**TEXT BASED STEGANOGRAPHY**

**R. J. Lockwood**

**BSc (Hons) Computer Science**

School of Computing and Intelligent Systems

University of Ulster Magee

[lockwood-r@email.ulster.ac.uk](mailto:lockwood-r@email.ulster.ac.uk)

Table of Contents

[Table of Figures 1](#_Toc386513983)

[Table of Tables 3](#_Toc386513984)

[Abstract 5](#_Toc386513985)

[Introduction 6](#_Toc386513986)

[Acknowledgements 7](#_Toc386513987)

[Declaration 7](#_Toc386513988)

[Acronyms and Definitions 8](#_Toc386513989)

[1 Steganography 9](#_Toc386513990)

[1.1 Image Based Steganography 9](#_Toc386513991)

[1.1.1 File Format Manipulation Methods 10](#_Toc386513992)

[1.1.2 LSB Encoding Schemes 10](#_Toc386513993)

[1.1.3 Other Methods 11](#_Toc386513994)

[1.2 Video/Animation Based Steganography 11](#_Toc386513995)

[1.2.1 Frame Manipulation 11](#_Toc386513996)

[1.2.2 Time Manipulation 11](#_Toc386513997)

[1.3 Executable File Formats and Binary Execution 11](#_Toc386513998)

[1.4 Audio Steganography 12](#_Toc386513999)

[1.5 Suitability Matrix 12](#_Toc386514000)

[2 Text Based Steganography 14](#_Toc386514001)

[2.1 Format Based Systems 14](#_Toc386514002)

[2.1.1 Punctuation Amendment 14](#_Toc386514003)

[2.1.2 Spelling Misappropriation 14](#_Toc386514004)

[2.1.3 Open Space 15](#_Toc386514005)

[2.1.4 Line Spacing 15](#_Toc386514006)

[2.1.5 Font Manipulation 15](#_Toc386514007)

[2.2 Random and Statistical Generation 16](#_Toc386514008)

[2.2.1 Markov Chains 16](#_Toc386514009)

[2.2.2 Context Free Grammar 17](#_Toc386514010)

[2.3 Linguistic Steganography 17](#_Toc386514011)

[2.3.1 Synonym Replacement (Lexical Substitution) 17](#_Toc386514012)

[2.3.2 Sentence Modification 18](#_Toc386514013)

[3 IEEE System Requirements Specification 19](#_Toc386514014)

[3.1 Introduction 19](#_Toc386514015)

[3.1.1 Purpose 19](#_Toc386514016)

[3.1.2 Definitions 19](#_Toc386514017)

[3.1.3 Scope 20](#_Toc386514018)

[3.1.4 Overview 20](#_Toc386514019)

[3.2 Overall Description 20](#_Toc386514020)

[3.2.1 Product Perspective 20](#_Toc386514021)

[3.2.2 System Interfaces 26](#_Toc386514022)

[3.2.3 User Interfaces 26](#_Toc386514023)

[3.2.4 Hardware interfaces 29](#_Toc386514024)

[3.2.5 Software interfaces 30](#_Toc386514025)

[3.2.6 Communication Interfaces 30](#_Toc386514026)

[3.2.7 Memory Constraints 31](#_Toc386514027)

[3.2.8 Operations 31](#_Toc386514028)

[3.2.9 Site Adaptation Requirements 32](#_Toc386514029)

[3.2.10 Product functions 32](#_Toc386514030)

[3.2.11 User characteristics 33](#_Toc386514031)

[3.2.12 Constraints, assumptions and dependencies 33](#_Toc386514032)

[3.3 Specific Requirements 34](#_Toc386514033)

[3.3.1 External interface requirements 34](#_Toc386514034)

[3.3.2 Functional requirements 35](#_Toc386514035)

[3.3.3 Performance requirements 35](#_Toc386514036)

[3.3.4 Design constraints 35](#_Toc386514037)

[3.3.5 Standards Compliance 36](#_Toc386514038)

[3.3.6 Logical database requirement 36](#_Toc386514039)

[3.3.7 Logical class diagram 37](#_Toc386514040)

[3.3.8 Reliability 37](#_Toc386514041)

[3.3.9 Availability 38](#_Toc386514042)

[3.3.10 Security 38](#_Toc386514043)

[3.3.11 Maintainability 38](#_Toc386514044)

[3.3.12 Portability 39](#_Toc386514045)

[3.3.13 Other requirements 39](#_Toc386514046)

[4. Project Plan Overview 40](#_Toc386514047)

[4.1 Introduction 40](#_Toc386514048)

[4.2 Work Packages 40](#_Toc386514049)

[4.3 Milestones and Deliverables 40](#_Toc386514050)

[4.3.1 M1 – Report Findings (Interim Report) 41](#_Toc386514051)

[4.3.2 M2 – Application Design Specification 41](#_Toc386514052)

[4.3.3 M3 – Implementation 41](#_Toc386514053)

[4.4 Project Plan 41](#_Toc386514054)

[4.5 Time Management 43](#_Toc386514055)

[4.6 Meeting Plan 44](#_Toc386514056)

[4.7 Risk Management 44](#_Toc386514057)

[4.7.1 Event Driven Risks 44](#_Toc386514058)

[4.7.2 Evolving Risks 45](#_Toc386514059)

[4.7.3 Critical Contingency 45](#_Toc386514060)

[4.7.3.1 Loss of Data 45](#_Toc386514061)

[4.7.3.2 Time Delays 45](#_Toc386514062)

[5 IEEE Software Design Description 46](#_Toc386514063)

[5.1 Revision History 46](#_Toc386514064)

[5.2 System Architectural Design 46](#_Toc386514065)

[1.2.1 System Description 47](#_Toc386514066)

[5.2.2 System Architecture 48](#_Toc386514067)

[5.2.3 Design Constraints 51](#_Toc386514068)

[5.2.3.1 General Constraints 51](#_Toc386514069)

[5.2.3.2 Hardware Constraints 51](#_Toc386514070)

[5.2.3.3 Software Constraints 51](#_Toc386514071)

[5.3 Components Description 51](#_Toc386514072)

[5.3.1 Decomposition Description 52](#_Toc386514073)

[5.3.1.1 User 53](#_Toc386514074)

[5.3.1.2 Group 54](#_Toc386514075)

[5.3.1.3 Steganographic Plugin Manager 55](#_Toc386514076)

[5.3.1.4 Setting 56](#_Toc386514077)

[5.3.1.5 Mail 57](#_Toc386514078)

[5.3.1.6 Data Plugin Manager 60](#_Toc386514079)

[5.3.1.7 StegoText 61](#_Toc386514080)

[5.3.2 Class Diagram 62](#_Toc386514081)

[5.4 External Interfaces 63](#_Toc386514082)

[5.4.1 User Interfaces 63](#_Toc386514083)

[5.4.1.1 Login 64](#_Toc386514084)

[5.4.1.2 Main Menu 64](#_Toc386514085)

[5.4.1.3 Encode Text 65](#_Toc386514086)

[5.4.1.4 Decode Text 66](#_Toc386514087)

[5.4.1.5 Send Mail 67](#_Toc386514088)

[5.4.1.6 Receive Mail 68](#_Toc386514089)

[5.4.1.7 Settings 69](#_Toc386514090)

[5.4.2 External System Interfaces 70](#_Toc386514091)

[5.4.2.1 Steganographic Plugin 70](#_Toc386514092)

[5.4.2.2 Data Plugin 71](#_Toc386514093)

[5.5 Data Description 76](#_Toc386514094)

[6 Implementation 80](#_Toc386514095)

[6.1 Compilation Guidelines 80](#_Toc386514096)

[6.1.1 Target Environments 80](#_Toc386514097)

[6.1.2 System Requirements 80](#_Toc386514098)

[6.1.3 Compilation Guidelines 82](#_Toc386514099)

[6.1.3.1 Ubuntu™ Host and Target 82](#_Toc386514100)

[6.1.3.2 Microsoft® Windows™ Host and Target 85](#_Toc386514101)

[6.2 System Implementation 87](#_Toc386514102)

[6.2.1 Splash Screen 87](#_Toc386514103)

[6.2.2 Authentication 88](#_Toc386514104)

[6.2.3 Main Menu 88](#_Toc386514105)

[6.2.4 Encode Text 89](#_Toc386514106)

[6.2.5 Decode Text 91](#_Toc386514107)

[6.2.6 Receiving Encoded Messages 92](#_Toc386514108)

[6.2.6 Sending Encoded Messages 93](#_Toc386514109)

[6.2.7 System Administration 93](#_Toc386514110)

[6.2.7.1 General Settings 94](#_Toc386514111)

[6.2.7.2 Groups 94](#_Toc386514112)

[6.2.7.3 Users 95](#_Toc386514113)

[6.2.7.4 Libraries 95](#_Toc386514114)

[6.7.2.5 Mail 96](#_Toc386514115)

[7 IEEE Software Test Plan 97](#_Toc386514116)

[7.1 Revision History 97](#_Toc386514117)

[7.2 Introduction 97](#_Toc386514118)

[7.3 Test Items 97](#_Toc386514119)

[7.4 Tests Not Carried Out 98](#_Toc386514120)

[7.5 User Interface Tests (Graphical User Interface) 99](#_Toc386514121)

[7.6 Test Result Execution 99](#_Toc386514122)

[8 Critical Evaluation 100](#_Toc386514123)

[8.1 Steganographic Methods 101](#_Toc386514124)

[8.1.1 Open Space Encoding 102](#_Toc386514125)

[8.1.2 Synonym Replacement 103](#_Toc386514126)

[8.1.3 English Diversity 104](#_Toc386514127)

[8.1.4 Wayner’s Mimic Functions 105](#_Toc386514128)

[8.2 Application Issues and Future Recommendations 106](#_Toc386514129)

[8.2.1 Unknown Encoding 107](#_Toc386514130)

[8.2.2 Library Not Registered 107](#_Toc386514131)

[8.2.3 Path Execution 107](#_Toc386514132)

[8.2.4 Other Future Enhancements 107](#_Toc386514133)

[8.3 User Evaluations 108](#_Toc386514134)

[8.3.1 Icons 108](#_Toc386514135)

[8.3.2 Colours 108](#_Toc386514136)

[8.3.3 Interface Sizing 109](#_Toc386514137)

[8.3.4 Other usability issues 109](#_Toc386514138)

[9 Conclusions 111](#_Toc386514139)

[References 113](#_Toc386514140)

[Appendix 116](#_Toc386514141)

[Appendix 1 Class Diagram 116](#_Toc386514142)

[Appendix 2 HTML Help 117](#_Toc386514143)

[Appendix 3 Class Identifiers 125](#_Toc386514144)

[Appendix 4 Licensing 126](#_Toc386514145)

[Appendix 5 Test Case Plan 127](#_Toc386514146)

[Appendix 6 Test Execution 137](#_Toc386514147)

[Appendix 7 User Evaluations 155](#_Toc386514148)

## Table of Figures

[Figure 1: Original and Difference encoding ‘A’ 10](#_Toc386513871)

[Figure 2: System Diagram 21](#_Toc386513872)

[Figure 3: Send message use case 22](#_Toc386513873)

[Figure 4: Send message use case 23](#_Toc386513874)

[Figure 5: Encode text 24](#_Toc386513875)

[Figure 6: Decode text 25](#_Toc386513876)

[Figure 7: Login Prompt 27](#_Toc386513877)

[Figure 8: Progress / Status 27](#_Toc386513878)

[Figure 9: Menu 27](#_Toc386513879)

[Figure 10: Send Message 27](#_Toc386513880)

[Figure 11: Inbox 27](#_Toc386513881)

[Figure 12: Encode Text 28](#_Toc386513882)

[Figure 13: Decode Text 28](#_Toc386513883)

[Figure 14: Animated Display 28](#_Toc386513884)

[Figure 15 Logical Database Structure 36](#_Toc386513885)

[Figure 16: Logical Class Hierarchy 37](#_Toc386513886)

[Figure 17: Work Packages 40](#_Toc386513887)

[Figure 18: Project Task Plan 43](#_Toc386513888)

[Figure 19: The Software Process 47](#_Toc386513889)

[Figure 20: System Overview 47](#_Toc386513890)

[Figure 21: Deployment Overview 48](#_Toc386513891)

[Figure 22: Decode Text Sequence Diagram 48](#_Toc386513892)

[Figure 23: Decode Text Sequence Diagram 49](#_Toc386513893)

[Figure 24: Send Message Sequence Diagram 50](#_Toc386513894)

[Figure 25: Receive Message Sequence Diagram 50](#_Toc386513895)

[Figure 26: Stegaid Core Application Class Diagram 63](#_Toc386513896)

[Figure 27: Design - frmLogin 64](#_Toc386513897)

[Figure 28: Design - frmMainMenu 65](#_Toc386513898)

[Figure 29: Design - frmEncode 66](#_Toc386513899)

[Figure 30: Design - frmDecode 67](#_Toc386513900)

[Figure 31: Design - frmSendMessage 68](#_Toc386513901)

[Figure 32: Design - frmReceiveMessages 69](#_Toc386513902)

[Figure 33: Design - frmSettings 70](#_Toc386513903)

[Figure 34: Entity Relationship Diagram 79](#_Toc386513904)

[Figure 35: Library/Executable Dependency Directory Structure 86](#_Toc386513905)

[Figure 36: Implementation - Splash Notification 87](#_Toc386513906)

[Figure 37: Implementation - Authentication Display 88](#_Toc386513907)

[Figure 38: Implementation - Main Menu 89](#_Toc386513908)

[Figure 39: Implementation - Encode Text File Selection 89](#_Toc386513909)

[Figure 40: Implementation - Encode Text Method 90](#_Toc386513910)

[Figure 41: Implementation - Encode Text Status 90](#_Toc386513911)

[Figure 42: Implementation - Encode Confirmation 90](#_Toc386513912)

[Figure 43: Implementation - Decode Text File Selection 91](#_Toc386513913)

[Figure 44: Implementation - Decode Text Status 91](#_Toc386513914)

[Figure 45: Implementation - Decode Text Confirmation 92](#_Toc386513915)

[Figure 46: Implementation - Receive Messages 92](#_Toc386513916)

[Figure 47: Implementation - Received Message View 93](#_Toc386513917)

[Figure 48: Implementation - No Account Set Up 93](#_Toc386513918)

[Figure 49: Implementation - New Message 93](#_Toc386513919)

[Figure 50: Implementation - General Settings 94](#_Toc386513920)

[Figure 51: Implementation - Groups 95](#_Toc386513921)

[Figure 52: Implementation – Users 95](#_Toc386513922)

[Figure 53: Implementation – Libraries 96](#_Toc386513923)

[Figure 54: Implementation - Mail 96](#_Toc386513924)

[Figure 55: Example Evaluation Radial Chart 100](#_Toc386513925)

[Figure 56: Target Score Radial Chart 101](#_Toc386513926)

[Figure 57: Open Space Algorithm Evaluation 103](#_Toc386513927)

[Figure 58: Synonym Replacement Algorithm Evaluation 104](#_Toc386513928)

[Figure 59: UK / US English Algorithm Evaluation 105](#_Toc386513929)

[Figure 60: Wayner's Mimic Functions Evaluation 106](#_Toc386513930)

[Figure 61: Top-Down Views and Error Prevention 110](#_Toc386513931)

## Table of Tables

[Table 1: Acronyms 8](#_Toc386513932)

[Table 2: Definitions 8](#_Toc386513933)

[Table 3: Suitability Matrix 13](#_Toc386513934)

[Table 4 System Definitions 20](#_Toc386513935)

[Table 5: Send message 23](#_Toc386513936)

[Table 6: Receive message 24](#_Toc386513937)

[Table 7: Encode Text 25](#_Toc386513938)

[Table 8: Decode text 26](#_Toc386513939)

[Table 9: Interaction guidelines 29](#_Toc386513940)

[Table 10: Recommended System Requirements 30](#_Toc386513941)

[Table 11 Communication Interfaces 31](#_Toc386513942)

[Table 12 Basic Operations 31](#_Toc386513943)

[Table 13 Administrative Options 32](#_Toc386513944)

[Table 14 User Characteristics 33](#_Toc386513945)

[Table 15 Operational Constraints 34](#_Toc386513946)

[Table 16 Failure Scenarios 38](#_Toc386513947)

[Table 17 System Maintainability 39](#_Toc386513948)

[Table 18 System Portability 39](#_Toc386513949)

[Table 19: Time Management 44](#_Toc386513950)

[Table 20: Partial Meeting Log 44](#_Toc386513951)

[Table 21: IEEE Software Design Specifications Revision History 46](#_Toc386513952)

[Table 22: Component Overview 52](#_Toc386513953)

[Table 23: Decomposition Description 52](#_Toc386513954)

[Table 24: The User Component 54](#_Toc386513955)

[Table 25: The Group Component 55](#_Toc386513956)

[Table 26: The Steganographic Plugin Manager Component 56](#_Toc386513957)

[Table 27: The Settings Component 56](#_Toc386513958)

[Table 28: The Mail Component 60](#_Toc386513959)

[Table 29: The Data Plugin Manager Component 61](#_Toc386513960)

[Table 30: The StegoText Component 62](#_Toc386513961)

[Table 31: User Interface Component frmLogin 64](#_Toc386513962)

[Table 32: User Interface Component frmMainMenu 65](#_Toc386513963)

[Table 33: User Interface Component frmEncodeText 66](#_Toc386513964)

[Table 34: User Interface Component frmDecodeText 67](#_Toc386513965)

[Table 35: User Interface Component frmSendMessage 68](#_Toc386513966)

[Table 36: User Interface Component frmReceiveMessages 69](#_Toc386513967)

[Table 37: User Interface Component frmSettings 70](#_Toc386513968)

[Table 38: The Steganographic Plugin Interface 71](#_Toc386513969)

[Table 39: The Data Plugin Interface 76](#_Toc386513970)

[Table 40: Data Description Overview 76](#_Toc386513971)

[Table 41: Data Dictionary 78](#_Toc386513972)

[Table 42: Implementation Compilation Guidelines 80](#_Toc386513973)

[Table 43: System Requirements 81](#_Toc386513974)

[Table 44: Library Requirements 82](#_Toc386513975)

[Table 45: Ubuntu Variant Compile Guidelines 85](#_Toc386513976)

[Table 46: Microsoft® Windows™ Build Prequisites 85](#_Toc386513977)

[Table 47: Microsoft® Windows™ Compilation Guidelines 87](#_Toc386513978)

[Table 48: Software Test Plan Revision History 97](#_Toc386513979)

[Table 49: Test Items 98](#_Toc386513980)

[Table 50: Items not to be tested 99](#_Toc386513981)

[Table 51: Evaluation Areas 101](#_Toc386513982)

## 

## Abstract

*Steganography is the art of hiding information within other less conspicuous information to prevent eavesdropping by way of hiding its existence in the first place. It is the focus of this research to investigate various methods of steganography in modern digital communication. At first the focus will be to investigate steganography as a whole before specifically investigating text based steganography. This specific area has been chosen because of the limited research in this area and text is everywhere from documents, to source code and the web. The area will conclude with the strengths and weaknesses within each method and suggest solutions as alternative methods. In order to complete this review extensive research of conference proceedings and journals related to steganography was investigated to identify in basic terms the methods that can be applied. The significance of this information is to provide a single document combining the basic methods that can be applied based on existing research, this can then be used by steganographers and steganalysts alike. It was found that whilst text based steganography has had lesser research than other methods such as with the use of images, many methods could be used to encode text to prevent awareness of hidden information. It was found in general format based systems are highly subject to the process of steganalysis and discovery. In general random and statistical generated methods create a cover text but do not necessarily make semantic sense; that is the subject matter of each sentence has little or no relation to the next sentence. Linguistic Steganography can use natural language processing to hide information but again is still subject to analysis particularly if the basis for the cover text is an existing document. It was found that the best methods to hide information should not use a single scheme, but a hybrid of many schemes. In order to further hide information, text should be compressed, encrypted and then hidden in a cover document.*

## Introduction

Encryption of secret messages occur all the time, when we authenticate against websites, perform banking and even google searches are now encrypted by default. Whilst we feel fairly secure in the knowledge that encryption takes place, the very existence of the encryption will alert network peers, rogue routers and so forth to the presence of hidden information. With more and more sophisticated ways of breaking encryption with the power of the cloud and the potential for man in the middle attacks the objective seems clear. Do not just to hide information, but to hide the existence of information. The field of steganography has had much research especially with image based steganography but lesser research has taken place with text based steganography, which the web is mostly composed of. Beyond email and watermarking, steganography has not become mainstream, yet the very purpose of steganography is not to secure information as encryption but to hide its very existence in the first place. It is therefore my intention to review steganography and common approaches to that can be applied to hide information. This document can be a basis for future research or application in this field. In the first part we will introduce methods categorised by cover medium and suggest ways this can be accomplished. The second part will focus specifically on text based steganography as this area has had the most limited research. To date (Bennett, 2004) has provided an overview of text steganography but limits research to the field of linguistic steganography. The difficulty in compiling this document arises when much research is based on existing identified algorithms, but essentially are the same approach. To overcome this, the intention is to review related literature and summarise the basic concepts to each approach. I will then highlight the basic problem with each method. To conclude I will summarise the issues identified and suggest approaches that may overcome these issues. A project plan will detail milestones and issues likely to be encountered in this project. This will deal with time constraints and an overview of the work to be undertaken. An analysis of the problem area will be investigated to form a Software Requirements Specification with a view to using the identified methods in a software application.

## Acknowledgements

I would like to thank my Supervisor, Dr. Kevin Curran for the opportunity to take on this interesting project and giving the guidance I need to complete this project. Further advice given by Dr. Mia Siddique for his input that aided in the requirements investigation such as language selection. I would also like to thank the Open Source community, namely the Qt Project and the Poco Project for the provision of Free and Open Source Software (FOSS) for which this project would be impossible.

Ultimately, special thanks are due to the University of Ulster, School of Computing and Intelligent Systems for the opportunity to study and investigate this valuable field.

## Declaration

This document is submitted to the University of Ulster in support of my application for the degree of Bachelor of Science in Computer Science. I declare that this is all my own work and does not contain unreferenced material copied from any other source. I have read the University’s policy on plagiarism and understand the definition of plagiarism. If it is shown that material has been plagiarised, or I have otherwise attempted to obtain an unfair advantage for myself or others, I understand that I may face sanctions in accordance with the policies and procedures of the University. A mark of zero may be awarded and the reason for that mark will be recorded on my file.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Robert Lockwood

## Acronyms and Definitions

|  |  |
| --- | --- |
| ASCII | American Standard Code for Information Interchange, specifically a defined 8 bit character set for encoding information. |
| BPP | Bits Per Pixel. |
| CBR | Constant Bit Rate |
| CFG | Context Free Grammar |
| DCT | Discrete Cosine Transform |
| DOCX | Office Open XML (OOXML) Document Format |
| DWT | Discrete Wave(let) Transform |
| EOF | End Of File. A marker in a file that signifies the end of the file on a file system or stream. |
| EXIF | EXchangable Image File format. |
| JPEG | Joint Photographic Expert Group (also the name of an image file format derived from the group) |
| LSB | Least Significant Bits or Least Significant Bytes. |
| MVC | Model View Controller Paradigm. In this case the Model; being the database, the View(s) being the User Interfaces and the Controller(s) being the objects that provide the validation and logic. |
| ODF | Open Document Format |
| PE | Portable Executable |
| VBR | Variable Bit Rate |

Table 1: Acronyms

|  |  |
| --- | --- |
| Bi-Gram | Is *(in the case of computer linguistics)* a window into a series of words, this window is two words wide, the test being the two words make sense logically although possibly nonsense. |
| N-gram | An n-gram is therefore n length window of words. As such unigram is just one word and a trigram is 3 words. |
| Steganalysis | The process of analysing a cover medium for presence of hidden data. |
| Steganalyst | An analyst performing tests in the case of steganalysis |
| Steganographer | A person undertaking the process of steganography. |

Table 2: Definitions

# 1 Steganography

The focus of this research is to investigate algorithms (best methods) of hiding information in other information in such a way a peer is unaware of its true content such as a third party on a network. The origins of steganography was first coined by Trithemus who coined “Steganographia” which means “Concealed Writing” (Bennett, 2004). Today steganography has been extended to not only include text but also images and any other object. For example, text can be embedded in images, video or other objects and vice versa with enough data to hide information in. Steganography can fall into five categories: Images, Video, Audio, Text (Bhattacharyya, et al., 2010) and other objects such as executables which does not fit into the four original categories that Bhattacharyya described.

In general no matter the cover medium, steganography can be classified into two areas; key based systems and keyless based systems. A key based system hides information in a cover medium and generates a key for transmission on a separate channel. Only the sender and target receiver are aware of this key which would be used to expose the hidden information in a cover material. Keyless systems employ only the insecure channel to transmit and receive information but the sender and the receiver must be aware of the encoding algorithm in order to decipher the original information.

In this review, an overview of the common methods in steganography will be investigated to draw understanding in the field. Further exploration will take place in text based steganographic systems due to the more limited research in this area. Most research to date has been completed in the field of digital steganography with particular emphasis on Images, Video and Audio.

## 1.1 Image Based Steganography

Image based steganography is *usually* the process of hiding text in an image by various means without distorting the picture noticeably to the user. Other information can also be inserted such as other images. Significant research has taken place in this area (Bennett, 2004) and as such a brief overview of the most common methods will be explained.

### 1.1.1 File Format Manipulation Methods

Some Image based methods do not employ modification of the image itself but can the file container in which the image is stored. One such scheme shown by (Cheddad, et al., 2010) explains that files can be appended to the EOF marker to hide data. Whilst this is ultimately very simple to implement for a small amount of information an image file significantly larger than the expected file size for the resolution may raise eyebrows and in itself cause further investigation. Certain Image formats also have areas within the format to hide small amount of data such as the EXIF field in images.

### 1.1.2 LSB Encoding Schemes

Various research papers have used the encoding of data within the Least Significant Bits (LSBs herein) within the pixels of the cover image. For example (Rig & Tuithung, 2012) also shown that the letter A can be coded into 3 pixels using the 3 LSBs of each pixel (3 BPP x 3 Pixels = 9 Bits which is enough to cover the 8 bits of the letter A). Figure 1: Original and Difference encoding ‘A’ shows 3 pixels one without encoding and one with the letter ‘A’ encoded (zoomed).

|  |  |
| --- | --- |
|  |  |

Figure 1: Original and Difference encoding ‘A’

As you can see this method cannot easily be identified to the person using viewing the image. You may just about see the far right pixel slightly discoloured. If a steganalyst could detect the hidden data if the image is significantly malformed. Detection of hidden information is even easier given the original image and comparing to the cover image. Given (in this case) a 1024 x 768 image, using 3 pixels per character, 262144 characters can be encoded or squashed together to form 294912. Given that much of the ASCII character set is unused a way to convert more information into fewer pixels would be use of a custom character set that omits unused characters. (Rig & Tuithung, 2012) does this by way of Huffman Encoding. In the case of (Rig & Tuithung, 2012), they modify the DCT blocks of pixels in JPEGs but in essence any format can be used to encode information such as within bitmaps. The frequency of the characters being used form shorter bit lengths (such as ‘A’). The letter ‘Z’ would less often be used so is located at the bottom of the binary tree and thus has a longer bit length.

### 1.1.3 Other Methods

Other methods of encoding information into images can be by manipulating the way the file is formatted by itself. (Rig & Tuithung, 2012) notes JPEG uses DCT blocks of 8x8 pixels as a form of compressing pixels and near pixels. Beyond JPEGs, different solutions can be applied to PNGs and other types.

## 1.2 Video/Animation Based Steganography

Videos on their very size make an attractive alternative to extremely large amounts of information in. For small amounts of data video based steganography would take a considerable amount of computational time (Balaji & Naveen, 2011) and network bandwidth, however, it can be suitable for large amounts of information. Depending on the format data can be held in frame by frame (within the pixels of the frame). Videos have another dimension in which information can be held, time.

### 1.2.1 Frame Manipulation

As with image based steganography, individual frames (which are images in their own right) can also be modified by changing the LSB pixels of the frame. As this has already been covered in Image Based Steganography, it will not be repeated here as the concept is the same.

### 1.2.2 Time Manipulation

Videos are divided into set of frames. Video formats can fall into one of two categories and some video formats support both: CBR and VBR. Beneath that frame rates can also vary. On high frame rate video formats, a single frame can contain a hidden frame. Due to the way our eyes work if the colour is nearly matching the rest of the frames the watcher would not notice. Whilst it can be used in steganography, it has also been used in subliminal messaging.

## 1.3 Executable File Formats and Binary Execution

Steganography can take place in other objects and in theory any object. Executable files for the most part can also hide data and often do. Executable files do not necessary harbour the main application program itself but in some cases viruses, spyware and adware also.

The Microsoft Portable Executable not only has sections for code (.text/.code segment) but data also; such as strings. Images are often included to form icons or *embedded resources* that are embedded into the application without having the resources externally stored. To the user the embedded content is hidden but exposable by using a resource extraction tool. (AL-Nabhani, et al., 2010) propose the use of header field of the portable executable. Immediately after the header, the hidden information would be stored. By updating the offsets of the starting program code, data and text segments, the capacity is high. (AL-Nabhani, et al., 2010) do note, however, in order for the application to execute it must first be downloaded and run (or installed and run).

## 1.4 Audio Steganography

Like images and video, the least significant bits of audio data can also be modified. Because of the minimal modification to the generated sound, to human ears no distortion is identified. As with all plain LSB methods, steganalysis can potentially uncover hidden information. To overcome this (Asad, et al., 2001) proposes selective modification of lower bits depending on the value of the most significant bits. This would make steganalysis more difficult but not immune particularly of the steganalyst is aware of the algorithm.

The audible range of human hearing is 20Hz (Cycles per Second) to 20KHz (Cuttnel & Johnson, 1998). Outside if this range humans cannot hear. If a significant safety margin is applied, we can encode audio below and above this range. In fact (Gopalan & Wenndt, 1998) did just this, while the selected frequencies in use are not outside of the audible range, low frequencies were used with the cover audio on top. (Gopalan & Wenndt, 1998) noted that this method is susceptible to noise and can cause the method to fail, any lossy medium could also cause data loss.

