CITRENZ
COMPUTING AND INFORMATION TECHNOLOGY RESEARCH AND EDUCATION NEW ZEALAND

Journal of Applied Computing and Information Technology

ISSN 2230-4398, Volume 12, Issue 1, 2008

Incorporating the NACCQ publications:
Bulletin of Applied Computing and Information Technology, ISSN 1176-4120
Journal of Applied Computing and Information Technology, ISSN 1174-0175

:: **Refereed Article A5:**

Mifrenz: Safe email for children

Tim Hunt
Waikato Institute of Technology, New Zealand
wtim.hunt@wintec.ac.nz

Hunt, T. (2008). Mifrenz: Safe email for children. *Journal of Applied Computing and Information Technology*, 12(1). Retrieved June 2, 2015 from http://www.citrenz.ac.nz/jacit/JACIT1201/2008Hunt_SafeEmail.html

Abstract

Products currently available for monitoring children's email usage are either considered to encourage dubious ethical behaviour or are time consuming for parents to administer. This paper describes the development of a new email client application for children called Mifrenz. This new application gives parents the ability to let their children safely use email, with the minimum of intervention. It was developed using mostly free software and also with the desire to provide real first hand programming examples to demonstrate to students.

Introduction

This work was first presented as a conference paper (Hunt, 2007). In the present paper the literature review has been brought up to date and the latest developments in the application are described. The most significant development in the area of children's email software has been the recent launch of 'Family Safety' software from Microsoft. This is described in the literature review.

For many adults, email has become the most common method of distant communication (and sometimes not so distant), despite the growing nuisance caused by unsolicited email - SPAM. Anti SPAM technology usually limits the quantity of SPAM to a level that still makes email worthwhile, at least for adults. This is not the case for children aged between 5 and 12 years old, where even one offensive email may be an unacceptable price to pay. If a parent wishes to let, or even encourage, their young children to use email, the safest approach is to provide continuous supervision of their email activities. Of course, finding time for this supervision may be an issue for many parents and so email use is either not supervised, or just does not happen.

A basic technique known as a 'white listr' does, however, provide an excellent way for the elimination of nearly all SPAM (Templeton, 2007). A white list is a list of email addresses from which email is accepted and all other email is automatically deleted. For many adults, a white list is not an acceptable solution to SPAM, as most adults expect and want to receive email from people who they have not yet added to their white list. For children, however, in theory, a white list provides an ideal method for parents to control whom their children communicate with. However in practice, a parent must still find the time to create and maintain (add new friends to) the white list in a timely manner - children will want it done NOW. Another major issue exists with white lists; they can be turned off quite easily once the child starts to explore what all the buttons do. It is therefore evident that email client applications designed to be used by adults are not a satisfactory solution for parental supervision of email usage by young children.

Although a number of products are available to address some of these issues (see literature review) the author proposes new functionality that if present, with the best features of the available products, would improve on the currently available solutions. This paper describes the design and development of such an email client, using Java and other free software development environments and tools. In addition, the work was performed in the context of teaching programming skills to undergraduate students and many of the design issues were tackled with teaching in mind.

Literature Review

Very few studies have been conducted that solely focus on the use of email by children, however, the use of email by children has been included in numerous studies on general Internet use by children. These studies tend to focus on children of at least 7 years of age.

A survey (Livingstone & Bober, 2004) of Internet use by children aged 9 to 19 in the United Kingdom found that 75% of them had access to the Internet from home; of those, 72% used the Internet for sending and receiving emails. Among the regular users of the Internet, 4% "have sent a message to make someone feel uncomfortable or threatened", and 25% had received pornographic SPAM via email or instant messaging. 79% of the children said that they normally used the Internet unsupervised. Beran and Li (2005) reported that in the 12 to 14 age group, 21% of children said that they have been cyber-harassed multiple times and 3% said they had actually committed this type of harassment.

Jackson et al. (2006) studied the benefits of Internet use by children aged 10 to 18 years from low income families. They investigated the academic benefits of home Internet use and concluded that the more the children used the Internet, the greater the positive benefit on their academic performance was. However, they found that children in that group made very little use of email and "chat tools". It was suggested that a major reason for this was that children from low income families had friends of similar backgrounds and so were also unlikely to have Internet access - therefore no one to 'talk' to online.

It is interesting to see the conflict that exists between childrens' and parents' expectations of Internet access software. Whereas 69% of children do not want their parents monitoring their online activities, over 50% of parents want access to improved control and monitoring tools (Livingstone & Bober, 2004). Monitoring of children seems to be a growing phenomenon (Delaney, 2006) but the ethics of this are starting to be questioned in the popular media; for example an article (Hymowitz, 2006) in the New York Times likened the practice of monitoring your child's email to reading their personal diary.

