Operating System for Kids

By
Rupal Gala

Supervisor: Dr Julie A. McCann
Second Marker: Mr Ian W. Moor

Individual Project
BEng Computing Degree

Department of Computing
Imperial College London

June 2003
Abstract

In today’s society, it is becoming increasingly important to become computer literate, and to start using computer technology from an early age is a big head start. The promise of the networked world of the Internet and the World Wide Web is that we have the potential to profoundly shape children’s education by developing educational materials that can more easily traverse the home to school environment.

The problem with this is that there are very few computer interfaces specifically designed to introduce children to the use of computers. From the child’s point of view, the commands can often be complicated, and using a mouse can be quite cumbersome. As for the parent, to see their child at a computer can be either a beautiful and proud sight to behold, or it can create absolute fear. This other side looks at computers as an expensive machine and are frightened by scenarios where their children are presented with the “Delete All” option, from which the child duly presses “OK”!

The aim of this project is to design and then evaluate a system suitable for children, which will address these and other problems.
Acknowledgements

First and foremost, I would like to thank those who were involved in the evaluation and testing and without whom this project would not have been possible:

Charlotte Ingram, Fiona Ingram, Jamie O’Neil,
The students at Stag Lane First School: Adil Mirza, Jazz Jordan, Vaishally Patel, Pooja Shah, Nina Cowlin, Justin Mulcahy
The staff at Stag Lane First School, especially the headmistress, Mrs Brownsell, and Mrs Madhu Shah

As well as giving up their time, they were also eager to offer advice and their ideas broadened my views on the various issues involved.

I would also like to thank my flatmates who have had to put up with me!

Last, but definitely not least, many thanks go to my supervisor, Dr. Julie A. McCann, whose idea the project originally was.
# Table of Contents

Abstract ........................................................................................................................ 2  
Acknowledgements...................................................................................................... 3  
1. **Introduction** ......................................................................................................... 6  
   1.1 Objectives ......................................................................................................... 7  
   1.2 Report structure .............................................................................................. 8  
2. **Background** ......................................................................................................... 9  
   2.1 Children and computers ................................................................................. 9  
   2.2 User Interface Paradigms ............................................................................. 10  
      2.2.1 Command-based Interaction ................................................................ 11  
      2.2.2 Direct Manipulation ............................................................................. 11  
   2.3 Some existing operating systems ................................................................. 13  
      2.3.1 Microsoft Windows .............................................................................. 14  
      2.3.2 KidDesk ................................................................................................ 15  
      2.3.3 Microsoft Bob ...................................................................................... 15  
3. **HCI** ..................................................................................................................... 17  
   3.1 What is HCI? .................................................................................................... 17  
   3.2 HCI principles .............................................................................................. 18  
      3.2.1 Usability ............................................................................................... 18  
      3.2.2 HCI for children ................................................................................... 19  
   3.3 What do children want? .................................................................................. 23  
   3.4 What do adults want? .................................................................................... 24  
4. **Design** ................................................................................................................. 26  
   4.1 Target audience .............................................................................................. 26  
   4.2 The Design .................................................................................................... 27  
      4.2.1 Workbench ........................................................................................... 30  
      4.2.2 Notepad ................................................................................................ 31  
      4.2.3 Paint ...................................................................................................... 33  
      4.2.4 Browser ................................................................................................ 34  
      4.2.5 Email .................................................................................................... 36  
      4.2.6 My work ............................................................................................... 37
1. Introduction

Whilst working in the children’s section at Harrods, staff were often asked by parents whether there was any software specifically designed to help children become familiar with using computers. There are a lot of computer-based educational software and games, but there are virtually no user interfaces for children from where they can perform the basic functionality that a normal Operating System can whilst providing the security required to protect both child and computer.

Children are becoming exposed to computers and technology at an increasingly early age, and most of the technology developed did not make the child-computer interaction optimal. For instance, early on in the design of children’s software products, the strategy used was to add a user interface with animation and primary colours on top of an existing adult product. However, this failed to deal with underlying issues such as complex error messages or detailed help systems too complicated for children to use and understand.

Further motivation was the opportunity to study and understand the relationship between the computer and the child. The idea of how computers have aided the development of children in their early years, and how they can be enhanced to provide even more progress, is exciting and this is something that is hoped to be touched upon during the course of the project.

These reasons prompted the undertaking of this project to investigate what exactly makes an Operating System suitable for a child, find out why current options fail to meet these needs, and hopefully create a satisfactory solution. The goal is to enable more young children to use a computer for their personal purpose with minimal training. There has been a lot of research conducted about children and computer interaction, but no satisfactory system design specification has been produced.

To do this, we will first conduct a study into understanding Human Computer Interaction
Operating System for Kids

1. Introduction

(HCI) principles, especially focusing on those specific to children. After looking at the two types of Operating Systems, the Command-based Interaction and the Direct Manipulation Systems, we will review two products developed to make using a computer less daunting for young, novice users.

Most of the HCI research and the resultant software have generally been targeted for adult applications. Furthermore, most of the research conducted on children has been in the school environment, where children find it much harder to express their natural freedom. They are not in control of when they can have a lesson such as Art, or what they can write about, or even when its time to go home. In school everything is much more sterile. They are not allowed to feel that they “owned” the technology, and teachers tend to rely on verbal instructions, which is not how the children are used to learning at home. As a result they tended to put their brains onto autopilot and let the lessons wash over them.

Evidence (including Anderson, Lundmark, Harris & Magnan, 1994; Comber, Colley, Hargreaves & Dorn 1997; Kirkman, 1993) [W13] suggests that children who own or have access to home computers, for example, demonstrate more positive attitudes towards computers, show more enthusiasm, and report more self-confidence and ease when using computers than those who do not have a computer in their home.

For these reasons, we intend to focus on children’s interaction with computers inside and outside of the school environment, where we feel we can get a more genuine image of what is needed to create a successful interface for the Operating System.

It has been noted in many papers that we don’t work with children when designing software for them (Druin 1996) [B1]. When we develop technologies for doctors, astronauts or investment bankers we work with them. For children’s applications, it is typically the teachers or parents that are consulted, whereas children are more commonly observed and tested to measure the impact of using the new technology. Instead we are aiming to produce a design based on children’s needs and wants, and then involve them into the testing process.

1.1 Objectives

The objectives for this project are as follows:

- Gather ideas and concepts of a system for children that have been suggested or recommended as being appealing, and decide on the main design features required.
• Implement a prototype of an operating system user interface that meets the design criteria.
• Test the prototype by using several methods, such as measuring the acceptance of children.
• Evaluate whether recommendations made by researchers are accurate and attempt to identify additional features that could be incorporated.

1.2 Report structure

• Chapter two looks at the effect of computers on children, and assesses some Operating Systems used by children.
• Chapter three describes some design principles and other research which influenced the design of the application.
• Chapter four then looks at some design constraints and then explains the various components created.
• Chapter five justifies why Flash MX was used for the creation of the software, and also looks at some of the techniques used.
• Chapter six describes the methods and results from the user and technical testing that was carried out.
• Chapter seven then evaluates these results, and also outlines suggestions from the users.
• Chapter eight then concludes the project, as well as looking at any future investigations and improvements that can be made.
2. Background

We begin by taking a look at the influence computers has had on the lives of our children and then follow this up with a study of user interfaces and the history of Operating Systems, stopping to look at applications designed to make children more comfortable with the use of computers.

2.1 Children and computers

Children have always used a wide assortment of tools to explore and understand the world. Technology offers new tools - including keyboards, computers, and software that allow for exploration and creation. A computer used with young children is found to be “a powerful learning device that facilitates cognitive development and positive social interaction without harm to young children” (Buckleitner & Hohmann, 1987, p.338.) [W1]. By using computers from an early age, children have been able to benefit from several advantages:

- Computers provide great opportunities to inspire children to learn. The computer helps develop children’s cognition. When computers are used appropriately, children can be encouraged to develop their freedom and creative thinking.
- Computers capture children’s interest and stimulate their imagination and creativity. Computer use helps children to move from concrete to symbolic representational thought, thus improving their ability to think abstractly.
- Computers enable children to learn from each other through cooperative learning. They can learn to share information, and to contribute to ideas. Thus, they are able to develop a high level of communication and cooperation skill, problem-solving skills in a collaborative manner, and higher order thinking skills.
- Computers satisfy individual students’ needs. Computers help children master concepts at their own pace. Individual students may choose to repeat a certain task several times until they feel comfortable about the intended tasks.
Today computers continue to foster a positive attitude to learning. The benefits above contribute to ensuring that children are exposed to computers in their early development years, as they make an ideal context for learning through play. In order to make a smooth transition into the technological environment, they should be provided with a suitable interface that would make their learning an easy progression.

Many children watch their parents and teachers using various tools on a computer and want to have the same experiences. However, this can get very frustrating and tiring when they have to spend more time learning to navigate around the system rather than enjoying the experimentation and exploration of these tools. There is a need for all the basic applications to be integrated together into a single, easy-to-use package.

2.2 User Interface Paradigms

A user interface paradigm is a model for human-computer interaction. They are often based on metaphors of real life situations, for example the desktop metaphor is the typical application for the graphical user interface paradigm.

A brief history of user interface

A 1970s user interface usually consisted of many fields with very cryptic and often unintelligible captions. It was visually cluttered, and often possessed a command field that challenged the user to remember what had to be keyed into it. Effectively using this kind of screen required a great deal of practice and patience and often required referral to a manual to interpret.

At the turn of the decade guidelines for text-based interface designs were finally made available and many screens began to be less cluttered and user memory was supported by providing clear and meaningful field captions and by listing commands on the screen, and enabling them to be accomplished through function keys.

The advent of graphics yielded another milestone in the evolution of screen design. Buttons and menus for implementing commands replaced function keys and multiple properties of elements were also provided, including different font sizes and styles.

Two main paradigms have dominated over the past 25 years – the Command line and the Desktop, which are both instances of more general classes of user interface – command-based and direct manipulation. Along with having core differences with the way they function, they
each have a distinctive user interface. In order to pick the most suitable paradigm for children, we need to compare the two areas, outlining their features.

2.2.1 Command-based Interaction

The first instance of what could reasonably be described as a user interface is the command line. The system itself is regarded as a black box, with discrete non-interactive commands taking inputs, manipulating the system somehow, and then generating outputs. Common examples are Unix, and MS-DOS.

Applications running under such systems also follow a command-based scheme, with complex key sequences needed to perform actions. The effect of these actions is often not acknowledged on-screen, and if they are then it is often in a confusing or non-intuitive manner. Due to the speed of older systems that often offer command-based interfaces, immediate feedback on what has been done as a result of a command is often not possible. It is not immediately obvious what commands were available or possible, and even if the user knows what has to be done, extensive prior knowledge and training is necessary to allow them to do it.

Later on, menu systems and form-filling interfaces came into use, which present available options from which one can be selected, and guided parameter entry with prompting. Often menu systems are tree structured, guiding the user through sensible sequences of actions. Although this partially solves the problem of remembering complex commands and their arguments, and generating sensible sequences of commands, there is still usually no immediate feedback about what has been done.