## 1.5 Suitability Matrix

In order to compare the methods identified, it must be compared to suitability for a given purpose.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cover Information | Text | Image | Audio | Executable | Video |
| Hidden Information |
| Text |  |  |  |  |  |
| Image |  |  |  |  |  |
| Audio |  |  |  |  |  |
| Executable |  |  |  |  |  |
| Video |  |  |  |  |  |

Table 3: Suitability Matrix

The matrix (traffic light system) in Table 3 shows the suitability of encoding a type of information in another type of information. It can be seen that hiding videos in Audio, Images and Text is possible but impractical because of the large size requirements. Images and Audio can be encoded in such a way that they can be embedded into text with enough cover text, but most of all, all information can be interchangeable.

# 2 Text Based Steganography

The focus of this research is to analyse methods with relation to text based steganography. This form of steganography has had lesser research with comparison of other methods such as images, therefore is the focus of this research. *Steganographia*, literally means “covered writing” (Bennett, 2004), and although this has now been extended to include images and other formats, the origins of steganography involve text. Text Based Steganography can fall into one of three categories: format based, linguistic and random/statistical generation (Bhattacharyya, et al., 2010). In this chapter, we uncover the basic methods that can be applied to any language and not a specific language such as Chinese or Indian.

## 2.1 Format Based Systems

Format based steganography relies on a selected cover text and changing properties within the cover text such as punctuation, or spelling to hide information. More commonly information can be held in white space and non-printing characters.

### 2.1.1 Punctuation Amendment

The use of punctuation has been suggested as a way of hiding bits. Commas and Full-stops are explored by (Agarwal, 2013). By selecting appropriate points of insertion of punctuation bits can be represented. For example, a full stop might represent 00, comma 01, exclamation 10 and question mark 11. Whilst, if the punctuation is logically correct, there is no reason to believe such an algorithm would ever be discovered.

### 2.1.2 Spelling Misappropriation

(Shirali-Shahreza, 2008)propose the way in which words are spelled is a method in which to hide information. For example, in UK English the word “favourite” and US “favorite” have the same meaning but is inconspicuous in that the difference could represent a zero or one. Whilst this method is very simple, a selective approach to chosen words would have to take place as there are more English word spellings in common than different. In order to encode a large message, it could be estimated you get one bit per word, eight words to make a single character. On its own, the method would not be feasible for a significant amount of text due to the low encode result. If the method was combined with other methods to form a hybrid more bits can be encoded.

### 2.1.3 Open Space

In some documents (namely HTML), spaces are ignored as are carriage returns. For example in order to structure the document <p> and <br> tags are used. The first space is accepted, but any additional spaces are explicitly ignored by the user’s browser. In order to overcome this website developers have to encode the &nbsp; code. Indeed (Barilnik, et al., 2007) has explored this and whilst not suitable for normal documents can be used to hide information in source code, html documents or anywhere formatting is ignored and justified text. This can be useful for hiding a signatures (a form of watermarking) for copyright or hidden data. To a trained eye, this can be easily be subjected to identification through steganalysis. (Barilnik, et al., 2007) also noted that opening a HTML document in Microsoft Word and enabling the formatting marks, spaces are easily identified.

Other ideas include the colouring of white spaces (a hybrid method of the open space method above and applying colour). Naturally a white space coloured red on a white background is still white. Consequently for each space 3 bytes of information can be encoded (R, G and B, 1 byte each not including a possible Alpha channel).

### 2.1.4 Line Spacing

Documents such as ODF and DOCX store document formatting by modulating the line spacing (by tiny amounts) to encode bits. (Jalil & Mirza, 2009) explains line shifting by a small amount of pixels can be used in watermarking as a form of document protection. Such a method can also be used in steganography to hide small amount of information or as part of a larger strategy in combination.

### 2.1.5 Font Manipulation

For printed material and direct to screen there are a number of using fonts to hide information. For some documents such as Microsoft Word, the user can easily see where the font changes by looking at the font field in the top icon bar. In HTML this could be used as viewers are not immediately aware of the font change. The following example contains two x characters, one bit 0 encoded and the other bit 1.

|  |  |
| --- | --- |
|  |  |

We can easily see the difference, but as part of a larger sentence it is difficult to pick up the differences. A computer application that can detect pixel level differences would easily pick up on these differences. Whilst these images are enlarged for visibility, what you may not notice is the additional character encoded in the “Hello Wor”. In this case the letter is “H”. Whilst un-zoomed and appropriately encoded, it is not immediately obvious. On paper, a pixel can be very small, for example a 600dpi Printer, 1dot equal 1/600th of an inch.

The above methods are all extremely simple and whilst can either on their own or by using hybridisation are all candidates for steganalysis. The more encoding schemes employed, the more difficult it is to decipher the hidden information. In these cases the cover text can already exist but must be modified to hide the information in.

## 2.2 Random and Statistical Generation

The second category of text based steganography involves generation of a cover text based on either randomisation or likelihood of correctness. This differs from linguistic based steganography which attempts to create a valid natural language text using a range of algorithms. This area borders the realms of computer science and language by way of natural language processing and computer generated texts.

### 2.2.1 Markov Chains

(Hernon Moraldo, 2012) suggests the use of generation of a cover text hiding the information by way of markov chains. Markov Chains are often used to generate language based on words, bi-grams and so forth). (Hernon Moraldo, 2012) uses such a method to create a markov chain based on a cover text. In the example provided, the book “War and Peace” is used as the markov generation source. It is noted, that whilst unigram based steganography is low quality, bigram generated texts are better. Despite this it can still be identified there are issues, take the following example:

”Be a square for fuel and kindled ﬁres there. Secondly it was hard to hide behind the cart and remained silent. He feels a pain in the now cold face appeared that the man continually glanced at her as though they stumbled and panted with fatigue. With a deep.”

Certain words are out of context in that “it was hard to hide behind the cart and remain silent.” In this case, this approach can fall under linguistics and statistical generation.

### 2.2.2 Context Free Grammar

(Wayner, 1991) created the well known mimic functions and has been cited by a large proportion of text steganography research papers. In this manual, he describes the process of the generating context free grammar using his mimic functions which are based on probabilities.

An example provided shows that whilst the generated text is legible, the result does not make sense:

Paul is dead! I am the walrus!

Buy something right now. Do not shoplift. Buy! Buy!

Here are the plans to the Overthruster, Sergei.

Yoyodyne forever.

## 2.3 Linguistic Steganography

Linguistic Steganography is the third major category, this involves the creation of a natural language text (or modification of an existing text) in order to hide information.

### 2.3.1 Synonym Replacement (Lexical Substitution)

Coupled with a thesaurus, where the sender and receiver have the same thesaurus. Words based on existing text can be modified to represent values. This was proposed by (Topkara, et al., 2006) and others to act as a watermarking technique to protect documents. It has come to the attention of this researcher that you can have multiple related words. A proposed method could be to use a thesaurus and based on the value of the word an alternative is used. For example “cat” == “feline”. This could be extended further, there are 16 words or more (synonyms) for cat:

bobcat cheetah cougar jaguar kitten leopard lion lynx panther puma puss pussy tabby tiger tom tomcat

In order to put things in context the thesaurus would have to have “context”. (Topkara, et al., 2006) notes that a generated sentence may lose its context and may not make semantic sense. You’ll note the following sentence contains a syntactically correct original sentence, a possible flawed sentence and a steganographic sentence:

*My pet cat has been to the vet today.*

*My pet tiger has been to the vet today.*

Tomcat is tagged with “cat”, “pet”and “tiger”, whilst tiger is still valid, it may raise some eyebrows. Tomcat is the 16th word (0b10000). Reverse lookup suggests (with some computation) tomcat is a synonym for cat.

The above method would work if the thesaurus on the sender and receiver are identical, and assuming we have a thesaurus capable of identifying “is a” relationships reverse lookup is not a problem. (Thinkmap Inc, 2013) has such an application that shows relationships between words such as “is part of”, “pertains to” and more specifically “is a type of”. Other methods can also be used such as negativity (antonyms) of positivity to represent bits also.

### 2.3.2 Sentence Modification

A variation on synonym replacement is to replace the whole structure of the sentence. The previous method focusses on the modification of words (either synonyms or antonyms), this method requires use of natural language processing tools. (Chang & Clark, 2010) proposes the use of Google n-gram data to verify the correctness of the sentence although they have not proposed any type of medium in which the cover text be a basis for steganography. They have suggested news articles, but comparison with the original article and the modified text would yield suspicion. Lets take a look at the following sentence:

*The beginning of this month*

The sentence can be modified whilst still meaning the same thing:

*This month in the beginning . . .*

The sentence is correct but have bits (zeroes and ones) encoded depending on its structure. This method is the simple whilst remaining understandable to us. The issue arises when, if based on an existing article such as a news story, differences would be noticeable.

# 3 IEEE System Requirements Specification

Document Revision

|  |  |
| --- | --- |
| 1.0 | Initial Specification |

## 3.1 Introduction

### 3.1.1 Purpose

The purpose of this document is to clearly define the requirements of the eventual to be designed, developed and tested software system with relation to text based steganography. This document *shall* conform to approximate industry standard conventions with regards to the IEEE System Requirements Specifications (Institute of Electrical and Electronics Engineers, 1998). The target audiences for this document is the stakeholders in this project, in this case, my project supervisor, any secondary auditors and the developer himself. This document will aid in agreement between stakeholders as to project requirements and will be iterated over to accommodate new facts identified through elicitation. The current version (1.0) is the initial version, further requirements elicitation will enable further revision to this document.

### 3.1.2 Definitions

Table 4 System Definitions defines keywords for those persons not directly involved in this area of computer science to aid in better understanding.

|  |  |
| --- | --- |
| Steganography | The process of hiding information in unrelated data to prevent “suspicion” of the very presence of this information. |
| Client | A person sending or receiving a steganographic message, or a educator learning the processes involved in text based steganography. |
| Operating System | Software which manages the physical hardware of a computer system. Examples include Microsoft® Windows™ and Linux (Ubuntu, Android (based on Linux)). |
| Runtime | A software system that enables software to run in a given format. It is usually installed as part of the operating system or packaged with the application. Examples include C++ Runtime. |

Table 4 System Definitions

### 3.1.3 Scope

This document will examine the requirements of the project, identifying key points identified through elicitation. An examination of the system is provided along with any constraints or issues. The purpose of this document is not to design to product but to confirm to a clearly defined specification to what is required of the product and is a basis for the future design.

### 3.1.4 Overview

This specification will firstly be targeted to all audiences (Section 3.2), to identify key requirements for the software system and the basis for the next section (3.3). Section 3.3 is primarily target towards software designers and developers of the software system. This section will be far more in depth to aid in generation of the design, implementation and testing phases.

## 3.2 Overall Description

This section is intended for all audiences and will detail the software product to be designed.

### 3.2.1 Product Perspective

A need was identified that a text based steganographic system, codenamed “stegaid” should be investigated and developed for two purposes:

1. An education tool to show methods that can be applied to texts in order to hide messages.
2. A general purpose tool, that can hide messages and recover messages across the internet using identified methods and as such a communication channel across the internet is insecure. The presence of encryption will alert others to the fact that information that may be sensitive. Obscurity through text based steganography may not. A mail transfer agent could act as a medium in which information can be transferred to others or an intermediate web server backed by a database.

The software system will be system independent and not play part of another system (as a subsystem). The system will act as a client for a database storage system and mail server system to facilitate storage and communications to other clients. The system is entirely new and will not be a replacement for an existing system.

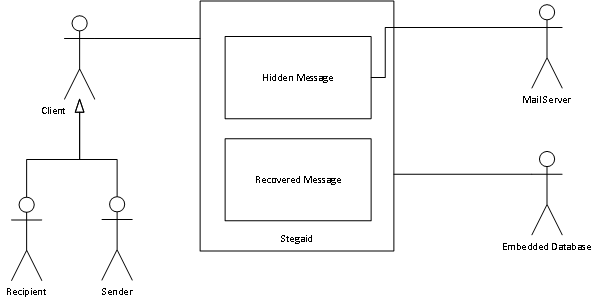


Figure 2: System Diagram

The diagram detailed in Figure 2: System Diagram shows an overview of the system with two actors, and two external interfaces. The two actors are the sender and the recipient of the message and the other significant actors are the mail server to facilitate the sending and receipt of messages. The final external actor is the database storage system to be used, whether embedded or external service with preference to both. Figure 3: Send message use case through to Figure 6: Decode text define the use cases for the target application.

1. Send Message

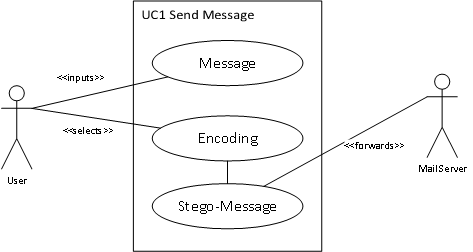


Figure 3: Send message use case

|  |  |
| --- | --- |
| Use Case Name | Send Message |
| Iteration | Focused |
| Summary | User chooses to send a hidden message to another person via the public internet. |
| Basic Course of Events | 1. Message is entered with recipient details 2. User selects encoding or full hybrid, (the method in which to hide the information) 3. The application forwards the message to the selected mail server for processing |
| Alternate Flows | - |
| Exception Paths | 1. Text is too large for mailbox to handle - (alert user) 2. Mail Server not available (offline) – (alert user) 3. Network Connectivity not available (offline) – (test network connectivity before message input, alert user) 4. Email Address Non Existent |
| Extension Points | - |
| Triggers | User selects option to send message |
| Assumptions | - |
| Pre-conditions | 1. Internet Connected 2. One steganographic method available |
| Post-conditions | 1. Message Sent Successfully 2. Steganographic-Key Generated |

Table 5: Send message

1. Receive Message

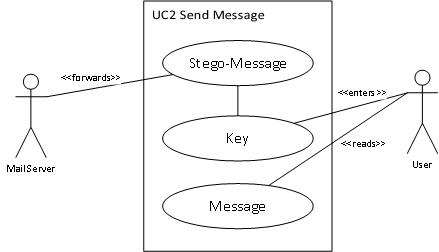


Figure 4: Send message use case

|  |  |
| --- | --- |
| Use Case Name | Receive Message |
| Iteration | Focused |
| Summary | User checks for message for reading after inputting steganographic key (if provided). |
| Basic Course of Events | 1. System checks mailserver for queued messages to be forwarded 2. User selects message and enters steganographic key 3. Message is decoded and displayed for reading |
| Alternate Flows | 1. User does not read message (unread) 2. No messages to forward (empty mail box) |
| Exception Paths | 1. Authentication Failure for Mail Server 2. Mail Server not available (offline) – (alert user) 3. Network Connectivity not available (offline) – (test network connectivity before attempting access, alert user) 4. Steganographic-Key invalid 5. Message Unable to be decoded (unknown algorithm) |
| Extension Points | - |
| Triggers | User selects menu option to receive message |
| Assumptions | - |
| Pre-conditions | 1. Internet Connected 2. One steganographic method available 3. Mail Server credentials stored for user |
| Post-conditions | 1. Hidden message displayed |

Table 6: Receive message

1. Encode Text

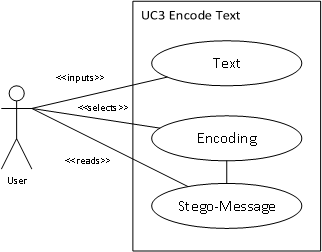


Figure 5: Encode text

|  |  |
| --- | --- |
| Use Case Name | Encode Text |
| Iteration | Focused |
| Summary | User chooses to encode text in some cover text (hide information) |
| Basic Course of Events | 1. User inputs text from input box or file 2. User selects encoding and output 3. Steganographic Text is displayed |
| Alternate Flows | 1. User cancels |
| Exception Paths | 1. File Not Readable |
| Extension Points | - |
| Triggers | User selects menu option to hide text |
| Assumptions | - |
| Pre-conditions | 1. One steganographic method available |
| Post-conditions | 1. Output steganographic text to screen or file |

Table 7: Encode Text

1. Decode Text

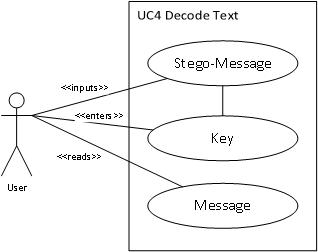


Figure 6: Decode text

|  |  |
| --- | --- |
| Use Case Name | Decode Text |
| Iteration | Focused |
| Summary | System decodes steganographic text with optional steganographic key (method dependent) and displays to user |
| Basic Course of Events | 1. User inputs steganographic text or selects file 2. User inputs steganographic key received on a separate channel 3. System decodes message for display or output to file |
| Alternate Flows | 1. User cancels |
| Exception Paths | 1. File cannot be read/written 2. Steganographic Key invalid 3. Steganographic Method not supported or text not decipherable |
| Extension Points | - |
| Triggers | User selects menu option to decode steganographic text |
| Assumptions | - |
| Pre-conditions | 1. One steganographic method (algorithm) available |
| Post-conditions | 1. Hidden text displayed or saved |

Table 8: Decode text

### 3.2.2 System Interfaces

The system should provide the capability of encoding messages using a number of steganographic techniques identified by literature review that can apply to text or the generation of text (encode).

The system will also provide the capability of decoding messages having identified the encoding method and display to the user the message, text or code (decode).

### 3.2.3 User Interfaces

In order to accomplish the two main user targets identified in the Scope of this specification, the application must target educational users explaining the process of encoding and decoding hidden text in a cover text. The application must also target users who wish to use the software system as a tool. Industry standard conventions with relation to human computer interaction shall be adhered to but due to the nature of the application certain features such as shortcuts via special function keys will be avoided and alternatives suggested.

Logical Flow

1. Upon launching of the application the user can expect a login dialog such as the one in Figure 7: Login Prompt, with guest mode enabled if the option is set.
2. Upon login or where heavy processing is taking place, a suitable message and progress bar should be indicated (Figure 8: Progress / Status).
3. A menu should display options for actions that can take place. An example of which is detailed in Figure 9: Menu. A toolbar or status bar has been indicated as not needed due to the unknown target device.
4. The user should be able to select an option but only if the option is available, for example, mail functions can only be accessed if mail server information is stored within the database.

|  |  |  |
| --- | --- | --- |
| Figure 7: Login Prompt | Figure 8: Progress / Status | Figure 9: Menu |

1. Email can be sent (Figure 10: Send Message) and received (Figure 11: Inbox)

|  |  |
| --- | --- |
| Figure 10: Send Message | Figure 11: Inbox |

1. Text can be encoded (Figure 12: Encode Text) and decoded (Figure 13: Decode Text)

|  |  |
| --- | --- |
| Figure 12: Encode Text | Figure 13: Decode Text |

1. The primary purpose of this software is to show how text based steganography takes place. Figure 14: Animated Display shows such an example animated display, however the screen is likely to change depending on the method being employed at the time. This feature can be disabled for users who wish to use the application as a tool in which case a progress bar will be displayed.

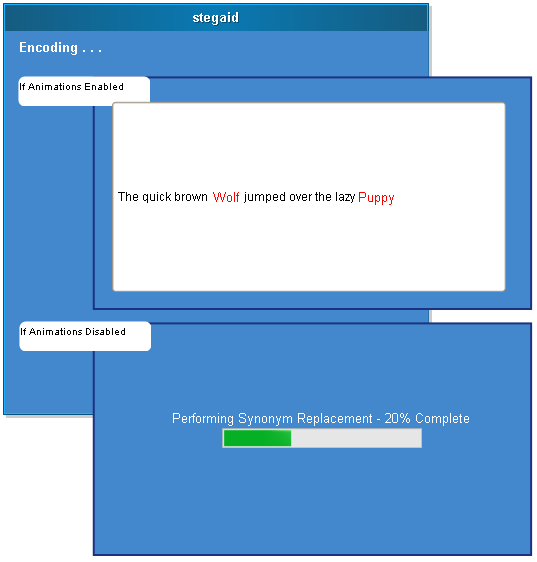


Figure 14: Animated Display

Interaction Guidelines

|  |  |
| --- | --- |
| Visibility of system status | The use of progress bars (or animations) will be displayed at all points significant processing is taking place. |
| Match between system and the real world | Industry standard controls will be used to aid familiarity to new users. Error messages will be displayed in clear English with preference to no errors. |
| User control and freedom | Abort should be implemented on timely operations such as encoding and decoding of texts. |
| Consistency and standards | Internal consistency should be applied across platforms, forms design and fonts are easy to read. Forms are dynamic to the size and resolution of the display in use. |
| Error prevention | The hiding of mail functions when no configuration is present. The hiding of guest mode if not enabled. |
| Recognition rather than recall | Wizard style interface presenting only appropriate display at that point in time. Text boxes clearly disabled when text cannot be entered. |
| Flexibility and efficiency of use | That lack of function keys on all devices prevent use. To alleviate this a console based system will also be implemented enabling input and output via batch/shell scripts enabling the application itself to be part of another program or independent. |
| Aesthetic and minimalist design | The clear looks design and minimalism will enable students learn rather than spending more time figuring out the interface. |
| Help users recognize, diagnose, and recover from errors | Simple dialog boxes will be shown for open and save file functions preventing errors. Error dialogs shall be displayed where appropriate. The system shall never crash except in situations beyond the developer’s control (Operating System failure). |
| Help and documentation | A help icon (using the industry format question mark) shall be displayed on each display with contextual help on the task in hand. |

Table 9: Interaction guidelines

### 3.2.4 Hardware interfaces

There are no specific hardware interfaces beyond that of a standard computational device. The target software system must support platform independence (i.e. not specific to x86 instruction set or ARM). The software will not directly interact with any specific hardware interfaces.

A recommended system requirement is therefore the defined in Table 10: Recommended System Requirements.

|  |  |
| --- | --- |
| Processor | Physical Processor capable of running a host Operating System with appropriate C++ runtime, i.e. Linux or Windows and derivatives. |
| Memory | The application will be as lightweight as possible thus memory requirements should be no greater than the host operating system requirement plus 64Mb RAM. |
| Hard Disk | The hard disk requirements will be minimal. 256Mb available for host application. |
| Network | Not required for the educational mode or the steganography portion of the application. For the application to make use of mail services network capabilities should be provided. |
| Input | A form of input would be needed. Attention should be applied during the design phase to allow for differing forms of input such as touch, mouse, keyboard and so forth. |

Table 10: Recommended System Requirements

### 3.2.5 Software interfaces

Internal Interfaces:

The application should be platform independent and should link against a suitable graphics library, network library and any other required software libraries to provide the expected output. The ability to load libraries on demand (plugin architecture) shall be necessary therefore preventing the need for a software recompile a new algorithm or method is implemented.

External Interfaces:

The application will make use of mail transports for the sending and receiving or steganographic texts and encode and decode as necessary. The use of a database management system for configuration, user authentication and such like will be necessary. The database management system will be external (set by options within a configuration portion of the application [multiple client shared configuration]) or internal (single client, not shared configuration).

### 3.2.6 Communication Interfaces

The application will use standard network capabilities such as the internet for the transmission and reception of information. As is standard with mail transfer agents, the network is unencrypted hence the use of text based steganography to conceal the presence of hidden information. The lack of networking for the other portions of the application should not inhibit the use of the software.

Table 11 Communication Interfaces, details the protocols that the system must support.

|  |  |
| --- | --- |
| SMTP | Simple Mail Transport Protocol. This protocol sends information to a mail server for delivery to that mail server, or to forward to other mail servers (relay). |
| POP3/IMAP | Either one of these protocols enable the receipt of mail from a mail server. One of these should be supported with preference to both. |

Table 11 Communication Interfaces

### 3.2.7 Memory Constraints

In order to minimise memory requirements, the application must be as light as possible to enable further porting of the application to other devices such as tablets or potentially mobile telephony devices. The enables greater portability to multiple host devices. A suggested requirement is limited to 64Mb in addition to the host operating system and runtime.

### 3.2.8 Operations

The system should support the operations (basic functionality) defined in Table 12 Basic Operations.

|  |  |
| --- | --- |
| Authenticate | Authenticate User Against a Database for the use of mail based steganography. Options for Guest Mode for educational use and non-mail based steganography. |
| Encode | Encode messages, text or code/scripts in a given method, or a combination method (hybrid text based steganography). |
| Decode | Decode message, text or code/scripts with a key (or without a key for keyless steganography). |
| Display | Display the result (animated if option enabled). |
| Send | Send result message via email. |
| Receive | Receive messages via email and decode appropriate messages. |

Table 12 Basic Operations

Additional operation that the system should enable are defined in Table 13 Administrative Options.

|  |  |
| --- | --- |
| Set Options | Set options for:   * Optional mail server * Animation Timings (or disablement) * Methods to enable * Database Storage, whilst the application will use an internally compiled database, a shared configuration method can be implemented for a multiple user environment. |

Table 13 Administrative Options

### 3.2.9 Site Adaptation Requirements

The very design of the application will alleviate the necessity to adapt the site in any way. Therefore during implementation, we cannot rely on network or database presence and should fall back as necessary.

### 3.2.10 Product functions

Stegaid have the following product functions:

* Authentication will be provided to enable a user to log in or out of the system to access the steganographic interface and encode or decode text or perform mail functions.
* The ability to encode text and perform steganography showing the user how the process is taking place and then providing the output and associated steganographic key.
* The ability to decode text and show the user the process that is taking place by way of animation upon the insertion of the cover text and key.
* The ability to send encoded mails to other people who can then receive the text and decode as such.
* The ability to receive emails, and decode encoded ones on input of the steganographic key.

### 3.2.11 User characteristics

In order to cater for two very difference user bases, a generic approach to application design should be followed. Table 14 User Characteristics describes the two user bases in more detail.

|  |  |
| --- | --- |
| Educational Users | People undertaking study wishing to learn text based steganography and how it is accomplished. |
| General Users | People wishing to use the system as a tool for sending mail via insecure mail servers to other people. Other users may wish to use the system to hide information and/or code over an unencrypted network. |

Table 14 User Characteristics

### 3.2.12 Constraints, assumptions and dependencies

The system should limit the amount of information being hidden (especially when animations are enabled) otherwise the user with be waiting a significant amount of time for the process to take place. To alleviate the problem with extremely large text the option to override this limit should be provided.

Table 15 Operational Constraints shows the constraints that must be considered.

|  |  |
| --- | --- |
| Regulatory | Certain States disallow the export of encryption materials. Steganography, whilst not encryption, is subject to the same law in that it is hiding information.  SPAM law varies by state, but attention should be applied for mail operations.  Affects: End User License Agreement |
| Hardware Limitations | No assumption should be applied to the host system to maximise portability, this includes embedded systems, low power systems and advanced systems.  Affects: Software Design |
| Parallel Operation | The application client should be able to run in parallel to enable users to multitask if it is required. |
| Safety and Security | The application should terminate upon inputting the incorrect username and password three times. Limits should be present to users people spamming others set by an Administrator. |
| Interfaces to other applications | There are no interfaces to other applications that are required beyond the communication with mail servers. Mail Servers use well defined protocols (SMTP/IMAP/POP) which will be provided to the application, |

Table 15 Operational Constraints

The following assumptions have been made:

* In order to accommodate a wide variety of users, we cannot know the host device, special care should be taken to maximise portability.

The following dependencies have been identified:

* The system’s design should assume no dependencies are present other than the standard host device and runtime.

## 3.3 Specific Requirements

This section is specific to designers and developers and will be used for design generation. It will contain more in depth information than the previous section along with various rule-bases.

### 3.3.1 External interface requirements

It was previously stated that there will be two external interfaces, a mail transport agent and a database management system for shared configuration (an internal database for single system configuration and fall back mechanism).

Mail Transport Agents:

The sending of mail will be accomplished using the standard port 25 using the Simple Mail Transport Protocol (SMTP) by using an existing library. The reception of mail shall be accomplished by library also supporting the POP3 and IMAP protocol. The settings for these external systems will be stored in the relational database and will be mapped to a user.

Database:

In order to add flexibility, additional libraries will enable the support of external database systems (i.e. on the Local Area Network) other than the inbuilt one. The support for mysql/mariadb will be implemented using standard sql syntax. Theoretically, with the appropriate library loaded any database management system can be used.

### 3.3.2 Functional requirements

Base System Requirements

The system *shall* enable the user to encode text using a variety of formats hiding the information within a cover text (and a combination thereof).

The system *shall* will enable the user to decode a cover text with the input of a steganographic key to reveal the hidden information within.

The system *will* support the sending of encoded email messages to others if the option has been enabled and mail settings are present.

The system *will* support the receipt and subsequent decoding of messages if the option has been enabled and mail settings are present.

The system *shall* enable an administrator to administer settings and options such as algorithm selection. Mail settings and security limits shall be adjustable to the required attributes when required.

### 3.3.3 Performance requirements

In order to improve the performance of encoding and decoding texts, animations can be disabled as appropriate. If animations are enabled the system naturally performs slower for the user to see the processes that are taking place.

The following performance characteristics were identified:

* Encode 100 words in a given cover text in 1 minute using 1 steganographic method. A hybrid steganographic scheme will take longer.
* Application launch visibility within 1 second regardless of PC platform.
* Decode 100 words from a given cover text (after identification of the steganographic method used) in 1 minute.

### 3.3.4 Design constraints

The application should be simple to use particularly when used in an educational context whilst remaining intuitive to those using the application as a tool. Plugin API should be simple to implement for future enhancement.

### 3.3.5 Standards Compliance

No standards have been targeted within this application.

### 3.3.6 Logical database requirement

The schematic in Figure 15 Logical Database Structure is a logical view of a database configuration file. It will be the basis for the design of the real database. Note there are no specific types associated with the diagram, this is for the design level stage.