It is now possible for a parent to monitor a child's Internet access very closely (Delaney, 2006):

> *"Features vary among the different offerings, but most can record every keystroke a computer user types and log every Web site he visits. Many log incoming and outgoing email and instant messages and record "screen shots,or images of what the user does online. The software - including packages called Specter Pro, eBlaster, and IamBigBrother -- is designed to prevent children from tampering with its operation. It allows parents to see sites, including blogs, their children are accessing. The keystroke logging permits them to capture login and password information their kids have entered for blogs. That could allow parents to access even blogs that are password protected or are otherwise restricted."*

Email Clients for Children

For young children, a different approach can be used to control email activity. Kidmail.net (2007) offers a client application that is designed to appeal to young children (5 to 9 years) and also an email server service that controls (for an annual fee) the email a child receives. This is basically a 'white-list' solution to preventing email from unknown persons reaching the child. The parent (with help from the software) creates and maintains the white-list and also has the ability to read the child's emails. The software does not delete emails that are not on the white list, but gives the parent a chance to read them and add them as a new contact to the white list.

Starfish Family Mail (Lincoln Beach Software Ltd, 2004) is an email client for children that uses a white list created by the administrator. The software can be set up so that any emails from addresses not on the white list are either hidden from the child or require parent intervention to be read. The software also allows a basic authentication method to make sure that the email has not been spoofed (see next section). This product requires a high parental involvement to keep the white list up to date.

Since the earlier version of this paper was published, Microsoft has launched 'Family Safety' as part of their 'Windows Live' product range (Microsoft Corporation, 2008). The Family Safety software allows parents to set up Hotmail email addresses that are linked to their own Hotmail address. When a child wishes to send email to a friend that is not on their contacts list, they are able to send a contact request to the parent who can either approve or decline this contact request. The Family Safety software also offers the ability for parents to control and monitor web page and chat access. Children access their email through the normal Hotmail website which is not specifically designed for children and contains advertising.

Spoofing

A potential flaw with relying on the white list to block SPAM is the concept of spoofing (E-mail spoofing, 2007). An email is said to be spoofed when the 'From' email address has been deliberately changed so that the email appears to be from someone other than the true sender.

A very basic way to spoof an email is to use a false 'From' address. This type of spoofing (see Table 1) is easy to detect. The table shows some of the header fields of an email that was sent from a Hotmail (Microsoft Corporation, 2007) account to a Gmail (Google, 2008) account. The sender was pretending to be a friend called Nice Person (nice.person@ happyserver.com), however the 'mailed-by' value is different to the 'from' and 'reply-to' values. Although easy to detect, Gmail did not pick this up as SPAM, presumably because for adults at least, this is not a definite sign of spoofing.

from	Nice Person <nice.person@happyserver.com>
reply-to	Nice Person <nice.person@happyserver.com>
to	timhot@gmail.com
date	Mar 3, 2007 1:51 PM
subject	Let's meet at the park
mailed-by	hotmail.com

Table 1. Detailed email header information from a spoofed email

Another method to reduce spoofing is to employ a 'hand shake' between the sender and receiver. Riyadh (2001) proposes the 'Correspondence Negotiation Protocol' to give more control to the receivers of email. Another report (Anti-Phishing Working Group, 2007) provides a comparison of various proposed methods of preventing email spoofing including checking the IP address of the sender and using digital signatures. The problem that they are trying to solve though is not the same as the problem for children, where email to children can be limited to an approved white list rather than trying to allow all non-spoofed emails through.

Literature Review Conclusion

The use of email can provide a valuable learning experience for young children, yet also presents a risk to those same children. Many parents are turning to spying on their children's online activities and this poses ethical considerations and dilemmas. Although the recent release of Microsoft's free Family Safety product is likely to be very popular, it also encourages parents to spy on their children's online activities. Two less intrusive solutions were identified based on the use of a white list; however neither solution provided all functionality in a low cost product. It is concluded that there is a gap in the market for a specialised email application for children that addresses the needs identified here.

Aims of This Work

The currently available email clients do not provide a satisfactory and low cost solution for protecting young children from SPAM. This work aimed to address this gap in the market with a new email application specifically designed with both children and parental supervision in mind. As identified in the literature review, a number of issues needed to be addressed.

1. The conflict between a child's need for privacy and the parent's responsibility to protect the child.

2. How to give a child the ability to email their friends in a timely manner while not requiring the parent to give constant supervision.

3. Keeping the cost of the solution low enough to allow widespread use.

In addition the following additional aims were identified:

4. Create a product that is both fun and simple to use for young children so they would want to use it.

5. Use techniques with the Java programming language that the author teaches in a number of computer programming courses so that they can be demonstrated to students.

The literature review identified that Internet use by young children is a potentially useful tool for improving their academic performance. By addressing the above aims, it was hoped that this work would create an application that does indeed encourage children to read and write (type) more than they otherwise would and therefore potentially improve those skills.

Design

A new email client application called Mifrenz was created to meet each of the aims identified above. An overview of how each aim was addressed follows.

Conflict Between Needs of Children and Parents

When a parent logs on to the software, they are presented with the 'parent' window and they create the initial contact list which acts as a white list. The parent window does not display any emails that are sent or received. When the child logs on, they are taken to the 'child' window and can then email anyone on the contact list and receive email from anyone on the list. The parent is thus able to control the child's contacts without being encouraged to read the email. The parent can easily view the child's password so they can retell it to the child without having to reset it. So, although the parent could logon as the child, the software does not encourage this behaviour and the parent would have to make a conscious decision to do so.

