does not add anything of importance to the way in which the program synchronises files or improves the GUI and therefore been omitted.

The final version of the programs can be found at http://javaprograms.netfirms.com.

5 Evaluation

5.1 The Criteria

To evaluate this software the criteria that is going to be used is the same as the criteria used to analyse the other file synchronisation products. It is important to use the same criteria as it will give a better representation of the software. The criteria are as follows:

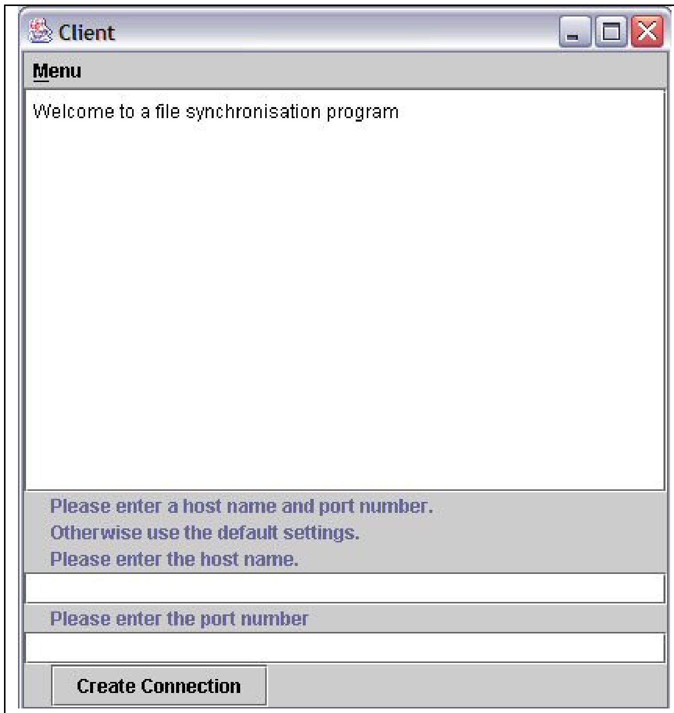
- How well presented is the software? This is important as it is what the user 'sees' it is important that software is presented well in order to give a professional look, as well as giving the user confidence to use it.
- Is the interface sufficient? This is important as if the interface is not capable of completing the tasks given then the program will not perform as well as it could do.
- Is the software interface easy to use, or is there a learning curve? This is another important criterion as if the user interface difficult to use then users may be dissatisfied with the program. On the other hand if the program is easy to use and self explanatory then the user will approve and not have difficulty in synchronising files.
- How portable is the program, is it capable of synchronisation over different systems? The importance of this criterion is that this is what the project is about; if the program cannot perform this task then the program has failed.
- How efficient is the synchronisation? This is another criterion as the more efficient the synchronisation is then the more efficient the will be.

5.2 Analysis of the Software

The software created meets both the minimum requirements and the criteria above. The first criterion is 'How well presented is the software?' the answer is that the software is well presented, but it could be better. See Figure 9 and Figure 10 for screenshots. The layout is set out in 'standard' windowed program style, i.e. a menu bar at the top and a text area in the middle. The down side of the presentation is the connection method at the bottom of the text area. This would have been more ideal to be placed in a separate frame; however because of time restrictions and problems with the frames and sockets in Java time to implement the connection button was limited.

The next question: 'Is the interface sufficient?' The simple answer to this question is, yes, the interface is more than sufficient for the job. Even though the interface is sufficient it does not mean it cannot be better, there are a number of improvements that can be made which are discussed in chapter six.

The next question: 'Is the software interface easy to use, or is there a learning curve?' the answer is as with all programs there is a learning curve, but the curve is not that steep and users who are familiar with GUI interfaces the program should be easy to use. The user interface is not complex and therefore easy to understand, once the user has performed the operations a number of times then it should be relatively easy to execute instruction.

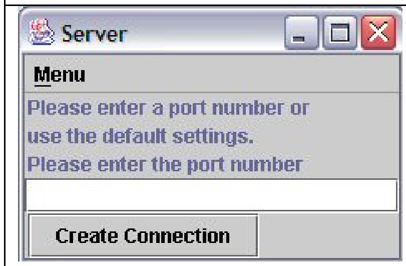


Welcome to a file synchronisation program

Please enter a host name and port number.
Otherwise use the default settings.
Please enter the host name.