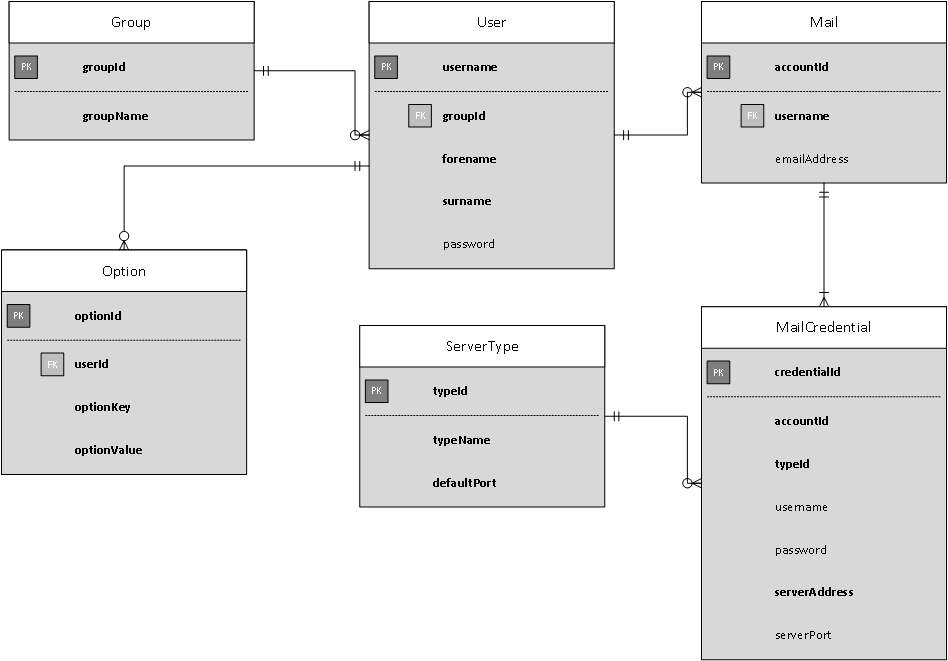


Figure 15 Logical Database Structure

### 3.3.7 Logical class diagram

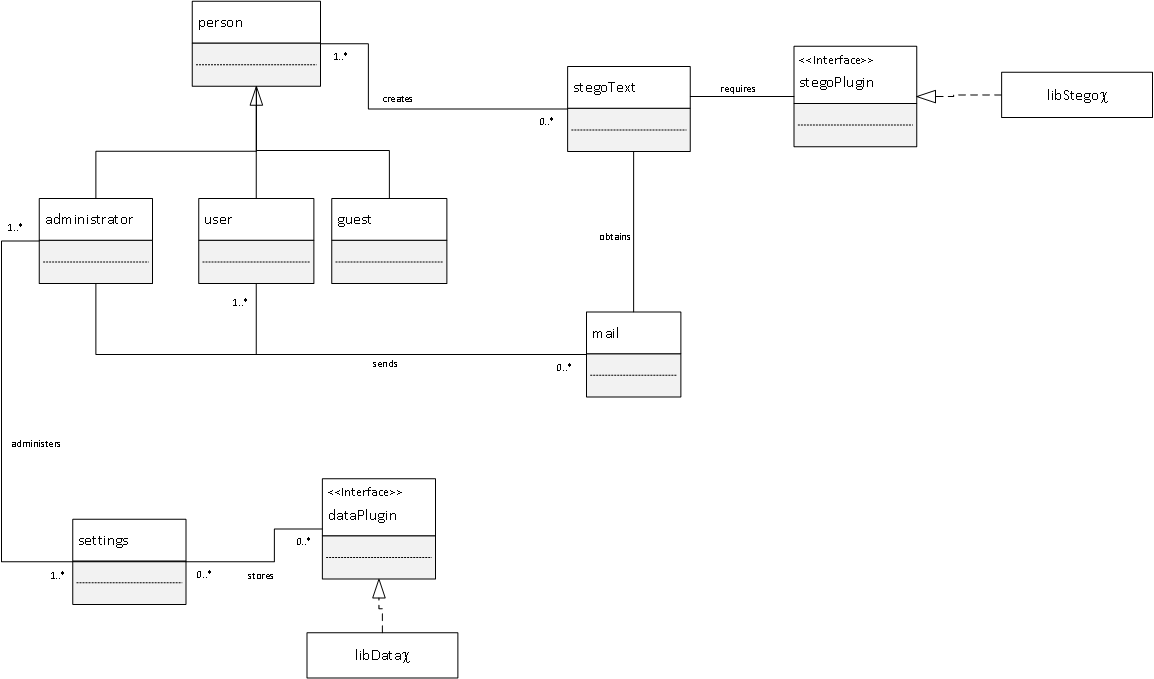


Figure 16: Logical Class Hierarchy

### 3.3.8 Reliability

With the exception of the host operating system and runtime, the application shall never fail as a result of poor coding. The low coupling of the application (and high cohesion) will enable us to handle exceptions in the following manner:

|  |  |
| --- | --- |
| Failure Scenario | Causes Crash / Threshold |
| Steganographic Library Error | Never; library should fail, but not the core application itself. If hybrid mode is selected, this method of steganography should be skipped and moved onto the next method. |
| Network failure | The application should abort network communications. Inform user of network issue. |
| Graphical Incapability | The application must not fail in the event of the user not having a graphical user interface (CLI). In this instance a command line interface shall be provided. |

Table 16 Failure Scenarios

Metrically, the application should commit to the following failure rate:

The target mean time between failure rate is *desired* to be approximately once in 5000 operations (that is once in 5000 time the application is run). The requested mean to recovery should be no more than 30 minutes.

### 3.3.9 Availability

The system should be available at any time, with no specific hours of operation in effect. As the application is a client and is portable. There should be no restrictions as to where the application can be operated.

### 3.3.10 Security

The following goals in relation to security should be adhered to:

Appropriate data held within both database files shall be encrypted prior to saving. This is to prevent information such as credentials, mail server settings and so forth being read and accessed via third parties.

Authentication will take place within the application. To improve security, after three unsuccessful login attempts, the application will terminate. Password text controls will be delimited with \* when a key is pressed. Password requirement validation should take place to ensure passwords are not too short, simple or based on a dictionary word.

### 3.3.11 Maintainability

To improve the core maintainability of the application, when the system is updated by a suitably competent administrator, the application will not need to be compiled to accommodate new features. A level of abstraction will enable future proofing as Table 17 System Maintainability will demonstrate.

|  |  |
| --- | --- |
| Database Abstraction Layer | The configuration of the required external library for database access (if required) will be stored within the internal database. In theory the plugin is not required to be a database, it could be a file store or otherwise. |
| Steganographic Library Layer | No methods of encoding or decoding of texts are built directly into the application. A cross platform library loader will load methods on demand as and when it is needed. Adding new libraries does not require a core program recompile. This is to reduce memory and enable extensibility. New methods of encoding text to perform data hiding are being discovered all the time, so this feature is essential. |

Table 17 System Maintainability

### 3.3.12 Portability

To aid in portability to as many devices as possible, the application design will from the outside bear portability in mind. To maximise portability a few early design choices have been made, which will be incorporated into the Design Document. In this version the main targets are Microsoft® Windows™ and Linux (and derivatives). Table 18 System Portability, details suggested libraries for the target system.

|  |  |  |
| --- | --- | --- |
| Internal Database | Sqlite (Suggested) | Cross Platform database system, runs on UNIX/Linux and Windows. Shall perform internal disk I/O operations for configuration purposes. |
| Dynamic Library Loader | Libltdl | Used for the plugin system, so that not all libraries are loaded at once. |
| Mail Provider System | Libvmime | Provision of SMTP(s), POP3(s) and IMAP(s). |

Table 18 System Portability

### 3.3.13 Other requirements

Usability requirements state that the application design should be simple to academic users exploring the field of steganography whilst being feature rich for advanced users (using the tool). To overcome this the application should be both a console application which can be called from the command line and by other programs and a graphical client.

# 4. Project Plan Overview

## 4.1 Introduction

In the research proposal it was identified that text based steganography would be investigated with particular interest into 4 criteria. Based on the findings of the research a system would be implemented that could use steganography to hide information as necessary. This document specifies a work plan in order to complete the project.

## 4.2 Work Packages

Within the project there are a series of Work Packages (Activities) that must take place for completion of the project (version 1.0). An activity consists of the work that must take place and the requirements for the operation to complete. Whilst the steps must be taken in order, it may be necessary to take a step back and forth and “re-iterate” over steps (as defined in the Research Project Proposal).

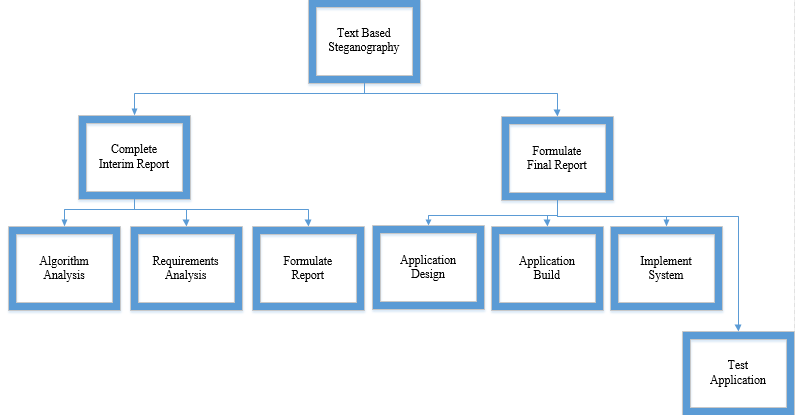


Figure 17: Work Packages

## 4.3 Milestones and Deliverables

There are three important parts to the work packages identified; these are significant milestones in the project:

### 4.3.1 M1 – Report Findings (Interim Report)

The review of known methods will investigate steganographic systems in research papers and journals. Evaluation based on set criteria will take place and form part of the Interim Report. An analysis of user requirements will also be conducted forming the second part of the report. This is the conclusion of Work Package 1 to 3.

### 4.3.2 M2 – Application Design Specification

The application design specification will form the first part of the final report. Having identified the requirements an object oriented design using UML will take place ensuring the user requirements are met. This concludes Work Package 4.

### 4.3.3 M3 – Implementation

The build of the associated system according to the agreed design by developers and stakeholders (the user, the supervisor) will take place. Testing according to the test plan (also Work Package 4) should be carried out and finally implementation. A final report will be submitted along with program code and evaluation which concludes Work Packages 5 to 7. There are two sets of deliverables identified. The interim report will be submitted forming D1 and the final report and associated program code along with demonstration and final report forming D2.

## 4.4 Project Plan

Having identified key work packages, these packages are further broken down into tasks and the estimated time to complete the task. The following details Tasks associated with each Work Package.

**WP 1 Algorithm and Method Research**

Tasks involve analysing suitable research papers and journals and identifying suitable algorithms and methods associated with text based steganography. These methods will then be evaluated based on a set criterion.

**WP 2 Requirements Analysis**

The analysis of the system to be designed software system will be investigated and conclusively identify requirements which will be used later in the lifecycle as a benchmark to software success.

**WP 3 Formulate Interim Report**

Based on WP1 and 2, the result will be formulated into a report forming Milestone 1 concluding findings identified in the research.

**WP 4 / M2 Application Design**

Data design, Application Design with regard to structures and the user interface will be developed based on the requirements specification document (SRS). Upon creation of relevant diagrams (not limited to) Class Diagrams, State and Activity diagrams. A suitable User Interface will be designed according to HCI Usability recommendations. A Test Plan will also be created to thoroughly Test the system.

**WP 5 System Build**

This work package involves the task of coding the application according to the specification WP 4. The code will then be debugged and tested according to the test plan.

**WP 6 Implementation**

The implementation will occur which includes creating installers and acceptance testing.

**WP 7 / M3 Produce Final Report**

The final report will be assembled with the design specification(s), results of the build (testing) and conclusion. The product and software process will be evaluated as required in CIP (Continuous Process Improvement).

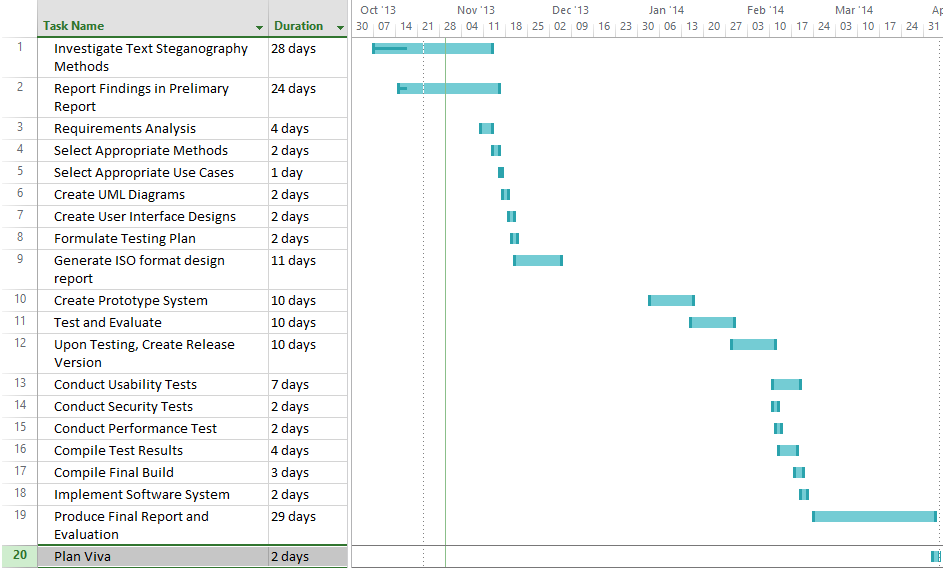


Figure 18: Project Task Plan

## 4.5 Time Management

The following table describes the time management of the associated researcher in developing this project. In order to meet deadlines certain work life rearrangements must take place.

Goal 1: Complete final year project on time to the desired level grade (74%).

In order to achieve this certain life choices must be analysed:

|  |  |  |  |
| --- | --- | --- | --- |
| Time | Activity | Effectiveness | Comments |
| 6-7 am | Shower/Dress | 100% | Sometimes Deadlines Missed |
| 7-8 am | Get Breakfast | 50% | Could Reduce |
| 8-11am | Entertainment | 30% | Move Entertainment to Night time. |
| 11-9pm | Work | 100% | Reduce travel times |
| 9-10pm | Eat | 50% | Could Reduce |
| 10-12am | College Assignments | 100% | Could move to morning to allow more time |

Table 19: Time Management

In order to meet deadlines rearrangement of existing duties and actions must take place in order to remain on track. It is now suggested I do College work in the morning each day to allow for increased hours.

## 4.6 Meeting Plan

Meetings will take place no later than every 2 weeks with specific preference to weekly meetings to be held every Monday or at Supervisory Request. Constant contact via email should also be considered to maintain good communication and to prevent misunderstanding.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Subject Matter | Date | | Completed | |
| Preliminary Meeting with Project Supervisor | 09/10/2013 | | 🗹 | |
| Interim Requirements | 28/10/2013 | | 🗹 | |
| Requirements Elicitation | 4/11/2013 | | 🞏 | |
| Progress Report | 11/11/2013 Weekly | | 🞏 | |
| Final Presentation | | --/04/2014 | | 🞏 |

Table 20: Partial Meeting Log

## 4.7 Risk Management

A risk analysis has been carried out and an analysis has identified a number of risks that could affect the project in a number of ways resulting in delay or potentially project failure. A criteria is set to analyse the likelihood of occurrence. The worst criteria will be mitigated by way of contingency planning. As with the traffic light system, green risks are low risk (unlikely to occur) whereas red are high risk (very likely or imminent). Risks are categorised into two key areas event driven or evolving risks.

### 4.7.1 Event Driven Risks

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Risk** | **Likelihood** | **x** | | **Consequence** | **Impact** | |
| Computer Crashing | 2 (Medium) | | 2 (Low) | | | 3 |
| Coding Skills Not Met | 1 (Low) | | 4 (High) | | | 4 |
| Requirements Not Met | 2 (Medium) | | 5 (Very High) | | | 10 |
| Called In To Work | 2 (Medium) | | 2 (Low) | | | 4 |
| Loss of Data | 3 (High) | | 5 (Very High) | | | 15 |
| Falling Ill | 3 (High) | | 3 (Medium) | | | 6 |

### 4.7.2 Evolving Risks

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Risk** | **Likelihood** | **x** | | **Consequence** | **Impact** | |
| Feature Creep | 2 (Medium) | | 4 (High) | | | 8 |
| Time Delays | 3 (High) | | 4 (High) | | | 12 |
| Complexity Increase | 2 (Medium) | | 3 (High) | | | 6 |

### 4.7.3 Critical Contingency

Having identified the most common risk likely in this project, an estimated score of impact to the project has taken place. In order to alleviate these risks it has been decided to identify ways around the problems surrounding the risk.

### 4.7.3.1 Loss of Data

A computer crash or loss of data will not occur by way of timely backups (risk avoidance). The project will be backed up to a separate remote data source weekly.

### 4.7.3.2 Time Delays

Time delay *will* occur. It is necessary to be dynamic in the approach to the project. In some cases certain tasks can be reduced in time to accommodate increased allocation on other tasks.

**4.7.3.3 Requirements Not Met**

Projects ultimately fail when a project does not entirely meets its original agreed requirements. Careful requirements checking must take place at various stages of the software development lifecycle.

# 5 IEEE Software Design Description

This document will detail the design of the “Stegaid” application which follows on from the IEEE Software Requirements Specification already agreed upon. Due to the fact that the target application is a new system and not a replacement system, as such as no external components or documents to base a design from, elicitation is **ongoing** and thus this design is based on a prototype already presented. This way features can be added, design modified to suit ongoing requirements change. Prototyping helps deal with unknowns, particularly is such an application has never been attempted before. This document shall detail the internal components of the system, its various interfaces and functions. This document follows the specification defined by (Institute of Electrical and Electronic Engineers, 2009) standard 1016 and the examples provided by (INSA, 2014). The intended audience for this document is for developers to implement the product and test planners to plan the various tests that should take place.

## 5.1 Revision History

|  |  |
| --- | --- |
| 1.0 | Initial Specification |
| 1.1 | Add Settings Support |

Table 21: IEEE Software Design Specifications Revision History

## 5.2 System Architectural Design

The Stegaid system will be a tool to aid learners in the processes involved in text based steganography. They can encode and decode information and save the results which can be read. Further to this core function, Stegaid will be a tool that can be used out of a classroom. As a secondary requirement the application can send and receive encoded messages and decode as necessary. The application whilst graphical in nature; has a command line capability so that it can be used on systems without a graphical user interface or part of another program (using “Shell Execute” depending on the language). As specified in the requirements the tool will be fully cross platform with the initial incarnation targeting both Linux derivatives such as Ubuntu™ and Microsoft® Windows™ (Vista or later).

In order to complete the aforementioned task, the application will firstly be prototyped from which a design can then be established and test plan carried out. Then, based on the design, the application is developed and tested. As stated in the project plan, an iterative methodology will be adopted, which means each development process can go back on forth, as new requirements are identified and as issues arise (Software Agility).:

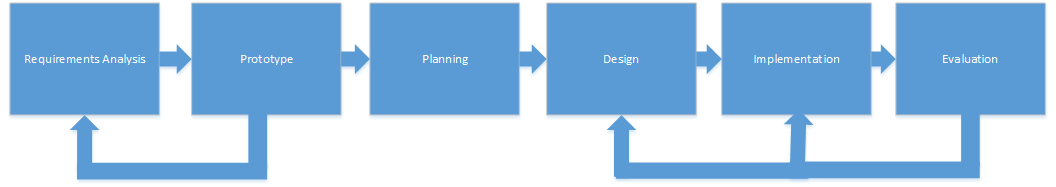


Figure 19: The Software Process

### 1.2.1 System Description

The software to be designed is an educational tool to show how steganography takes place. Multiple methods of text based steganography will be enabled. To enable a wider audience beyond the classroom, certain features could be enabled and the system be made extensible enough to enable new features and methods. The system overview in Figure 20: System Overview describes the target system.

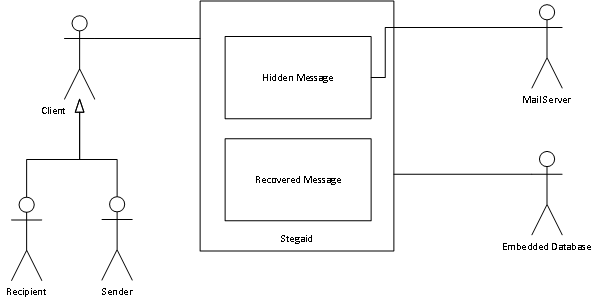


Figure 20: System Overview

As can be seen in Figure 20: System Overview, a client can send and receive mail or encode and decode text and/or messages. The embedded database performs authentication and stores settings, and the **optional** mail server performs mail requests. Students (or indeed general purpose) users can learn the concepts of text based steganography by using this system. Analysis, found that there are a variety of techniques available, so it was decided that the system should be extensible and libraries be loaded on demand as opposed to compiled as a singular application.

### 5.2.2 System Architecture

The diagram in Figure 21: Deployment Overview shows the environment in which the application exists. The target for this diagram are deployment managers and system administrators, which shows that the client interface (namely the core application) has optional integration with an external authentication server and mail server.

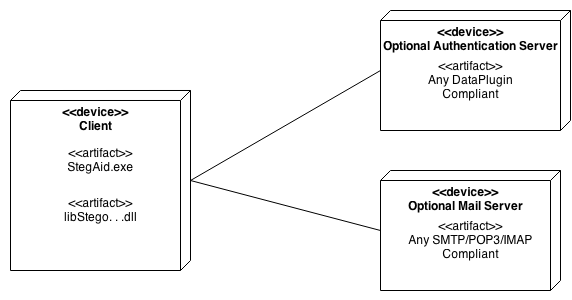


Figure 21: Deployment Overview

The objects in Figure 22: Decode Text Sequence Diagram through to Figure 25: Receive Message Sequence Diagram details the inner workings of the desired system. The target audience for this diagram are the stakeholders, developers and testing personnel. Implementers can also use this diagram to understand how the application works.

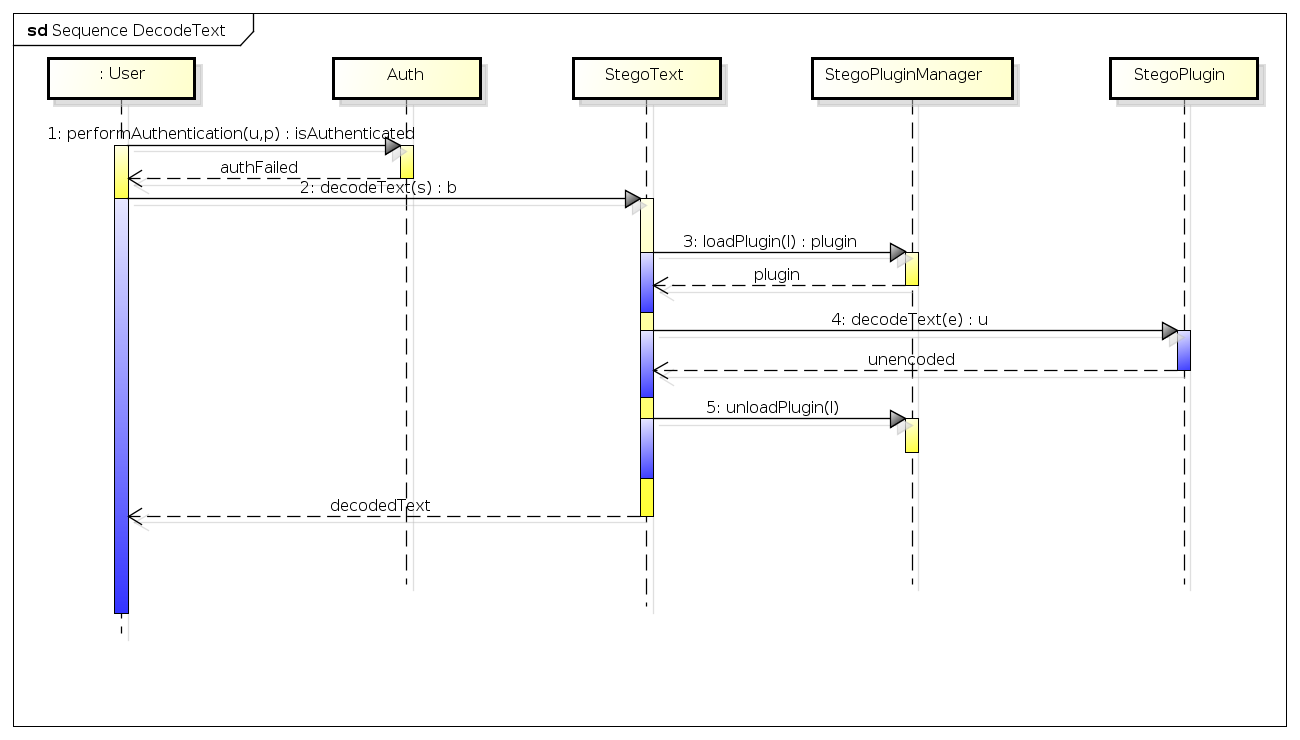


Figure 22: Decode Text Sequence Diagram

The object in Figure 22: Decode Text Sequence Diagram shows the process of decoding a text to uncover the hidden information beneath it. It can be seen that after authentication takes place and the user selects the decode text option, a plug in is loaded that performs the decoding. The information is then returned to the core application for the user to save this file.

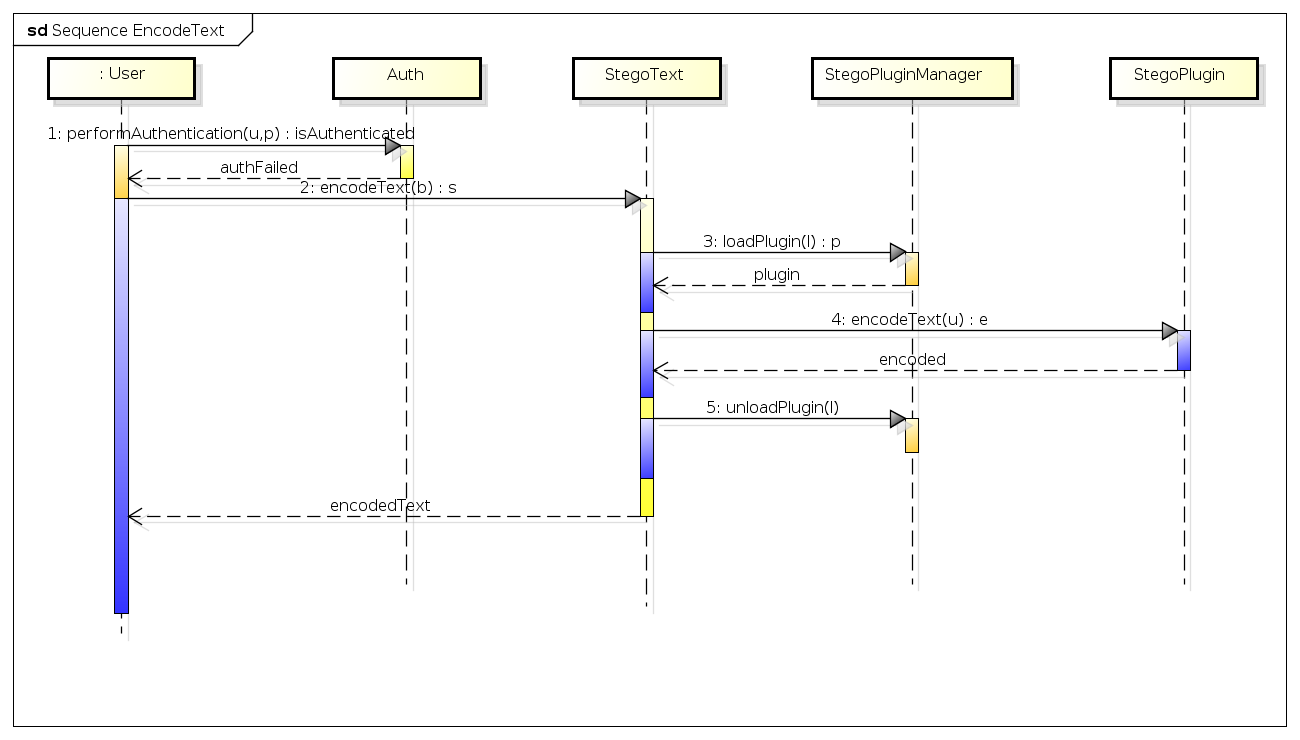


Figure 23: Decode Text Sequence Diagram

The object in Figure 23: Decode Text Sequence Diagram shows the process of encoding a text to hide information in randomly selected cover text. The cover text after hiding information within it is returned to the core application for the user to save the file.