Timely Manner

Once the initial list has been created, the child can then create new contacts himself or herself. The parent is automatically sent a 'contact approval request' to their normal (for example work) address, to which they only have to reply with a yes or no, to activate or block the new contact and any current or new emails. During the time interval between a child creating a new contact and the parent either approving or declining the contact, the contact is in a 'pending' state. While in this state, the child can use the new contact as normal, except when they press the send button, the email is held back (pending) behind the scenes, until the parent approves the contact. Emails received from a pending contact are also held back out of view until the contact is approved. However, if the contact is declined, all pending emails to and from this contact are deleted.

Low Cost

Mifrenz uses a third party email service (initially Gmail) to send and receive email, but all 'intelligence' is on the client application. This avoids the need to provide a server to supply the email service and so keeps the potential cost of the product low. Also, Mifrenz does not provide the extensive 'spying' capabilities of many other products and so the development time was relatively short.

As far as possible, Mifrenz was developed using free software development tools and environments. The main development environment used Netbeans (Netbeans, n.d.) and Java (Sun Microsystems, 2007c), both are free to use. To distribute the software, a website was created using the free Netbeans plug-in, Visual Web Application which allows WYSIWYG (What You See Is What You Get) web page development. Sun's free "Sun Java System Application Server" (Sun Microsystems, 2007a) provided the server platform. The first iteration of the website used "Java Web Start" (Sun Microsystems 2007b), a relatively new and free technology, to distribute and install the software to users. However, Java Web Start proved to be unsuitable and has now been replaced with two other free products, Launch4J (Launch4J, 2007) and NSIS (Nullsoft Scriptabe Install System, n.d.). MySQL (MySQL.com, n.d.), an open source database, is used to keep track of who downloads the software. Also various other free utilities are used to maintain the web server remotely, for example LogMeIn Free (LogMeIn.com, 2007), a remote control tool, and FileZilla (FileZilla, n.d.), an FTP tool.

Fun and Simple to Use

Young children do not need the sophistication of many current email client applications (see, for example, Microsoft Corporation, 2007), and in fact the author believes that too much choice is an impediment to children using email. For example, young children only require an 'In box' and a 'Sent box', they do not need other folders to organise email. They do not need extensive information about each email such as date, importance, size etc.

Young children love jokes and collecting things. Mifrenz provides a simple way for them to create jokes and then to easily share them with their friends. The software automatically builds a joke list from jokes that are sent to them (or they can create new jokes), and they can then attach these new jokes to emails to share with others.

Teaching Material

The author teaches Java programming and was keen to demonstrate actual use of the techniques being taught. For example, students are taught how to write objects to disk and this is demonstrated by writing an email message object to disk, and threads are used to make the GUI interface more responsive by connecting with the email server on one thread, while a separate thread allows the child to write or read another email.

Major Design Decisions

The following sections describe some of the major design decisions made.

Three Layer Design

A well known design pattern is to separate the business logic of an application from the user interface and database/disk access code. This is typically implemented as three separate layers: the presentation layer, the business logic layer and the data access layer. Mifrenz used this pattern with each layer represented by a package of classes. Akin to a three tier client/server architecture design, each layer was designed to allow its possible replacement at a future date. For example, all data flow and method calls from the presentation layer are to the business logic layer and not to the data access layer, and the data access layer implements all calls to the disk storage and all connections to the email server.

An example of this design is seen when new emails are retrieved from the server. Although the dataAccessLogic package handles connections to the server, it is the EmailLogic class in the businessLogic package that determines if the email is from an approved contact and then determines what to do with the email.

Other examples of this design occur in the presentationLogic package which is responsible for all user interaction. Whenever possible, this package passes all decisions to the businessLogic package. This design solution decouples the GUI from the rest of the application. Object oriented analysis and design

Object Oriented (OO) techniques are currently seen as 'best practice' (Perks, 2006) for creating software and as Mifrenz is intended to be used as a teaching tool, OO techniques such as data encapsulation and inheritance were utilised throughout the process. For example, objects are used to store, manipulate and pass data and many of the interface classes inherit from one of the predefined Java classes.

Data Storage Design

The following considerations were taken into account when choosing a storage solution:

- To keep the final cost of the product to a minimum, only free solutions were considered.
- As Mifrenz is to be distributed via a website, a large database application such as MySQL (MySQL.com, n.d.) would have resulted in an unacceptably long download time.
- The author wished to demonstrate techniques taught to his students which include object serialization, but not SQL databases.

Based on the above considerations, and because there are no concurrency and transaction management issues, it was decided to create a proprietary pseudo database using object serialisation techniques. In essence, each email is stored as a separate object on disk and is read into a list structure when the application is opened. The DatabaseWrapper class handles all read and writes to the disk, for example, a method called getAllContacts(User user) returns a list of all the contacts for the user called 'user'.