Command systems lack interactivity. Experts are able to immediately recall commands without being directed by an often restrictive user interface dialogue. These factors, together with their inability to read and understand commands and instructions, make the command line the least practical for children.

2.2.2 Direct Manipulation

This is the general name for the class of user interfaces seen in the vast majority of present day operating environments. A common example is Microsoft Windows.

The system is portrayed as an extension of the real world. It is assumed that a person is already familiar with the objects and actions in his or her environment of interest. The system
simply replicates them and portrays them on a different medium, i.e. the screen. A person has
the power to access and modify these objects, among which are windows. The user is allowed
to work in a familiar environment and in a familiar way, focussing on the data, not the
application and tools. The physical organisation of the system, which is most often
unfamiliar, is hidden from view and not a distraction.

The Desktop metaphor and the “graphical user interface” are examples of a direct
manipulation interfaces. Ben Schneiderman (1982) [B3] first presented Direct Manipulation,
as a collective term for user interfaces exhibiting the following characteristics:

- Continuous representation of the object of interest.
- Physical actions or labelled button presses instead of complex syntax and command
  names.
- Rapid incremental reversible operations whose impacts on the objects of interest are
  immediately visible.

Such a system is thought to reduce the memory requirements imposed on the user, make more
effective use of one’s information processing capabilities, and dramatically reduce the
system’s learning requirements.

The success of Direct Manipulation operating system interfaces can be attributed to a host of
factors. The following are a summary of the advantages of these systems, as offered by
Wilbert O. Galitz:

- *Symbols recognized faster than text*. The graphical attributes of icons such as shape
  and colour are very useful for quickly classifying objects, elements or test by some
  common property.
- *Faster Learning*. Symbols can be easily comprehended and learned. Because of the
greater simplicity it is easier for novice users to retain operational concepts.
Predictable system responses also speed learning.
- *Fosters more Concrete Thinking*. Displayed objects are directly usable in their
  presented form. There is no need to mentally decompose tasks into multiple
  commands with complex syntactic form.
- *Exploits visual/spatial cues*. Spatial relationships are usually understood quicker than
  verbal representations.
- Increased feeling of control. The user initiates actions and feels in control. This increases user confidence and hastens system mastery.

- Immediate feedback, predictable system responses and easily reversible actions.

- More attractive. Systems are more entertaining, clever and appealing. This is especially important for the younger age group.

Frese, Schulte-Gocking, and Altmann (1987) [W8] compared learning and performance for direct-manipulation and command-based word processing systems. While no difference existed after the first experimental session, the direct-manipulation system user’s performance became increasingly superior as the study progressed (and task complexity increased). It appeared the direct-manipulation system facilitated the learning process as the task complexity increased.

A study done by Brown and Schneider (1992) [B3] in which a direct manipulation interface was compared to a conversational computer interface using elementary school students grades three to six. The children were given basic arithmetic problems. It was found that the direct manipulation interface was more comfortable and enhanced the speed of completing the basic arithmetic tasks. Informal observations showed that students experienced more difficulty and frustration with the conversational computer interface.

2.3 Some existing operating systems

The one area in the literature that covers learners in general is the area that looks at the needs of the novice user versus the experienced user. It is safe to assume that most children are novices. Joanna Lynne McGrenere (1996) [W6] covers the areas of novice vs. experienced users and some specific issues in child-centred HCI. When users are new to an application they generally have to learn and understand the interface first. Most user-interface research concentrated on expert users despite the fact that the more difficult issues, namely ease of learning and skill acquisition, arose with the novice users. This is highly counter-intuitive. The application, whilst encouraging exploratory learning, should be designed to match to the user’s skill level.

Even though there are such interfaces available, with a great deal of importance given to the direct manipulation paradigm, it is disappointing that the knowledge gathered over the last thirty or so years has had so little influence on the design of a system specific to children.
There are many educational software and games aimed for children below the age of 10, but few interfaces for helping novice users with the computer. Two such programs are Microsoft Bob and KidDesk; both are based on the Direct Manipulation paradigm. Before discussing these, we shall look at Microsoft Windows briefly.

2.3.1 Microsoft Windows

Microsoft Windows has been the most popular operating environment over the past several years. Most children start learning computers on a machine running Windows OS and it is imperative to consider its usage by children.

Windows uses icons to represent objects that appear on the desktop, but some of these can be unclear, especially to children. For example, the icon for Internet Explorer does not in any way imply that this is an application through which to browse the Web. The hieroglyphic icon lacks intuitiveness and defeats its purpose. The user has to learn what the icon represents, as there is no direct mapping between its appearance and functionality.

By removing the administrative capabilities so that the children can use the computer in user mode, there are still several problems that they can encounter. Each user, by default, gets the whole array of programs installed on their system, and these are automatically entered into the menu system accessible via the Start button. From my experiences and observations, some useful breaking down of these into categories is done, but most of these categories have no meaning to the children. Children are probably more interested in the “Games” folder, and it is not intuitive to them to look under “Programs --> Accessories”.

Another feature of Windows is that users can have more than one window on the screen at a time and can move from the active window to any other window with the click of a mouse on the Taskbar. Children do not understand that these windows are “open” on the screen, and thus this feature useful to adults may be completely disregarded by children, and just adds to the complexity of the system.

Error messages occur when a user, a software, or hardware does not do what the computer is expecting it to do. Most error messages are simple enough to deal with and involve such mundane messages as "The printer is out of paper." or "Please insert the disk *DISKNAME*." Unfortunately, sometimes error messages make no sense at all to the average user, let alone a child. The machine may display a message ‘Fatal Exception in 0D’ or something similar, such as how an error of Type -3 has occurred.
Windows Help and Support Centre is well documented but consists of hundreds of lines of instructions and to children in just beginning to read, instead of being a source of help, it is just another mind-boggling distraction.

2.3.2 KidDesk

KidDesk was designed by Edmark, and works on the Windows and MAC platforms. Originally designed and marketed as a child-friendly desktop protection and management program, it helps keep children out of places where adults don’t want them to be by creating a child-friendly environment that they can't leave. An analogy is that of a personalised prison, where the parent can customise the desktop for individual children to suit their likes, abilities and needs, whilst restricting access to files and programs. This is done during setup, where the adult decides which programs the child will have access to, and these programs are subsequently added to the desktop. Adding / Restricting a program will not affect its usage to normal adult users in any way, it simply renders it accessible from KidDesk.

KidDesk also controls what websites children are able to visit. With its child-friendly Internet Browser, parents are given the tools to create their own list of positive sites for children, or alternatively can download sites that Edmark (or any other website which has a directory of child-friendly sites) has selected.

KidDesk also has the ability to be configured with an auto-start option that provides constant hard drive security; any time the computer is switched on, KidDesk is the first thing that comes up, and the last thing children see before they log off.

KidDesk only works well with Internet Explorer. It can be run with Netscape, but this usually results in some conflicts and time-consuming reconfigurations, many of which the average parent may not be able to do. Such an application should have been programmed to run with any browser equally well.

The desktop on KidDesk is cluttered, with items scattered everywhere. Some of the icons lacked intuitiveness. This can make the desktop unattractive to the child, and discourage them from using it.

2.3.3 Microsoft Bob

Microsoft Bob was a front-end application for Windows 3.1 designed to make computers less threatening to children and the technologically bewildered. Microsoft Bob consisted of eight
integrated programs / characters - Letter Writer, Calendar, Checkbook, Household Manager, Address Book, E-mail, Financial Guide and the GeoSafari quiz game.

The difference between common existing user interfaces and Microsoft Bob is primarily the characters, which are almost human-like in the way they respond to the operator. This is an example of Affective Computing, which takes into account that human beings are emotional and social beings by nature. Each character within Microsoft Bob has a different comical personality and is programmed to recognize any usage patterns.

Microsoft claimed that it is significant that help is offered automatically and in an encouraging, humorous manner. The result, it says, is users interact with the computer as if it were a living, responsive thing, which leads, ultimately, to a more pleasant computing experience.

Initially, Microsoft Bob performed well in usability tests, and users felt it was "easy" to understand and use. However, over days of usage people found it to be annoying and the "ease of use" features that helped them get started were now in the way of getting things done. Most notably, it was uncovered that for Microsoft Bob's e-mail to work the computer owner had to take out an account with MCI - no other on-line services provider would do. This inflexibility and "heavy handedness" didn’t go down well with customers.
3. HCI

To design an ‘Operating System for Kids’, it is necessary to understand the Human-Computer Interaction (HCI) principles in general, and especially in relation to children and to research and discuss those features required for the design.

"All art is an illusion" so Picasso reputedly once claimed. Although the great master was not thinking in digital terms at the time, his maxim nonetheless captures the essence of human-computer interface design.

Much time, money and effort has been put into the field of user interfaces – how they should look and feel, how they should be designed and implemented and what should define these things. The look and feel of a modern user interface – its appearance, how it operates – is the product of many thousands of hours of experiments and surveys. Establishing an understanding of HCI issues that are specific to children is an important area of our research.

3.1 What is HCI?

The term Human-Computer Interaction (HCI) was adopted in the mid 1980’s to describe the emerging field of study that focuses on all aspects of interaction between users and computers. This includes, amongst others, the design of the computer interface. HCI is concerned with understanding, designing, evaluating and implementing interactive computing systems for human use (Preece, 1994) [B3].

Through its first few decades, a computer’s ability to deal with human communication was inversely related to how simple a task was to carry out. The computer demanded rigid typed input through a keyboard; people responded slowly using this device and with varying degrees of skill.
An underlying theme that has emerged from HCI research is that users come first. The concept of usability is central to HCI and refers to making systems easy to use – users should not need to adapt to a system, but rather the system should be built to suit the needs of the user. Given that the users’ needs are paramount, one important aspect of HCI is to understand the context and environment in which systems will be used.

Designing an interface fits into the broad domain of HCI. The users are children and the goal is to produce an interface that is friendly, aesthetically pleasing and most importantly easy to understand.

### 3.2 HCI principles

A good interface is not simply something that makes working with the computer easier. The key to good interface design is to "empower" the users by giving them control to perform functions based on their particular needs [B2]. The field of HCI has many general principles and heuristics that can be applied to interface design.

#### 3.2.1 Usability

It is important to follow usability heuristics, thus ensuring that the interface is user-friendly. Nielsen (1994) [W9] proposed the following ‘Ten Usability Heuristics’:

1. **Visibility of system status**
   
The system should continuously inform the user about what it is doing and how it is interpreting the user’s input through appropriate and immediate feedback. Microsoft Windows illustrates the transfer of files by using a progress bar and animation showing paper moving from one folder to another.

2. **Match between system and the real world**
   
The system should speak the users' language, with words, phrases and concepts familiar to the user.

3. **User control and freedom**
   
User should not be too constrained by the system. For example, the system should have clearly marked exits which offer the user an easy way out of as many situations as possible, including ways to undo and redo.

4. **Consistency and standards**
   
The same command or action should always have the same effect. The user should not have to wonder whether different words or actions mean the same thing.
5. Error prevention
Where possible, the user interface should be structured to avoid error situations. This is one of the most important factors when designing for children. One method can be to ensure that users never lose their work as a result of error on their part, or any other reason other than the completely unavoidable, such as sudden loss of power to the client computer.