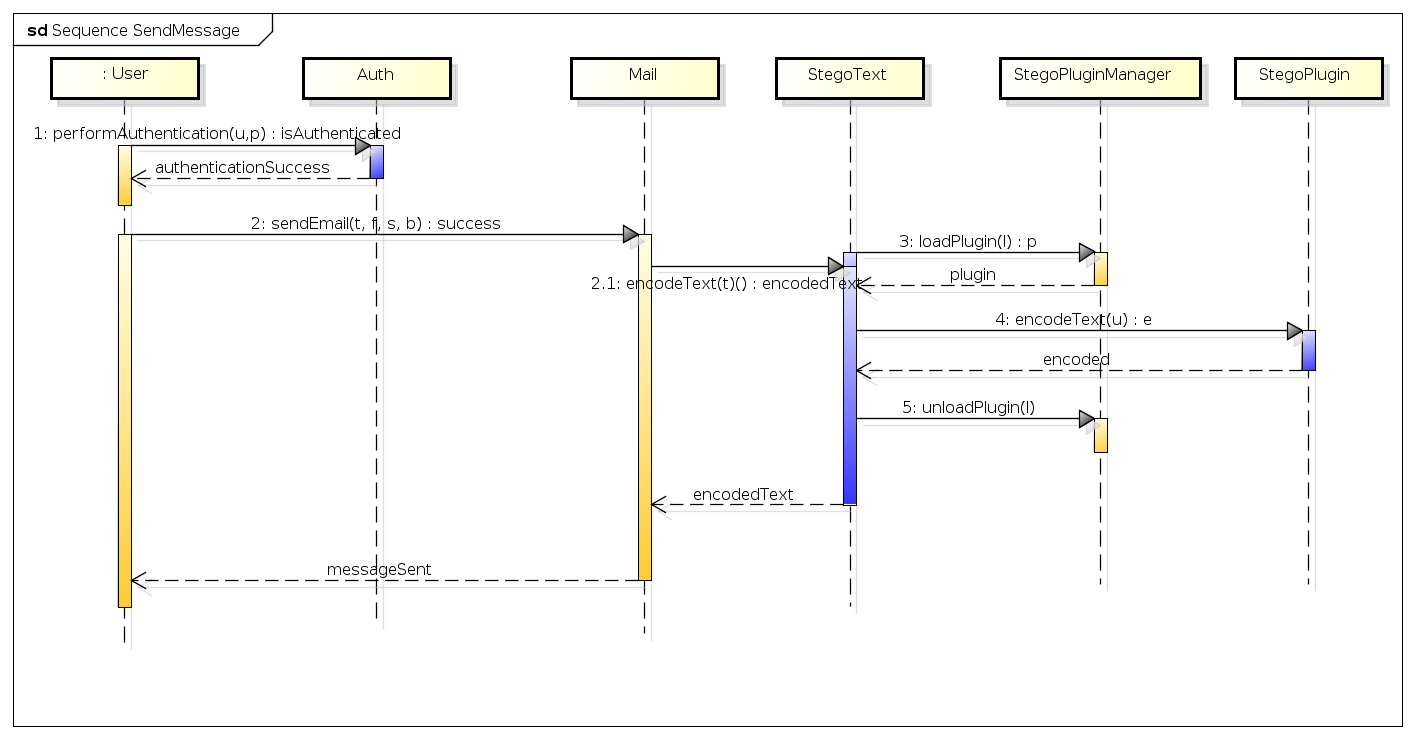


Figure 24: Send Message Sequence Diagram

The object in Figure 24: Send Message Sequence Diagram shows the process or sequence of events that must take place in order to send an encoded mail message. This (upon authentication) requires the user to select the send a message option and enter to, subject and the message. The information is encoded into a cover text and sent via email.

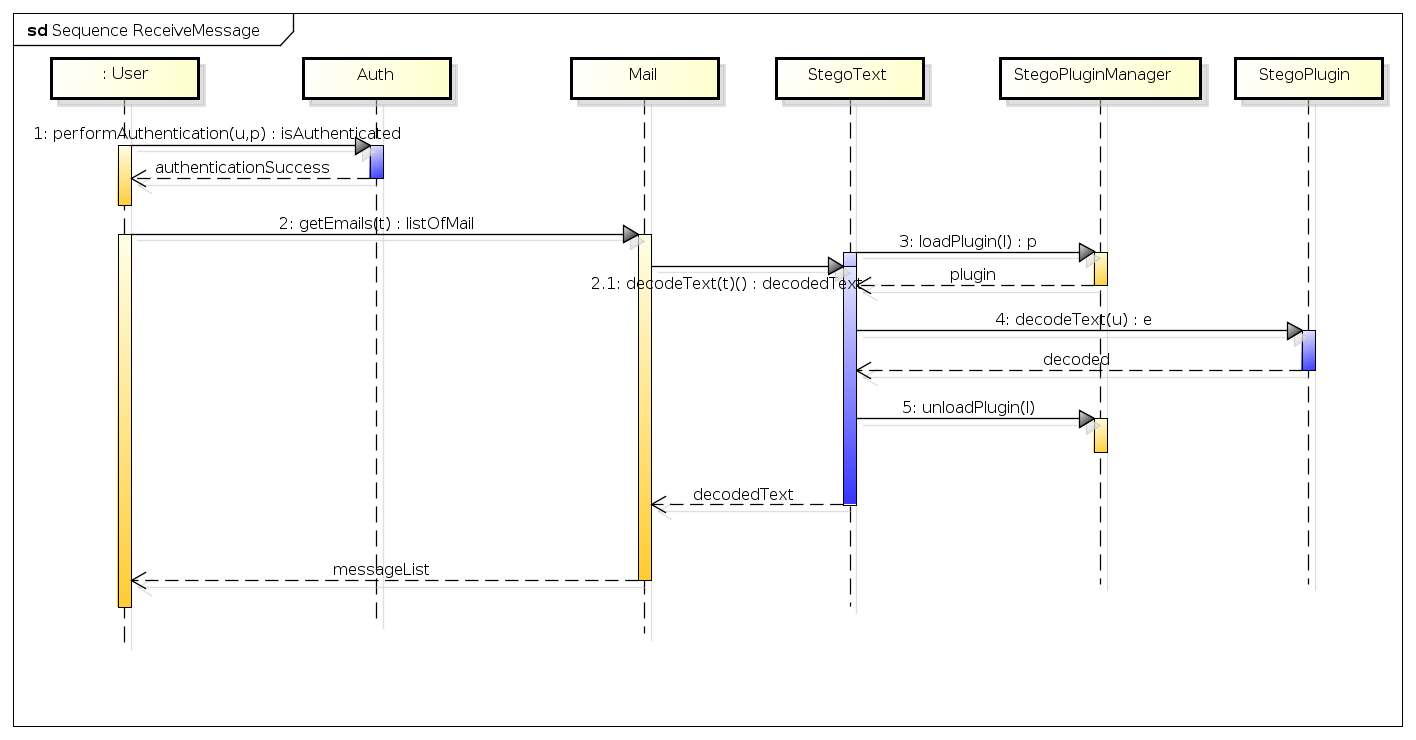


Figure 25: Receive Message Sequence Diagram

The object in Figure 25: Receive Message Sequence Diagram shows the sequence of event that must take place when receiving a message. The application will receive a list of messages from the server and display for selection. The message is decoded as appropriate and displayed to the user.

### 5.2.3 Design Constraints

The constraints will provide limits for the target software system.

### 5.2.3.1 General Constraints

The general constraints are boundaries to which the application will conform to that do not fall in the category of Hardware or Software Constraints. Many of these were defined in 3.2.12 Constraints, assumptions and dependencies.

### 5.2.3.2 Hardware Constraints

This specific version shall meet certain hardware characteristics, in that is shall be as lightweight as possible to maximise the target device base.

* The initial incarnation *shall* target all x86 complaint processors using both the 32 bit and 64 bit architectures.
* Memory usage *shall* not exceed 64 Megabytes of Random Access Memory excluding swap usage.

The final implementation with associated text files *will* not exceed 100 Megabytes *sans* runtime support libraries.

### 5.2.3.3 Software Constraints

To maximise target devices beyond the specific hardware constraints certain software constraints must be defined.

* The initial implementation *shall* target both Linux™ and Microsoft® Windows 7™ or later.
* Specific attention *shall* be applied to target to other host operating systems therefore special attention should be applied to the choice of runtime libraries.

## 5.3 Components Description

Through identifying the processes of the core application (that it must encode, decode, send and receive texts), additional functionality will be required in order to support these functions. Additional Support Functions are detailed in Table 22: Component Overview.

|  |  |
| --- | --- |
| Function | Dependency |
| Encode Messages | * A method to load and unload Plugins * A method to register Plugins |
| Decode Messages | * A method to load and unload Plugins * A method to register Plugins |
| Send Messages | * A method to manage mail accounts (Authentication) |
| Receive Messages | * A method to manage mail accounts (Authentication) |

Table 22: Component Overview

### 5.3.1 Decomposition Description

Having identified additional support functions of the desired system beyond the core application it can be identified the logical groups of information that should exist. Table 23: Decomposition Description details these logical groups of information that must be processed.

|  |  |
| --- | --- |
| Component | Functions |
| User / Group | Provide Authentication |
| StegoText | Provide Encode and Decode capabilities |
| StegoPluginManager | Provide the Registration, Deregistration, Load and Unload of Steganographic Plugins. |
| Setting | Provide a Global Method to store information about Users, Plugins and other Settings |
| Mail | Provide Mail Sending and Receiving Capabilities |
| DataPluginManager | Provide the Registration, Deregistration, Load and Unload of External Data Source Plugins. |

Table 23: Decomposition Description

### 5.3.1.1 User

The User component represents the users within the system. Users can be added and deleted on demand by a suitable administrator. The user component should also perform the necessary authentication in accessing the system. The logical information and processed that must take place within the User component is detailed in Table 24: The User Component.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| perform  Authentication() | Authenticates User   |  |  | | --- | --- | | Requires | Username: String  Password: String | | Return | Success: Boolean | | Validation | Username and Password present | |
| addUser() | Add User To Data Store   |  |  | | --- | --- | | Requires | Username: String  Forename: String  Surname: String  Password: String  GroupId: Integer | | Return | Success: Boolean | | Validation | Username: Min 6, Max 50, Unique  Forename: Min 6, Max 50  Surname: Min 6, Max 50  Password: Min 6, Max 50, Alpha Num Mix  GroupId: Exists | |
| deleteUser() | Delete User From Data Store   |  |  | | --- | --- | | Requires | Username: String | | Return | Success: Boolean | | Validation | All references to user delete elsewhere in datastore (such as Mail Credentials) - Keep Relational Integrity | |
| listUsers() | List Users In Data Store   |  |  | | --- | --- | | Requires | None | | Return | List of Users | | Validation | None | |

Table 24: The User Component

### 5.3.1.2 Group

The Group components shall enable a permission set to be applied to a group of users as opposed to per user. The information that must be processed with relation to Group management is detailed in Table 25: The Group Component.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| getPermissions() | Get the permissions allocated to the group   |  |  | | --- | --- | | Requires | GroupName: String | | Return | Permissions (Unix Style) | | Validation | None | |
| getGroupId() | Get Group Id given a Group Name   |  |  | | --- | --- | | Requires | GroupName: String | | Return | Success: Group Id  Fail: Zero | | Validation | Group Exists | |
| addGroup() | Add Group To Data Store   |  |  | | --- | --- | | Requires | GroupName: String | | Return | Success: Boolean | | Validation | GroupName: Min 6, Max 30, Unique | |
| deleteGroup() | Delete Group From Data Store   |  |  | | --- | --- | | Requires | GroupName: String | | Return | Success: Boolean | | Validation | Group exists,  No relational dependencies, i.e. No Users in Group | |
| listGroups() | Get Group Id given a Group Name   |  |  | | --- | --- | | Requires | GroupName: String | | Return | Success: Group Id, Integer  Fail: Zero | | Validation | Group Exists | |

Table 25: The Group Component

### 5.3.1.3 Steganographic Plugin Manager

The Steganographic Plugin Manager performs the necessary functionality to enable the system to load, unload, register or deregister plugins. It is the plugins that actually perform the operations required to hide information making the system very extensible. The Steganographic Plugin Manager shall be composed of the capabilities defined in Table 26: The Steganographic Plugin Manager Component.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| registerLibrary() | Registers a Plugin to the system   |  |  | | --- | --- | | Requires | LibPath: String | | Return | Success: Boolean | | Validation | File Exists,  Plugin Passes Test | |
| unregisterLibrary() | Deregisters a Plugin from the system   |  |  | | --- | --- | | Requires | ClassID: String | | Return | Success: Boolean | | Validation | Library exists | |
| testLibrary() | Tests a Plugin in the system   |  |  | | --- | --- | | Requires | ClassID: String | | Return | Success: Boolean | | Validation | Library exists | |
| listPlugins() | List Plugins Registered in the system   |  |  | | --- | --- | | Requires | None | | Return | List of Plugins | | Validation | None | |
| loadPlugin() | Load a Plugin for usage   |  |  | | --- | --- | | Requires | ClassID: String | | Return | Success: \* Reference to Plugin  Fail: Zero | | Validation | Library exists | |
| unloadPlugin() | Unload a loaded Plugin   |  |  | | --- | --- | | Requires | ClassID: String | | Return | void | | Validation | None. If plugin not loaded, ignore. | |

Table 26: The Steganographic Plugin Manager Component

### 5.3.1.4 Setting

The system settings component was presented later as a unified way to provide access to system settings to separate the Code (aka Controller) from the View; the model being the database. This is a Key Value store that when provided with a Key, the value is returned. The value is set on provision of the key and value. Table 27: The Settings Component explains these processes in more detail.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| getSetting() | Gets some form of system setting   |  |  | | --- | --- | | Requires | Key: String  GroupName: String [Optional] | | Return | String: The Value with reference to the Key [a key value store], “False” if no setting. | | Validation | Key Value exists or “False” | |
| setSetting() | Sets some form of system setting   |  |  | | --- | --- | | Requires | Key: String  Value: String  GroupName: String [Optional] | | Return | void | | Validation | None | |

Table 27: The Settings Component

### 5.3.1.5 Mail

The mail system shall call the Application Programming Interface of the Poco Mail Client libraries to both send encoded messages, and receive messages and decode as necessary. In order to send and receive messages, email accounts must be added with support for differing protocols and standards (and deleted as necessary). Table 28: The Mail Component details these operations.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| sendEmail() | Perform the Send Mail Process   |  |  | | --- | --- | | Requires | ToAddress: String  FromAddress: String  Subject: String  MailBody: String | | Return | Success: Boolean | | Validation | FromAddress exists with defined format:  protocol:user@mail:port  ToAddress defined format: user@mail  Mail Settings Exist | |
| getEmails() | Receive Emails from Mail Server   |  |  | | --- | --- | | Requires | ToAddress: String | | Return | List of Emails | | Validation | ToAddress exists with defined format:  protocol:user@mail:port  Mail Settings Exist | |
| getSupportedProtocols() | Protocols change, the view needs to know supported mail protocols   |  |  | | --- | --- | | Requires | None | | Return | List of Protocols | | Validation | None | |
| getProtocolByName() | Get the Protocol given its name   |  |  | | --- | --- | | Requires | ProtocolName: String | | Return | Protocol Port, ID and relevant information or Zero if not exist | | Validation | Protocol Exists | |
| getProtocolById() | Get the Protocol given it’s ID   |  |  | | --- | --- | | Requires | ProtocolID: Integer | | Return | Protocol Port, ID and relevant information or Zero if not exist | | Validation | Protocol Exists | |
| addMailUser() | Add a mail user to the system   |  |  | | --- | --- | | Requires | EmailAddress: String  Username: String | | Return | Success: Boolean | | Validation | Username does not already have an email address.  EmailAddress unique.  User has Mail Permissions | |
| emailAddressExists() | Checks if an email address exists   |  |  | | --- | --- | | Requires | EmailAddress: String | | Return | Exists: Boolean | | Validation | None | |
| removeMailUser() | Remove a mail user from the system   |  |  | | --- | --- | | Requires | Username: String | | Return | Success: Boolean | | Validation | If email address has registered protocols/servers (remember there are multiple protocols involved in the mail process), these will need deletion first. | |
| getMailUsers() | Lists Mail Users   |  |  | | --- | --- | | Requires | None | | Return | List of Mail Users and List of Mail Protocols to each User | | Validation | None | |
| addMailOption() | Adds a Mail Option [Protocol] to a User   |  |  | | --- | --- | | Requires | ProtocolID: Integer  EmailAddress: String  Hostname: String  Port: Integer  Username: String  Password: String | | Return | Success: Boolean | | Validation | ProtocolID: Protocol Exists  EmailAddress: Exists  Hostname: Min 6, Max 50  Port: > 1, <65535  Username: No Validation, validation done at authentication stage of mail requests, not all smtp relays (internal ones) have authentication  Password: No Validation, validation done at authentication stage of mail requests, not all smtp relays (internal ones) have authentication | |
| removeMailOption() | Remove a Mail Option [Protocol] from a User   |  |  | | --- | --- | | Requires | EmailAddress: String  ProtocolID: Integer  Hostname: String  Port: Integer  Username: String | | Return | Success: Boolean | | Validation | Fields exist. Note: In theory only the MailOptionID could be used BUT, the system shall not allow an arbitrary delete command given any number. | |

Table 28: The Mail Component

### 5.3.1.6 Data Plugin Manager

By default this component is not used unless External Authentication has specifically been requested by Administrator. A Data Plugin is an alternate handler for Users, Groups and so forth. If enabled data requests are processed via the plugin otherwise internal data management will take place. The Data Plugin Manager therefore is the handler to enable those plugins to be registered, loaded and unloaded as necessary. Whilst similar to the Steganographic Plugin Manager, they are not compatible. For the sake of simplicity they have been purposefully kept separate. This will be part of Stegaid extensibility along with the Steganographic Plugin capability. Table 29: The Data Plugin Manager Component, shows the logical operations that must take place.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| registerLibrary() | Registers a Plugin to the system   |  |  | | --- | --- | | Requires | LibPath: String | | Return | Success: Boolean | | Validation | File Exists,  Plugin Passes Test  Library not already registed, There can only be one Data Plugin. | |
| unregisterLibrary() | Deregisters a Plugin from the system   |  |  | | --- | --- | | Requires | None | | Return | void | | Validation | None | |
| testLibrary() | Tests a Plugin in the system   |  |  | | --- | --- | | Requires | LibPath: String | | Return | Success: Boolean | | Validation | Library exists | |
| getPluginName() | Gets the Plugin Information   |  |  | | --- | --- | | Requires | None | | Return | Plugin Name: String or “” | | Validation | There is a Plugin Registered | |
| loadLibrary() | Load a Plugin for usage   |  |  | | --- | --- | | Requires | None | | Return | \* Reference to Plugin | | Validation | Library Registered | |
| unloadLibrary() | Unload a loaded Plugin   |  |  | | --- | --- | | Requires | None | | Return | void | | Validation | None. If plugin not loaded, ignore. | |

Table 29: The Data Plugin Manager Component

### 5.3.1.7 StegoText

This component shall not perform any steganography. The role of this component is to load a plugin via the Steganographic Plugin Manager, call methods within that plugin to perform steganography using the defined external interface in Table 38: The Steganographic Plugin Interface and return the information back to caller. Table 30: The StegoText Component details the operations the object must perform.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| stegoText() | Constructs a stegoText Object   |  |  | | --- | --- | | Requires | SelectedLib: String (The Plugin Class ID to load)  Containe: Widget \* [Optional] An optional reference as to where to render output. | | Return | None | | Validation | None | |
| encodeText() | Requests the StegoPluginManager to load the referenced plugin into memory. Calls encodeText() functionality of the plugin.   |  |  | | --- | --- | | Requires | A stream of bits to encode: iostream | | Return | Encoded bits in the form of a text: String | | Validation | Plugin Loaded Successfully  Plugin Unloaded after use | |
| decodeText | Requests the StegoPluginManager to load the referenced plugin into memory. Calls decodeText() functionality of the plugin.   |  |  | | --- | --- | | Requires | Encoded bits in the form of a text: String (aka A Cover Text) | | Return | A stream of bits decoded: iostream | | Validation | Plugin Loaded Successfully  Plugin Unloaded after use | |

Table 30: The StegoText Component

**Where Success is returned in all cases, the value shall be False (Zero), or True or (Non Zero) when failure occurs.**

### 5.3.2 Class Diagram

With the information obtained in Section 5.3.1 Decomposition Description a class diagram can be formulated. The class diagram shows the relationship (call hierarchy) from one object to another. Minimisation of global variables (within the scope of the class has taken place) favouring object creation at point of necessity and destructing the object as soon as possible to minimise memory requirements.

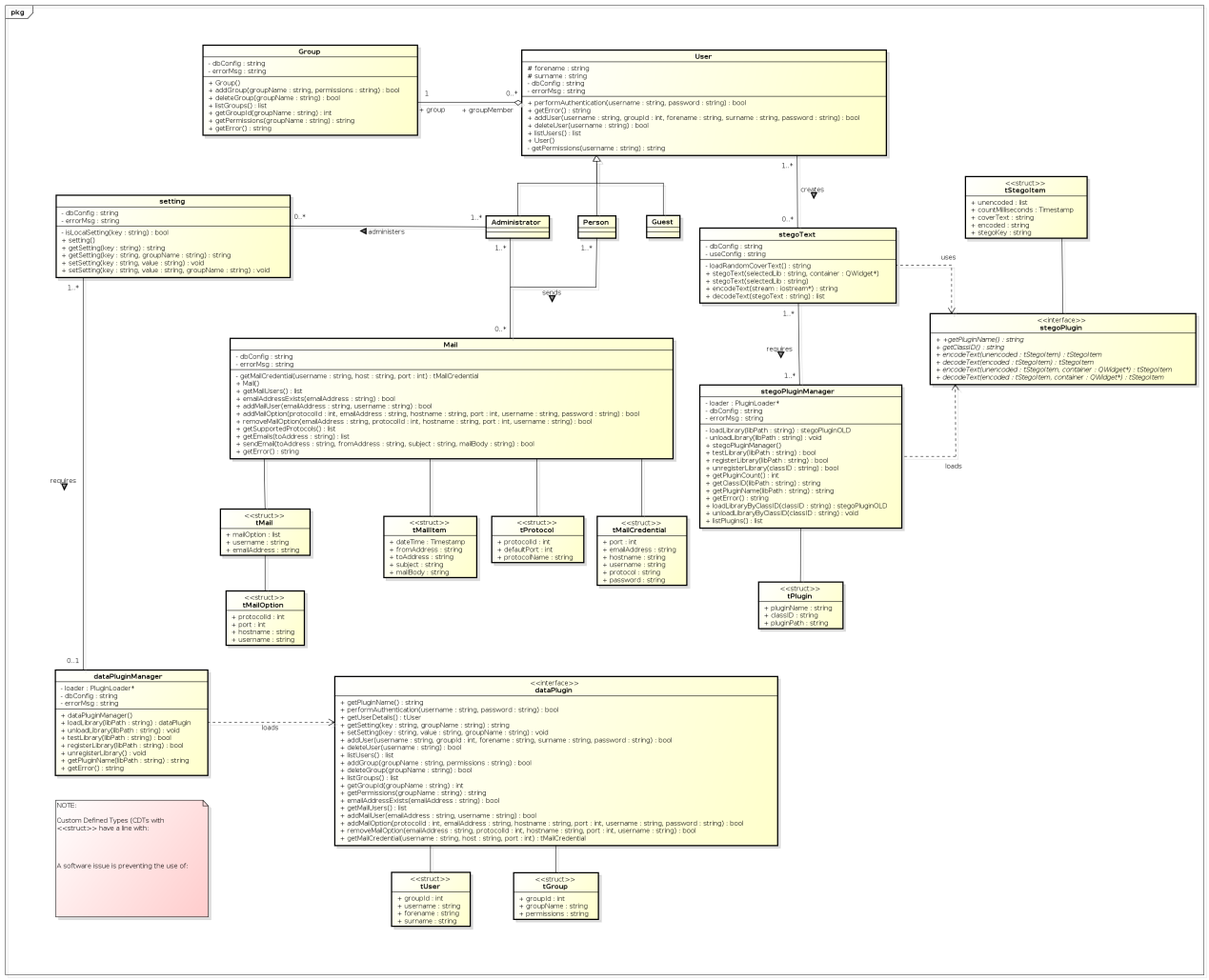


Figure 26: Stegaid Core Application Class Diagram

Appendix 1 contains a full size view of the class diagram for better viewing.

## 5.4 External Interfaces

External Interfaces can be defined as, the external interface the user sees (the View) and the Plug in interfaces. The Plug in interfaces’ are well defined protocols for access to dynamic link libraries that are loaded (or unloaded) on demand. These will have the extension .dll for Microsoft Windows platforms and .so on Unix/Linux.

### 5.4.1 User Interfaces

There shall be a number of User Interfaces to the system. This is the “View” portion of the application. Stegaid’s “Views” shall be based on Qt although any library could be used in the future as the Controller defined in (5.3 Components Description) actually performs all operations including validation.

Where possible, the application view should provide shortcuts. Also displays should flow in a specific order; for example, when the tab is pressed the next logical control should be activated.

### 5.4.1.1 Login

The Login Screen will enable authentication to the system. The authentication can [optionally] be bypassed by activating the “Guest” Mode feature if enabled.

|  |  |  |
| --- | --- | --- |
| Picture1  Figure 27: Design - frmLogin | | Close after three authentication failures  Only show if Guest Mode enabled  Password Characters |
| Name | frmLogin | |
| Shortcuts | Alt-L Login  Alt-G Guest  Alt-X Exit | |
| Additional Points | Password Text Field is Password Char.  Objects should grow depending on the resolution of the screen. | |
| Logical Flow | Username -> Password -> Login | |

Table 31: User Interface Component frmLogin

### 5.4.1.2 Main Menu

The Main Menu provides access to the main functions of the application, namely provide Encode and Decode capabilities, Send and Receive Mail and perform System Wide Configuration.

|  |  |  |
| --- | --- | --- |
| Picture3  Figure 28: Design - frmMainMenu | | Only show if user has permissions to amend Settings  Only show if Mail Enabled |
| Name | frmMainMenu | |
| Shortcuts | Alt-V View Messages  Alt-S Send Message  Alt-E Encode Text  Alt-D Decode Text  Alt-T Settings  Alt-X Exit | |
| Additional Points | Depending on the settings not all areas are visible. Objects should grow depending on the resolution of the screen. | |
| Logical Flow | View Messages -> Send Message -> Encode -> Decode -> Settings -> Exit | |

Table 32: User Interface Component frmMainMenu

### 5.4.1.3 Encode Text

The Encode Screen shall enable the user to encode a series of bits by calling the controller’s application programming interface and allowing the user to save the results.

|  |  |  |
| --- | --- | --- |
| Picture6  Figure 29: Design - frmEncode | | Close if no methods available. |
| Name | frmEncode | |
| Shortcuts | Alt-C Close  Alt-. Select File  Alt-E Encode | |
| Additional Points | The Encode Button should disappear and/or Transform into a Save As button, so the user can save the generated file which can then be decoded later. Note; that whilst the application is suited for smaller amounts of information that application will be able to operate on any file (in theory). | |
| Logical Flow | ... -> Method -> Encode -> Close | |

Table 33: User Interface Component frmEncodeText

### 5.4.1.4 Decode Text

The Decode Screen shall enable the user to decode a cover text by calling the controller’s application programming interface and allowing the user to save the results.

|  |  |  |
| --- | --- | --- |
| Picture8  Close if no methods available.  Figure 30: Design - frmDecode | |  |
| Name | frmDecode | |
| Shortcuts | Alt-C Close  Alt-. Select File  Alt-D Decode | |
| Additional Points | The Decode Button should disappear and/or Transform into a Save As button, so the user can save the generated file. Note that whilst the application is suited for smaller amounts of information that application will be able to operate on any file (in theory). | |
| Logical Flow | ... -> Method -> Decode -> Close | |

Table 34: User Interface Component frmDecodeText

### 5.4.1.5 Send Mail

The send mail function shall enable the user to send an encoded text based email to a recipient by calling the controller’s application programming interface. The recipient can then receive the email and decode as necessary.

|  |  |  |
| --- | --- | --- |
| Picture4  Close if no mail account set up.  Close if no methods available.  Figure 31: Design - frmSendMessage | |  |
| Name | frmSendMessage | |
| Shortcuts | Alt-C Close  Alt-S Send | |
| Additional Points | The application should via the API, Encode the Text and Call the Mail API for Sending. To maximise usability, if the user has no mail account set up, inform the user and close the display prior to allowing any fields being filled out. | |
| Logical Flow | Email Address -> Enter Message -> Method -> Send -> Close | |

Table 35: User Interface Component frmSendMessage

### 5.4.1.6 Receive Mail

The receive mail interface shall enable the user to download messages from a compatible mail server. Valid emails shall then enable the application to decode and display the hidden message.

|  |  |  |
| --- | --- | --- |
| Picture5  Close if no email account set up.  Close if no methods available.  Figure 32: Design - frmReceiveMessages | |  |
| Name | frmReceiveMessages | |
| Shortcuts | Alt-C Close  Alt-D Decode | |
| Additional Points | The email shall be displayed and after decode the result of the email shall be displayed when selecting a message. | |

Table 36: User Interface Component frmReceiveMessages

### 5.4.1.7 Settings

A tabbed interface shall separate the logical groups for administering system settings. Administrative functions shall exist for any general settings, addition and deletion of Groups (with selected permissions) and Users. If the mail option is enabled the application shall also permit the addition of mail servers for each user.

|  |  |  |
| --- | --- | --- |
| Picture9  Figure 33: Design - frmSettings | |  |
| Name | frmSettings | |
| Shortcuts | Alt-C Close | |
| Notes | There is no “Save” button to save system settings. This should occur when either a tab switch takes place or the form is closed. | |

Table 37: User Interface Component frmSettings

### 5.4.2 External System Interfaces

There shall be a number of external interfaces to the system. There is the Steganographic Plugin and the Data Plugin. Plugins shall be loaded and unloaded on the fly thus minimising memory requirements and enabling portions of the program to be updated without requiring a whole system upgrade.

### 5.4.2.1 Steganographic Plugin

The Steganographic Plugin interface shall be used for implementing plugins that the Steganographic Plugin Manager shall load and unload on demand. The structure of the interface is defined in Table 38: The Steganographic Plugin Interface.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| getPluginName() | Gets a User Friendly name for the plugin   |  |  | | --- | --- | | Requires | None | | Return | Plugin Name: String | | Validation | None | |
| getClassID() | Gets a unique class identifier for the system to use. These shall be listed in Appendix 3.   |  |  | | --- | --- | | Requires | None | | Return | Plugin’s Registered Class ID: String | | Validation | None | |
| encodeText() | Encodes Text as appropriate in the format defined by the plugin.   |  |  | | --- | --- | | Requires | Unencoded: StegoItem  Container: \* Widget Reference [Optional but required to be implemented]. An area to output information to the user. If Zero use std::out. | | Return | Encoded: StegoItem | | Validation | Perform a Decode after Encoding to ensure that the original bits can be recovered. | |
| decodeText() | Decodes Text as appropriate in the format defined by the plugin.   |  |  | | --- | --- | | Requires | Encoded: StegoItem  Container: \* Widget Reference [Optional but required to be implemented]. An area to output information to the user. If Zero use std::out. | | Return | Decoded: StegoItem | | Validation | None | |

Table 38: The Steganographic Plugin Interface

### 5.4.2.2 Data Plugin

The Data Plugin Interface shall provide an alternative data provider to the built in one. The built in data provider will use SQLite Embedded and the use of this plug-in will bypass the built in system as much as feasibly possible enabling a different data store. Such a data store could be anything from Text files, XML, or another database management system.