Email Server

h4

A decision was made to use a third party email server, so there would not be an ongoing service requirement. This keeps the cost of providing the product to a one-off cost, rather than having to have a yearly fee as other similar solutions require. Initially Mifrenz was 'hard coded' to work with a Gmail email account so as to reduce any setup issues and testing requirements. Gmail was chosen as it is a global service, provides 2 GB of storage and allows POP3 (Post Office Protocol version 3) email access. Gmail now also offers IMAP (Internet Message Access Protocol) access which opens up new opportunities as discussed in the Future work section below. Before Mifrenz can be used, the parent must first create a Gmail account for their child and enable it for POP access (instructions are provided to help do this). Mifrenz uses the Java email API (Application Programming Interface) to connect to the Gmail email server, allowing the sending and receiving of emails. Gmail requires 'Secure Socket Layer' (SSL) access which the Java API also provides.

User Interface Design

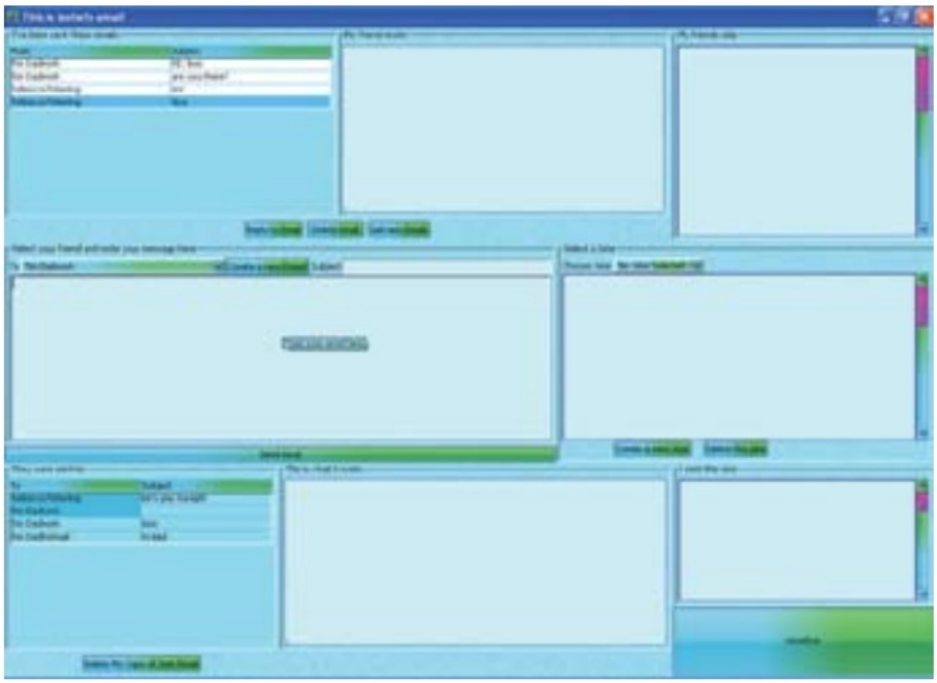
Mifrenz has a separate user interface for the child and the parent. It also has an email notification for parents.

Child Interface Design

The overriding factor in this design was to make it as simple as possible. Again, the product was designed with teaching in mind, in that some basic design principles were followed e.g. the users focus naturally starts at the top left of the screen and moves towards the lower right. As mentioned earlier, it was felt that many of the features of 'normal' email applications were not necessary for young children, and therefore these were not included. Figure 1 shows the layout of the child interface. To keep it simple, there is only the one window (except for the 'pop up' joke creation window) that is split into three horizontal layers: the top layer is for emails that have

been received, the middle layer is for creating new emails, and the bottom layer is for emails that have been sent.

Figure 1. The child interface



The top layer has three panels: the left panel shows a list of received emails, the centre panel displays the text of the email and the right panel displays the joke (if any) that was attached. Unread emails are differentiated from read emails by a different colour. There are three buttons for replying to the selected email, deleting an email, and checking for new emails.

The middle layer is split into two panels: the left panel is used to compose a new email, and the right panel is used to choose and display a joke to attach to the email. The child chooses whom to email by either pressing the 'Reply to Email' button, or by choosing the contact from the 'To' drop down list. The bottom layer is basically a repeat of the top layer, but instead displays the emails that have been sent.

Scroll bars appear automatically when there is more information than can be entirely displayed in a panel. The GUI layout uses a Java technology known as 'Layout Managers' that provides windows that automatically change size to suit the size of the available monitor resolution and the information being displayed. This avoids the display problems that are inherent with fixed position by pixel design that many other applications use. This is another example of good design that can be demonstrated to students.

Look and Feel

It is possible to manually set the colour behaviour of each of the visual components; however this is a laborious process and doesn't lend itself to ease of change. Java provides the class 'SynthLookAndFeel' which along with a configuration file, can be used to globally set how a component will look and behave. After a considerable amount of time had been spent on trying to get this technology to work in a satisfactory manner, it was eventually abandoned. One of the major reasons was the inability to draw scroll bars that looked professional. While searching for help with the SynthLookAndFeel, another Java 'Look and Feel' technology was discovered, called 'SubstanceLookAndFeel'. This technology proved to be much easier to use and was successfully implemented. A range of predefined themes are available and initially Mifrenz was hard coded to just use one of them, although it is intended to give the child a choice.

In addition to the SubstanceLookAndFeel 'tooltips' were added to many of the controls so that when a child hovers the mouse over a control, help appears for that control.