6. Recognition rather than recall
The system should minimize user memory load by taking over the burden of memory from the user. For example, a button needs to have an image that gives more information about its function so that a child can immediately recognize its purpose instead of having to memorize it with repeated use.

7. Flexibility and efficiency of use
The use of accelerators may often speed up the interaction for the expert user but it should be such that the system is able to cater for both experienced and inexperienced users.

8. Aesthetic and minimalist design
The screen should not be cluttered with unnecessary tools or messages.

9. Help users recognize, diagnose, and recover from errors
Error messages should be simple, indicate the problem concisely and constructively suggest a solution.

10. Help and documentation
It was observed by many researchers that, in general, children turn off the help systems in most software products because they were disrupting game play. Thus, it is important to increase the relevance and decrease the intrusiveness of the help system.

When evaluating the final product these points should be taken into account to ensure that they have been fully addressed. If testing highlights any of the above problems action will be taken to ensure the interface is revised.

3.2.2 HCI for children
Most of the HCI research has generally been targeted for adult applications. However children have needs that are different from those of adults. For children, most of the research done to date has focussed on designing educational software, and evaluation is primarily of learning outcomes, not usability. Usability is similarly important for entertainment, communications, and other applications, especially operating systems. In order for the
interface to be effective and appealing, it is necessary to understand the heuristics involved when children use computer technology.

How are children different?
Children do not just lack knowledge and experience, but also fundamentally experience and understand the world differently than adults. The Swiss psychologist Jean Piaget was a leading figure in analyzing how children’s cognition evolves (Piaget, 1970) [W8]. He divided children’s development into a series of changes:

- **Sensori-motor (birth – 2 years).** Children’s cognition is heavily dependent on what their senses immediately perceive. Little interaction can be expected from the child.
- **Pre-operational (2 – 7 years).** Children’s attention span is brief. They can only hold one thing in memory at a time. They think concretely and have difficulty with abstractions, and cannot understand situations from another person’s point of view.
- **Concrete Operational (7 – 11 years).** “We see children maturing on the brink of adult cognitive abilities. Though they cannot formulate hypotheses, and through abstract concepts such as range of numbers are still difficult, they are able to group like items and categorize.” (Schneider, 1996) [W7].
- **Formal Operational (age 11+).** By the time a child reaches this stage, their thinking is similar to adults, but their interests and tastes remain different.

These stages can be used as a basis to establish which age group to target and what features to provide for them.

The differences between an adult and child can be explored by looking at several basic characteristics:

**Psychological**
Children have different attention and interaction patterns compared to those of adults. Children are easily distracted, and are drawn to arrows and other such basic objects. For example, if there are big buttons with arrows on them, they are going to click them; a red stop sign would mean stop [W5]. Animated buttons can be as distracting to children just as they are to adults. When animating on rollover, children will often not hover over a button long enough to see the animation. However, this feature should still be implemented as it makes the interface more attractive and appealing to children.
Another difference between adults and children is the mental models that they form. Many user interfaces are based on metaphors from the real world – something that the user already knows. A popular metaphor is that of the desktop, but Jones (1990) [W10] says that this is not appropriate for children, as they include icons for such things as file folders and in-out trays. Jones argues that for a child to understand what these items represent, the child will need some knowledge of the office environment, which most children do not have. However, providing the icons are kept simple and intuitive, such a desktop could succeed with children.

Certain facilities such as Search and Index are available on most interfaces and are key functions for adults, but would not be useful for children. This is because it may involve too much reading, and may be difficult to understand. Children like to be in control of their world - and this means learning things as quickly as possible. When there is something to do that is meaningful and makes sense, children are more likely to sit there and pay attention. However, if the interface offers little control, they are likely to lose interest and patience quickly [B1].

Other basic psychological differences between children and adults include:

- Children like sound while adults want their computers to be quiet, so as not to disturb them or their co-workers [W14].
- Children like fantasy in their software whereas adults prefer reality and utility [W5].
- Children do not mind repetition unless it is complicated, while adults do not like it at all. Consistency is very important as children learn by recurrence [W5].
- Children are naturally in a learning mode while adults feel they are past that stage in life [W13].
- Children enjoy creative activities like colouring [W12].
- Children are more likely than adults to work with more than one person at a single computer [B1].

**Physical**

Young children’s motor control is not equal to that of adults (Thomas, 1980) [W11], and they are physically smaller. Devices designed for adults are difficult for children to use. Dragging and double-clicking are skills that take time to develop. Dragging requires continuous pressure on a mouse button and accurate destination targeting. Compared to adults, children have smaller hands and thus greater difficulty holding down the mouse button for extended periods and performing a dragging motion (Stronmen, 1994) [W8]. This makes things like marquee selection very problematical (Marquee selection is a technique for selecting several objects in one go using a dynamic selection shape. This is done by clicking on the screen on
the initial, static corner of the selection shape and dragging the mouse diagonally towards the opposite corner of the shape). A badly placed initial corner can make it difficult and sometimes impossible to select/encapsulate all the objects.

Knowing when to double-click is not obvious to the user and difficulties in distinguishing timing differences between two clicks and double-clicks exist. Also, users often erroneously assume that all actions require double-clicking. Children often have trouble double-clicking, and their small hands have trouble using a three-button mouse. Children will have problems with identifying the motion of on-screen cursor with the mouse’s movement and clicking the correct button on the mouse – most will click the right hand button first. This leads to an increased possibility of incorrectly clicking on an unwanted option (e.g. Delete).

As with adults, point-and-click interfaces are easier to use than drag-and-drop (Inkpen, 2001; Joiner, 1998) [W7]. With children though, it may be useful to have large clickable areas. Strømnen (1998) [W8] notes that since young children cannot reliably tell their left from their right; interfaces for children should not rely on that distinction. Instead, options such as left-clicking and right-clicking should perform identical functions.

Similarly children have trouble holding down more than two buttons at a time, for example, having to press CTRL+ALT+DEL, since their hands are too small to stretch across the keyboard.

**Other HCI features for children**

**Visual**

Children are continuously developing their reading skills presents, and hence the text presented must be of the appropriate reading level for the target population. Children who can read, understand simple words like *Go, Next, Back, Stop*. Larger font sizes are generally preferred. Bernard, Mills et al. (2001) [W10] found that children 9 to 11 years old prefer 14 point fonts over 12-point. Another possible technique is to convey messages in a pictorial format, as this may be easier for the child to understand.

**Audio**

Speech recognition has intriguing potential for a wide variety of applications for children. The ability for a child to simply say a command, and for the user interface to recognise and execute it, could greatly aid a child to carry out a task correctly. However, would a child be able to recall all the commands?
Children like sound effects and audio instructions. Therefore, another option is to give an audio output every time the user executes a command, for example, when opening a Microsoft Excel document, the interface would give an audio output specifying which file was being opened.

### 3.3 What do children want?

With research methodologies of contextual inquiry, technology immersion and participatory design, Allison Druin, a renowned researcher in the design of children’s technology, and her team came to three conclusions in what children want in technology experiences. These were as follows:

- **Control:** The nature of being a child is such that they are dependent on others. Children are empowered when they feel in control of their environment and when they feel they “own” the environment. With new software that offered children limited paths of interaction, children often became bored and uninterested. When software offered options for varied interaction, children spent a considerable amount of time exploring and actively engaged.

- **Social Experiences:** Children naturally want to be with other children. They generally do not create in isolation: they want to share, show, and use technologies with others. It is important for children that their tool offers social opportunities.

- **Expressive tools:** Children like telling stories, make up games, and build things. They enjoy many different forms of expressions: sound, visuals, movement, and physical appearance.

After researching several studies into the design of children’s software, further criteria and principles emerged that are absolutely essential, including:

- **Ease of use:** Research by Microsoft usability engineers demonstrated that ease of use is a critical determinant of engagement and as such is key to every child’s product if it is to be a success [B1].

- **Fun to use:** They do not like something that is tedious or boring. They prefer something which is eye-catching, and which flows at a speed that they are comfortable with.

- The system should support a degree of fantasy, which can be accomplished partly by the absence of text and distracting widgets such as menus.
• **Challenging**: The design should be inherently interesting with an element of surprise, and should give the child a sense of satisfaction after accomplishing tasks.

• **Dependability**: Adults expect things to break but children expect things to work, young children do not have patience for systems that crash and reboot. The system should act in a predictable manner and the child should be able to easily figure out how to do things that he/she wants to do.

• **Responsiveness**: Children expect immediate response and feedback or reactions.

### 3.4 What do adults want?

Children use computers both at home and at school. Researchers say children learn very differently when using computers at home than they do in school. The use of computers at home tends not to include educational software, certainly at the younger ages – but children did write stories, draw, and play games on computers for fun and learn through “playful discovery” [W3]. At home, children are surrounded by rich resources. They can use a variety of software, read magazines, and ideally, are able to ask parents and friends for assistance. However, in practice it is more likely that a child would be working on a computer alone and without the necessary guidance of an adult. From this, we can see that software for home use should be designed such that, if parents do not have the time to spare, children should be able to teach and use it themselves, resulting in a richer, independent experience, whilst giving parents the peace of mind that the software will not allow the child the opportunity to meddle with other parts of the system.

Parents want a safe environment where their children are able to explore his/her world on the computer. They want to be able to provide a selection of educational and fun places for their children to learn and discover on the web, without all the fuss of a filtered adult search engine and places where the children will not be immediately accosted by web garbage. Parents are worried that computers will provide children with inappropriate content. Children can easily access violence and other inappropriate content by using the Internet. While filters may be able to block most violent and pornographic content, they cannot do much to ensure that children access valuable and accurate information, or that they avoid websites that bombard them with advertisements for toys and candy (Alliance for Childhood, 2001) [W15]. Parents want their children to be provided with a curated collection that is accessible through developmentally appropriate interfaces and is large enough to sustain children’s interests.
At the same time, parents want their children to use an assortment of software, both educational and fun. Thus, they want to be given the option to easily update and change to cater for the child’s personal needs.

In conclusion, they want their children to be able to roam free and learn how to operate their computers.
4. Design

Before coming to a decision on the design of the application, it is necessary to specify the target audience. Thereafter, we discuss the design and the issues involved, taking an in-depth look at the justifications for each component.

4.1 Target audience

The application will be aimed at the age group between the pre-operational and the concrete operational stage, since in general; this is when they are initially exposed to computers. More specifically, the targeted audience will be between the ages of six and ten. This age range was chosen on purpose as their experiences in school will make them want to learn new things, but can be quickly discouraged if the system proves to be too complex for them. Their aspirations are creative and ambitious. Our goal is to provide an easy entry into using computers. However to the extent possible, we will try to create a system that will provide them with a seamless transition to a more common Operating System at a later stage. An added advantage of children in this age range is that they are relatively easy to include in software usability testing and it is hoped they will give honest answers.