The implemented data plugin shall use the format defined in Table 39: The Data Plugin Interface.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| getPluginName() | Gets a User Friendly name for the plugin   |  |  | | --- | --- | | Requires | None | | Return | Plugin Name: String | | Validation | None | |
| performAuthentication() | Authenticates a User given Credentials   |  |  | | --- | --- | | Requires | UserName: String  Password: String | | Return | Success: Boolean | | Validation | Plugin Dependent | |
| getUserDetails() | Get Details About Users   |  |  | | --- | --- | | Requires | None | | Return | User Details of the Current User | | Validation | None | |
| getSetting() | Gets a System Setting. In this scenario a system setting will apply to all systems using the plugin. In a shared environment with multiple Personal Computers, settings, authentication and so forth can be accessible from other computers using the same data source.   |  |  | | --- | --- | | Requires | Key: String  GroupName: String [Optional] | | Return | Value to the setting or “False” if not set | | Validation | Plugin Dependent | |
| setSetting() | Sets a System Setting.   |  |  | | --- | --- | | Requires | Key: String  GroupName: String [Optional]  Value: String | | Return | void | | Validation | None | |
| addUser() | Add a User to the System   |  |  | | --- | --- | | Requires | Username: String  GroupId: Integer  Forename: String  Surname: String  Password: String | | Return | Success: Boolean | | Validation | Plugin Dependent | |
| deleteUser() | Delete a User from the System   |  |  | | --- | --- | | Requires | Username: String | | Return | Success: Boolean | | Validation | Plugin Dependent | |
| listUsers() | Lists Users in the System   |  |  | | --- | --- | | Requires | None | | Return | List of Users | | Validation | None | |
| addGroup() | Add a Group to the System   |  |  | | --- | --- | | Requires | GroupName: String  Permissions: String | | Return | Success: Boolean | | Validation | Plugin Dependent but GroupName must be Unique | |
| deleteGroup() | Delete a Group from the System   |  |  | | --- | --- | | Requires | GroupName: String | | Return | Success: Boolean | | Validation | Plugin Dependent | |
| listGroups() | List Groups in the System   |  |  | | --- | --- | | Requires | None | | Return | List of Groups | | Validation | None | |
| getGroupId() | Provide the Group ID given the Group Name   |  |  | | --- | --- | | Requires | GroupName: String | | Return | GroupID: Integer | | Validation | Plugin Dependent | |
| getPermissions() | Gets the Permissions associated with a Group   |  |  | | --- | --- | | Requires | GroupName: String | | Return | Unix Style Permissions: String | | Validation | Plugin Dependent | |
| emailAddressExists() | Get whether an email address exists within the system   |  |  | | --- | --- | | Requires | EmailAddress: String | | Return | Exists [True]: Boolean | | Validation | Plugin Dependent | |
| getMailUsers() | Lists Mail Users in the System   |  |  | | --- | --- | | Requires | None | | Return | List of Mail Users as not all listUsers() have mail accounts. | | Validation | None | |
| addMailUser() | Add Mail User to the System   |  |  | | --- | --- | | Requires | EmailAddress: String  UserName: String | | Return | Success: Boolean | | Validation | Plugin Dependent. A Username may or may not be the Email Address. This system shall be flexible enough to allow them to be separate entities. | |
| addMailOption() | Add a protocol to an email address. An email address will generally have two (or more) protocols associated. Sometimes this is SMTP and POP but could be SMTP/TLS or POP3/TLS or multiple variations thereof.   |  |  | | --- | --- | | Requires | ProtocolId: Integer  EmailAddress: String  HostName: String  Port: Integer  UserName: String  Password: String | | Return | Success: Boolean | | Validation | Plugin Dependent | |
| removeMailOption() | Remove a Protocol for a specified email address.   |  |  | | --- | --- | | Requires | EmailAddress: String  ProtocolId: Integer  HostName: String  Port: Integer  UserName: String | | Return | Success: Boolean | | Validation | Plugin Dependent | |
| removeMailUser() | Removes a Mail User   |  |  | | --- | --- | | Requires | UserName: String | | Return | Success: Boolean | | Validation | Plugin Dependent | |
| getMailCredential() | Provide the Mail Credentials for a specified User, given the Host and Port   |  |  | | --- | --- | | Requires | UserName: String  HostName: String  Port: Integer | | Return | Success: Boolean | | Validation | Plugin Dependent | |

Table 39: The Data Plugin Interface

**Where Success is returned in all cases, the value shall be False (Zero), or True or (Non Zero) when failure occurs.**

## 5.5 Data Description

The application will require the storage of various settings. These general settings include mail credentials, authentication settings and libraries (plugins) that the system can load. The logical groups of settings are defined in Table 40: Data Description Overview.

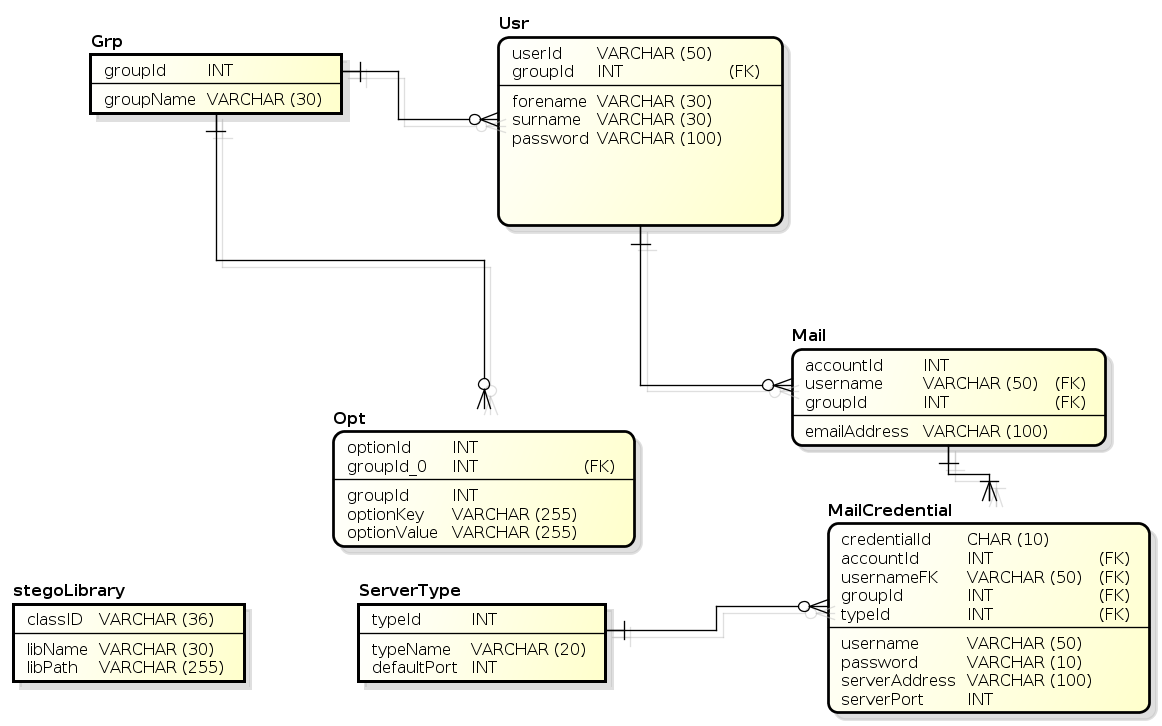
|  |  |
| --- | --- |
| Category | Description |
| User Settings | User authentication and preferences enables the security of the application (if required by systems administrators). Without authentication mail cannot be accessed as there is no way to store email account information. |
| Data Library Settings | Both authentication and user preferences are stored internally within an embedded database management system. Alternatively the application can use an external authentication source, which could be anything from a flat file to another database or SOAP provider. |
| Steganographic Library Settings | The nature of the application enables the use of different libraries and load them on demand. In the context of this application it is limited to steganography, but libraries can be updated without recompiling the core program every time (plug in architecture). Libraries can be extended beyond steganography to encryption, compression or any other operation imaginable that a person may wish to perform. |
| Data Library Settings | An external authentication module enables the application to be used in a shared environment with a single data source. |

Table 40: Data Description Overview

The embedded database management system shall have the following characteristics: This data dictionary displays the data type, limits and options that should be followed. Whilst this design is used internally, it can also be used as a basis for an external database management plan.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Variable | Type | Require | Limit | Options |
| Grp (alias Group) | | | | |
| groupId | Integer | Y | - | Primary Key |
| groupName | Varchar(30) | Y | >3  <30 | Unique |
| Usr (alias User) | | | | |
| userId | Integer | Y | - | Primary Key |
| Forename | Varchar(30) | Y | >3  <30 |  |
| Surname | Varchar(30) | Y | >3  <30 |  |
| Password | Varchar(100) | Y | >6  <30 | Encrypt |
| Opt (alias option) | | | | |
| optionId | Integer | Y | - | Primary Key |
| optionKey | Text | Y |  |  |
| optionValue | Text | Y |  |  |
| Mail | | | | |
| accountId | Integer | Y |  | Primary Key |
| emailAddress | Varchar(100) | Y | >6  <30 | Unique |
| MailCredential | | | | |
| credentialId | Integer | Y | - | Primary Key |
| Username | Varchar(50) | N |  |  |
| Password | Varchar(100) | N |  | Encrypt |
| serverAddress | Varchar(100) | Y | >8  <100 |  |
| serverPort | Integer | N | Min 1  Max 65535 |  |
| serverType |  |  |  |  |
| typeId | Integer | Y | - | Primary Key |
| typeName | Varchar(20) | Y | >3  <20 |  |
| defaultPort | Integer | Y | Min 1  Max 65535 |  |
| stegoLibrary | | | | |
| classID | Char(36) | Y | Fixed Length | GUID |
| libName | Varchar(30) | Y | >6  <30 |  |
| libPath | Varchar(255) | Y | >6  <255 |  |

Table 41: Data Dictionary

Figure 34: Entity Relationship Diagram

# 6 Implementation

## 6.1 Compilation Guidelines

### 6.1.1 Target Environments

In order to build the target application, a number of different compilation tests have been executed on differing platforms. The target platforms requirements/theoretically supported are listed in Table 42: Implementation Compilation Guidelines.

|  |  |  |  |
| --- | --- | --- | --- |
| Target Platform | Core Application [Console View] | Core Application [GUI View] | Tests Executed  [U denotes untested] |
| Linux | ✓ | ✓ | GCC 4.8 |
| Microsoft Windows | ✓ | ✓ | MSVC 2012 SP1  MinGW 4.8 (x64) |
| Android | ✓ | Requires a new view better suited to small devices | [U] |
| Solaris | ✓ | ✓ | [U] |
| HP-UX | ✓ | ✓ | [U] |
| Embedded Systems | As part of another application | ✓ | [U] |

Table 42: Implementation Compilation Guidelines

### 6.1.2 System Requirements

The system requirements are extremely light beyond compilation of the core application. If the target device is hardware limited a cross compiler should be used on a different platform and computer.

|  |  |  |
| --- | --- | --- |
| Hardware | Target device | Build device |
| Memory | <64Mb Available RAM [Console]  512Mb Available RAM [GUI] | Recommended compile environment has a minimum of 2Gb or RAM. If you wish to compile all libraries (namely Qt) from source, the task was only successful with more than 10Gb Swap Space. |
| Processor | Single Core 1GHz | Dual Core Recommended or more. |
| Hard Disk | 200Mb Free. | 10Gb recommended. |
| Graphical Processor | None if using core application as part of another application [Console]  Required [GUI] | Console (Shell TTY), or Microsoft Windows GUI. |

Table 43: System Requirements

The system will also require a number of runtime libraries and software in order for it to operate. These runtime libraries are listed in Table 44: Library Requirements.

|  |  |  |
| --- | --- | --- |
| Software | Target device | Build device |
| Console View | Poco 1.5.2 Minimum  C++ Runtime | Poco Development Libraries [1.5.2]  Qmake [Qt Qmake from Qt4 or Qt5]  C++ Compiler MSVC or GCC [tested] |
| GUI View | Poco 1.5.2 Minimum  C++ Runtime  Qt 5 Runtime [5.2 or higher is recommended, as the view look different with different versions making this project design somewhat difficult] | Poco Development Libraries [1.5.2]  Qmake [Qt5]  C++ Compiler MSVC or GCC [tested] |

Table 44: Library Requirements

### 6.1.3 Compilation Guidelines

### 6.1.3.1 Ubuntu™ Host and Target

The following instructions were tested using a straight compile i.e. not cross compile on Ubuntu 64 bit 13.04 and compiles everything from source including runtime libraries. The application will fail using current versions of Poco and Qt that are held within the Ubuntu repositories as they are not using the latest APIs.

|  |  |
| --- | --- |
| Step 1 | System Update:  From a terminal window (or shell): |
| 2 | Prerequisite Install:  These include SSL development libraries required for the TLS functions within Stegaid. |
| 3 | Qt Compilation:  Firstly Qt Sources must be downloaded. Whilst Qt can remain in your home directory a better option is to install globally for all users. |
| 4 | Unzip the tar ball. |
| 5 | Move to Global Directory /Opt [Optional]    Change permissions. The following assumes a single user environment. Alternatively use sudo. |
| 6 | Configure |
| 7 | Make |
| 8 | If you are constrained on memory, the likelihood of build failure due to lack of memory is high during the link phase. A message will be shown like so:    To alleviate this temporary virtual memory can be created and released manually:    This creates an image 8 Gigabytes in size for the creation of a swap image. Now the swap must be enabled:      If the error was encountered, make must be called again:    To recover disk space after allocation: |
| 9 | After Qt has been compiled, the Poco libraries and runtime are a core requirement of the system. First return to your home directory: |
| 10 | Download Sources (1.5.2 Minimum) |
| 11 | Unzip tar ball |
| 12 | Enter directory and configure:      In this example ODBC is omitted as it is not required by the application. |
| 13 | Now the libraries have to be compiled: |
| 14 | In order for the application to access them, these libraries must be installed: |
| 15 | Stegaid must now be compiled:  Stegaid has different parts to the application. There is the Core Program that shall be compiled first and the plugins which are compiled independently.  Firstly the tar ball must be unzipped: |
| 16 | A build directory should be created, this is to keep a clean source directory: |
| 17 | And compiled: |
| 18 | Executing qmake will generate a Makefile (on Linux and Unix).  From here, you can type “make” to compile the application, which will result in:    From here the application is ready to run:  ./stegaid If not Window Manager is running the application will fail, If command line arguments are present the application will not attempt to create a Window and run from the command line instead. |
| 19 | Plugins will also be needed to operate the application (in our case Steganographic plugins).  For example: /opt/Qt/qtbase/bin/qmake \ ../../plugins/libStegoOpenSpace/libStegoOpenSpace.pro will build the Open Space plugin in the current application directory.  Certain files are also necessary such as thesauruses and dictionaries (in data.tar.gz), these should be unzipped and copied over to your application directory also. |
| 20 | Please check Appendix 2 Administrative Guidelines for recommendations regarding the application. |

Table 45: Ubuntu Variant Compile Guidelines

### 6.1.3.2 Microsoft® Windows™ Host and Target

This example uses the freely available Microsoft C++ Express Version 2012. It is noted that whilst the Linux Version generates nearly zero compiler warnings, due to differences within platforms more compiler warnings are generated with Microsoft’s C++ Compiler.

Install Microsoft Visual Studio Express (2012 Desktop Express or Professional is recommended to use the scripts). On the Resources DVD are the following prebuilt binaries for Microsoft Windows 32 bit and 64 bit systems consisting of the following dependencies:

|  |  |
| --- | --- |
| Dependency | Included Version |
| Cygwin | 64 bit make, dos2unix (Build Tool) |
| ICU (International Components for Unicode) | 53.1 (Library) |
| MySQL Client Libraries | 6.1.3 (Library) |
| Perl | 64 bit (Build Tool) |
| Poco | 1.5.2 (Library) |
| Qt | 5.2.1 (Library) |
| Ruby | 64 bit (Build Tool) |
| Qt Creator | 3.0.1 (Build Tool) |
| StegAid | (Target Application) |

Table 46: Microsoft® Windows™ Build Prequisites

To build, all libraries are pre-compiled although you can compile each if chosen to do so. All build tools are 64 bit; all libraries are 32 bit and 64 bit (depending on the target required). Software Development Kit Directory Structure is shown in Figure 35: Library/Executable Dependency Directory Structure.

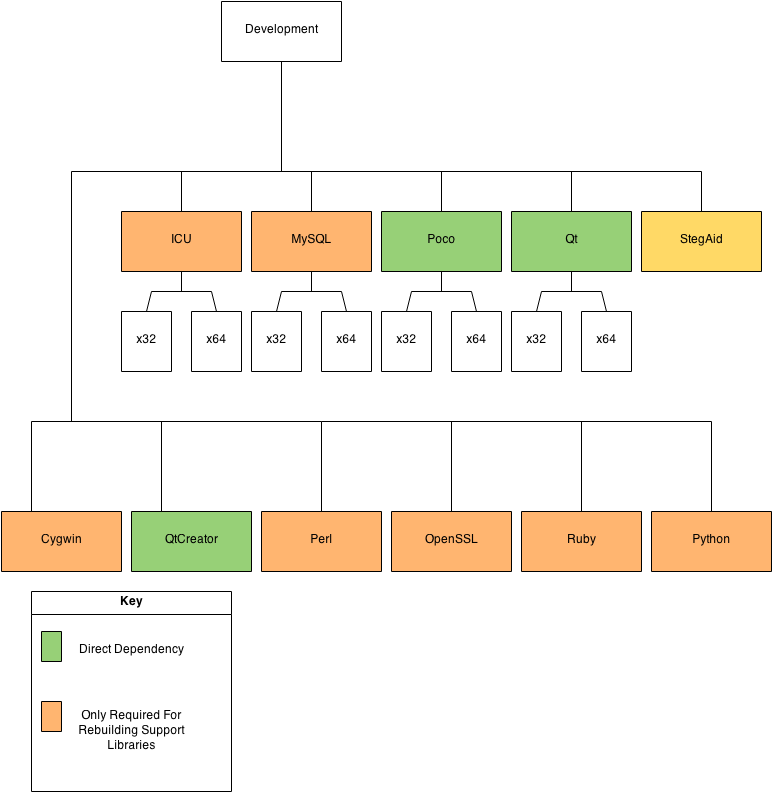


Figure 35: Library/Executable Dependency Directory Structure

Areas marked in green represent the components used directly (a Direct Dependency). Amber represents tool chains and libraries used by the libraries we are using (Indirect Dependency), these are all included in the SDK. In the case that the Visual C++ Compiler is a different version, you will need to rebuild the libraries for that version. You should not mix 2012 runtime with other versions, nor mix architectures between 32 bit and 64 bit.

|  |  |
| --- | --- |
| Step 1 | In order to compile, the Development Folder should be copied to your C:\ drive, this enables us to use the batch scripts and sets up Paths as appropriate. |
| 2 | Enter the Development Folder in your C:\ drive and double click cmd:  Screenshot from 2014-04-08 15:53:50 |
| 3 | Depending on the type of build you wish type in x64.bat (not IA-64 but x86\_64 architecture) or x86.bat in the console prompt:  Selection_001 |
| 4 | In this example, this will set up the system for a 64 bit build.  Let’s create a build directory and enter it: mkdir build && cd build:  Build qmake ..\StegAid\src\stegaidunix.pro -p |

Table 47: Microsoft® Windows™ Compilation Guidelines

## 6.2 System Implementation

The final implemented system is extremely similar to the design; however, due to User Evaluations that have taken place small changes have occurred to try and improve usability. It was recognised that issues existed with the colour scheme chosen, so that background was made a lighter pastel shade and the font darker.

### 6.2.1 Splash Screen

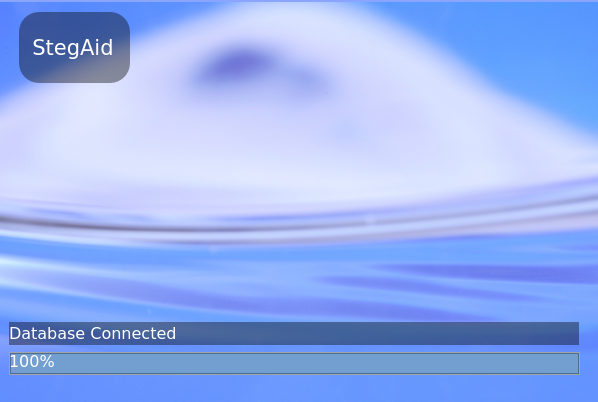


Figure 36: Implementation - Splash Notification

The splash screen displays status messages whilst the application is loading. If that application has not finished executing after two minutes the splash screen remains in place whilst loading is being carried out, otherwise it closes.

### 6.2.2 Authentication

After the Splash Screen has loaded, the system prompts for user credentials. After the failure of input of credentials 3 times, the application will terminate. If authentication is successful, the user is passed through to the Main Menu Interface. Specifically the Password has hidden characters for security. A guest mode button appears if guest mode is checked in the system settings.

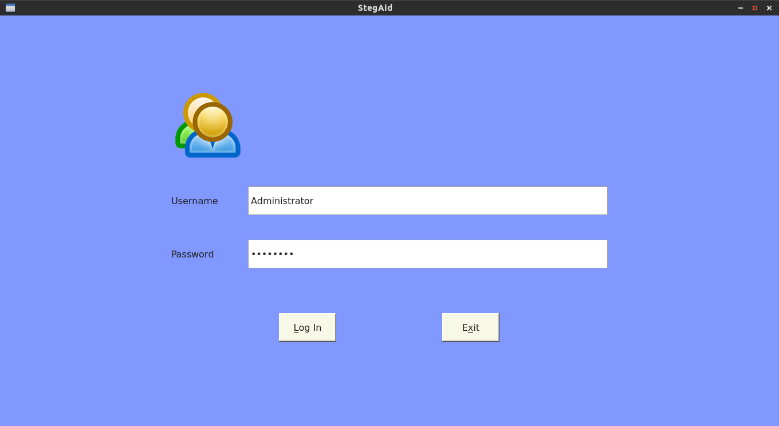


Figure 37: Implementation - Authentication Display

### 6.2.3 Main Menu

If successful authentication occurs within three attempts, the Main Menu displays. The Main Menu allows for the sending and receipt of mail, the encoding and decoding of files and access to settings. If you belong to a Group elevated permissions you will see the Settings area, likewise, if you do not have access to mail, these options will also not be visible. In the example provided in Figure 38: Implementation - Main Menu the user has full administrative privileges. The buttons stand out significantly as users felt they could not identify these controls. Icons were also added as appropriate (along with the original planned shortcuts).

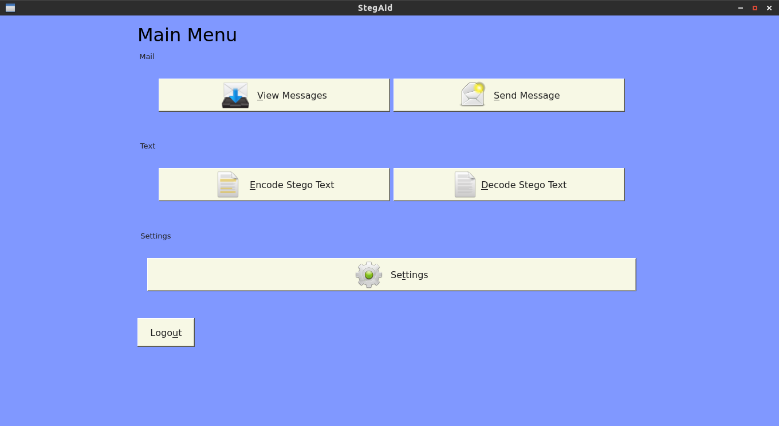


Figure 38: Implementation - Main Menu

### 6.2.4 Encode Text

The primary capability of Stegaid is to encode and decode data using a variety of formats by way of text based steganography. Four selected methods currently exist using the plugin architecture. To hide information, the Encode button can be selected. At first all controls are hidden and activated only as appropriate to prevent confusion to the user, although a HTML Help File is also provided.

Firstly the user should select a file to be hidden, be it some text, an image or other document. The only way to enter file is to click “Select File” which will provide the user with an Open File Dialogue.



Figure 39: Implementation - Encode Text File Selection

Upon selection of a file, more controls become active. All registered libraries (plugins) are shown using a radio option select. If an option has been selected, an “Encode” button appears.

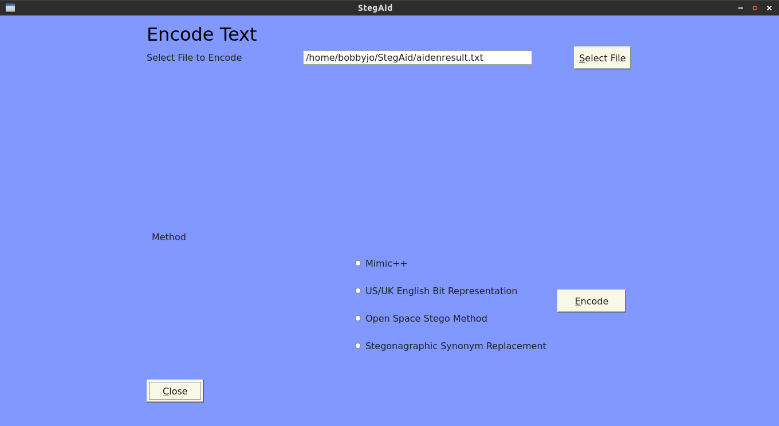


Figure 40: Implementation - Encode Text Method

Once Encode has been selected, the Core application, (Controller) loads the selected plugin, Encodes the information before returning with the result and status messages. Encoding provides additional information such as the time taken to encode a file which can be useful for evaluation algorithms. In order to validate the encoding was successful, the performance timer is stopped and the cover text decoded and validated against the original file. There is currently no way to disable this feature and is considered essential to inform the user when things fail. Also some algorithms have capacity limits, these checks also occur.

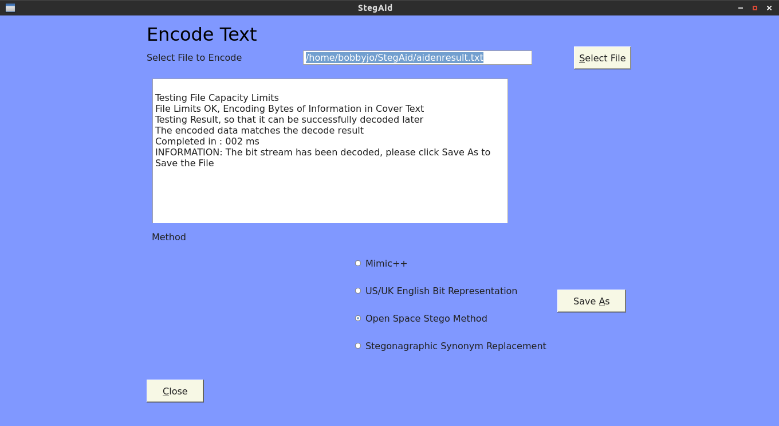


Figure 41: Implementation - Encode Text Status

After Encoding has taken place, all invalid controls become disabled, and a “Save As” button becomes activated. The user can then activate this button to show a “Save As” dialogue and save the resulting cover text. Upon success confirmation is provided as to the status of the save process.

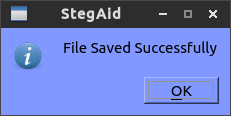


Figure 42: Implementation - Encode Confirmation

### 6.2.5 Decode Text

The decode text, opens a cover text and attempts to decode as appropriate. Like the Encode Text function, a stepwise approach to information elicitation from the user takes place to help avoid confusion to new users.

After selecting the appropriate option from the Main Menu a Decode Text display is provided.

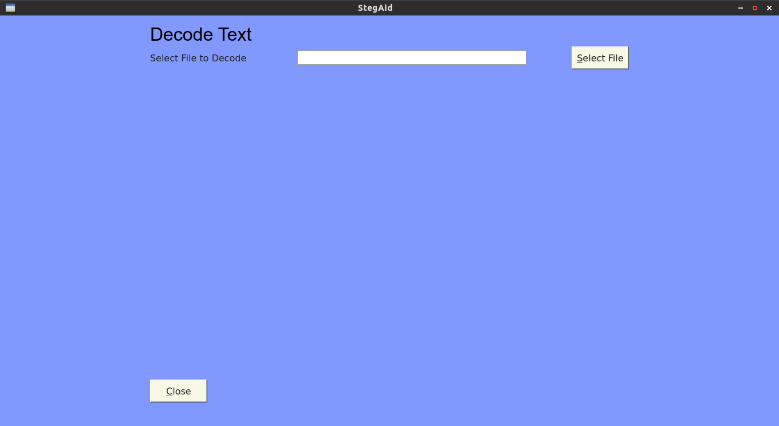


Figure 43: Implementation - Decode Text File Selection

After a file has been selected the additional options become active. The user must know how the file was encoded. Most steganographic systems have a single method, Stegaid has many. Consideration was made to placing a signature or embedded header into the cover text, however, this approach was not adopted in this release as it was felt a security risk to put implementation details of how a file is encoded into the cover text. If an incorrect method is selected, the decoding process will still occur, except the returned bytes of information will differ from the originally encoded data.

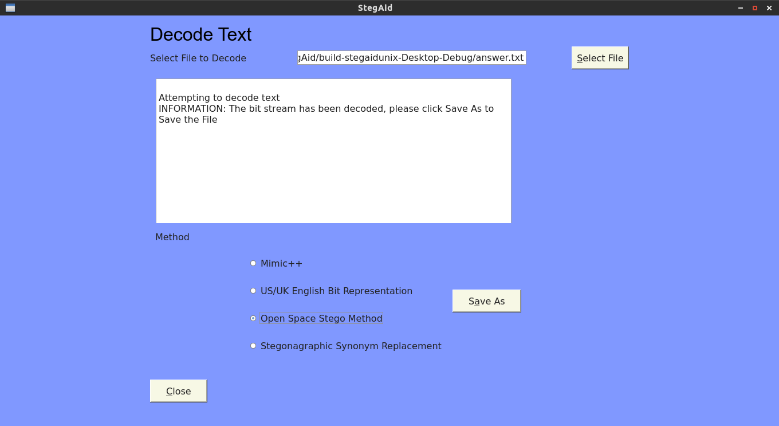


Figure 44: Implementation - Decode Text Status

After the file selection and appropriate method selected the user can save the result file be it an image, or document. If the result was successful confirmation is provided as appropriate.