Parent Interface Design

The first time that Mifrenz is used, the 'Parent (Administrator) Details' window opens automatically. The parent enters their own password and email address details. They are then taken to the main parent window where they can enter the details for each of their children including the initial contacts for each child.

Parental Supervision Design

The author believes that it is not ethical for parents to read their children's email, but that it is a parent's duty to protect their children from SPAM email. The use of a white list is an ideal way to block email from all sources except from those on the list. The white list (i.e. the child's list of contacts) is imbedded into Mifrenz and there is no option to not use it. To prevent the child accessing Gmail directly, i.e. using the Gmail web site, the child's Gmail password is hidden in the parent's password protected area of Mifrenz.

A child's friends are likely to be changing at a rate higher than a parent wishes to enter the new contact details, and certainly faster than a child is willing to wait until their parent gets around to entering the new contacts. Mifrenz was designed to make it as easy as possible for parents to vet their child's new contacts and update the white list. This is achieved by allowing the child to create the new contact themselves, and even write and 'send' the email before their parent approves the contact. However, 'send' is not what actually happens, as that email is held back until the contact is approved. A design was sought that would allow this approval process to occur as quickly and as easily for the parent as possible - parents should not be required to use the same PC as the child (as one reviewed product required).

As the design of Mifrenz did not allow for a central server, this approval process could not be achieved by logging on to a server, as required by one of the other products reviewed. Instead, when a new contact is created by a child, an email is sent to the parent's specified email address. The subject of the email is "Contact request from Mifrenz" so that the parent can easily prioritise these requests if they want to. The email contains the name and email address of the new contact and asks the parent to reply with either yes or no on the first line of the body of the email. This email is then returned to the child's Gmail account and is downloaded to Mifrenz the next time the child checks their email. Mifrenz intercepts this email by looking for the string "Contact request from Mifrenz" in the subject line. Mifrenz then processes the email, determining whether the contact has been approved or declined. If the contact has been approved, Mifrenz then checks to see if there are any emails waiting to be sent to this contact and either sends them, or deletes them if the contact request was declined.

This design allows a parent to help their children choose who they send and receive emails from. The parent has the opportunity to discuss with their children who their new friends are and how they met them. The child knows that their parent is being asked to approve their new contacts and no deceit is happening. Of course, a parent could still go and read their children's email by either logging on to Mifrenz with the child's password, or by using the Web interface of Gmail. Just as the author believes a parent should not be reading a child's diary, it is up to the parent to refrain from snooping on their child's email activities. Most of the other email supervision products that have been identified seem to actively encourage a parent to snoop on their child's emails.

Robustness of Design

Although Mifrenz is not a mission-critical application, its design had to be robust enough to not frustrate either parents or their children. It certainly aimed to have a 100% guarantee of not letting through emails that were from addresses not on the white list. The reliance on a white list to block unwanted emails still leaves the potential for receiving spoofed emails. See the section on spoofing in 'Future Work' later in this paper.

It is worth pointing out that Mifrenz also benefits from the anti-SPAM tools deployed by Gmail, although this gives no guarantee of completely eliminating SPAM. In addition to keeping the users happy (by not behaving unexpectedly), Mifrenz was also created as a teaching aide and so was required to demonstrate good design practice. Java in itself is an excellent language for robust design and so a good foundation to build on. Extensive use of the try-catch-throw structure was used with errors thrown to the business logic layer when ever possible, thus keeping decisions in this layer.

Further Functionality

This section lists some of the other functionality of Mifrenz that has not yet been described.

Emails from Unknown Friends (Strangers?)

As with adults, children give out their email addresses to friends and people they meet. However, with children, the new friend is quite likely to not know their own email address and so the Mifrenz user cannot add the new friend to their contact (white) list. This poses a problem for users of Mifrenz, as any email that arrives that is from someone who is not on the contact list could automatically be deleted. The following feature was added to Mifrenz to mitigate this scenario.

When an email arrives from an unknown person (i.e. the email address is not in the contact list), a reply is automatically sent to that person with a message explaining that Mifrenz has intercepted their email and that they should reply back if they really want to email the child. If they do reply, Mifrenz again intercepts the email and then creates a new contact for this 'Stranger'. An email is then sent to the parent (note, the child is totally unaware at this stage),

who can decided if they want to approve this 'Stranger'. If they do, they just reply back to Mifrenz with a 'y' on the first line (as for a normal contact request). The child is then informed that a new contact has been added and the original email message from the 'Stranger' is displayed. This process relieves the parent from having to read all of the SPAM that a child might receive in order to check that it is not from a genuine friend.

Multiple Children

Mifrenz can be used by multiple children on the same PC. When a parent creates a child user, a new directory is created in the file system, and all emails associated with that child are stored in sub-directories for that child.

All Grown Up

Because Mifrenz uses a Gmail account, when the child reaches a more independent age, parents can tell their children that they can now access their email directly through the Gmail web interface (without using the Mifrenz client). This means that children do not have to change email addresses but can continue to use the same Gmail email address.