**Breakdown of tasks we envision the child to carry out**

Increasingly, young children observe adults and older children working on computers, and they want to do the same. They want to be able to do their homework on a computer on their own without constant supervision. Children love expressing themselves by drawing pictures and writing stories. We would like to see the children confidently using a word processor and a paint program for their homework and other recreational tasks.

When given access to a computer at home, children tend to spend a lot of their time playing games, especially nowadays when the market is flooded with an extensive choice of PC games.
Children who are geographically confined to a particular area because they do not have the means to travel, can, using the Internet, visit far away lands that they might not have been able to visit otherwise without the computer. Thus, a browser to be able to use the Internet safely and securely is required.

Children who can read and write want to collaborate beyond their immediate classroom environment with children in other classrooms, or relatives in other cities or countries. With the potential of access to the Internet or other on-line “user friendly” networks, young children can use electronic mail through the Internet that facilitates direct communication and promotes social interactions previously limited by the physical location of their peers.

4.2 The Design

After examining the recommendation of researchers in the field of children’s software, we need to consider how all this information can be incorporated into a suitable system.

- Children are already bombarded with too much information, so the system should be made as simple as possible.
- Children have difficulty learning and understanding abstract and unfamiliar concepts, so the system should avoid these wherever possible.
- Novice users often have trouble remembering the names and/or locations of applications, thus the system should not expect the child to be able to do so, but try to encourage their learning ability.

The software should regulate the use of the computer, just like an Operating System, but with controls appropriate to children. The application, which will be called DeskToon, must install and work with any machine. This was done to prevent any inconvenience to adults; it is a lot more sensible to design a program that installs and works with the host operating system. This also takes into consideration that the majority of households own only one computer [W3], and most parents (or at least those who are not computer literate enough to) would not like to partition the hard drive so that it can run two or more operating systems.

The benefits of Direct Manipulation make its use the best solution for children. A colourful interface with pictures and sound, which this type of interface can offer, would captivate a child and hold their interest. It is important that this personalized Operating System interface makes the child feel good, comfortable and empowered when using the entire system. The "feel" has to be a mix of powerful, intuitive simplicity that should also scale with the users knowledge (c.f. Section 3.3)
By incorporating the use of Direct Manipulation, DeskToon can build on the human visual and cognitive capabilities and allow, with a certain level of abstraction, to leverage knowledge about real world things and their behaviour for the interaction with a virtual environment or tool. They can provide visual clues about possible features and choices, deal with more complex interfaces and provide a lower learning curve. We shall use the heuristics previously discussed as a basis for design guidelines for the interface.

While there were many possible formats that could have been used in designing the child-friendly environment, it was thought to be important to stick to the concept of the ‘desktop metaphor’. DeskToon can be configured to auto-start, when the computer is switched on, to provide constant hard drive security or as a stand-alone application. After the child has switched on the computer and launched their personal user profile, the application will start automatically. After logging on, the child will see a desk, which is intended to convey the idea of work (i.e. a computer is not just for play, but also for work).

There were several key design goals, but usability and security issues were given precedence over other objectives. Usability issues included making the application as child-friendly as possible through ease of learning and ease of use. The ability to start working productively within a few minutes, and to be proficient without the need for a manual or ongoing help from an expert would give the child greater self-confidence in the use of computers. Security is maintained by giving full administrative rights to the parent, who can then dictate what the child is able to do. By limiting the child’s access to only their personal workspace, it maintains the integrity of any important files located elsewhere.

HCI research shows that children are able to recognise and remember icons more easily than commands. Many of the general principles are encapsulated by the use of icons – the foreground metaphor for focusing attention, imagery and prompts recognition rather than recall, which is one of the heuristics discussed in Section 3.2.1. Icons are compact, and contribute greatly to clarity and attractiveness. The design principles for ‘meaningful’ icons include context of use, its representational form (resemblance, exemplary, symbolic, arbitrary), and its underlying concept. However it is important to be careful when creating icons, as poorly designed icons are not intuitive. The icons must be designed to allow the user to match between the system and the real world, which meets the second usability heuristic (c.f. Section 3.2.1).
Graphical icons are used to represent applications and commands such as open, save, new and exit. As stated in our research, the icons in DeskToon are big and have large clickable areas with a three-dimensional effect. To indicate to the user that the mouse is over the icon, some animation effects are added to the icon, and text appears in large letters to explain to them what the specific icon stands for. This meets the HCI usability heuristic of maintaining the visibility of system status (c.f. Section 3.2.1). Doing so, will hopefully encourage the child to investigate what all the other icons stand for before deciding what they want to do. Further aid provided for a child, is by having the cursor change to the shape of a hand to alert them that they are over a clickable area. This method is already used on the World Wide Web and has been proven to be extremely useful.

We decided not to use pull-down menus or drop-down menus, since they consume screen space and children find it difficult to navigate through them due to categorical ambiguity. With children, when an application is minimised, they tend to forget about it, or at the very least lose interest in it. Window management is confusing. Many children do not always realise that overlapping windows represent a three-dimensional space and they assume that hidden windows no longer exist. The expression “out of sight, out of mind” springs to mind. As a result, components like taskbar and functions for minimising and maximising screens, which can be found with most Operating Systems, have been discarded from our interface. Since this also means that children are not likely to switch between windows; they will only be allowed to open one window at a time. Other features like keyboard shortcuts such as ALT+TAB or CTRL+ALT+DEL have also been removed since children may have trouble stretching their fingers on the keyboard.

We also decided to make the interactions with the mouse as simple as possible. As mentioned earlier, children have trouble with double clicking. This problem will be tackled by using a one-click interface (which would mean that the children get an immediate response to any action) with little dragging and the right mouse button is disabled so that it does not have any functionality. This could have also been met by Windows, which can be customized to meet this requirement. However, as adults are already accustomed to double-clicking, having to change their habits can cause them some annoyance. It was not possible to duplicate the actions of the left and right mouse click due to the limitations of the software used.

All users make mistakes and appropriate error messages are needed. DeskToon will have as few but sufficient, simple error messages where required. Some situations where an error
message will be required are whilst trying to open non-existent files, or when a child clicks on New before saving his current work. This is an example of the error prevention heuristic.

The following section describes the set of interconnected components that comprise a ‘desktop environment’.

1. A **workbench** that is a base for all the things that is possible for the child to do.
2. A **Notepad** for doing their homework, writing essays or stories.
3. A **Paint** program that provides a range of tools that a child can use to draw and paint with.
4. A **Browser** with controlled access to the Internet.
5. An **Email** program that allows them to send and receive emails to a specified group of contacts.
6. An area from which they can access their **games**.
7. A section from where they can access all their **external applications**, which have been approved and installed by their parents / administrator.
8. **My work** which will give children a list of all their stored work, and enable them to delete or open their work.
9. Other useful accessories, such as a **calculator** and the **date and time**.

In addition, it will provide a special login mechanism for adults where they can control and personalise their children’s environment. For the child’s login, they are presented with personalised icons that they click on to enter their workbench. No passwords are used as children are likely to forget them, causing unnecessary hassle for the adult administrator.

### 4.2.1 Workbench

Following the login process, children enter their **workspace**. After considering the many possible ways in which to design the workspace, such as a palace where each room represented a different function, it was settled on using a single room containing a desk, which will have items appropriate for the children placed on top of it. These items will be icons that the children can relate to and recognise easily. Clicking on an icon will launch a component provided as part of the system. The components available are the notepad, paint program, web browser, and an email facility. A section is also provided to house external applications. There will be an analogue clock and a calendar at the top left hand corner of the room. This was placed here based on studies of eye-coordination, which state that the first place anyone looks at is the top left-hand corner [B5]. An analogue clock was chosen so children can learn to tell the time and give the room a more ‘homely’ look and feel.
The interface is very visual, avoiding the use of text as much as possible and therefore reducing the cognitive load.

### 4.2.2 Notepad

It is essential to have a notepad as part of an Operating System for many reasons. Children write freely on the word processor. They quickly learn to use word processing software and often write syntactically better work than with pencil and paper. Writing and revising can be difficult as children struggle with letter formation and fine motor skills. A word processor lets them focus on ideas, and more easily compose and revise text. This encourages children to view writing as a process, and to refine their work [W14].
Compared to Microsoft Word and WordPad found on Windows systems, it offers less functionality, but meets the basic requirements for the child. This was done so not to bombard the child with too many options. By providing them with this basic application, it is hoped it will give them the confidence to later move onto more advanced word processors.

The workbench will typically contain an icon, which is a piece of paper with a pen to write on it. On clicking this icon, the notepad will be launched. This will be divided into two segments; on the left hand side of the screen are buttons that enable the child to carry out normal functions such as opening a new or existing document, saving documents or exiting the notepad. Similarly to above, this was done as this is the location that users tend to notice first, and this theme was consistent throughout all the components. This was done to conform to the consistency and standards heuristic (c.f. Sections 3.2.1). The other segment is a large text area where they can type in their stories or homework in a large font.

Upon clicking on the ‘New Document’ icon, a pop-up message asking whether the user wants to save is displayed to avoid them losing any of their work. After the appropriate response, a new blank document is created. The save button prompts the user to enter a filename (if the file has been previously saved, the existing filename is displayed by default). If the filename already exists, the user is asked to confirm whether they wish to overwrite this file. For the open button, if any files exist, a list of filenames is presented to the user and they are able to open one of their choice. If no files exist, an error message informing them of this is
displayed. The exit, after checking whether the document has been saved, returns the user to the workbench. These command icons have the same functionality in all of the components.

4.2.3 Paint

The Paint program allows children to be entirely creative, and has excellent tools for developing fine motor skills. Creative and interactive software contribute to the effectiveness of an early childhood arts program, providing another media for creative expression. It can help children express their imagination in ways not possible by other materials. Children are able to select a range of exciting colours from the palette to use in their work, and there is no mess! Drawing on a computer can give children the ability to draw in any colour they want, on a large canvas, where they can use special effects that would never be available on paper.

The icon to launch this is a pot of pencils and paintbrushes, representing the tools by which children create paintings. Again, the screen is split into a drawing area and a segment that contains the options such as Save, Open and Exit. In addition, there is a tool palette instead of menus to provide a concrete visual interface that is very easy to learn for young children.

Paint provides several tools to the user. They can change the tool they are using by clicking on an unused tool. Children can choose from a pencil, a straight line, a curved line, a circle, a square and a polygon ranging from a triangle to a dodecahedron, and an undo button. The research showed that it was important to give the user control and freedom of a system, and this is met by the provision of the undo functionality.

The children are also allowed to generate whatever colour they want for their lines or as a fill-in colour. By moving any or all of the three separate sliders (representing red, green and blue) from left to right, they were able to generate any colour they want. They were also able to pick whether they wanted a linear, gradient or normal fill style. This introduces them to the idea of shading and gives them more freedom in their pictures. In addition they were also given an option to change the thickness and opaqueness of any object in the drawing.