Please enter the port number

Figure 9: Screenshot of the client program.



Please enter a port number or
use the default settings.
Please enter the port number

Figure 10: Screenshot of the server program.

'How portable is the program, is it capable of synchronisation over different systems' the answer is that the software is capable of synchronising over different systems so long as the systems are compatible with Java. If a system does not support Java then the software cannot synchronises with it, this does limit the number of systems the program can synchronise with to Window, Linux, MacOS, and Solaris.

The final question 'How efficient is the synchronisation?' is that the software is capable of synchronisation but its effectiveness is not as good as other programs (see below). The software has to transfer the entire file, when two files are out on sync this means that if on one kilobyte is different between then in a one megabyte file then most of the data transfer is wasted. A more effective algorithm would be to transfer the one kilobyte that is different.

5.3 The Software Compared to Other Solutions

The following section compares the software created to the software analysed in chapter three. The software is again compared on the criteria above.

The first question: 'How well presented is the software?' of all the software analysed the one that is the best presented is Backer 6.1 and the worst is probably Rsync or Unison that use command based inputs. The software created is probably in the middle of all the GUI interfaces available, the software does not have the professional of programs such as Backer 6.1 and Novell IFolder but the general layout and appearance is better than programs like Xfiles.

The second question: 'Is the interface sufficient?' the software created is more than capable of completing the tasks with the interface used. However other programs, especially Backer 6.1, have a superior interface that is capable of so many more tasks than the software created. Having said this, the interface designed is more sufficient than others, such as the Xfiles software which offer an overly complex interface.

The next question: 'Is the software interface easy to use, or is there a learning curve?' of all the programs tested the most simple and easy to use was Backer 6.1, as it offered a simple to understand user interface with a synchronisation wizard that took the user through the tasks necessary to synchronise. The programs with the more difficult interfaces were Rsync and Unison with command based approach to synchronisation. Looking at the software analysed and the software created it is apparent that the software created is one of the most simple and easy to use.

'How portable is the program, is it capable of synchronisation over different systems?' None of the programs analysed are capable of synchronisation over every type of systems, Novell IFolder was capable of synchronisation with the most systems, and programs such as Backer 6.1 and Xfiles were only able to synchronise with one type of system. The software developed is capable of synchronisation with four systems (Windows, Linux, Solaris, and MacOS), which makes the software designed one of the most successful analysed.

How efficient is the synchronisation? Of all the software analysed the software created had the least efficient synchronisation method. The reason for this is because the transfer method had to transfer there entire file, as opposed to programs such as Rsync and Novell IFolder that transferred just the difference in the files.

5.4 Conclusion

Of all the software analysed they all have there advantages and disadvantages. No one program added was superior to the once created in every aspect, there are some features that are better implemented that the once in the created software, such as better synchronisation method, or better user interface. It should be noted that the software created is only the first version, many of the programs analysed have multiple version, and each version improves upon the last. If the software created was continued to be develop newer version would add more features and improve, this would in turn reduce the gap between the commercial products available and the software developed.

6 Conclusion

6.1 Minimum Requirements

The minimum requirements for the project are as follows:

- To produce a stable piece of software that can synchronise files between two different platforms. The software should be able detect when a file has changed and decide which copy to update on either computer.
- To produce a simple user interface to the software. The interface should be easy to use and intuitive for the user.
- To product basic documentation for the software. The documentation should be easy to follow, easy to understand and without too many technical terms for the novice user. The documentation should be in a text file.

All of the above requirements have been met, and below are the possible enhancements that are in the mid-project report:

- Further the software by allowing it to be synchronised over more that two operating systems such as Microsoft Windows, Linux, and Palm OS for example.
- Produce a more comprehensive user manual, through the use of an HTML page, or a printed document bound together.
- Improve the efficiency of the program for example automatically detect when a file has been updated and update the file on the other computer.