Testing

The author teaches the use of JUnit (JUnit, n.d.) testing and intended to use Mifrenz to demonstrate its use. The initial idea was to follow the Extreme Programming (Extreme Programming, n.d.) approach of writing tests first, that is before a Class is written. However, the phrase "Do as I say, not as I do" springs to mind as the author found that at the start of the coding process, the Classes were not well defined, and writing tests for these Classes was very time consuming. Testing developed into a more manual checking of system functionality. Once the code has 'settled' down, it is intended to write a more comprehensive set of JUnit tests, so that further work can progress in the knowledge that it has not broken existing code.

Testing therefore produced a number of discussion points for students. What was it about this particular development that meant the theory was not followed? To answer this question the following points should be considered: 1) this was a one person project, 2) the developer was still learning, 3) the functionality was changing as the project progressed. The author's two young children have been using the software to email friends and family. Although a formal test plan was not followed (ethical issues have been raised!), their use of the software was observed and many issues encountered that allowed fixes and enhancements to be made. In fact, these alpha testers provided many suggestions for improving the product.

Software Distribution

A major issue was how to distribute and install Mifrenz, a Java application, via a download from a website. Although commercial products exist for installing Java applications, their cost precluded their use. The initial approach taken was to have a web page that checked that the correct version of Java was installed on the client PC and, if it was not, direct the user to the Sun website to upgrade their PC to the correct version. Once this was completed, the user could then see a Java applet in their browser that allowed them to download a Mifrenz jar file to their desktop. The user could launch Mifrenz by double clicking on the downloaded file. Although this approach worked, the whole process and end result did not give the desired professional look and feel.

Java Web Start

Java Web Start (Sun Microsystems, 2007b) is a relatively new technology that addresses the need for a free and professional method of distributing Java programs. This technology allows the version of Java on the user's PC to be checked and upgraded automatically if required. Java Web Start then downloads the Java program (i.e. Mifrenz) and installs it on the user's PC and a shortcut is placed in the 'Start' menu. There are various configuration settings, but a major one is that it can be set to automatically check the availability of new versions of the Mifrenz software. Note Java Web Start also works with other operating systems such as Mac OS X.

A major drawback of using Java Web Start was that the free Web Service (Geocities, n.d.) that was hosting the Mifrenz website did not allow the use of Java Web Start technology. It was time to spend some money! A local web hosting company (Treehouse Networks Ltd, n.d.) was found that would host a dedicated Mifrenz server. This gave the author complete control over the server setup. Sun's Application Server was used to provide the necessary environment for Java Web Start.

It became apparent that all was not well with this distribution solution. Although all worked well when a single account was used on the client PC, when the software was installed and used by separate accounts (e.g. the parent installs the software when they are logged in, and the child tries to use the software when they are logged in), Mifrenz was not available for the child to use.

The author discovered that Java Web Start has been deliberately designed to operate in this manner and so was not a suitable technology for Mifrenz distribution.

'Standard' installers

Java Web Start was replaced with a combination of Launch4J (Launch4J, 2007), NSSI (Nullsoft Scriptable Install System 2.33, n.d.), and HM NSIS Edit (HM NIS Edit, 2005) all free products for commercial use. Launch4J was used to build an MSWindows executable file (.exe) from the Mifrenz java jar file. Launch4J can specify the required minimum version of the Java Runtime Environment and if it is not present on the client PC, a web browser is automatically opened and takes the user to the appropriate Java download website page. NSIS provides the ability to create a MSWindows installer, including the ability to display splash screens, choose the install location for the executable file and supporting files (in this case the java library files), require a license agreement acceptance, and create 'Start Menu' and 'Desktop' icons. NSIS is configured using a text-based script and this can initially be created with the help of the wizard in HM NSIS Edit.

The result was the ability to install Mifrenz to the normal MSWindows location for programs that are shared by multiple users (typically c:\ program files). However, only users with Administrator privileges have write access to this location by default. This meant that although the child could use the Mifrenz software, when they tried to save either an email or a joke, the operating system prevented this from happening. What was needed was for the user data to be stored in a location that non-administrators had access to by default. NSIS provides a variable that refers to the shared documents directory on the MSWindows operating system. On testing however, it was discovered that for the MSWindows XP operating system, users still could not by default write to this directory but instead to a subdirectory. As Mifrenz contact files had to be created and modified by both the parent and child, this directory was unsuitable. Instead, the Java System.exec method was used to create a directory that all users of the PC would have read and write access to. This setup should have allowed children to log on to a shared family PC and use Mifrenz; however the author was unable to get this to work with the MS Vista operating system. The author believes that this problem may be common, as it has not been solved in another software application for children that he has bought recently. A solution was finally reached by determining the operating system at runtime, and using the known shared directory, for that operating system, that allows all users to read, write and delete file access.

Website and Database

The Mifrenz website (http://mifrenz.com/) was developed using the Netbeans plug-in "Visual Web Application". A brief description of the software is given and users are required to fill in a form (name and email address) to get access to the download link. A MySQL database collects the form information. The software is currently available free of charge.