This Paint component is very different from the paint application incorporated within Windows. DeskToon’s icons have larger clickable areas, which were designed to cope with children who are not very dexterous with the mouse since they have smaller hands compared to adults (c.f. Section 3.2.2). Paint also gives them a degree of freedom in that they can easily make their own mixture of colours, whereas the corresponding ability in Windows Paint application is found through a mixture of menu items and buttons.
4.2.4 Browser

According to NOP, three-quarters of British children aged between 7 and 16 now have Internet access. Twenty-three percent of young users go online for exam revision, while 53 percent of 15 and 16 year olds do so. Nine in 10 of all users aged between 7 and 16 say the Internet helps them with their learning [W16]. This shows that it is absolutely needed that children are provided with a safe and secure browser.

Connections to the Internet can give children the ability to access content far beyond what a school library can provide. Just imagine being able to read a one of a kind book that only Kings in a far away country had access to; or being able to see videos of that animal that so intrigued you when you read about it in a book; or having your favourite book being read by its author. Internet connections can also provide children with opportunity interact with family, friends or other children and teachers anywhere in the world. Children in the United Kingdom and Brazil could connect to learn about each other’s cultures.
However, at the same time, the Internet provides access to pornography and other such websites containing inappropriate content. To counter this, the browser only lets the child navigate to pre-specified web pages.

The children can browse by clicking on an icon representing the globe with WWW written on top of it. WWW is synonymous with the Internet, and the globe represents the world. To the child, this means they can explore the world.

To go to a particular site, the child has to click on the button with the name of the site on it. They are not allowed to manually enter the site address. This is to ensure that they can only visit ‘safe’ Internet sites. Like the Windows Start menu, these sites will be divided into different categories, such as sports, entertainment, etc. This is to avoid having a long continuous list. It was necessary to ensure these categories remain as simple and intuitive as possible since we have already said most of the items on the Windows Start menu have no meaning to children. The Browser also includes back and forward buttons to give the user more freedom in the way he/she ‘surfs’ the Internet. This ensures that the user is not too constrained by the system; a point raised during the HCI research (c.f. point 3, Section 3.2.1).
4.2.5 Email

Children could send the artwork they create in class to a parent in a humanitarian mission halfway round the world. They can write to their pen pals more often. Email provides a motivation for children to write well because they know that others will be reading their work.

By providing children with a personal email account, we are exposing them to the risk of being bombarded by spam advertising inappropriate products and services. Four out of five children receive inappropriate spam e-mails touting online drugs, get-rich-quick schemes and pornography, a survey has suggested. Internet security firm Symantec found that more than 80% of those interviewed received inappropriate spam on a daily basis, with half saying it made them feel uncomfortable and offended [W17]. However, locking any email sent to the child from a friend or family unknown to the parent can prevent this. These emails can be unlocked at a later stage once the parent has approved the sender.

Children can go to their email account by clicking on a ‘Post Box’ icon. To them, this means sending mail and in a way relates to email. It will also indicate to them that sending an email is a form of communication.

A simple and straightforward interface is used so children can enjoy the thrills of email. Again it is divided into two segments; one area for enabling functionality such as checking for new mail, composing mail, an address book and exit. The other segment is the work area itself to write their emails and replies. Commands to send, add attachments, delete and reply to emails are at the bottom of the screen so as to guide them in a step-by-step way that they can only send an email after it has been written. The address book will consist of names and email addresses of approved contacts. They can enter email addresses directly from the address book by clicking on the name or address.

No save email button was implemented as research shows children generally concentrate on doing one thing at a time, making the save button largely unemployed, and would only serve to clutter up the screen, thus avoiding contradicting the aesthetic and minimalist design heuristic.
4.2.6 My work

Using Windows Explorer can also be very confusing to children. The hierarchical organisation of files and folders is difficult to understand because it is not as obvious as in the real world.

A typical scenario where a child would be bemused is when he/she is using an application, say WordPad, and creates and saves his work. Most applications tend to save the file in the last folder accessed. A few days later, when trying to retrieve the file, he is unable to find it as WordPad looks in a different folder to that where the file has been saved, and the child is thus unable to locate his work. This can become very frustrating for the child, and the parent, who may find that their meticulously organised work has been mixed up with their children’s work.
Therefore ‘My work’ is implemented to allow children to view a list of all their work stored on the machine. The list has no hierarchical structure to it. A scroll bar is used to browse up and down through the list of files.

We thought about removing the delete function completely from the child’s desktop, but if a child is using the computer often, he/she is bound to create a lot of files, which will eventually become redundant. This could become annoying for a parent who would have to consistently check that there is enough space for the children to store their work! Therefore, the child needs to be allowed to delete. A rubbish bin is provided for this functionality. To solve the problem of accidentally deleting important files, the child is only allowed access to his/her files only, and they cannot add/remove new applications on DeskToon. This is solely the responsibility of an adult.
4.2.7 Games

The way to any child’s heart is to provide them with some games! Most children will own PC games but as with most Operating Systems, we must provide them with some games. Solitaire on Windows was designed not just for the joy of the game but also to learn how to better handle the mouse. In the same way, we will design a Tic-Tac-Toe, which will help the children improve their dragging and dropping skills. This will be implemented by letting them drag and drop the noughts and crosses to a box on the grid. This program will not allow them to cheat and move when it is not their turn. If not placed entirely on the box or if placed on an occupied box, it bounces right back to the original location and the child will have to drag and drop it again.

Research shows that children work better in pairs and like to share and use technology with others. Therefore, we decided to cater for both one player and two players.

Figure 4.7: Tic-Tac-Toe

Indicates whose turn it is

Shows counter being dragged into position
4.2.8 Administrative functions

The final component of the system allows the parent to perform administrative functions (see Figure 4.8). The parent is able to access this component by pressing CTRL+ALT+ENTER from the child’s workbench. This was used because the chances that a child would press these keys together are highly unlikely, and our research has shown that they would find it difficult to do so. Even if they managed to, a login and password must be entered before access is granted to the administrative section. From here, parents will be able to create new accounts if they have more than one child, control what the child has access to, what applications and games the children are allowed to use. They will also be able to manage which pre-selected websites they want their children to browse, or whether their children should have access to the entire World Wide Web.

The parent has the option of approving every contact in the address book, so that the child can only send/receive emails to people that the parent approved. This way, they are able to monitor to whom their children can send and receive emails from. Once a sender is approved, their email address will be added to the child’s address book, so that future emails from these senders can be sent / received with no problems. Thus parents are able to prevent any unsolicited emails reaching the child. However, this may involve extra effort for the adult, as they will have to regularly check if the children have any locked mail. To address this, we decided to make the screening of emails optional, i.e. the parent can disable this option.
Figure 4.8: Administrative Section – Adding a new website

Tabs which correspond to the various components in DeskToon.

Allows user to choose which sites should be removed from the approved sites list.

This shows the category where the new site will be added.

To which children the changes will apply to.
4.3 Design challenges

Every step during the design involved trade-offs between making the application powerful enough that children would find it expressive and engaging and making it simple enough that they would not find it frustrating or confusing.

One of the main aims was to involve children as much as possible during the design process. However we were not able to do so, which meant all design decisions were based on our judgement and ideas recommended by research carried out, and not on suggestions made by the children themselves.
5. Implementation

This chapter looks at some of the reasons behind why the project was programmed in Macromedia Flash MX, and then gives at some of the software engineering techniques employed in the application.

5.1 Choice of technology

Object Desktop

Initially we decided to employ a product created by Stardock [W18] that was a program that allowed the user to completely customize the desktop. Object Desktop is a collection of graphics applied by a particular program to customize a specific element (such as the Start bar, the fonts used in menus, etc) of the Operating System.

Although the software offered much, after purchasing it, a lot of problems became apparent, many of which contradicted the many points raised in the specification. The program was very inflexible with what it would work with. For example, it would only work with the Windows Operating System – something we wanted to avoid. Also its graphics could only be read / stored in .bmp or .png files. Animation became a slow and tedious process, and after taking a long time to load, the application was very unresponsive to user input. There was very little documentation available to support the software, which created a steep learning curve.

However, perhaps the worst feature was that the program contained many bugs which often caused it to crash. This was undesirable as children would not have the patience or understanding to work with such defects, nor should they have to.
Why Flash MX
We then decided to use Flash MX instead. Flash MX is an application used by many
developers to create web-based applications, but it can also be used to build stand-alone
applications on a computer running Windows or Mac Operating System.

It was decided to use Flash MX for several reasons. A Flash application is a platform that
provides sophisticated interaction through a GUI. It is possible to create a purpose-built
usable interface, as the data is separate from the display mechanism. The display can be
customized and can create three-dimensional worlds with moving interaction. This has clear
benefits in terms of usability. Flash uses vector graphics and output applications can be
compressed to a very small size thus not taking up a lot of hard drive space on the parent’s
computer.

To run Flash MX applications, Flash Player 6 is required. This is freely available to most
browsers and all major platforms with support for Internet Explorer, Netscape, and Opera
being the most widely used browsers, and are compatible with Windows, Apple Macintosh,
Linux and Solaris platforms. Thus, it allows cross-platform compatibility and it will always
render exactly the same, wherever it is placed.

As Flash MX was previously used, it meant that there was a less steep learning curve. This
also contributed to the decision to use Macromedia Flash MX.

5.2 Technical overview
Action Script is the ‘code’ that is given to a Flash application, so that the objects within it can
be told what to do and when to do it. Action Scripting is an event-driven language, in that it is
set up to wait for something to happen and then react to this event.

Layers are like transparent sheets stacked on top of each other. Layers help organize the
artwork in the application. Editing objects on one layer do not affect objects on other layers.

The timeline in Flash MX relates to the elusive fourth dimension, time (see Figure 5.1).
Actions are used to control the timeline and are always targeted at the objects they are
attached to.
There are two main ways of breaking down design problems: top down or bottom up. Top down design routes involve looking at the overall task and breaking the problem down into smaller chunks, whereas bottom up design would mean starting by looking at the basic building blocks, adapting them and building towards the solution. We decided to use the top down approach since it is more functional and allows us to look at what we have to do at each stage. Since we had already planned what would comprise an ideal ‘Operating System for Kids’, it was decided to start with the implementation of the workbench.

Given that we will have many functions and effects that would require the Flash application to do many things at the same time, we would be working with many timelines, each one handling a bit of the problem, and all working concurrently. Since we were bound to end up with an enormous amount of code, making it difficult to structure the project or to add new functionality, we decided to use a modular approach. In this way, we separated components and then tackled a subtask at a time. For example, the notepad is programmed separate from the workbench and all the functionality of the notepad is self-contained. This approach applies to all the interconnected components on the interface.
These approaches to the design of DeskToon meant it became easily extensible. Adding new components was made simpler due the modularity of the design. By using the top down approach, it was straightforward to integrate the new segments into the overall application.

For the general functions such as open, exit and save, a black box programming approach was used (Black box programming is code that can be re-used without necessarily knowing all the details of how it works). This makes the individual functions portable in that they can be used with the paint and notepad and does not require the code to be rewritten.

![Figure 5.2: Overview of the system architecture](image-url)
Each component was created in a separate flash file and tested thoroughly before being integrated with the workbench. This method allowed for loose coupling between the different components. However, when integrating, we found that it was impossible to pass variables between different files. For instance, if a document had been created in Notepad, it was not possible to update My Work accordingly. To get round this, we decided to embed each component within a single Flash file, which allowed us to then pass data between components and still preserve the modularity.