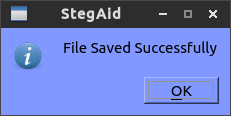


Figure 45: Implementation - Decode Text Confirmation

### 6.2.6 Receiving Encoded Messages

The Stegaid application toolkit provides functionality for encoded communications. Currently through the use of support libraries, SMTP, SMTP/TLS, POP3 and POP3/TLS email protocols are supported. To receive messages, selecting the appropriate option on the menu.

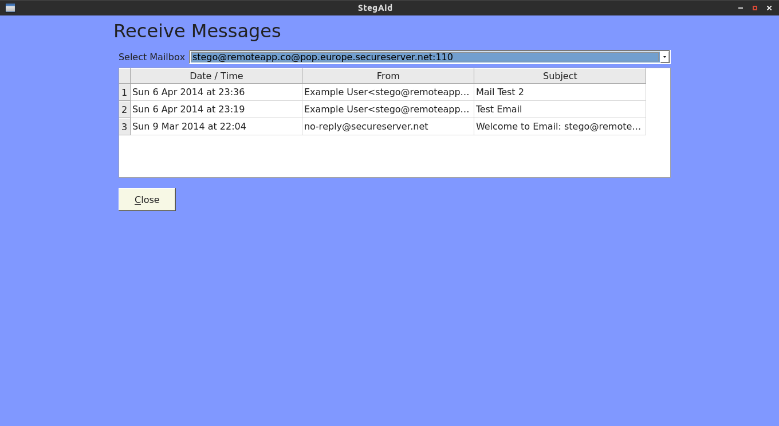


Figure 46: Implementation - Receive Messages

Currently as no message has been selected, no operations can be performed, hence no options have been enabled. Once a message is clicked relevant options become available.

The transformed display in Figure 47: Implementation - Received Message View shows once such decoded message. As a rule messages are downloaded but not deleted, this is intentional as Stegaid only supports encoded messages. Multipart MIME type messages are listed as un-encoded messages as they were not encoded by Stegaid.



Figure 47: Implementation - Received Message View

If the user does not have any mail accounts set up but does have mail permissions, then a message box is displayed as soon as the display is loaded.

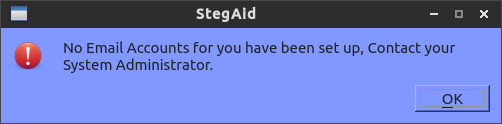


Figure 48: Implementation - No Account Set Up

### 6.2.6 Sending Encoded Messages

As well as receiving encoded messages, mail messages can be encoded and sent using a compatible mail server such as Gmail™, and Go Daddy™. As with the receiving of messages, the system checks for presence of mail accounts for the current user (along with other tests). This is to prevent the user typing a mail message only to find out that the message cannot be sent. Upon the successful sending of the message a confirmation message is displayed as appropriate.

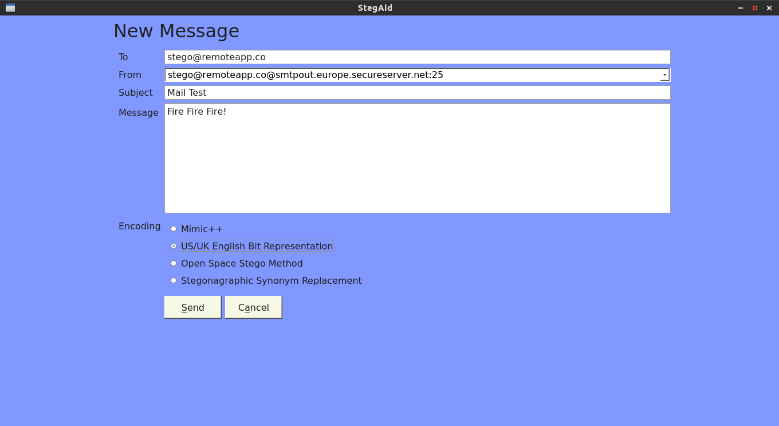


Figure 49: Implementation - New Message

### 6.2.7 System Administration

To manage the system and enable some customisability a “Settings” feature has been implemented. This section has been locked down to any Group that has the appropriate permissions. Using a tabbed interface, items are in related groups, such as “General Settings”, “Groups”, “Users”, “Libraries” and “Mail”.

### 6.2.7.1 General Settings

The general settings tab enable the ability to disable the console, allow for external authentication using a data plugin, enable mail and guest mode. Significantly there is no “Save” button, save occur internally either when the display is closed or a tab switch takes place.

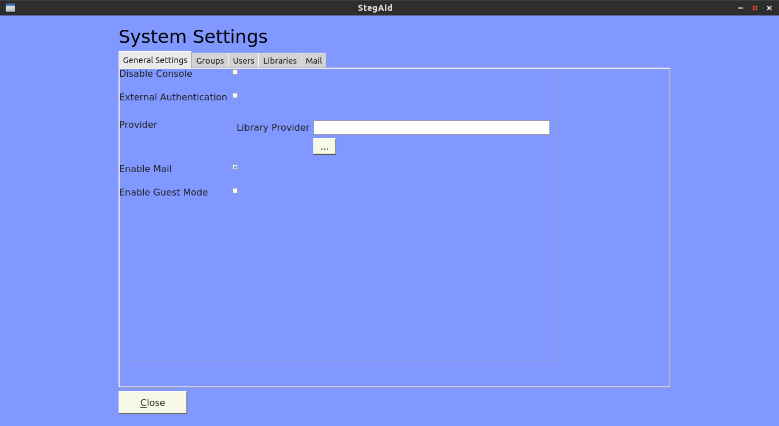


Figure 50: Implementation - General Settings

### 6.2.7.2 Groups

Groups are logical grouping of Users that have shared permissions. A certain group may have access to Mail but not System Settings, likewise, another Group may have access to no Mail or System Settings. Group Names must be unique and are case insensitive along with other validation requirements. Plain English helpers will always alert the user to any issue before appending to the database. Administration is designed to be as simple as possible so the only options are to Delete and Add using the + and – icons, showing only relevant fields at the time. The system also checks for orphan Users that are a member of the Group to be deleted and will prevent Group deletion until they are deleted.

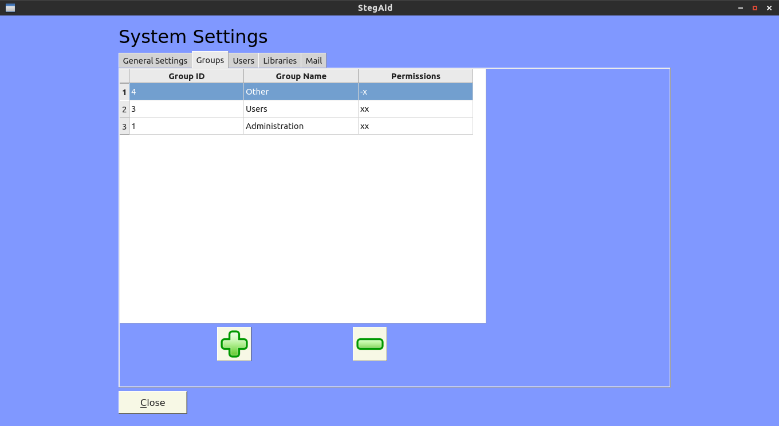


Figure 51: Implementation - Groups

### 6.2.7.3 Users

There can be many users to the system and they can be managed from this tab. A simple Create, Read and Delete interface has been provided for User Administration. Users can be added if certain validation proves true such as the password containing at least 6 character is contains both alphabetic and numeric characters.

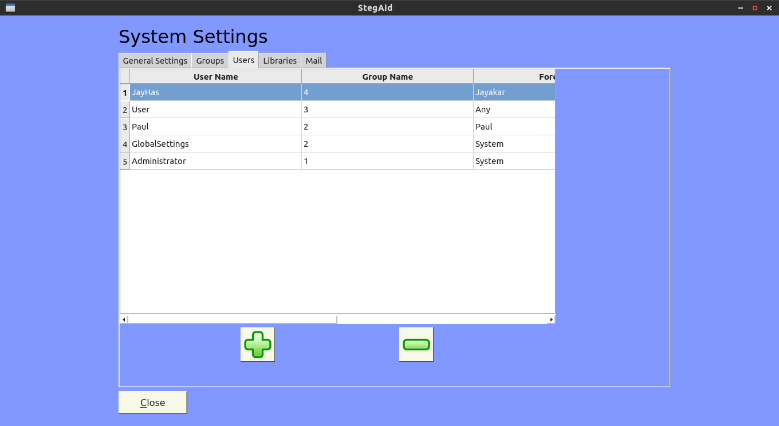


Figure 52: Implementation – Users

As with groups, no fields are visible until a request to add a User has been made. This is to simplify the interface to the System Administrator.

### 6.2.7.4 Libraries

The system was designed to be extensible from the outset, the System Administrator can on demand register and de-register libraries as needed. Buggy libraries can also be removed if necessary such as development versions and new versions be loaded on the fly without restarting the whole application. More importantly whilst version 1.2 solely targets text based steganography, version 2 will support other plugins such as Encryption and Compression by using a prioritisation system (only minimal change in the core application is required to accomplish this).

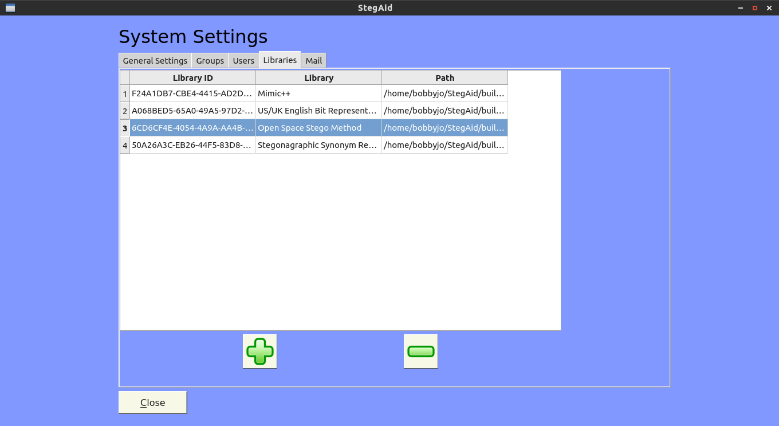


Figure 53: Implementation – Libraries

To add a library, the + icon is clicked. A “Select File” option is provided that loads a File Dialogue box showing only relevant \*.dll or \*.so files. The library is then loaded and tested to ensure it meets the criteria before registering it to the system. Valid libraries have a unique class identifier and a useful description.

### 6.7.2.5 Mail

If the mail option is enabled under 6.2.7.1 General Settings another interface becomes available, the Mail administration area. A user can have one email address but many protocols to support the email capability. Further to this there may also be backup servers and so forth. Upon changing the protocol, the default standard port changes as appropriate but is amendable as some servers have irregular ports.

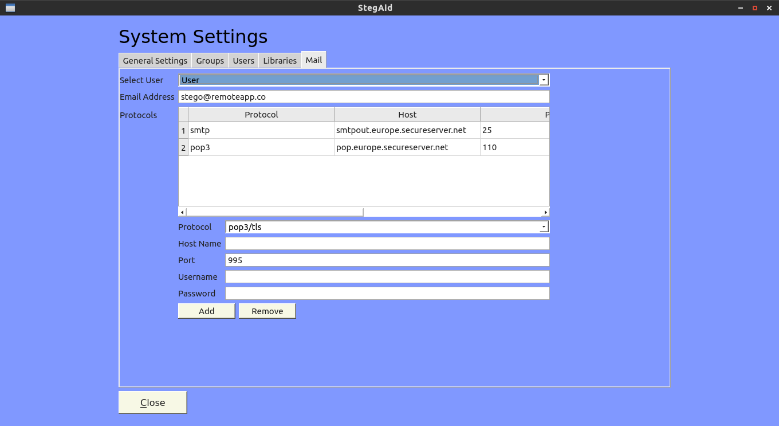


Figure 54: Implementation - Mail

# 7 IEEE Software Test Plan

This document shall detail the planning and the execution of testing procedures as defined by (Institute of Electrical and Electronic Engineers, 1998) standard 829.

## 7.1 Revision History

Also known as the Test Plan Identifier, as the software version increases, the test plan is revised. As each amendment takes place, a full test shall be executed.

|  |  |
| --- | --- |
| Revision | Details |
| 1.0 | Initial Specification |

Table 48: Software Test Plan Revision History

## 7.2 Introduction

In order to successfully test the software application, tests will occur on each form, testing each unit from start to finish. This is a traditional approach to software testing. The test process shall test the failure that will exist within the system. There are a variety of tests that will take place and are not limited to the core application itself.

## **7.3 Test Items**

The software system inclusive of the associated documentation shall be tested in a variety of ways (in order). This involves method testing within the application within each class (Unit Tests), through to documentation tests (Help Guides and so forth). Table 49: Test Items shows the items to be tested visually.

|  |  |
| --- | --- |
| Test | Description |
| Core Tests | Test of the base classes and methods (Unit Tests) for:  stegoText  stegoPluginManager  Group  User  Mail  Setting |
| User Interface Tests | Tests perform basic Human Computer Interface tests which involve third parties performing evaluation of the software system (Black Box).  Console Environment  Graphical Environment |
| Compilation Tests | Due to the nature of the application, cross platform tests will be carried out. The requirements for the initial version will target Microsoft® Windows™ and Linux™ Operating Systems with preference to supporting a wide array of future platforms.  Target MSVC Compiler  Target GCC Based Compiler |
| Library Tests | It is noted that each Stegaid method is external from the main application for upgrades, extensibility (and customisability). Indeed Stegaid is not limited to Steganography but could support compression and encryption in the future. Each method must be tested to verify output. |
| Requirements Validation | As stated, in order to ensure requirements are met/or not met, a preliminary prototype was presented in January 2014 based on requirements investigation in the previous December. This checklist will test each piece of information received and note issues arising. |

Table 49: Test Items

## 7.4 Tests Not Carried Out

There are a number of tests that will not be carried out. It is not feasible or in some cases appropriate to perform some tests for the reasons as stated.

|  |  |  |
| --- | --- | --- |
| Test | Description | Reason |
| Performance Tests | Tests the performance of the system in a given environment. | Tests of this nature are internal. Performance is dependent on the library in use and in each case a “trade off” occurs between performance and utilisation. In many cases in order to increase performance, more objects are loaded in memory to increase system response (such as dictionary search). In each library method performance is tested, and returned to the user in milliseconds. |
| Stress Tests | Tests the system under load or beyond the designed system capacity. | Again limited tests of this nature will take place. Many steganographic implementations have a limited storage capacity (can only embed so much information [bits]). In each case the library is expected to return that the information cannot be encoded. Unlike server software, Stegaid is purely client software. |

Table 50: Items not to be tested

## 7.5 User Interface Tests (Graphical User Interface)

The User Interface tests have a Test Number which is equal to one form (interface). Each subordinate number represents the test case. Upon execution of the testing phase a Pass or Fail should be noted and the steps taken to resolve the issue.

The table in Appendix 5 details items to be tested in this test plan. Each test item, will execute certain failure scenarios such as failure in validation to ensure a suitable response to the user, before executing a success scenario.

## 7.6 Test Result Execution

The results of the execution of these tests shall be placed in Appendix 6.

# 8 Critical Evaluation

There are various ways in which evaluation can take place and in this case what the evaluation can take place on. In order to evaluate the project as a whole; a review will consist of the areas defined in Table 51: Evaluation Areas.

|  |  |
| --- | --- |
| Area | Evaluate What |
| Steganographic Methods | The methods implemented within the application. Each method shall be evaluated using set criterion (both qualitative and quantitative) and displayed via a result set like the in figu  20140407041948  Figure 55: Example Evaluation Radial Chart  We can see from the example diagram the evaluation is based on five benchmarks:   * Speed - The speed in milliseconds for the encoding to take place. Note in the interested of fairness, our versions are all single thread and uncompressed. The more speed the lower the score. * Capacity - The number of bits that can be encoded within set limit. The less capacity the lower the score. * Complexity - The difficulty (estimated) in implementing the algorithm. The more complexity the lower the score. * Compromisability - The difficulty (without knowing the algorithm) on suspecting hidden information. The ease of breaking the algorithm the lower the score. * Size - The result set size in comparison to other methods. The more the result set size the lower the score.   The target for any algorithm is to have a score of 5 in all key areas:    Figure 56: Target Score Radial Chart |
| Application | The application evaluation is the process of applying critique to the resulting software. Evaluation takes multiple forms and includes the following methods:   * Usability Testing - The evaluation of responses by ten users who have tried this software in order to obtain directions for either improvement in the current release or a future release. * Project Evaluation - The personal evaluation of the project detailing issues that have arisen, what should be done to improve and recommendations should a future release occur. Time constraints or other reasons will be noted as to why these aren’t incorporated into this first release. |

Table 51: Evaluation Areas

## 8.1 Steganographic Methods

In order for a fair testing and evaluation take place all methods use a single thread (as some can only be performed sequentially). Also whilst Stegaid is perfectly capable of Compression and Encryption (should a plugin be developed), no algorithms shall make use of a these technologies. Some research papers were using compression and/or encryption in their tests and are disregarded in this application.

### 8.1.1 Open Space Encoding

This method is the simplest method that was chosen, which involves the encoding of bits between the spaces of words. Depending on the implementation, the encoding can be different. In this implementation, a simple option was chosen whereby a zero bit (off) has one space and a one bit (on) has two spaces.

The encode limit is 1 bit per word and across 1000 words, therefore 125 bytes of information can be encoded.

The current implementation using a randomly generated document from Project Gutenberg, an image of 513 Bytes could be encoded into 5205 bytes.

|  |  |  |
| --- | --- | --- |
| Encode | Partial Result | Decode |
| C:\downloads\openspacedecodedog.png | The Project Gutenberg EBook of Beyond Good and Evil, by Friedrich Nietzsche  This eBook is for the use of anyone anywhere at no cost and with almost no restrictions whatsoever. You may copy it, give it away or re-use it under the terms of the Project Gutenberg License included with this eBook or online at www.gutenberg.org | C:\downloads\openspacedecodedog.png |

The tests in this implementation show that whilst Open Space is a simple solution however only small images such as icons and/or a small amount of textual information can be stored.

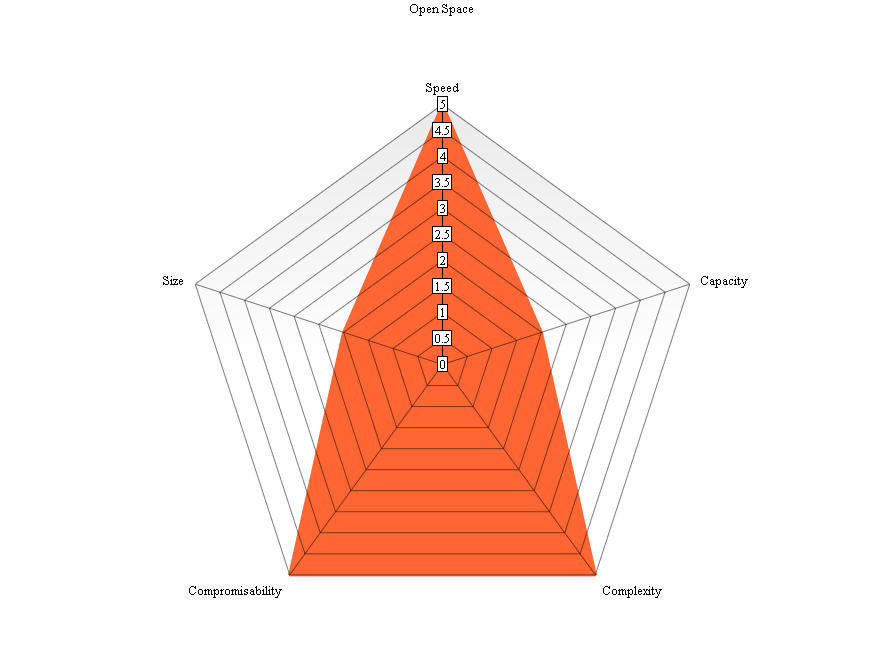


Figure 57: Open Space Algorithm Evaluation

### 8.1.2 Synonym Replacement

The process of using a thesaurus which is identical on both clients can be used to store information. For each word in a given text a word is replaced with a synonym of that word. Again implementations differ, such as XXXXX which uses frequency analysis. Our implementation is simpler in that a given word is searched through the thesaurus to generate a suitable word list. The greater the number of words in the generated list, the more bits can be encoded (variable bit encoding). If there are three synonyms, two bits can be encoded or seven synonyms 3 bits. In some cases there are more than 60 synonyms for a given word so the number of bits that can be encoded is higher.

It is noted that currently this plugin does not identify a context for the given synonym, for example:

The “cat” sat on my lap

The “lion” sat on my lap

Logically this sentence is correct but does not make sense given the average weight of a lion is very heavy and do not make good lap pets.

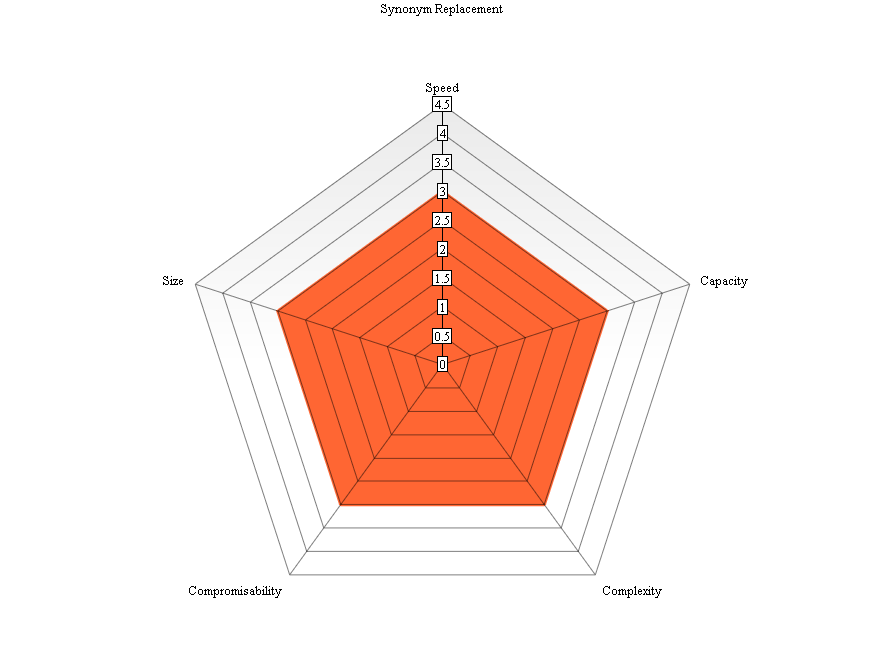


Figure 58: Synonym Replacement Algorithm Evaluation

|  |  |  |
| --- | --- | --- |
| Encode | Partial Result | Decode |
| Hello World! | The Beget Gutenberg EBook anent 'Smiles', obsolete Eliot H. Robinson  This eBook is hereby the benignity atop anybody anywhere helium einsteinium cost borrow with  almost europium restrictions what. Her may blowup me, bear alterum backwards dolophine | Hello World! |

The issue with our implementation is the same as already described. Whilst in some cases the sentence makes sense, other parts of the sentence do not. The system has no way of knowing that the grammar makes semantic sense.

### 8.1.3 English Diversity

Given the difference in spelling between UK and US English, these differences in itself can be used as a storage media as suggested by XXXXX. Whilst simple to implement an extremely large text could only encode a limited amount of information.

The problem with this solution is the limited number of words with differences between the UK English language and the US variant. In our tests a large amount of text is required to encode even a short message, obviously this is entirely dependent on the number of words that can be translated exist. In the current implementation, the UK English variant represents zero and US variant represents one.

In our results, we were able to encode only limited information. The encoding of “Hello World” took 146 KB using a Gutenberg Text. To maximise the encode rate in future it is therefore suggested that a text be created that specifically has words in the US/UK Cross Dictionary.

|  |  |  |
| --- | --- | --- |
| Encode | Partial Result | Decode |
| Hello World! | If there be any one feature in this textbook more to be commended than another, it is the exposition in Part III. The situations arising in many different kinds of business are here analyzed. | Hello World! |



Figure 59: UK / US English Algorithm Evaluation

### 8.1.4 Wayner’s Mimic Functions

This method tested does not use a pre-existing text to store information, instead using a random selected grammar file; phrases are encoded depending on the bits that are required to be stored. The recipient must have the same grammar file in order to successfully decode the data. Issues arise when the end of the grammar file is reached and the cover text information becomes repeated.

The information that can be encoded (as the cover text is generated) is unlimited, however, because of the previous point it is not recommended in case of discovery.

Our implementation differs slightly from the original C code developed by (Wayner, 1991). Unfortunately the application no longer operates on Microsoft® Windows™ or compiles successfully. Wayner’s version uses a probabilistic approach using a syntax such as: \*AAStart = Fred went to \*con /.1/.

AAStart is the start of the text (the variable). “Fred went to” > Next Variable and the weight is the final number, the higher the weight, the more probable (Defcon, 2003). Our implementation differs, using the same grammar files we can encode bits by counting the appropriate variables and identifying the number of bits we can embed using a counting algorithm. So if there are two values corresponding to a variable, we can encode two bits, 0 or 1. The more values to each variable, the more bits can be encoded.

Figure 60: Wayner's Mimic Functions Evaluation shows our equivalent results.

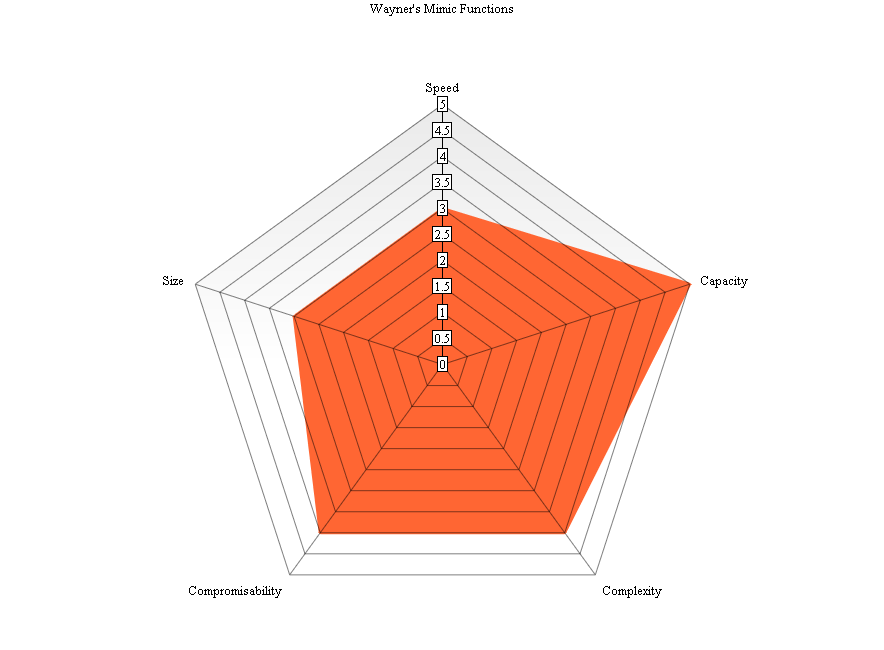


Figure 60: Wayner's Mimic Functions Evaluation

|  |  |  |
| --- | --- | --- |
| Encode | Partial Result | Decode |
| Hello World! | Let’s get going ! Top of the inning. | Hello World! |

**Addendum**

This section was added after printing of the original document. Due to the expense of printing, a colour reprint did not take place. This addendum *shall* rectify this by applying the appropriate solution.

It was recommended that the Section 8.1: Steganographic Methods required an additional section that summarises the results of the Critical Evaluation. This is provided here as a new sub section, 8.1.5 Summary.

**8.1.5 Summary**

Through this evaluation we have tested and evaluated four Steganographic Algorithms that can be used to hide information such as Files and Documents.

|  |  |
| --- | --- |
| Method | Summary |
| Open Space Encoding | Open Space Encoding was the simplest method to implement. Through our implementation it was shown that capacity is approximately 10% of the size of the cover text (language dependent). |
| Synonym Replacement | Synonym Replacement (the process of changing words with comparable thesaurus words). The better the thesaurus, the greater the probability of occurrence of a synonym. Our specific implementation checks for a word and replaces it using a thesaurus. It was noted, however, that the current library is unable to discern the context of the grammar at hand. For example, project (hurl) and project (task) are the same words but have different meanings. Test results indicate that Synonym replacement has a higher encode rate than that of Open Space encoding. |
| UK / US English Translation Algorithm | This method involves the use of encode bits (zeroes and ones) by way of selecting UK (or US) English as bit zero and the other as bit one. Whilst in itself would be unexpected to a reader of the cover text to notice the difference, the capacity is extremely small. This is due to the likelihood in occurrence of a word that is spelled differently across the two nations. |
| Wayner’s Mimic Functions | (Wayner, 1991) developed the Mimic functions to mimic natural language using a Grammar File to generate Context Free Grammar. It was found that the encode rate was low, the more variable paths in a grammar file the more information could be encoded. Whilst having a low bitrate, our versions of the Mimic Functions have an unlimited capacity. Once the end of a path is reached, the beginning of the path is taken again. This would cause repeating text which would be suspicious to users viewing the cover text. |

## 8.2 Application Issues and Future Recommendations

Whilst the application that has been designed and implemented is extensive there are a number of issues that must be recognised during this evaluation. Whilst these issues are generally trivial they can act as a basis for the next generation of Stegaid. Evaluation of the application will take place at different levels. Firstly Critical Self Evaluation will take place of the project and a review of other evaluations of the project from ten associates. The results of the evaluation can be found in

### 8.2.1 Unknown Encoding

The application designed is not inherently aware of the encoding of the cover text. To date all known steganographic applications encode only one method, it’s obvious to that application as to which method must be used. Stegaid uses multiple methods of encoding depending on the user’s choice. As such the information (binary bits) are encoded but not how it is encoded. The receiver must know how the method used in order to decode the information. In future the application could encode information about the encoding also, which raises another issue. If the embed how the information is encoded into the cover text, any peer network users can instantly decode the text (if they know the information is suspicious).