Future Work

Testing

At the time of writing, ethical approval is being sought to test the software with third parties. Only adults will be approached via friends of the author, schools, youth organisations and the website. Data on how the software performs will be emailed to the author. Before someone can download the software they will need to 'sign' a consent form giving permission for this data to be collected and used for research purposes.

The JUnit environment will be used to setup an extensive set of tests. This will enable the maintenance and development to proceed with the confidence that software changes will not break current functionality.

Object Orientated Design

As stated earlier, object oriented analysis and design was an aim of this work. On reflection, although many classes inherit other classes, and objects are used to encapsulate the data with methods, there are many more opportunities to use object oriented techniques such as reducing the coupling between components.

Joke Attachment

Currently jokes are sent from one Mifrenz user to another as a binary file attachment (a serialized joke object). As intended, this prevents non-Mifrenz users from sharing the jokes however it also gives some problems.

When a child sends a joke to an adult, say their grandmother, it is unlikely that their grandmother will be using Mifrenz. The joke file will appear as an unrecognised binary file and if it can be displayed, will not make much sense. Another problem (identified by one of the author's students)

is that as Mifrenz is improved and new versions are released, the joke file may well not be compatible between different versions.

In order to solve these problems it is intended to relax the initial concept of not sharing jokes with non-Mifrenz users. Instead jokes will be incorporated into the body of the text and Mifrenz will extract the joke to store it in the user's lists of jokes, otherwise it will just be left in the email for the non-Mifrenz user to read.

3D Interface

To improve the appeal to children, a 3D world environment will be developed for the children's interface. The Java 3D API (Java.net, 2007) will be used to develop this interface. An initial proof of concept has been completed by a final year 'capstone' project student.

Spoofing

To prevent basic spoofing, the 'From' domain and the 'mailed-by' domain could be checked to see if they are the same before allowing the email to be delivered to the user's 'In Box'. Mifrenz could be modified to use a simple authentication system where senders are told to use a certain text string in their email (or this could be done automatically between two Mifrenz users). A more sophisticated handshaking solution based on previously sent emails could be employed, although this type of authentication is more suited to server-based solutions or 'always on' clients - requiring Mifrenz to be similarly deployed.

Fault and Usage Logging

Reporting errors via 'calling home' has become a common feature of software applications. As already mentioned above, errors that are thrown are caught at the business logic layer. This error information is currently written to a text file on the local disk drive. It is intended that this information will be sent back to the Mifrenz website server. Implementing a web service is one obvious method of doing this, although just sending the data as an email may avoid any issues with the user's personal firewall such as displaying a warning message that the user doesn't understand.

The literature review indicated that the use of email may improve the academic achievement of children. As a first step towards further investigation of this idea, Mifrenz will incorporate usage data reporting. There are obvious privacy issues with this and so work needs to be done on what data can safely be collected: for example it may be possible to log and report on the level of spelling and grammar, without reporting the actual words used or the child's name.

Automatic Creation of Email Account

To improve the ease with which a parent can set up Mifrenz, it is intended to have an option that automatically creates an email account and configures Mifrenz to use it. Users will have to pay a third party to use this account after an initial trial period.

Multiple Point of Access

Initial testing by users has highlighted the requirement for children to access their email from two separate homes. Currently Mifrenz uses the POP protocol to download all email from the email server to the local client. Although POP can be configured to leave a copy of the email on the server, the IMAP protocol is better suited for synchronising email access from multiple devices. How this can be made to work, with the additional requirement that children only see email from their contacts, needs to be investigated.

Summary and Conclusions

This paper has described the development of an email client application for children - Mifrenz. The application was developed to provide a low cost, low maintenance and ethically acceptable way for parents to guide young children's use of email and so protect them from the risk of receiving harmful SPAM. The aim of creating a low cost solution was a major guiding force on how the product was developed. To this end, the software does not rely on a dedicated email server, and only free development technologies were used. The application's Java source code provides numerous programming examples and can be used as teaching aids. The application is currently in a working beta stage and is available for free download from http://mifrenz.com/

Acknowledgements

Thanks are due to Rebecca and Isobel, who were great testers, and to the students who have provided suggestions for improving Mifrenz and discovering various bugs and problems.

References

Anti-Phishing Working Group (2007). Proposed solutions to address the threat of email spoofing scams. Retrieved March 3, 2007, from http://www.antiphishing.org/Proposed_Solutions_to_Address_the_Threat_of_Email_Spoofing_Scams_White_Paper.p

Beran, T., & Li, Q. (2005). Cyber-harassment: A study of a new method for an old behavior. Journal of Educational Computing Research, 32(3), 265-277

Delaney, K. (2006). Spy vs. spy: Tools for shadowing teens online. Wall Street Journal. (Eastern edition). Retrieved February 28, 2007, from ProQuest database.