Within each file, the code for each object was, by default, made private, which meant it was not possible to pass variables to between objects. This was solved by writing global functions which applied to the whole timeline and the required function was called by an object whenever necessary. (Sample code can be found in Appendix A)

To extend the application so that it can send and receive messages from its host environment, namely the Windows Operating System, the "Exec" fscommand in Flash can be used. However, due to the changes between Flash 5 and Flash MX, this command has been severely crippled for security reasons and it does not allow passing arguments anymore. By making use of a third party tool, Flash Studio Pro, we were able to make use of the functionality provided by "exec", and thus were able to communicate with the host environment. This allowed us to save and retrieve .txt and .bmp files. We were also able to create a working Browser.
6. Testing

The chapter describes the method and the outcome from user testing, as well as evaluating technical testing.

6.1 Method used

During April 2003, the application was tested with Jamie O’Neill, Charlotte and Fiona Ingram in the Department Of Computing, Imperial College. A further testing session was carried out on 6th June 2003 with children from the Stag Lane First School to test how the application would fare in the school environment. In total, there were five girls and four boys. All nine children were within the target audience of six to ten years old.

At the school we asked for children who had at least some experience with the computer so we would not be spending our testing time teaching a child how to use a mouse. We also requested if the children could be in pairs of varying intellectual levels, i.e. below average, average and above average. This was in order to get a broader assessment of the application. Approximately forty minutes were spent with each child.

When we first met the children, we tried to establish a relationship with them by engaging in some ‘small talk’ to make them feel comfortable and relaxed. To get an idea of the child’s experience with a computer, and to get them accustomed to it, we let them use the Paint application or the Word Processor on Windows OS at the beginning of the testing session. We gauged their understanding of these applications and noted where they had problems and what they were confident with. After about ten minutes, they were shown the new application. The criteria used to judge the application were:

- Where do they excel?
- Where do they become engrossed, animated, excited?
• Where do they struggle, sigh or get frustrated?
• Where do they lose interest?
• Where did they ask for help?

The children were asked to carry out a series of tasks, which included creating a painting, typing a few lines in the Notepad and then saving and re-opening this file, sending an email, browsing the Internet and playing Tic-Tac-Toe. This was done so we could judge how suitable DeskToon was for children. Thereafter the children were given a free run of the application. This was to investigate whether the children wanted to explore some more, reuse components like Paint or the Browser, or were tired and bored and not sure what to do.

When conducting the series of tasks, we made sure to switch the order around for different children so that the same tasks did not always come to the end of the test when children are tired. To keep the children encouraged, they were constantly offered generic positive feedback. Statements like “you did all that in your own!” or “that’s really pretty” were used to help keep the children motivated.

Initially, it was decided to get the children to fill out a questionnaire, but when asked to do this we realised it was not practical as it took up a lot of the limited time we had with them and also the children only jotted down one-word answers. We managed to solve this by us asking them the questions and filling in their responses whilst they were carrying out the testing of the application. This produced better results, as they did not have to concentrate on writing answers and worrying about spelling mistakes or tidiness. At the same time, they could continue using DeskToon and have fun.

The questionnaire was designed to elicit a substantial body of information about the children’s use of computers and their opinions about DeskToon, without overloading them with too many questions. The questions had to be short, and written in simple English, so that the children could understand what was being asked of them. So that the flow of questions would be intuitive; we started by asking them their name, age and class. The next set of questions probed the children’s present access to computers, the frequency with which they used the computer for and the type of usage. The most important section was the questions on DeskToon. They were asked what they liked/ disliked about the application, what they would have preferred to have been different, and finally asked them to give an analysis of each individual component (The questionnaires can be found in the Appendix A).
6.2 Results
The outcome of the testing sessions proved to be very insightful. They can be grouped into two categories, namely the questionnaires and the observations of the children using DeskToon.

6.2.1 Questionnaire
The results from the nine questionnaires is summarised as follows:

1. Do you have a computer at home? If so, how often do you use it?
   - Only one of the children did not have access to a computer at home. All the others used computers on a regular basis, but more often during weekends.

2. What do you use the computer for at school or at home?
   - At school, the younger children used programs like TooSimple, which was a group of applications designed for use by children in schools. The older children were just beginning to learn how to use the Internet.
   - 89% of the children said that they used the computers at home for games; 67% children said they used it for painting or doing their homework.
   - One girl said that she was only allowed to use the computer when she was with her older sister.
   - Three of the nine children said that they were allowed to email their friends, and one of them was only allowed to check his email when his parents were around because “I get some funny emails sometimes.”

3. If you could design your own computer, what would you have on it?
   - This question was not well answered as the children did not seem to understand what was being required of them.
   - Pooja: “I want to have lots and lots of colourful pictures and I want to be able to do all of my homework on the computer.”
   - 67% of the children, especially the boys, wanted to have games while 60% of the girls said they wanted to be able to draw pretty pictures easily.
4. **What do you like about the new program?**
   - All of them liked the entire application as a whole. They commented on the fact that everything was so big and colourful. Nina: “I would never make the mistake of clicking the wrong icon.”
   - The games section, not surprisingly turned out to be the most popular. Some of the children were more specific, saying that they really enjoyed using the Paint. One boy enjoyed using the calculator.
   - Jamie: “I really enjoyed emailing. At school, I always forget my password and there are no pictures to tell me what to do. It’s all in words like ‘if you want to send email, click here’.”
   - They were thrilled at being able to use the calculator.

5. **What part did you dislike? How would you have done this differently?**
   - A couple of children reflected on the fact that they could not change the font, or the colour of their writing.
   - One boy said he wanted to be able to write and draw pictures at the same time.
   - Other suggestions were to have an eraser in Paint together with the existing undo button.
   - Whilst playing Tic-Tac-Toe, they found it annoying that player one always started.

6. **Did you understand what all the pictures meant?**
   - It was very impressive to note that all the children had no qualms about the icons. Each icon was recognised immediately by the children. They did not require any prompting or hints on what the icons stood for.

7. **For each of the following, write what you found easy/hard to use?**

<table>
<thead>
<tr>
<th>Easy to use</th>
<th>Hard to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td></td>
</tr>
<tr>
<td>Easy to enter addresses, as they did not have to remember the address.</td>
<td>Were not sure whether they had to click on subject or name to open an email. Recognising that the mouse pointer changed on clickable areas solved this.</td>
</tr>
<tr>
<td>Understood the error messages when clicking on send without entering a valid address.</td>
<td></td>
</tr>
<tr>
<td>Liked the spacious layout and easier to read their emails.</td>
<td></td>
</tr>
</tbody>
</table>
Games | This was the most liked part of the application. | Did not realise immediately that they had to drag and drop the noughts and crosses.
---|---|---
Internet | Liked not having to remember and type in web addresses. | None
Notepad | When using this, they remarked on how much easier it was with the larger fonts. Understood what the animated commands stood for. It was simpler for them to find and open their files. | None
Paint | Again, they like having the large buttons, which made it easier for them to pick the right tool. Some thought it was much simpler to click on the undo button rather than having to press CTRL + Z. | Found it difficult to change colours initially, but after experimenting, they became confident and thoroughly enjoyed the freedom of creating their own colour. Had a bit of trouble when they tried to erase, since Paint does not provide this functionality.

The results from the questionnaire suggest that the children thoroughly enjoyed using DeskToon. It supports some of the finding made in research which state children are eager to learn new technology as long as it is fun and easy to use [B1]. The children were always quick to point out what they liked / disliked, which we found useful as possible future improvements (c.f. Section 7.1).

6.2.2 Observational findings
To compare the impact DeskToon has compared to the normal Operating Systems used, we asked the children to carry out identical tasks using both systems.

Firstly, we observed the children using the desktop environment of a computer running Windows OS. We studied the children’s use of the mouse and keyboard as they used either Windows Paint or WordPad applications. Most of the children had used Windows Paint and Word Processor at home to draw pictures and do their homework respectively. Some of the
younger children had difficulty in controlling the mouse. They had trouble holding down the mouse and dragging for a long periods at a time, especially when drawing. Even though many of them said they had used Windows Paint, it was seen that they did not make full use of the many features provided by both Windows Paint and WordPad. When asked, they did not recognise many of the buttons on the tool palette. Some of the children got up from their seats to squint at the screen to try and recognise some of the smaller buttons and some of them hovered over the buttons until a tool tip appeared. While selecting a colour from the palette, there were several occasions when the children picked incorrect colours, mainly because the children were not able to control the mouse adequately enough to select the small squares on the colour palette. Another significant observation made was that many children could not hold down two keys at the same time, which confirmed the research studies.

Many of the children mentioned that whilst trying to open an application, they would end up with more than one instances of the application being opened. This proved that children did indeed have difficulty mastering the skill of double-clicking. Some of the children said they found the Windows Paint and WordPad dull to use. Another thing that arose from questioning them was that they were sometimes scared to explore on the computer, as they did not want to mess up their parent’s work.

A further area where children seemed to struggle was whilst navigating through the Start menu. Many of them did not know under which program item an application could be found. A poorly structured Start menu would confuse even some of the more experienced children. Ten-year old Jamie mentioned how cluttered the desktop on his parent’s machine looked, as they had made desktop shortcuts for all his applications.

These observations further confirmed the research that children do indeed have difficulties in using computers. Their problems were similar to those faced by novice users, but even though children are normally better than adults at picking up new technologies, they lack dexterity. The applications provided by Windows do not help them to significantly improve their control of the mouse.

When DeskToon was launched, the children’s initial impressions were very optimistic. There were lots of ‘oohs’ and ‘ahs’, which was encouraging. Each part of the application was well received, there being no particular section that was disliked or found to be boring. They were occupied throughout the entire testing session, and it was encouraging to see that they were completely engrossed in what they were doing throughout, especially given that children at
this age have a relatively short attention span. This showed that they really enjoyed using the computer with DeskToon acting as the OS interface.

We gauged how much children like a program by observing signs of engagement such as smiles and laughs or leaning forward to try things, and signs of disengagement such as frowns, sighs, yawns or turning away from the computer.

We were pleased to note that DeskToon had a positive visual impact on the children. As soon as DeskToon was launched, it was interesting to see the children grab the mouse and get ready to explore their ‘little world’. This showed that the colourful, attractive interface appealed to the children. However, they were not too happy to find out that they had to do some tasks before they were given a free run of the application, showing how keen they were to be allowed to discover what DeskToon had in store.

They seemed to excel in the usage of DeskToon as a whole, especially when using Notepad. Jamie said, “It is easier to write here rather than [on my word processor] at home. It is a lot easier to read as well…” This was because of its simple layout, and they preferred having command icons rather than text and complicated pull-down menus.

The email and game component was perhaps the parts that most excited the children. They seemed thrilled to be able to send and receive their own emails in a simple yet attractive manner. They enjoyed playing Tic-Tac-Toe against the computer and against each other. After drawing their pictures in Paint, they were also delighted to be able to print their work.