Of these enhancements only one has been achieved; produce a more comprehensive user manual. The manual is now available in a number of different formats; PDF, Microsoft Word and is also available on the internet at http://javaprograms.netfirms.com.

6.2 Enhancements

Having completed the project there are a number of enhancement and future work that can be made in this subject. Examining other software showed just how many different solutions there are to this problem and each have there own advantages and disadvantages.

There are a number of enhancements that could be made to the program that has been created. The main enhancement that can be made is in the file checking. The enhancement would give the user a greater freedom over the checking that can be performed, at the moment its limited to time last modified. Another possible area is in the synchronisation process, at the moment the whole file has to be transferred when synchronising, however if only part of the file needs to be updated then it would be more efficient and quicker just to transfer the differences in the file.

Another major improvement that could be made is the user interface. Even though it was more than capable of the job improvements could have in giving the user more feed back. For example more information could have been given to the user about what file was being transferred and in what direction. This would have improved the usability of the software as well. Also in the way in which the user connects to the server could be improved by including a separate window for the connection for example.

6.3 Future Work

There is a wide range of work that can be done if the file synchronisation sector. The main area to look at is in the user interface, in the programs analysed it is apparent that they do not convey to the user adequately what files are being transfer and in what direction. The only one that can close was Baker 6.1, and even sometime this was to confusing by overloading the user with too much information.

Another crucial area in which more work is done is the ability for the software to synchronise over multiple systems. A lot of the analysed software was not capable of effectively synchronising files over different types of system. This has to include PDA, a topic not closely looked at but many programs did not allow synchronisation with them.

Other enhancements that can be made are: built in help files with the program, similar to Microsoft Word. Improved checking algorithms, and give the user greater control over the types of checking. Improve the synchronisation methods so files can be sent over multiple sources i.e. the Internet.

The above improvements are only a handful that can be made, no matter how complete the software is there can always be improvements made, take Microsoft office for example. The improvements mentioned are the main improvements that should be concentrated on perfecting first before adding more features to the program.

References

[1] PDA OS Overview Cnet:
A review of the three main PDA operating systems: Palm OS, Windows CE, and EPOC.
http://computers.cnet.com/hardware/0-1087-7-4916659.html?tag=redirect
[8th Dec 2002]

[2] Palm:
The Palm OS homepage that details information and libraries for developing programs for the Palm OS.
http://www.palm.com
[8th Dec 2002]

[3] Java:
The Java Sun homepage listing the API's available for use.
http://java.sun.com/j2se/1.3/docs/api/
[9th Dec 2002]

[4] MSDN:
The Microsoft Development Network which includes information and tutorials for programming in Visual C++.
support.microsoft.com/default.aspx?scid=fh%3Ben-us%3Bvcc&x=11&y=16
[9th Dec 2002]

[5] Penguin Backup (for Linux/Palm OS):
The Penguin Backup homepage includes information about the Penguin Backup program and downloads.
http://penguinbackup.sourceforge.net/
[5th Dec 2002]

[6] Peersoftware:
Homepage for different commercial product that can synchronise files between different sources.
http://www.peersoftware.com
[5th Dec 2002]

[7] Zip-n-Go:

Synchronisation software that zips files before synchronisation.
http://zip-n-go.com/
[5th Dec 2002]

[8] Linux File Synchronisation Tools
http://linux.web.cern.ch/linux/redhat6/laptop/ftools/ftools.html
[5th Dec 2002]

[9] Rsync:
Homepage for Rsync includes information and downloads.
http://samba.anu.edu.au/rsync/
[10th Dec 2002]

[10] Unison
Homepage for Unison includes information and downloads.
http://www.cis.upenn.edu/~bcpierce/unison/
[10th December 2002]

[12] Review of Operating Systems:
A review of the different operating systems available for public use.
http://tunes.org/Review/OSes.html
[7th Dec 2002]

[13] Linux Devices
Gives brief information about the Linux PDA operating systems and PDA's that include the operating system.
http://www.linuxdevices.com/articles/AT8728350077.html
[11th Dec 2002] Updated [7th Nov 2002]