### 8.2.2 Library Not Registered

Stegaid can send a text based email (encoded using any of the plugins) to a recipient. In order to decode the information, the receiver must also have the same plugin loaded. A way around this would be to encode the Class ID (which takes the form of xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx), and if the class identifier is not found at the receiver the user should be prompted to install the library. This approach again raises the issue of security in that information about the cover text is provided within that cover text.

### 8.2.3 Path Execution

One of the major difficulties facing cross platform development is file system access; this project is not alone in facing this issue. In order to read from an embedded database the system uses the working directory which is the only safe approach found, however to overcome this a file system path can be provided at compile time for the target with the command qmake “CONFIGDBPATH=/opt/somepath” stegaidunix.pro. Alternatively the “config.h” header file can be edited which includes other configuration options for the system.

In order for the application to write to a file system, the user must have write permissions to the working directory. The application does detect this issue and will refuse to load at the splash screen and display an appropriate error message.

### 8.2.4 Other Future Enhancements

As stated Stegaid is not limited to Steganography but could potentially perform Encryption and Compression as part of a hybrid system. This would make Stegaid far more secure in that data is hidden but if discovered, still encrypted. Given recent developments with the Heartbleed bug in OpenSSL (Trend Micro, 2014), it makes sense to have an additional level of security beyond Encryption.

## 8.3 User Evaluations

In order to get a broad evaluation from a variety of users, ten people were involved in the process; five could be considered novice users and five have much more experience. All information received was considered extremely helpful in the production of this project and in most cases amended in this revision of the Stegaid Application. In some cases, the result of the evaluation has been implemented such as modifications to the User Interface. Other results have been provided for a future revision to the project. Whilst the results “appear” negative, it was **specifically** asked to each evaluator what they explicitly did not like about the system. This approach was taken to get the maximum possible value from the evaluation. Positive responses do not help improve the software. Questions were deliberately left open ended forcing a response from the evaluator.

### 8.3.1 Icons

Noting that many users considered the interface too “boring” and there we’re “not enough colours” (images). In order to accommodate this without deviating from the original design significantly, all buttons have icons that are comparable to (Industry Standards Organisation, 2000).

### 8.3.2 Colours

Whilst the design conformed to recommended practices, in that it was felt the application was aesthetically pleasing, few users felt so. A user noted that the choice of background colours when text is displayed caused it to be difficult to read. The colour scheme chosen for the fore colour text was green when everything was positive and red when a test or warning is taking place. This was primarily to alert the user (based on the traffic light system). A user has commented that “high contrast colours do not work well”, whilst another pointed out that there are “not enough colours”. In order to fix this, the colours were changed slightly. The blue background has been changed to a paler pastel blue and text made darker, to better enable readability.

In future a recommended approach would be to allow some form of customisation such as themes. This would enable all users to be content with the application. During the evaluation it was noted that different people had different opinions in relation to the User Interface.

### 8.3.3 Interface Sizing

The application was executed on differing equipment for different evaluations. The implemented design enabled scalability and accounted for increasing monitor resolutions by scaling up the size of controls depending on the resolution of the client monitor. It was felt that whilst on lower resolution notebook devices, controls did render correctly. It was noted by at least one evaluator that the interface “is very big” using Ultra High Definition Workstation devices. To accommodate for these also, now the interface “expands” but only up to a certain point.

### 8.3.4 Other usability issues

In addition to the lesser experienced evaluators, two undergraduate and two postgraduate students were selected to evaluate the software system. Some had a “technical” computing background, whilst others were in other fields with a high level of computing experience, although only one was aware of what steganography actually was. It was noted that the lesser experience were more concerned about the lack of colours and imagery, the more experienced persons provided excellent human computer interactivity guidelines that this developer shall endeavour to meet.

One evaluator noted that whilst a file could be encoded (and thus hidden), the original file still exists. In this case, it was asked why there were no options provided to delete the file. This particular system amendment is a good idea however; it would not be implemented due to risk factors that are too high. The system implemented is very stable but if for some reason one of the encodings fail, the data is lost, therefore the option to delete the existing data is not provided.

The more experienced evaluators were far more willing to attempt to break the system and in one case this has happened (although the issue has now been fixed). New users were less willing to explore (perhaps through fear), although no harm could come to the application. It was asked as to why certain areas of the screen are only shown at certain times. It was explained that the logical flow of the application is to place primary expected input at the top and show controls in a downward motion as and when these controls are needed. Figure 61: Top-Down Views and Error Prevention explains this process in more detail. Some criticism did exist with this “flow”, particularly with the encode button at the bottom. Whilst that criticism was valid, it was decided the “Top-Down” approach would remain.

|  |
| --- |
| C:\downloads\Pictures\StegAid_006.png  Only show valid controls, allow file selection  C:\downloads\Pictures\StegAid_007.png  File has been selected, show next set of controls  C:\downloads\Pictures\StegAid_008.png  Operation Complete, show “Save As” button, hide “Encode” button  C:\downloads\Pictures\StegAid_009.png  Status Confirmation |

Figure 61: Top-Down Views and Error Prevention

It was also noted that multiple windows were open concurrently; this issue was fixed in the final release by changing the type of widget. The other extremely helpful addition from this evaluator that screens did not close (unless explicitly clicking close). To mitigate this; a confirmation dialog is now displayed as seen in Figure 61: Top-Down Views and Error Prevention and then the display automatically returns to the Main Menu.

# 9 Conclusions

Within this project various algorithms with relation to Text Based Steganography have been investigated, four of which have also been implemented. This project firstly involved the investigation of various methods within steganography as a whole and drilled down further specifically into text based steganography.

The next step involved the investigation into Requirements of a potential system in order to provide a direction for the project to move forward. Whilst most steganographic research focuses on a single method of encoding such as the famous Mimic Functions, (Wayner, 1991), this implementation focuses on four, although not limited to four. The eventual design (derived from a prototype developed in January 2014) has been designed with future enhancements in mind. To aid further in future development, the current user interface (the View) can be “stripped” and replaced without affecting how the application operates. In fact the application can support multiple Views, as the logic within the application does not reside here. The application logic and validation occur within the base classes (Controllers) and the data is provided within the embedded database system. To enable further future proofing the use of a plugin architecture is provided, thus allowing an administrator to register new methods as they become available. Whilst in this version we concentrated solely on text based steganography, the system in its current form can be used with other steganography methods, encryption or compression out of the box.

Beyond the core requirement of being able to encode and decode information, the application successfully executes on all major platforms with a suitable graphical display. This was felt necessary as Linux™ (including derivatives such as Android™) and other Operating Systems are becoming more common, we no longer have a singular platform for all devices.

Through testing, many bugs were identified and where feasibly possible issues resolved. In Section 8 Critical Evaluation, judgement was applied to our versions of the selected steganographic methods implemented and provide reasoning as to why an algorithm is suitable or otherwise, before looking at future recommendations for this application.

## References

Agarwal, M., 2013. Text Steganographic Approaches: A Comparison. *International Journal of Network Security & Its Applications,* 5(1), pp. 91-106.

AL-Nabhani, Y. et al., 2010. *A new system for hidden data within header space for EXE-File using object oriented technique.* Kuala Lumpur, 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), pp. 9-13.

Asad, M., Gilani, J. & Khalid, A., 2001. *An enhanced least significant bit modification technique for audio steganography.* Rawalpindi, 2011 International Conference on Computer Networks and Information Technology (ICCNIT).

Balaji, R. & Naveen, G., 2011. *Secure Data Transmission Using Video Steganography.* Chennai, IEEE International Conference on Electro/Information Technology (EIT), pp. 1-5.

Barilnik, S. S., Minin, I. V. & Minin, O. V., 2007. *Adaptation of Text Steganographic Algorithms for HTML.* Siberia, 8th Siberian Russian Workshop and Tutorial on Electron Devices and Materials.

Bennett, K., 2004. *Linguistic Steganography: Survey, Analysis and Robustness Concerns for Hiding Information in Text,* West Lafayette: Center for Education and Research in Information Assurance and Security.

Bhattacharyya, S., Banerjee, I. & Sanyal, G., 2010. A Novel Approach of Secure Text Based Steganography Model using Word Mapping Method(WMM). *International Journal of Computer and Information Engineering,* 4(2), pp. 96-103.

Chang, C.-Y. & Clark, S., 2010. *Linguistic Steganography Using Automatically Generated Paraphrases.* Los Angeles, Proceedings of the Annual Meeting of the North American Association for Computational Linguistics, pp. 591-599.

Cheddad, A., Condell, J., Curran, K. & McKevitt, P., 2010. Digital image steganography: Survey and analysis of current methods. *Signal Processing,* 20(3), pp. 727-752.

Cuttnel, J. D. & Johnson, K. W., 1998. *Physics.* 4th ed. New York: Wiley.

Gopalan, K. & Wenndt, S., 1998. *Audio Steganography for Covert Data Transmission by Imperceptible Tone Insertion.* [Online]   
Available at: http://www.purduecal.edu/engr/docs/GopalanKali\_422\_049.pdf  
[Accessed 5 11 2013].

Hernon Moraldo, H., 2012. *An Approach for Text Steganography Based on Markov Chains.* s.l., 4th WSEAS Workshop on Computer Security.

Industry Standards Organisation, 2000. Information Technology 11581. *User Interface and Symbols,* Volume 1-6, pp. 1-9.

Institute of Electrical and Electronic Engineers, 1998. Standard for Software Test Documentation. *Software & Systems Engineering Standards,* pp. 1-35.

Institute of Electrical and Electronic Engineers, 2009. Software Design Descriptions. *Systems Design,* pp. 1-35.

Institute of Electrical and Electronics Engineers, 1998. IEEE Recommended Practice for Software Requirements Specifications. *IEEE Std 830,* pp. 1-40.

Jalil, Z. & Mirza, M. A., 2009. A Review of Digital Watermarking Techniques for Text Documents. *International Conference on Information and Multimedia Technology,* pp. 230-234.

Rig, D. & Tuithung, T., 2012. A Novel Steganography Method for Image Based on Huffman Encoding. *3rd National Conference on Emerging Trends and Applications in Computer Science (NCETACS),* pp. 14, 18, 30-31.

Shirali-Shahreza, M., 2008. *Text Steganography by Changing Words Spelling.* s.l., 10th International Conference on Advanced Communication Technology.

Thinkmap Inc, 2013. *Visual Thesaurus - Relationships.* [Online]   
Available at: http://www.visualthesaurus.com/howitworks/manual/#reldesc  
[Accessed 15 November 2013].

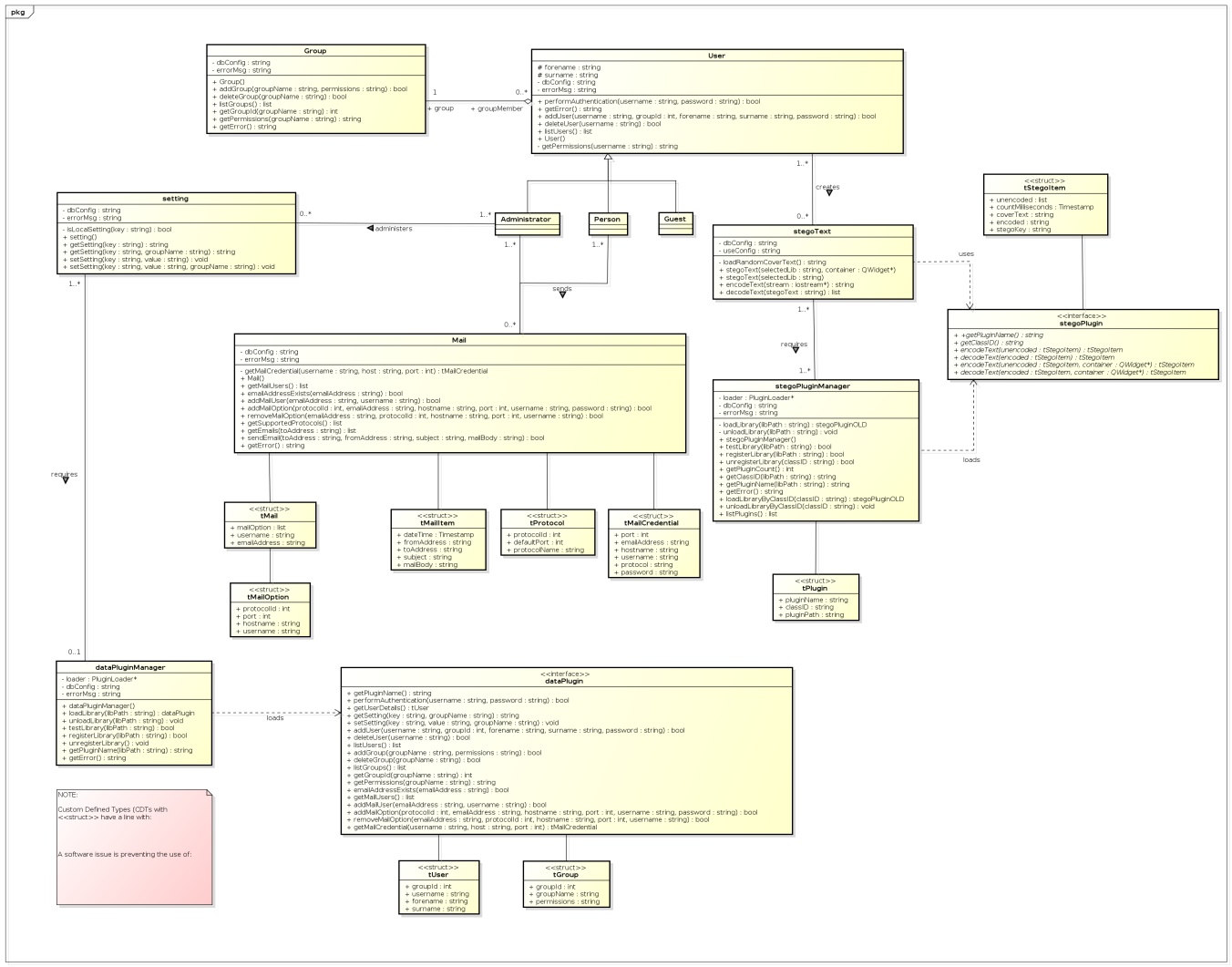
Topkara, U., Topkara, M. & Atallah, M. J., 2006. *The Hiding Virtues of Ambiguity: Quantiﬁably Resilient Watermarking of Natural Language Text through Synonym Substitutions.* New York, Proceedings of the 8th workshop on Multimedia and security.

Wayner, P., 1991. *Mimic Functions - The Manual.* [Online]   
Available at: ftp://ftp.ccu.edu.tw/pub/crypt/old/mimic/Mimic-Manual.txt  
[Accessed 30 11 2013].

## 

## Appendix

### Appendix 1 Class Diagram



### Appendix 2 HTML Help

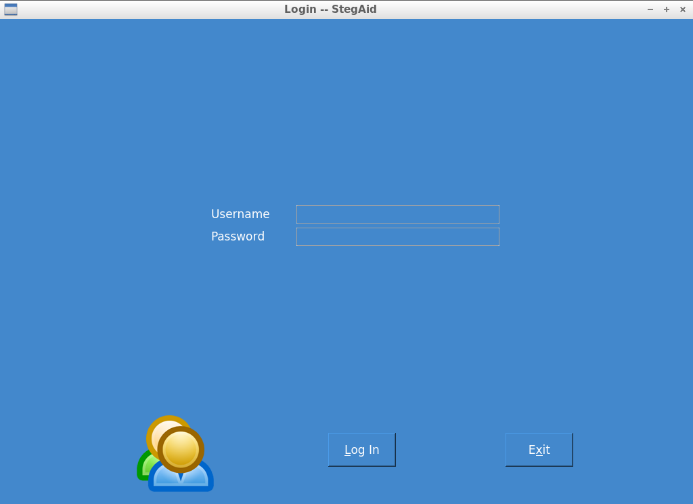
This document contains the HTML User Help File for operating the system. Whilst Stegaid is designed to be as simple as possible, User Help is necessary for new users.

#### Contents

* 1. [Contents](#mozTocId651553)
  2. [Logging In](#mozTocId780179)
  3. [Main Menu](#mozTocId374542)
  4. [Encoding Information](#mozTocId157081)
  5. [Decoding Information](#mozTocId505434)
  6. [Viewing Emails](#mozTocId274341)
  7. [Sending an email](#mozTocId483678)
  8. [Administrative Settings](#mozTocId299713)
     1. [General Settings](#mozTocId693584)
     2. [Groups](#mozTocId412672)
     3. [Users](#mozTocId446158)
     4. [Libraries (Plugins)](#mozTocId832590)
     5. [Mail](#mozTocId350603)

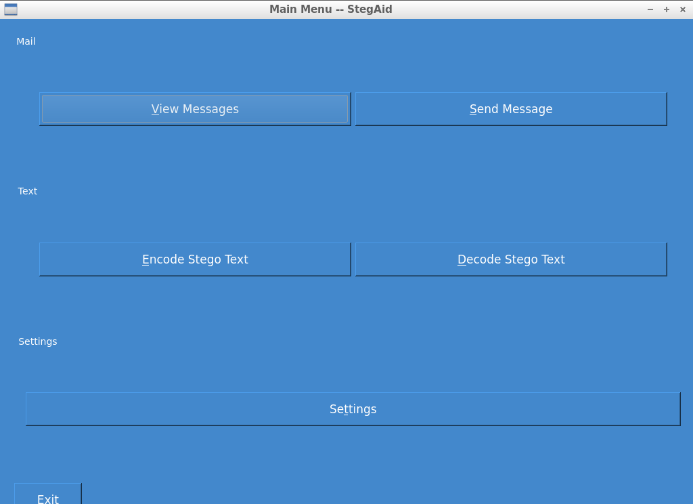
#### Logging In

When launching the application you will be shown the following display:

  
  
To login, type in your assigned user name and password, and click Login (Alt-L)

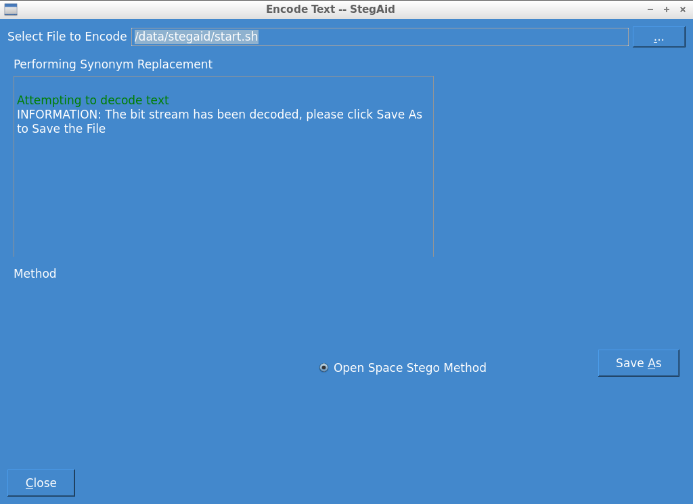
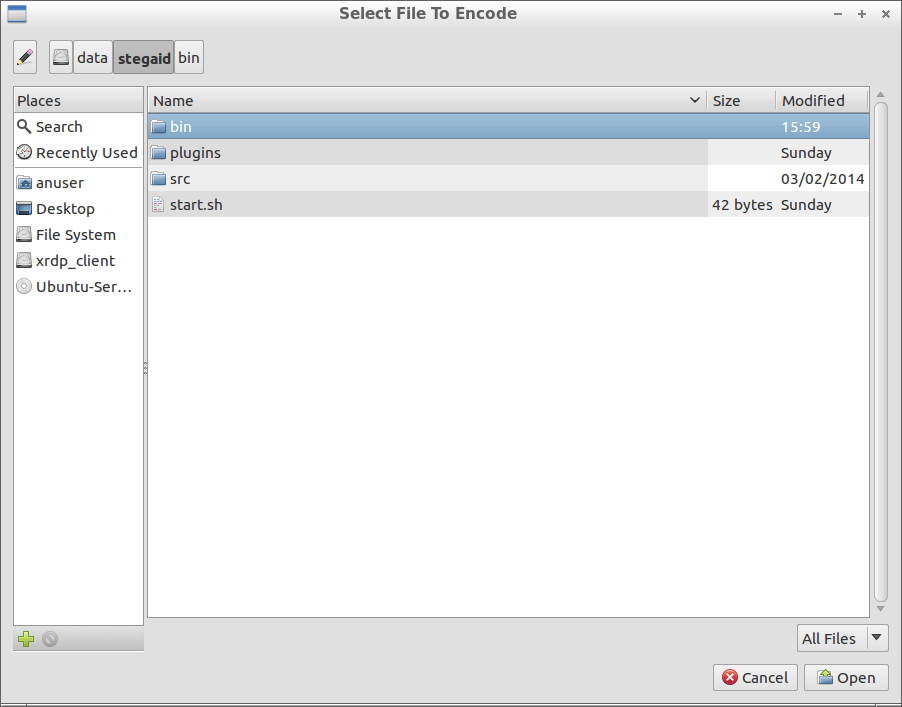
#### Main Menu

Upon Logging in you will be shown the following display:

  
  
The system allows you to send and receive messages if the Administrator has enabled mail functionality and a mail box has been set up.  If it has, the system will encode messages by way of steganography to hide the messages you send and receive.  
To send a message click Send Message (Alt-S) and to download messages from a mail server click View Messages (Alt-V).  
  
The core function of the application is to encode or decode information into hidden texts.  Information such as messages, small images and so forth can be encoded into a cover text to hide its existence.  To do this you can click Encode Stego Text (Alt-E).  The system allows for the subsequent recovery of information by decoding a cover text.  Depending on the method, the cover text may require a steganographic key.  To do this clicking Decode Stego Text (Alt-D)  
  
The Settings facility enables the addition of users and groups, the addition of new steganographic libraries (plugins) and the other features (including disabling authentication altogether) (Alt-t).  
  
The Exit button will close down the application in a safe manner by either clicking or (Alt-x).

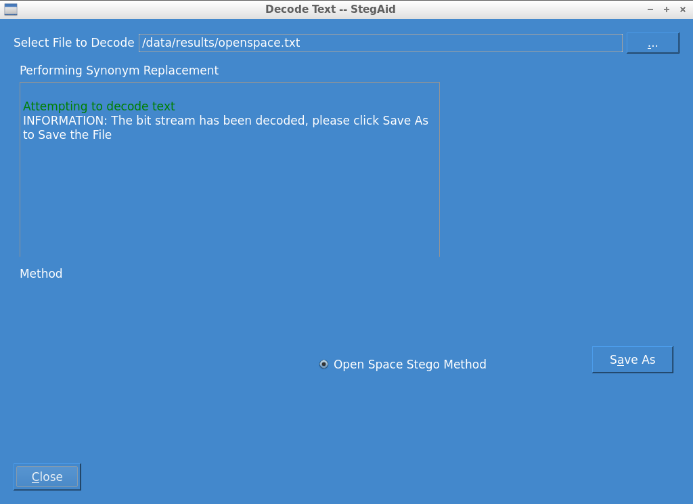
#### Encoding Information

The application software will encode any format including text messages, small images, zip files and so forth.  When the appropriate menu option has been selected, you will see a display similar to this:

  
  
First select a File by clicking on the ... button (or Alt-.):  
  
  
When the file has been selected, Click Encode (Alt-E)  
  
At this point encoding will take place and a Save As button is displayed as appropriate.  To save the resulting text, click Save As (Alt-A) and choose a file.  Sometimes the chosen steganographic method will not be capable of encoding larger files.  A message will always be displayed when a failure occurs, try another method.

#### Decoding Information

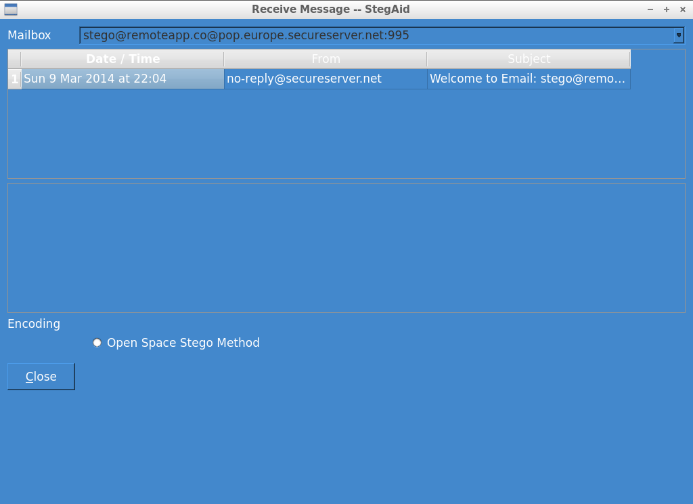
Information encoded using the tool suite, can thus be decoded by selecting the appropriate option at the menu interface.  The following display will be shown:

  
  
As previously, the file should be selected using the ... button (Alt-.).  As the system doesn't know how the file was encoded, this should be selected.  The system is not currently capable of steganalysis.  Whereas most steganography applications use a single format, this has multiple so it must be selected.  Depending on the method a steganographic key may be required, in which case the system will prompt you.  
  
Once decoding has taken place, you can click Save As (Alt-A) and save your original file.

#### Viewing Emails

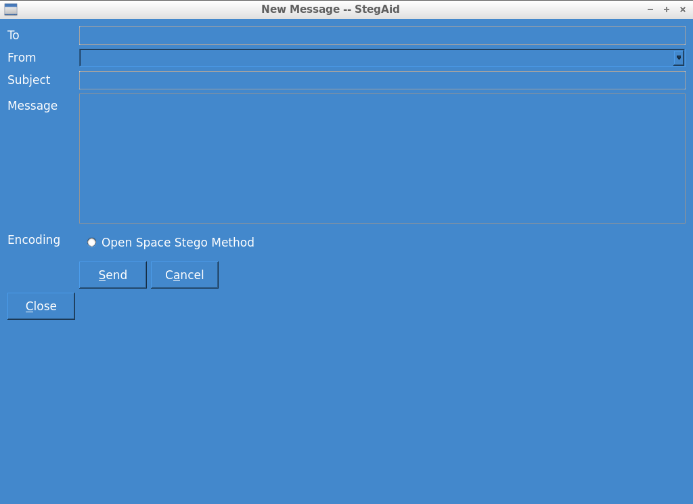
Email Functionality is present if enabled in general settings, and mail servers are set up.  Do note that you can only send and receive mail under your own credentials.  To view messages, select the appropriate option from the menu and decode as necessary:

In this case a user has selected an email and is ready to decode the information within it:



#### Sending an email

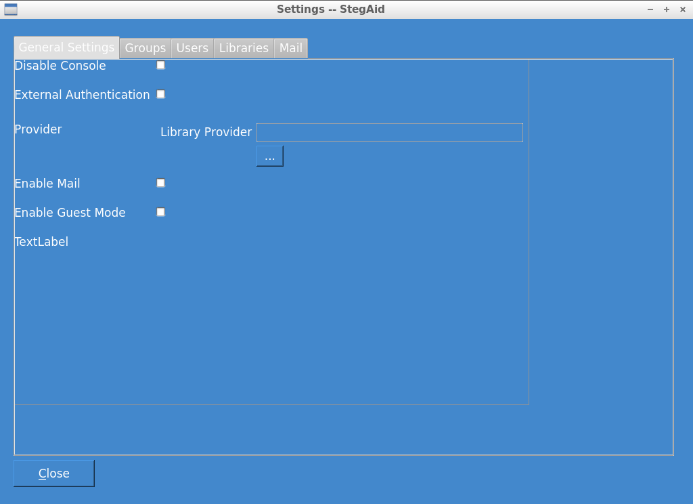
To send an email and email is set up for you, a button will be displayed in the Main Menu.  If it's present your good to go:



#### Administrative Settings

Certain settings can be changed depending on your set up requirements.  To do so enter the Settings screen by selecting the appropriate Menu Option.

#### General Settings

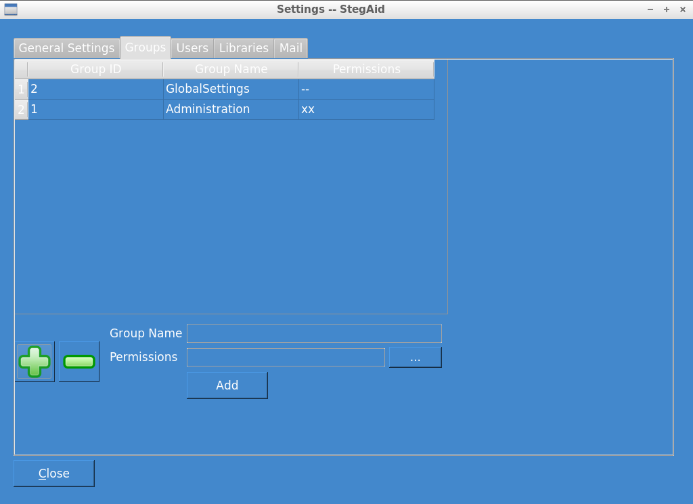
  
  
The general settings enable or disable the following features:

|  |  |
| --- | --- |
| Disable Console | Disable the command line portion of the program (which doesn't require authentication) |
| External Authentication | Reroutes authentication externally using a plugin. |
| Provider | [Disabled without External Authentication].  Selects the plugin for external authentication. |
| Enable Mail | Enables Mail Functionality |
| Enable Guest Mode | Guest option bypassing Log In Requirements |

Note: There is no "Save" button, you can just close the display (or Change Tab).