Extreme Programming. (2006). Extreme Programming: A Gentle Introduction. Retrieved February 28, 2007, from http://www. extremeprogramming.org/

E-mail spoofing. (2007). Wikipedia, The Free Encyclopedia. Retrieved March 1, 2007, from http://en.wikipedia.org/wiki/E-mail_spoofing

FileZilla. (n.d.). FileZilla, The Free FTP Solution. Retrieved February 28, 2007, from http://filezilla.sourceforge.net/

Geocities. (2007). Yahoo Geocities. Retrieved February 28, 2007, fromhttp://geocities.yahoo.com/

Google. (2008). GMail Beta, Welcome to Gmail. Retrieved April 21, 2008, from http://Gmail.com

HM NIS Edit. (2005). HM NIS Edit: A Free NSIS Editor/IDE. Retrieved February 28, 2007, from http://hmne.sourceforge.net/

Hunt, T.D. (2007). Mifrenz: A new email client application for children. In S. Mann & N. Bridgeman (Eds.) Proceedings of 20th Annual Conference of the National Advisory Committee on Computing Qualifications (pp. 99-105). Nelson, New Zealand: NACCQ.

Hymowitz, K.S. (2006). Taste: Big mother is watching. Wall Street Journal. (Eastern edition). Retrieved 28 February, 2007, from ProQuest database.

Jackson, L.A., von Eye, A., Biocca, F.A., Barbatsis, G., Zhao, Y. & Fitzgerald, H.E. (2006). Does Home Internet Use Influence the Academic Performance of Low-Income Children? Developmental Psychology, 42(3), 187-201

Java.net. (2007). Java.net, The Source for Java Technology Collaboration. Retrieved February 28, 2007, from https://java3d.dev.java.net/

JUnit. (n.d.) JUnit. Retrieved February 28, 2007, from http://junit.sourceforge.net/

Kidmail.net. (2007) Kidmail.net, The Safe E-Mail Service. Retrieved February 28, 2007, from http://www.kidmail.net/default.asp

Launch4J (2007) Launch4J 2.1.5, Cross platform Java Executable Wrapper. Retrieved November 27, 2007, from http://launch4j.sourceforge.net/

Lincoln Beach Software Ltd. (2004) Star Fish Family Mail. Retrieved March 4, 2007, from http://www.lincolnbeach.com/sfm_main.asp

Livingstone, S., & Bober, M. (2004). UK children go online : Surveying the experiences of young people and their parents. Retrieved February 18, 2008, from http://eprints.lse.ac.uk/395/

LogMeIn.com (2007). LogMeIn Free. Retrieved February 28, 2007, from https://secure.logmein.com/home.asp

Microsoft Corporation. (2007). Microsoft Office Online, Microsoft Office Outlook. Retrieved February 28, 2007, from http://office.microsoft.com/ en-au/outlook/ default.aspx

Microsoft Corporation. (2008). Windows Live. Retrieved April 21, 2008, from http://www.windowslive.com/overview.html

MySQL.com (n.d.) MySQL. Retrieved February 28, 2007, from http://www.mysql.com/

Netbeans. (n.d.) Netbeans IDE 5.5. Retrieved February 28, 2007, fromhttp://www.netbeans.org/

Nullsoft Scriptabe Install System. (2007). Nullsoft Scriptabe Install System 2.33. Retrieved November 27, 2007, from http://nsis.sourceforge.net/Main_Page.

Perks, M. (2006). Best practices for software development projects, IBM Software Services for WebSphere. Retrieved March 6, 2007, from http://www-128.ibm.com/developerworks/websphere/library/techarticles/0306_perks/perks2.html

Riyadh, M.M. (2001). Towards security and balance in email through correspondence negotiation. Master of Science Thesis, School of Computer Science, University of Windsor, 2005. Retrieved February 28, 2007, from http://ir.lib.sfu.ca:8080/retrieve/2180/etd1827.pdf

Sun Microsystems. (2007a). Sun Java System Application Server. Retrieved February 28, 2007, from http://www.sun.com/software/products/appsrvr/index.xml

Sun Microsystems. (2007b). Java Web Start. Retrieved February 28, 2007, from http://www.sun.com/software/products/appsrvr/index.xml

Sun Microsystems. (2007c) Sun Developer Network (SDN), Java.sun. com, The source for Java developers. Retrieved February 28, 2007, from http://java.sun.com/

Templeton, B. (2007) A variety of weapons against SPAM that don't require laws. Retrieved March 4, 2007, from http://www.templetons.com/brad/spam/spamfix.html

Treehouse NetWorks Ltd. (2008). Treehouse NetWorks Ltd. Retrieved February 28, 2007, from http://treenetnz.com/

erence">LogMeIn.com (2007). LogMeIn Free. Retrieved February 28, 2007, from https://secure.logmein.com/home.asp

---

Copyright © 2008 Hunt, T.

(Incorporating the Bulletin of Applied Computing and Information Technology, NACCQ: ISSN 1176-4120 and
Journal of Applied Computing and Information Technology, NACCQ: ISSN 1174-0175)
Copyright ©2008 CITRENZ.
The author(s) assign to CITRENZ and educational non-profit institutions a non-exclusive licence to use this document for personal use and in courses of instruction provided that the article is used in full and this copyright statement is reproduced.
The author(s) also grant a non-exclusive licence to CITRENZ to publish this document in full on the World Wide Web (prime sites and mirrors) and in printed form within the Journal of Applied Computing and Information Technology. Authors retain their individual intellectual property rights.
Donald Joyce (Editor).

An Open Access Journal, DOAJ #22304398, (✓zotero)