[14] Shneiderman Ben, 1992, Designing the User Interface: Strategies for Effective Human-Computer Interaction 2nd Edition

[15] Mercer R, 2000, Computing Fundamentals with C++ 2nd Edition

[16] Python Homepage
Information about the Python language.
http://www.python.org

[24th April 2003]

[17] Rossum Guido van, Comparing Python to Other Languages
Essay written by Guido van Rossum which compares Python to other languages available.
http://www.python.org/doc/essays/comparisons.html
[24th April 2003]

[18] Perl Homepage
The official homepage of Perl.
http://www.perl.org
[24th April 2003]

[19] Xfiles 1.4 Homepage
Information about the Linux software synchronisation tool Xfiles
http://www.idiom.com/~zilla/xfiles.html
[26th April 2003]

[20] Novell iFolder Homepage
Information about the Novell IFolder synchronisation software
http://www.novell.com/products/ifolder/
[26th April 2003]

[21] Backer 6.1 Homepage
Information about Backer 6.1 synchronisation software.
http://www.cordes-dev.com/
[27th April 2003]

[22] Orfali Rober, Harkey Dan, 1997, Client/Server Programming with Java and CORBA

[23] Gouda, Mohammed G, 1998, Elements of Network Protocol Design

Appendix A – Reflection

Research into file synchronisation has been an interesting, yet more complex subject than I originally though, to look into. The topic is very large and includes many aspects that I did not expect, initially, for example the greater importance of HCI. Many things have been learnt during this project each one valuable. I think the main lesson learned is the importance of research in a project, and how it is related to the final product. Without completing the research implementing the design would have been virtually impossible. The research gave insight into different techniques and methods, and gave more possibilities in creating the product.

Another area that has been of great importance in creating this project is the analysis of other software. The analysis allowed me to see what other people/companies had done and how they had tacked certain problems, for example the user interface.

During this project the topic of design has shown its importance, its shown that it is important to plan the design and implementation before starting to program. If this is done time can be saved in programming as not as much time is wasted trying different ways to do certain jobs. The design is especially useful for designing a user interface as it allows the designer to 'view' mock interfaces and then decide which is best to implement.

This project has also given me valuable programming knowledge; it has furthered my knowledge of Java by including GUI design and net features (i.e. the socket). It has also given me a greater insight into communicating over a network in any computer language and the techniques that are used to do it. One of the biggest tasks in the project was creating a method that was capable of sending files across a socket.

Much has been learnt in this project, however of it was done again a number of changes would be made: the first is the choice of programming language to use, if it was done again I think that either Python or Perl would be used as the 'glue' language and then java would create the GUI. This would give the program more flexibility. Also I would change the GUI design slightly and make it more user friendly by giving the user more control and relay more information back to the user showing what is happening.

Overall the project has been very interesting and enjoyable. Many lessons have been learnt for the creation of software, be it either design or research or the other topics, and I'm sure the lessons learned will be useful in the industry and will improve my design and programming techniques.

Appendix B – Schedule

Objectives	December				January						February				March				April				May	
Mid-project Report	🟥	🟥					🟦	🟦																
Background research	🟩	🟩	🟩	🟩	🟩	🟩	🟦	🟦			🟩	🟩	🟩											
Understanding the problem and learning new techniques			🟥	🟥	🟥	🟥	🟦	🟦			🟥	🟥	🟥	🟥										
Understanding HCI and computer interfaces					🟩	🟩	🟦 E X	🟦			🟩	🟩	🟩	🟩	🟩	🟩								
Implementing the software							🟦 A	🟦							🟥	🟥	🟥	🟥	🟥	🟥				
Testing the software							🟦 M	🟦								🟩	🟩	🟩	🟩	🟩	🟩			
Creating the documentation							🟦 S	🟦											🟥	🟥	🟥			
Further Research (for enhancements)							🟦	🟦													🟩	🟩		
Implementing the enhancements							🟦	🟦														🟥	🟥	
Testing the enhancements							🟦	🟦															🟩	
Updating the documentation							🟦	🟦																🟥
Final Report							🟦	🟦												🟩	🟩	🟩	🟩	🟩