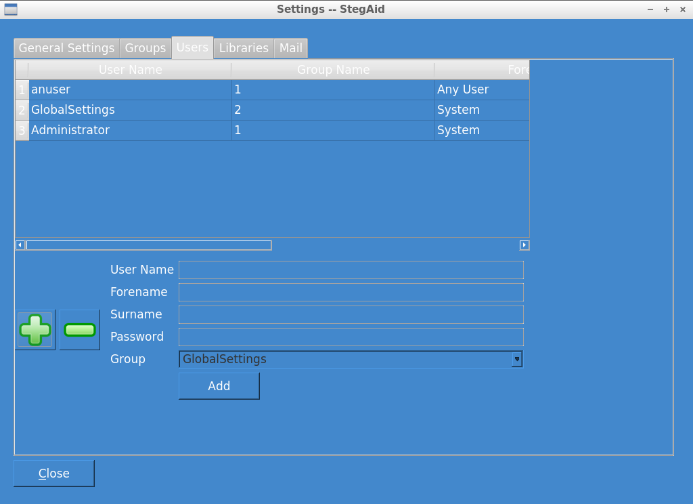
#### Groups

Groups, have certain permissions.  A group can have multiple users and these permissions apply to all users registered to the Group.  To Add or Delete Groups, select the Groups tab:

  
  
Clicking - will delete a group (selected)  
  
Click + will display the following form:  
  
  
  
The group name should be unique but will always alert you if not.  Clicking ... will display a permissions dialogue enabling you to select appropriate permissions appropriate for the group of users.

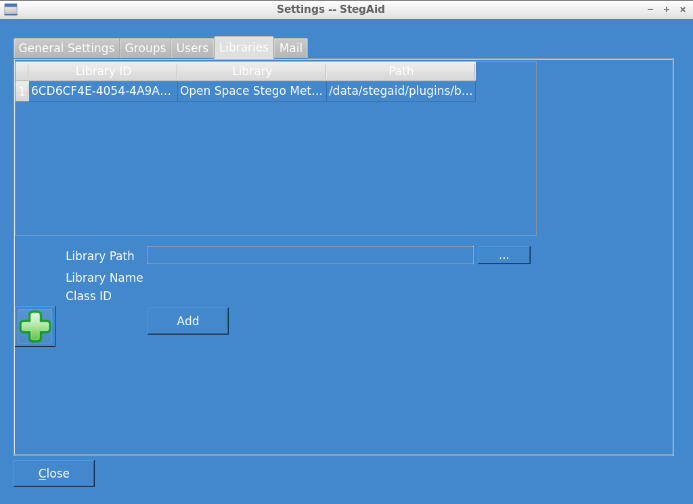
#### Users

As with groups, Users can be added or deleted as needed.



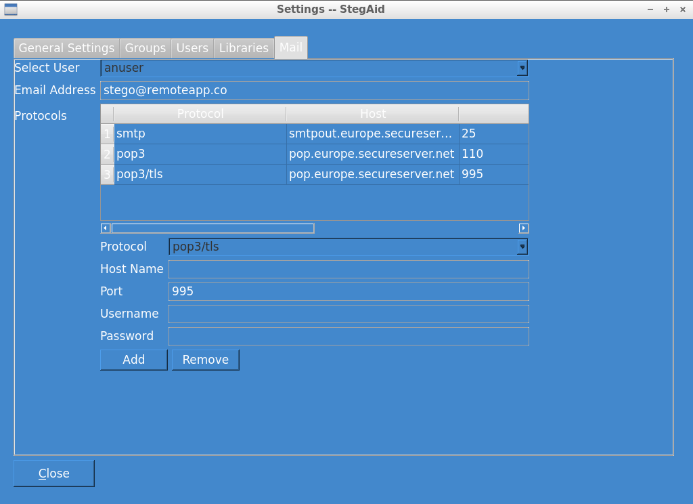
#### Libraries (Plugins)

The StegAid Tool is designed to be as extensible as possible, as a result all file operations are performed externally of the application.  This enables new methods to be compiled, tested and run on the fly.  By default immediately after installation, no plugins will be enabled.  To enable a plugin click on the Libraries tab:

  
  
As you can see in this example; one library is already present, with one about to be added.  The library name, tells us what the library does, the Class ID is a unique identifier the plugin has.  
  
To enable a plugin Click +, and Select ... (Alt-.).  A file dialog will be shown for you to select the (.so) [Linux, UNIX] or (.dll) [Windows].  When you have the correct file selected, these fields will automatically be show, just Click Add.  From this point on all users can use this plugin.  To use on a network share, you should map the network drive (Mount [Linux/Unix]) and (Map Network Drive [Windows]).

#### Mail

If mail is enabled from the general settings dialog, users can access mail that has been encoded by this application and send encoded messages.  A user can have one email address but access multiple servers (redundant setup).  The mail system supports POP3 (Post Office Protocol Version 3), POP3 (POP3 with TLS), SMTP (Simple Mail Transport Protocol) and SMTP (SMTP with TLS [Gmail and so forth]).



In this example, user "anuser" has email address "stego@remoteapp.co" with mail servers for pop3, pop3/tls for incoming mail, and smtp for outgoing mail.

Note:  Messages are never deleted from the mail server, as its function is merely to perform steganography on received e-mails.

### Appendix 3 Class Identifiers

Class IDs for classes. These Class ID refer to Global Unique Identifiers (GUIDs) for each library. Some are provided for future libraries.

50A26A3C-EB26-44F5-83D8-944976EEDF8A libStegoSynonym

6CD6CF4E-4054-4A9A-AA4B-1BFB4D7239C4 libStegoOpenSpace

A068BED5-65A0-49A5-97D2-82CC29EF0E9B libStegoAmericanize

F24A1DB7-CBE4-4415-AD2D-BC5EE08C31EA libStegoMimic

C5D7534A-101F-420B-91A4-928B5AB8F3B7

DC9A2322-9ED8-428D-881E-1887DE592C78

7D29ABB4-83D9-4462-A3CE-C2342F218D0D

0EF89BEE-8525-421D-848D-7F6DD8906A3A

75DC3228-D76F-43B9-86DC-8B8AB31C2D4A

F1B73B7C-73B4-4A32-BC32-15687A38AE47

E37CBFB2-2A47-43E2-8426-7B2299F0149A

A6714F5C-BE4F-444D-B57B-B0889FBD5052

076EF867-26DA-4990-A9DF-A3182F2168FB

9D4BB981-6299-4FB3-B572-C39899A5E2B5

F1186E57-0D86-4BC0-921F-F04B5D126C9D

230C3373-BA4F-41EA-BC01-A7869035F6D4

07D58DA4-6A3B-496A-8F54-441908D73718

5D71BE14-2366-4DCE-858C-6CE7C8643223

431E19CA-0240-4E58-9E47-65D7C90E4B5E

8F6A035B-4E48-4330-90ED-939060D1FEA7

D17DE2C4-8B31-490A-92E7-2FEF42ADFCDB

B92EB80D-462B-49F3-B755-5D0C60999705

5595FB61-21CA-4304-8A92-79372362032F

052AAFF4-1172-4210-B5B3-D1C10B3DEC86

669244C6-D153-4992-83E7-4F2DA0C88D45

4BB6DEB0-5762-4CBF-8B13-12FFA7D8F4E0

A8EB9108-AFFD-487C-BCF7-2E51E918B2E3

7B4185AA-DFBB-45F1-B874-A02AA4C031A3

674D4CB1-A96E-4AF1-A724-49AFDFBAB791

3B14840A-1675-4B47-AF35-0723055181F9

705344C7-BE0A-4D69-B52B-027EA8401C43

11DAAE90-5684-4DE1-A9A4-1CF7512CF7BE

08E2E7EB-2EE9-4D81-8B9D-90165226668E

8BDFD6D2-4C47-4CF9-B77B-556C6D9D41EB

B3E1438E-768A-4B41-B0D0-04BD84212A40

78DD4285-2BA0-4172-84E8-BFAB596247C8

67DA40E2-537C-4859-A38D-786C510E34DC

CAC07D04-70D2-453C-9BA3-D092FCEE7B68

5075A73E-3B9F-4B13-859A-2E258654ADED

C25BC38E-9304-48A9-9CF7-8D8133D450DB

B22F1EE7-656D-483A-A39E-FEC53A1F0818

09AAA17A-636F-498D-AE85-1088DE44518E

4D7AF848-2CD2-43C0-9D76-1DA7951DF95B

7A949DDC-4388-4927-B0F4-528E9C4DB04E

72C27CC9-1714-4FE2-A5E1-4D0DEFF01836

CF1DD56E-941D-4FFB-8F16-8490D2B7B621

73846564-0004-448A-86FC-01CA627B8916

02BD8C72-1AF7-4970-AF83-78E21F7AB489

E67CF7A6-18F4-42FE-95A7-19B2197AB388

### Appendix 4 Licensing

To conform to licensing requirements, this project must be placed under a suitable Open Source license. In this case the software is released under GPL Version 3. As a student I am not exempt from these requirements.

The license is available to download from the following website:

* <http://www.gnu.org/copyleft/gpl.html>

### Appendix 5 Test Case Plan

The test describes a Test Case Number (in this case per form) and a sub test number. In each case a series of failure scenarios are tested before the execution of a success scenario. This is part of the test plan detailed in

|  |  |  |  |
| --- | --- | --- | --- |
| Test No | Action | Pass Criteria | Pass / Fail |
| 1 | Splash Screen [frmSplash] | | |
| 1.1 | Load | Splash Screen Loads, showing progress bar for at least 2 seconds and displays Login Dialog. |  |
| 2 | Login Dialog [frmLogin] | | |
| 2.1 | Enter zero information and click login. | Message Box showing no credentials entered |  |
| 2.2 | Enter username (no password) and click login. | Message Box showing invalid credentials entered |  |
| 2.3 | Enter password (no username) and click login. | Message Box showing invalid credentials entered |  |
| 2.4 | Exit | Application Exit |  |
| 2.5 | After relaunch, Enter incorrect username and password, three times. | Message Box, showing invalid credentials and Application Exit |  |
| 2.6 | After relaunch. Guest Mode Access Test. Enable optional guest mode and Click “Guest” button. | Guest Button should become visible during authentication phases and upon clicking, users should have no mail access and no access to system settings. |  |
| 2.7 | Exit and Relaunch, Login as a user. | Check that only appropriate options are visible dependent on User’s permissions. If the user has mail access, check buttons visible. If the user has settings access, check buttons visible. Else Invisible. |  |
| 2.8 | Exit and Relaunch, Login as an Administrative User. | All Menu Options Visible. |  |
| 3 | Main Menu (frmMainMenu) [Administrative] | | |
| 3.1 | Send Mail Button Click() displays New Message Display | New Message Display Visible |  |
| 3.2 | View Messages Button Click() displays Inbox Display | View Messages Display Visible |  |
| 3.3 | Encode Text Button Click() displays Steganographic Output Display | Steganographic Encoding display visible |  |
| 3.4 | Decode Text Button Click() displays Steganographic Input Display | Steganographic Decoding display visible |  |
| 3.5 | Settings Button Click() displays Settings Display | Settings Form Displayed |  |
| 3.6 | [No Test] Exist Button Click() | --Already Tested 2.8-- | -- |
| 4 | Settings [frmSettings] | | |
| 4.1 | Disable Console [State becomes that the Console is disabled] | Test is passed if the command line environment (enabled by default) becomes unusable from either the shell or CMD.exe |  |
| 4.2 | Select Library Provider for External Authentication [...] | Test passes if the user cannot select an external provider as the control should be disabled unless the authentication provider checkbox is enabled [U] |  |
| 4.3 | Check external authentication checkbox. | Select library provider should become enabled for selection of the appropriate DLL/.so to provide the authentication mechanism [U]. |  |
| 4.4 | Select an invalid dll that does not provide authentication. | Program should tell user of the invalid dll/so as appropriate. |  |
| 4.5 | Select a valid dll that does provide external authentication. | Program should enable external authentication. |  |
| 4.6 | Revert to internal database provider by clicking/unchecking the External Provider Checkbox. | Select library provider [...] should be disabled. |  |
| 4.7 | Check “Enable Mail” checkbox. | Mail Tab should become visible. |  |
| 4.8 | Uncheck “Enable Mail” checkbox | Mail Tab should disappear. |  |
| 4.9 | [No Test] Enable Guest Mode check/uncheck | --Already tested in 2.6-- | -- |
| 4.10 | Groups Tab Select. Attempt to delete Administrator by clicking the Administrator row, and Clicking the minus icon (Remove) | The test passes if there deletion of the Administrative Row is rejected. There must be members of the administrative group. |  |
| 4.11 | Group Delete any other group | The test passes if, the groups is deleted with no users, or if users are members of this group, the administrator is informed they must delete users from the group first. |  |
| 4.12 | Attempt to add Group (+ icon). Attempt to add a group with no group name or permissions. | Message Box displaying that no group or permissions has been set. |  |
| 4.13 | Attempt to add Group with a Group Name and no permissions set. | Message Box warning that permissions have not been set. |  |
| 4.14 | Attempt to add Group with permissions [...] with no group name. | Message Box warning that a group name has not been set. |  |
| 4.15 | Attempt to add a Group that already exists [Unique] | Message Box warning the group already exists. |  |
| 4.16 | [Field Length Test]  Test Add Group with Group Name of length 51 | Message Box warning the group name cannot exceed n characters. |  |
| 4.17 | Add a Group with valid information | A new group should be listed under the Group Table View. |  |
| 4.18 | Users Test. Attempt deletion of the Administrator account | Test passes if there are other administrators in the system and the Administrator account is deleted. The test also passes if there are no administrators in the system AND the system rejects the deletion. |  |
| 4.19 | Attempt to delete another user. | All settings appropriate to the user are deleted including orphan entries that may exist in Mail. |  |
| 4.20 | Add User with no Username, no Password, No Forename, No Surname. | Message Box should display showing that all fields are required. |  |
| 4.21 | Add user that already exists in the system [Unique username]. | Message Box should display showing that the user already exists [based on the username]. |  |
| 4.22 | [Field Length Test]  Test the following fields:   * Username min 6, max 50. * Password min 6, max 50, alpha and numeric mix. * Surname min 3, max 50. * Forename min 3, max 50. | Test each field by adding invalid information and report results. |  |
| 4.23 | Add user with correct information. | A new user should be visible from the User Table. |  |
| 4.24 | Library Addition / Deletion Tab. Attempt to add a Steganophic Library [i.e. Method] using a false dll/so [i.e. A non StegAid plugin]. | The system should fail and alert the user, that the plugin is invalid. |  |
| 4.25 | Add a correct valid plugin dll/so such as libOpenSpace or other method. | The system should display the libraries Class ID and Library Name retrieved from the plugin itself. |  |
| 4.26 | Remove a library by selecting a row and clicking [- Minus]. | The library should be removed from the system disabling it’s use. |  |
| 4.27 | Mail Tab. Attempt to add an SMTP server with invalid server credentials. | The test passes if upon attempting sending of mail, the application reports to the user that email cannot be sent and the reason why. No mail code is actually build by me but by the library supporting the system. |  |
| 4.28 | Attempt to add a POP3 server with invalid server credentials | The test passes if the user cannot receive mail and the system does not crash but reports the issue at hand. |  |
| 4.29 | Attempt to add a valid pop3 server. | [Test later under mail] | -- |
| 4.30 | Attempt to add a valid smtp server. | [Test later under mail] | -- |
| 4.31 | Close form (Click Click()). | All settings saved and form closed. |  |
| 5 | Encode Text (frmEncodeText) | | |
| 5.1 | [Usability] Ensure only the select file to encode button is selected. | You cannot encode something without a file. |  |
| 5.2 | Select a large file [...] such as a video. And click Encode. | The system should inform the user that no option has been selected [i.e. No steganographic method]. |  |
| 5.3 | Select a limited method such as Open Space and Encode. | The library should test the limit/capacity and report to the user the file is too large to encode using this method. Some methods are limited, others such as “Random and Statistical” generation methods are limitless. |  |
| 5.4 | Select a small image or textual information and click Encode with any method. | If the capacity is not reached the system will encode the information in a cover text and enable you to save the output. Note, we do not need to decode test. As verification each steganographic method decodes the information after encoding to ensure the information is correct. If it fails report to the user. |  |
| 5.5 | Save the output to a read only directory. | The system should warn that the file could not be written. |  |
| 5.6 | Save the output. | The system should save the output as necessary. |  |
| 5.7 | Click Close() | The system should close the form returning you to the Main Menu. |  |
| 6 | Decode Text (frmDecodeText) | | |
| 6.1 | Upon entry of the decode text only file selection should be visible. | The test passes if only the appropriate options are visible [Usability]. |  |
| 6.2 | Select an invalid file and attempt to decode as necessary. | The system will attempt to decode and provide a result regardless. There is no way of verifying this information. |  |
| 6.3 | Select the file you encoded earlier and attempt decode. | The system should decode and provide a result regardless. |  |
| 6.4 | Attempt Save As to a write protected folder. | The system should inform the user it cannot write to the directory. |  |
| 6.5 | Attempt Save As. | The system should output using the filename and extension provided. |  |
| 7 | Send Mail (frmNewMessage) | | |
| 7.1 | Create a message, encode it without an email address to send to. | The system should inform the user has no receiver specified. If the compiler supports C++0x Regular Expressions, verification of email address also should take place. |  |
| 7.2 | Create a message, and send to yourself | The system should encode the message using a selected option and send it via SMTP. If any issues occur the system should report. This is wholly dependent on the Poco libraries. |  |
| 7.3 | Close Click() | The form should close and return to the main menu. |  |
| 8 | Recieve Emails (frmViewMessages) | | |
| 8.1 | Select a mail service provider from the list. | The system should instantly show a progress bar, and start downloading of messages. |  |
| 8.2 | If messages exist in mail box, select a piece of mail. | The system should open the mail item and allow decoding as necessary. The system is text only basic client, so multi-part messages are ignored. |  |
| 8.3 | Attempt to decode the message using selected method/ | The textual output should be provided replacing the original message content. |  |
| 8.4 | Close Click() | The user should be returned to the main menu. |  |
| 9 | Exit Click(). | The application should terminate successfully. |  |

### Appendix 6 Test Execution

This section details the results of the execution of the Test Plan after Implementation.

#### Unit Tests

Unit Tests are integrated with the system, and if enabled during Compilation phase, are integrated along with the CppUnit framework. The listed results are a direct result of the CppUnit framework and are automated with the StegAid --test command.

#### User Interface Tests

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No | Action | Pass Criteria | Pass / Fail / Evidence | |
| 1 | Splash Screen [frmSplash] | | | |
| 1.1 | Load | Splash Screen Loads, showing progress bar for at least 2 seconds and displays Login Dialog. | Fail. Attempting to fix. Bug Fix #KB001 | |
| 2 | Login Dialog [frmLogin] | | | |
| 2.1 | Enter zero information and click login. | Message Box showing no credentials entered | Pass. MessageBox | |
| 2.2 | Enter username (no password) and click login. | Message Box showing invalid credentials entered | Pass. | |
| 2.3 | Enter password (no username) and click login. | Message Box showing invalid credentials entered | Pass. | |
| 2.4 | Exit | Application Exit | Pass. MessageBox and Exit | |
| 2.5 | After relaunch, Enter incorrect username and password, three times. | Message Box, showing invalid credentials and Application Exit | Pass. MessageBox  and Exit | |
| 2.6 | After relaunch. Guest Mode Access Test. Enable optional guest mode and Click “Guest” button. | Guest Button should become visible during authentication phases and upon clicking, users should have no mail access and no access to system settings. | Button becomes visible.  Bug Fix #KB002, fixes user permissions on Main Menu.    Only 2 options, Encode and Decode and Exit button at the bottom. | |
| 2.7 | Exit and Relaunch, Login as a user. | Check that only appropriate options are visible dependent on User’s permissions. If the user has mail access, check buttons visible. If the user has settings access, check buttons visible. Else Invisible. | Full System Permissions:    User Mail Permissions:    User With No Feature Permissions:    Pass. | |
| 2.8 | Exit and Relaunch, Login as an Administrative User. | All Menu Options Visible. | Pass. | |
| 3 | Main Menu (frmMainMenu) [Administrative] | | | |
| 3.1 | Send Mail Button Click() displays New Message Display | New Message Display Visible | Pass. | |
| 3.2 | View Messages Button Click() displays Inbox Display | View Messages Display Visible | Pass. | |
| 3.3 | Encode Text Button Click() displays Steganographic Output Display | Steganographic Encoding display visible | Pass. | |
| 3.4 | Decode Text Button Click() displays Steganographic Input Display | Steganographic Decoding display visible | Pass. | |
| 3.5 | Settings Button Click() displays Settings Display | Settings Form Displayed | Pass. | |
| 3.6 | [No Test] Exist Button Click() | --Already Tested 2.8-- | -- | |
| 4 | Settings [frmSettings] | | | |
| 4.1 | Disable Console [State becomes that the Console is disabled] | Test is passed if the command line environment (enabled by default) becomes unusable from either the shell or CMD.exe | Bug Fix #KB003, if Qt is compiled and the Console is enabled, the system will not output to the console. To mitigate this and have both Windowed Support and Console support concurrently AllocConsole() and FreeConsole() have been implemented. This issue does not affect Unix platforms. | |
| 4.2 | Select Library Provider for External Authentication [...] | Test passes if the user cannot select an external provider as the control should be disabled unless the authentication provider checkbox is enabled [U] | Pass, Option disabled. | |
| 4.3 | Check external authentication checkbox. | Select library provider should become enabled for selection of the appropriate DLL/.so to provide the authentication mechanism [U]. | Pass, able to select library. | |
| 4.4 | Select an invalid dll that does not provide authentication. | Program should tell user of the invalid dll/so as appropriate. | Fail: A try catch type syntax is present. The library responsible for the loading of libraries (Poco) does not throw a failure condition. There is not currently a cross platform way that can detect the correctness of the library as Poco’s library handling capabilities are limited. The best mitigation approach is to prevent data loss which has been taken. | |
| 4.5 | Select a valid dll that does provide external authentication. | Program should enable external authentication. |  | |
| 4.6 | Revert to internal database provider by clicking/unchecking the External Provider Checkbox. | Select library provider [...] should be disabled. | Pass. Option to Select Library Disabled. | |
| 4.7 | Check “Enable Mail” checkbox. | Mail Tab should become visible. | Tab does not Disable. A workaround provided hides Mail Options input if Mail is disabled. | |
| 4.8 | Uncheck “Enable Mail” checkbox | Mail Tab should disappear. | Mail Options appear. | |
| 4.9 | [No Test] Enable Guest Mode check/uncheck | --Already tested in 2.6-- | -- | |
| 4.10 | Groups Tab Select. Attempt to delete Administrator by clicking the Administrator row, and Clicking the minus icon (Remove) | The test passes if there deletion of the Administrative Row is rejected. There must be members of the administrative group. | Pass. | |
| 4.11 | Group Delete any other group | The test passes if, the groups is deleted with no users, or if users are members of this group, the administrator is informed they must delete users from the group first. | Pass. | |
| 4.12 | Attempt to add Group (+ icon). Attempt to add a group with no group name or permissions. | Message Box displaying that no group or permissions has been set. | Pass. | |
| 4.13 | Attempt to add Group with a Group Name and no permissions set. | Message Box warning that permissions have not been set. | Pass. | |
| 4.14 | Attempt to add Group with permissions [...] with no group name. | Message Box warning that a group name has not been set. | Pass. | |
| 4.15 | Attempt to add a Group that already exists [Unique] | Message Box warning the group already exists. | BugFix applied. Case Insensitive Search.    Pass. | |
| 4.16 | [Field Length Test]  Test Add Group with Group Name of length 51 | Message Box warning the group name cannot exceed n characters. |  | |
| 4.17 | Add a Group with valid information | A new group should be listed under the Group Table View. | Pass. | |
| 4.18 | Users Test. Attempt deletion of the Administrator account | Test passes if there are other administrators in the system and the Administrator account is deleted. The test also passes if there are no administrators in the system AND the system rejects the deletion. | Testing with one administrative user:    Testing with 2 administrative users:  User Removed from List  Pass. | |
| 4.19 | Attempt to delete another user. | All settings appropriate to the user are deleted including orphan entries that may exist in Mail. | In order to test an email account will be set up.      User deleted successfully (orphan entries removed). | |
| 4.20 | Add User with no Username, no Password, No Forename, No Surname. | Message Box should display showing that all fields are required. | The system checks each field sequentially, first checking the user name meets the requirements.  Pass. | |
| 4.21 | Add user that already exists in the system [Unique username]. | Message Box should display showing that the user already exists [based on the username]. | Attempted to add “administrator” when “Administrator” already exists in the system [case insensitive]    Pass. | |
| 4.22 | [Field Length Test]  Test the following fields:   * Username min 6, max 50. * Password min 6, max 50, alpha and numeric mix. * Surname min 3, max 50. * Forename min 3, max 50. | Test each field by adding invalid information and report results. | Username field check ok.  Forename field check ok,  Surname field check ok,  Password field check ok.  Pass.  BugFix #KB005, Password field does not check for mixed characters. The fix check to see that atleast one letter and one number exists.    Retest Pass. | |
| 4.23 | Add user with correct information. | A new user should be visible from the User Table. | Pass. | |
| 4.24 | Library Addition / Deletion Tab. Attempt to add a Steganophic Library [i.e. Method] using a false dll/so [i.e. A non StegAid plugin]. | The system should fail and alert the user, that the plugin is invalid. | Pass. Libeay32 is not a valid stego plugin. Exception message generated needs to change to become more Human Friendly. | |
| 4.25 | Add a correct valid plugin dll/so such as libOpenSpace or other method. | The system should display the libraries Class ID and Library Name retrieved from the plugin itself. | Pass. | |
| 4.26 | Remove a library by selecting a row and clicking [- Minus]. | The library should be removed from the system disabling it’s use. | BugFix #KB006. Fixes refreshing the datatable.  Retest Pass. | |
| 4.27 | Mail Tab. Attempt to add an SMTP server with invalid server credentials. | The test passes if upon attempting sending of mail, the application reports to the user that email cannot be sent and the reason why. No mail code is actually build by me but by the library supporting the system. | Attempt to send mail to smtp.example.com, example.com is a reserved domain for example usage. This should fail because the port should be closed.    KB #007 Make Failure Message Make Sense. | |
| 4.28 | Attempt to add a POP3 server with invalid server credentials | The test passes if the user cannot receive mail and the system does not crash but reports the issue at hand. | No Messages. | |
| 4.29 | Attempt to add a valid pop3 server. | [Test later under mail] | -- | |
| 4.30 | Attempt to add a valid smtp server. | [Test later under mail] | -- | |
| 4.31 | Close form (Click Click()). | All settings saved and form closed. | All settings saved. | |
| 5 | Encode Text (frmEncodeText) | | | |
| 5.1 | [Usability] Ensure only the select file to encode button is selected. | You cannot encode something without a file. | Pass | |
| 5.2 | Select a large file [...] such as a video. And click Encode. | The system should inform the user that no option has been selected [i.e. No steganographic method]. | Pass | |
| 5.3 | Select a limited method such as Open Space and Encode. | The library should test the limit/capacity and report to the user the file is too large to encode using this method. Some methods are limited, others such as “Random and Statistical” generation methods are limitless. | Pass | |
| 5.4 | Select a small image or textual information and click Encode with any method. | If the capacity is not reached the system will encode the information in a cover text and enable you to save the output. Note, we do not need to decode test. As verification each steganographic method decodes the information after encoding to ensure the information is correct. If it fails report to the user. | Pass | |
| 5.5 | Save the output to a read only directory. | The system should warn that the file could not be written. | Pass | |
| 5.6 | Save the output. | The system should save the output as necessary. | Pass | |
| 5.7 | Click Close() | The system should close the form returning you to the Main Menu. | Automatic | |
| 6 | Decode Text (frmDecodeText) | | | |
| 6.1 | Upon entry of the decode text only file selection should be visible. | The test passes if only the appropriate options are visible [Usability]. | | Pass |
| 6.2 | Select an invalid file and attempt to decode as necessary. | The system will attempt to decode and provide a result regardless. There is no way of verifying this information. | | Application decodes |
| 6.3 | Select the file you encoded earlier and attempt decode. | The system should decode and provide a result regardless. | | Pass |
| 6.4 | Attempt Save As to a write protected folder. | The system should inform the user it cannot write to the directory. | |  |
| 6.5 | Attempt Save As. | The system should output using the filename and extension provided. | |  |
| 7 | Send Mail (frmNewMessage) | | | |
| 7.1 | Create a message, encode it without an email address to send to. | The system should inform the user has no receiver specified. If the compiler supports C++0x Regular Expressions, verification of email address also should take place. | |  |
| 7.2 | Create a message, and send to yourself | The system should encode the message using a selected option and send it via SMTP. If any issues occur the system should report. This is wholly dependent on the Poco libraries. | |  |
| 7.3 | Close Click() | The form should close and return to the main menu. | |  |
| 8 | Recieve Emails (frmViewMessages) | | | |
| 8.1 | Select a mail service provider from the list. | The system should instantly show a progress bar, and start downloading of messages. | |  |
| 8.2 | If messages exist in mail box, select a piece of mail. | The system should open the mail item and allow decoding as necessary. The system is text only basic client, so multi-part messages are ignored. | | The Project Gutenberg EBook of Five of Maxwell's Papers, by James Clerk Maxwell  (#1 in our series by James Clerk Maxwell)  Copyright laws are changing all over the world. Be sure to check the  copyright laws for your country before downloading or redistributing  this or any other Project Gutenberg eBook….  Pass |
| 8.3 | Attempt to decode the message using selected method/ | The textual output should be provided replacing the original message content. | | Hello All  If you can read this text, you have successfully decoded this message!Fire Fire Fire  Pass |
| 8.4 | Close Click() | The user should be returned to the main menu. | | Returned |
| 9 | Exit Click(). | The application should terminate successfully. | | Application Exits |

### Appendix 7 User Evaluations

The resulting documents are the results of evaluations by ten users. Five users were selected with a limited computing background with experience limited to checking emails and browsing the internet. The other five users have considerable computing background and have experience in a range of areas including Human Computer Interaction, Software Engineering and Psychology. It was felt that it was needed to get a sample from a wide spectrum of users in order to improve the product as much as possible within the time constraints available.