Many of the children said they needed to have the website addresses written down on paper before they began surfing the Web. By using the browser provided by DeskToon, they did not have to worry about making spelling mistakes. Instead, via the much liked sliding bar, they could find the website within the obvious categories. This definitely indicated a significant improvement over Internet Explorer and other commonly used browsers.

However, there were sections that the children found confusing, for example, choosing colours in Paint. The fact that it was not immediately obvious to them that they had to move the sliders to create a colour meant that it did not meet the initial objective of the children not requiring help at any stage. However, once it was explained to them how it worked, they enjoyed the freedom to be able to generate their own colours. They tried several combinations till they were able to achieve the colour they wanted. As they used Paint a bit more, it
became obvious that they remembered the various positions the sliders needed to be in for a particular colour.

Another unclear feature was highlighted when the children were playing Tic-Tac-Toe. They did not know that the playing pieces had to be dragged and dropped into position; instead they were clicking on the grid position and expecting the nought or cross to appear in the area where they had just clicked on. This was the only section where they seemed to search for instructions to guide them. Through our observations of watching them play other games on the Internet, we realised this was because the children were used to having to read instructions whenever starting a new game.

Thereafter, the children were given the freedom to do whatever they wanted. It was pleasant to see that none of the children were bored or uninterested. Neither did they appear to be unsure on what to do. The way the children reacted to this was different for each individual. Quite a few of the children returned to the workbench to explore further. They discovered that there was a calculator at their disposal. They also noticed the folder My Work, and were delighted to see that the work they had saved earlier could be found from here as well. Others generally went to Paint, email or the browser to check which other sites they were allowed to visit.

Jazz tried to send an email without specifying an email address. A message outlining this error popped up onto the screen. Jazz stopped to read this message and responded to it accordingly. This stressed the effectiveness of having simple error messages.

Another interesting observation was when one of the girls tried to click on the floppy disk, a boy pointed out to her that she could only click on icons when the mouse changed to the shape of a hand. This showed that they recognised these changes and used them to aid in their navigation of the application.

On being told that their time was up, they seemed genuinely disappointed and wanted to carry on.

6.3 Other testing

The above investigates whether the user can use DeskToon, but we also need to look at whether the user can break DeskToon. Children are notoriously good at breaking things, and we need to ensure that the system can cope with all types of user input. This is known as
**Monkey testing.** We tested the effect of the user clicking on wrong areas of the screen, or performing nonsensical operations, for example, trying to send emails, without specifying an address or opening non-existent files. In this instance, the user is informed that they have no files to open (see Figure 6.1).

![Figure 6.1: Pop-up error message displayed after attempting to open non-existing files](image)
7. Evaluation

This evaluates the findings and tries to explain them using the research carried out earlier. It also outlines suggestions made by the users.

The main objective of this project was to produce a system by which children could use computers more easily and safely, which includes providing hard-drive security and safe Internet browsing. Through DeskToon, we have managed to meet these specifications.

7.1 Evaluation of DeskToon

The children’s impressions of the application were mostly positive. It was inspiring to see that the children thoroughly enjoyed using DeskToon. They noticed what was ‘cool’. They found the application easy to navigate around and easy to learn. When asked whether they would prefer to use Windows OS or DeskToon, all of them chose DeskToon. This we believe is because of its bright colours, and animated graphics, ease of use and better setup.

**DeskToon is easy to learn and use**

The children’s feedback suggested that we succeeded in our goal of making computers easy for novices to learn and use:

Vaishally: “I don’t have a computer at home and we don’t use it much at our school that much. For my first time using this, I thought it was pretty cool and easy.”

Children want to be in control of their world as quickly as possible - and that means learning something quickly. If it is easy to learn they will quickly become immersed in the experience [B1]. This was definitely the case with DeskToon.

All of the children found it easy to interpret the error messages, which were displayed using large, brightly coloured fonts. Children did not mind having to read some words, as long they were not pages and pages of instructions.
**DeskToon helps with navigation and control**

The application’s structured approach allows novices to get a grasp of the entire system quickly and facilitates navigations and access to its segments. The intuitive icons meant that the children were recognising the icon’s functionality rather than having to recall what it did. This supported points raised during the research:

Jazz: “I like that I do not have to spend time looking for my program. With this, I know exactly where everything is and if I am not too sure I can just put my arrow over it to be sure.”

Charlotte: “Everything is there right in front of you…”

They found the layout of the application as a whole very helpful. DeskToon helps users in manipulating all aspects of the application, encouraging experimentation and thereby improving their usage of the computer. We felt DeskToon left the children more comfortable in their exploring:

Fiona: “It is better with the pictures there because I understand what goes with the pictures…”

**Supports creativity**

Components within DeskToon, such as Paint, permit more functionality in a simple way and give incentive to the children to be more imaginative. For example, a child can choose to draw n-sided polygon, and as discussed before, can create their own colour. With all these tools available to them, the children do not have to concentrate on how to do something but gives them freedom to express themselves.

Ultimately creativity lies within the person and the environment they work in. While such programs can inspire them to become creative, there is a limit to how big an effect it can have.

**DeskToon accommodates as many user groups within the target audience as possible**

The application has been designed to cater for as many user groups as possible. One such user group is colour-blind children who would not be able to enjoy some of the bright colours used in the interface, but this does not prevent them from using it, since there are very few sudden colour changes used to give feedback. Any time we used colour to convey information in the interface, we also used clear, secondary cues to convey the information to those who will not
be experiencing any colour coding. Secondary cues constituted a variety of actions, ranging from the subtlety of grey scale differentiation to having a different graphics or different text labels associated with each colour presented. We tested the software with children from a range of aptitudes, and the results were similar between these groups. This shows that DeskToon caters for children of all abilities.

To summarise our findings, we felt that the children found DeskToon extremely exciting and fun to use. Comments such as “can I stay a little while longer!” were common. Jamie [in response to question “would you like to use DeskToon at home?”]: “I would never get to use it because my [younger] sister will be stuck to it.”

7.2 HCI issues not implemented

There are several recommendations suggested by the HCI research, but which could not be incorporated within the scope of this project. Children like to collaborate with other users, and as of yet, DeskToon does not allow any form of networking or communication. The only area where children can work with other users is whilst playing the two-player version of Tic-Tac-Toe.

Our research states that, ideally, the right and left mouse buttons should perform the same functions, but due to the limitations in Flash MX, we were unable to duplicate the actions of the left mouse button. Instead, we disabled the right click menu, which meant options such as ‘Delete’ no longer appeared.

There are also recommendations concerning the use of sound as a method of feedback to the user. The current software does not allow this, but a fully working system should incorporate this feature.

7.3 Effect of work with children on our design

By watching, listening, and thinking about what we had seen and heard, we learned a lot that will allow us to improve the design. This prompted the following suggestions:

- The children requested new functionality - for example, the ability to play some music or to be able to watch a movie.
• Some form of visual instruction demonstrating how to play the game should be provided. The instructions could be displayed in a pictorial format, as suggested in the HCI research. Also, Tic-Tac-Toe should allow alternative players to start.

• Watching our users work suggested that as they became confident with the application, they would enjoy it more if as they grew older some more functionality were given to the internal components like Notepad.

• An eraser tool should be made available together with the undo function already existing in Paint. During the testing, we saw several children search for a way to delete bits of their drawing.

• Since the children often made spelling mistakes, there should be some method of checking their spellings.

• The children should be allowed to enter a website address since searching through a list can be quite tedious. However, the typed addresses must be in the list of approved websites. To assist them in typing in the address, the browser should have a sort of ‘auto-complete’ function, which suggests possible sites based on the user’s input.

• A messenger service should be made available so that children can communicate with parent-approved users in real-time.

• A search functionality provided in the Browser that looks for the keyword in the list of approved sites and returns links to matching websites.
8. Conclusion

This summarizes the project, and then looks at possible future investigations and enhancements that can be carried out.

8.1 Conclusions of the project

The key aims of this project were to find a way in which children can easily and safely use a computer. We looked at the type of interface most appropriate for children and then the HCI related concepts and features specific to children. These were incorporated into DeskToon to make computers attractive and appealing to children. With the successful user testing of DeskToon, an attempt to achieve these aims has been made.

DeskToon features include a notepad, paint, browser, email, calendar, and clock. These components entice children to explore and experiment with tools they see their parents and teachers using, whilst developing their literacy skills that can be further built upon through other classroom activities. It will help keep children out of places where adults don’t want them to be by creating a secure yet child-friendly environment.

Such a system must be easy and fun to use, evoking a sense of pleasure and accomplishment, not tedium and frustration. Children are sensitive to what they see, more than adults imagine. They care what something looks like just as much as how it works or what it does. They don’t want the visual look of things to question their intelligence. They want things that look good [B1]. As a result, the application must be visually attractive but at the same time easy to use. They also enjoy many other different forms of expressions like sound and movement. Children want to be in control of their environment. They do not want to be told what to do, but like to explore at their own pace. A system that restricts the children from having fun will not be received well. The system should work in a predictable way, making it obvious for the child to do whatever he/she wants to do.
Children are impatient, and expect an immediate response and feedback to their actions. There should be minimal errors, and when they do occur, the system should have a simple method of recovery. Children do want systems that crash and reboot. To reach a wide range of users, the application should be gender-neutral through its concepts, features and graphical elements. It must be designed so that it does not make it difficult for the child to learn how to use other Operating Systems like Windows at a later stage in their life.

DeskToon has been designed such that the addition of further components will be simple to integrate into the application. This is as a result of the software engineering approaches used during the implementation.

DeskToon demonstrates all these characteristics and others outlined in our research. From the success of the testing, we have verified that such an application would be ideal for parents who are worried about the safety of their computers but also want their children to use computers easily, learn and have fun.

Our findings from the testing verify many of the points raised during the research regarding what constitutes a successful system for children. One argument that was contradicted concerned animated buttons. The research indicated that these would be distracting for children, but our observations found that the children seemed to respond well to them. However, our testing was limited and was not enough to concretely suggest that the research was incorrect.

8.2 Future work

There are many areas that still need to be investigated further. The possibilities for DeskToon’s development are boundless.

Had we been able to get some children during the design stage, we would have been able to incorporate more of their needs and wants. More time is required to pinpoint the areas where the children have trouble using a computer, in particular when using a Direct Manipulation paradigm. Studies should be conducted with larger groups of children to gain more systematic information. Other areas to investigate further would be the hardware used by children. Children have trouble with mice and keyboards. Is there any existing of hardware made specifically for children? If not, is it viable to produce such hardware?
To assess whether DeskToon is a long-term success takes a lot longer than forty minutes of user testing. A real measure of success would be to see how long it would sustain in a home environment. Would parents be willing to carry out the administrative functions? Furthermore, measuring how long it takes before the children become bored with the existing functionality can only be done with time.

By turning the prototype software into a product, we hope that it can be widely used at home, and we can gather further evidence about the value of educating children in the use of computers.

One suggestion that would undoubtedly make DeskToon a better application would be to implement a simple Control Panel for the children’s exploration. Children should be able to "Add/Remove" only the programs that they have installed and parents should have the ultimate say in uninstalling a program after they have scanned a logbook. The Control Panel for children would also include child-specific desktop themes, backgrounds, or icons to personalize each child's desktop. This would give children more control and freedom.

Currently, when using the Browser, the user starts off in an approved website and is able to follow any hyperlinks embedded within the webpage. Whilst allowing the child complete access to related websites, it cannot ensure that he/she will not visit an unapproved site. A possible solution to this is to allow the child to navigate through web pages within the same domain of an approved site, and any other hyperlinks are disabled.

In conclusion, what has become apparent through this project is that children like computers, and would like it to be a more child-friendly environment and easier to use. It is now the task of Operating System developers to design and implement a completely perfect family program personalized for every child’s need, and at the same time satisfies adult’s fears concerning security and other similar issues.
Bibliography

Websites


[W17]. BBC news, (2003). **Children upset by spam mail.**

[W18]. Stardock. **Object Desktop**
http://www.stardock.com, (accessed on 8\textsuperscript{th} Jan 2003)

**Books**


Appendix A

A1. Sample code for drawing in Paint

onClipEvent(enterFrame){
    // Update X,Y positions and HEXcolor
    lastX = x;
    lastY = y;
    X = _root._xmouse;
    Y = _root._ymouse;

    // If penmode is set to free hand drawing (draw)
    if(pendown && penmode == "draw"){
        with(_root.drawingPlane["layer"+_root.layer]){/
            lineStyle( thickness, HEXcolorLine, cA);
            lineTo(X,Y);
        }
    } else {
        // If penmode is: line drawing, curve drawing or box drawing
        if(pendown && (penmode == "line" ||
                      penmode == "curve" ||
                      penmode == "box" ||
                      penmode == "circle" ||
                      penmode == "polygon")){
            _root.drawingPlane["layer"+_root.layer+"_"+(i-1)].removeMovieClip();
            _root.drawingPlane.createEmptyMovieClip("layer"+_root.layer+"_"+i, _root.layer);

            with(_root.drawingPlane["layer"+_root.layer+"_"+i]){/
                //Draws the thickness
                if(thickness > 0){
                    lineStyle( thickness, HEXcolorLine, cA);
                } else {
                    lineStyle( 0, 0x000000, 0);
                }

                //Checks whether fill-in type is solid, linear or gradient
                if(penmode == "box" || penmode == "circle" || penmode == "polygon"){
                    if(fillType == "solid"){
                        beginFill( HEXcolorFill, fA);
                    } else {
                        if(fillType == "linear"){
                            gColors = [ HEXcolorGrad1, HEXcolorGrad2 ];
                            gAlphas = [ g1A, g2A ];
                            gRatios = [ 0, 255 ];
                            gMatrix = { matrixType:"box", x: initX, y: initY, w: (X - initX), h: (Y-initY), r:0 };
                            beginGradientFill( fillType, gColors, gAlphas, gRatios, gMatrix );
                        } else {
                            if(fillType == "radial"){
                                gColors = [ HEXcolorGrad1, HEXcolorGrad2 ];
                                gAlphas = [ g1A, g2A ];
                                gRatios = [ 0, 255 ];
                                gMatrix = { matrixType:"box", x: initX, y: initY, w: (X - initX), h: (Y-initY), r:0 };
                                beginGradientFill( fillType, gColors, gAlphas, gRatios, gMatrix );
                            }
                        }
                    }
                }
            }
        }
    }
}
//Draws line

if(penmode == "line"){
    moveTo(initX,initY);
    lineTo(X,Y);
}

// Draws curve

if(penmode == "curve"){
    moveTo(initX,initY);
    curveTo(X,initY,X,Y);
}

// Draws box

if(penmode == "box"){
    moveTo(initX,initY);
    lineTo(X,initY);
    lineTo(X,Y);
    lineTo(initX,Y);
    lineTo(initX,initY);
}

// Draws circle

if(penmode == "circle"){
    difX = X - initX;
    difY = Y - initY;
    midX = (X + initX)/2;
    midY = (Y + initY)/2;
    offX = difX * 0.03;
    offY = difY * 0.03;

    moveTo( midX, initY );
    curveTo( X - offX, initY + offY, X, midY );
    curveTo( X - offX, Y - offY, midX, Y);
    curveTo( initX + offX, Y - offY, initX, midY);
    curveTo( initX + offX, initY + offY, midX, initY);
}

//Draws polygon

if(penmode == "polygon"){
    midX = (X + initX)/2;
    midY = (Y + initY)/2;
    difX = X - initX;
    difY = Y - initY;

    moveTo(midX,initY);

    for(s=0; s<(polypoints + 1); s++){
        nextX = (Math.sin(s * (360 / polypoints) * Math.PI / 180)+ 1) * difX / 2;
        nextY = (-Math.cos(s * (360 / polypoints) * Math.PI / 180)+ 1) * difY / 2;
        lineTo(initX + nextX, initY + nextY);
    }
}

- 68 -
endFill();
}
i++;
if(i > 10){i -= 10;}

// Show selected colour and HEX code.

HEXcolorLine = (cR << 16 | cG << 8 | cB);
HEXcolorFill = (eval(fillEdit + "R") << 16 | eval(fillEdit + "G") << 8 | eval(fillEdit + "B"));
HEXcolorGrad1 = (g1R << 16 | g1G << 8 | g1B);
HEXcolorGrad2 = (g2R << 16 | g2G << 8 | g2B);

if(fillType == "none"){
    myColorFill.setRGB(0xFF0000);
    _root.fBlock.gotoAndStop("nofill");
} else {
    myColorFill.setRGB(HEXcolorFill);
    _root.fBlock.gotoAndStop("fill");
}

if(thickness > 0){
    myColorLine.setRGB(HEXcolorLine);
    _root.cBlock.gotoAndStop("fill");
} else {
    myColorLine.setRGB(0xFF0000);
    _root.cBlock.gotoAndStop("nofill");
}

myColorGrad1.setRGB(HEXcolorGrad1);
myColorGrad2.setRGB(HEXcolorGrad2);

_root.cBlock._alpha = cA;
_root.fBlock._alpha = fA;
_root.gradMarker1._alpha = g1A;
_root.gradMarker2._alpha = g2A;

HEXcodeLine = "";
HEXcodeLine += (cR < 16) ? "0" + cR.toString(16) : cR.toString(16);
HEXcodeLine += (cG < 16) ? "0" + cG.toString(16) : cG.toString(16);
HEXcodeLine += (cB < 16) ? "0" + cB.toString(16) : cB.toString(16);

HEXcodeFill = "";
HEXcodeFill += (fR < 16) ? "0" + fR.toString(16) : fR.toString(16);
HEXcodeFill += (fG < 16) ? "0" + fG.toString(16) : fG.toString(16);
HEXcodeFill += (fB < 16) ? "0" + fB.toString(16) : fB.toString(16);

fillR = eval(fillEdit + "R");
fillG = eval(fillEdit + "G");
fillB = eval(fillEdit + "B");
fillA = eval(fillEdit + "A");

updateGradBar();
}
A2. Sample code from Tic-Tac-Toe

//Checks position in grid where cross is placed by player and updates array.

function place_cross(x_pos, y_pos, cross_name){
    insquare_cross = false;
    for (i = 0; i < 9 ; i++) {
        sq_cross = eval("b"+i);
        if (play_game[i] == 0) {
            if ((x_pos > sq_cross._x) & (x_pos < (sq_cross._x+45))) {
                if((y_pos > sq_cross._y) & (y_pos < (sq_cross._y+45))) {
                    insquare_cross = true;
                    play_game[i] = 1;
                    count +=1;
                }
            }
        }
    }
    return insquare_cross;
}

//Checks position in grid where circle is placed by player and updates array.

function place_circle(x_pos, y_pos, circle_name){
    insquare_circle = false;
    for (i = 0; i < 9 ; i++) {
        sq_circ = eval("b"+i);
        if (play_game[i] == 0) {
            if ((x_pos > sq_circ._x) & (x_pos < (sq_circ._x+45))) {
                if((y_pos > sq_circ._y) & (y_pos < (sq_circ._y+45))) {
                    insquare_circle = true;
                    play_game[i] = 2;
                    count += 1;
                }
            }
        }
    }
    return insquare_circle;
}

//To indicate which player’s turn it is.

function turn() { 
    if (myturn) { 
        player_turn = play_2 +"s turn";
    }
    else { 
        player_turn = play_1 +"s turn";
    }
}

//Checks for winner and which direction, i.e. horizontal, vertical or diagonal

function horizontal() {
    horiz = false;
    if ((play_game[0] == play_game[1]) & (play_game[0] == play_game[2]) & (play_game[0] != 0)) {
        setProperty("horz", _x, "97.5");
    }
setProperty("horz", _y, "160");
setProperty("horz", _visible, true);
horiz = true;
}
  setProperty("horz", _x, "97.5");
  setProperty("horz", _y, "285");
  setProperty("horz", _visible, true);
  horiz = true;
}
  setProperty("horz", _x, "97.5");
  setProperty("horz", _y, "405");
  setProperty("horz", _visible, true);
  horiz = true;
}
return horiz;
}

function vertical() {
  vertic = false;
  if ((play_game[0] == play_game[3]) & (play_game[0] == play_game[6]) & (play_game[0] != 0)) {
    setProperty("vert", _x, "145");
    setProperty("vert", _y, "112.5");
    setProperty("vert", _visible, true);
    vertic = true;
  }
    setProperty("vert", _x, "270");
    setProperty("vert", _y, "112.5");
    setProperty("vert", _visible, true);
    vertic = true;
  }
    setProperty("vert", _x, "395");
    setProperty("vert", _y, "112.5");
    setProperty("vert", _visible, true);
    vertic = true;
  }
  return vertic;
}

function diagonal1() {
  diagon1 = false;
  if ((play_game[0] == play_game[4]) & (play_game[0] == play_game[8]) & (play_game[0] != 0)) {
    setProperty("diag1", _x, "97.5");
    setProperty("diag1", _y, "112.5");
    setProperty("diag1", _visible, true);
    diagon1 = true;
  }
  return diagon1;
}

function diagonal2() {
  diagon2 = false;
    setProperty("diag2", _x, "97.5");
    setProperty("diag2", _y, "112.5");
    setProperty("diag2", _visible, true);
    diagon2 = true;
  }
//If no wins, and no grids left, checks for a tie

function tie() {
    empty = false;
    for (j=0; j < 9; j++) {
        if (play_game[j] == 0) {
            empty = true;
            return empty;
        }
    }
}

//Returns winner

function win(player) {
    if (count > 4) {
        h = horizontal();
        v = vertical();
        d1 = diagonal1();
        d2 = diagonal2();
        check_tie = tie();

        if (h|v|d1|d2) {
            score(player);
            player_turn = player + " wins!!!";
            return;
        } else if (!check_tie) {
            player_turn = "Tie";
            return;
        }
    }
}

//Updates score

function score(player) {
    if (player == play_1) {
        score1 += 1;
        no_wins1 = score1;
    } else if(player == play_2) {
        score2 +=1;
        no_wins2 = score2;
    }
}
Appendix B

B1. Questionnaires