INSTITUTE of TECHNOLOGY CARLOW

Institiúid Teicneolaíochta Cheatharlach

STEGANOGRAPHY MESSAGING TOOL FUNCTIONAL SPECIFICATION

Institute of Technology Carlow

C00241234 – Luke Byrne

Table of Contents

Abstract 2

Introduction 2

Project Scope 3

- **Objective:** 3
- **Deliverables:** 3

Technologies 3

- **Language:** 3
- **Operating System:** 3
- **Programming Environment:** 4
- **Java Libraries:** 4

Assumptions 4

Risks 4

Risk Mitigation 4

System Functions 5

- **System Funtionality Breakdown 1 – 8 :** 5

Furps 8

- **Functionality :** 8
- **Usability :** 8
- **Reliability :** 8
- **Performance :** 8
- **Supportability :** 8

Abstract

The main objective of this functional specification document is to outline the main intentions of the steganography messaging tool, the various features of the tool and a high-level overview of the user's interaction with the program. UML use case diagrams will be used to show the users interaction with the program and the various behaviours of the application in order to perform the steganographic embedding and extraction. This document will outline an accurate and realistic project plan by determining a solid project scope and identifying all the major risks involved to minimise the scope creep risk, project performance and the operational risk of the project.

Introduction

The main objective of this functional specification is to obtain a concrete structure in which can be used for the development of this steganography messaging tool. This functional specification will outline the various components and their core functionalities in which are utilized in this tool. After discussing the functionalities of this tool, I aim to give a high-level overview of the main tools and technologies used to create this tool such as the specific programming language in which the tool will be developed in and which platforms in which it will be available on.

This steganographic messaging tool is used to embed and to obscure the presence of a secret message within a carrier image file. The main use of the steganography messaging tool is to conceal the existence of any sender-receiver communication occurring within the image cover file from a third party. With the use of a covert channel, the steganography tool can allow a recipient to retrieve and extract the hidden message in secrecy. This tool will allow a user to embed an encrypted message into the least significant bits of the cover image through the use of the LSB image embedding algorithm. Before the message into the contents of the cover image it will be encrypted using AES 128-bit encryption to provide secure transmission of the data from an adversary. In order for the retrieve or extract the embedded hidden message from the image, the user will need to read in the image using the steganography messaging tool then extract it using the decryption function which essentially reverses the encrypted concealed messages ciphertext into its relevant plaintext to transform the concealed message into a human readable format.

Project Scope

Objective:

The main end goal of this project is to create a fully functionable and easy to use steganography messaging tool to conceal the existence of a message being embedded into an image file. The hidden message will be encrypted with symmetric cryptography using the AES 128-bit encryption algorithm. The user will need to obtain the same key in which the message was encrypted with in order to retrieve the original plaintext message.

Deliverables:

The prime deliverables of this project consist of the following:

- To provide an user-friendly windows-based desktop tool with a graphical user interface (GUI)
- To allow a user to choose a cover image in a specific directory path on their machine in which they will use as the transmission medium (Cover image) to contain the steganographic message.
- User has the option to use plaintext or encrypt the message then embed it within the least significant bits of the image.
- Encrypt the users message using AES using a key then embed the ciphertext of the secret message into the least significant bits of the chosen cover media.
- To allow a user to extract the secret message in plaintext from the least significant bits of the image.
- To allow a user to decrypt the ciphertext of the secret message from the least significant bits of the cover image using the initial key in which the message was encrypted with.

Technologies

Language:

The language is which this tool will be developed in is Java. The reason why I am developing this tool in Java is because I am more familiar with Java. I also chose Java because this language is platform independent, meaning that Java programs can be complied and ran on multiple operating systems and environments. Another reason why I intend to use Java is because I want to use a particular cryptographic library compatible with Java called Bouncy Castle.

Operating System:

The operating system in which I will make this program available on is Microsoft windows 10.

Programming Environment:

The particular IDE in which I have chosen to use for the development of this tool is Eclipse IDE. I chose to use Eclipse because it contains a built-in GUI builder called Windows Builder which allows developers to create GUI based applications in a more efficient manner with regards to time. I will be using Windows Builder for the development of my GUI because it simplifies GUI development, and it will save me a lot of time.

Java Libraries:

The particular Java library in which I am going to use for the encryption and decryption functions of the program is the Bouncy Castle Java cryptographic library. Bouncy Castle is an open-source provider which includes an API for AES 128-bit encryption in Java, along with many other cipher suites and algorithms. Bouncy Castle is essentially an extended version of the default Java Cryptographic Extension (JCE).

Assumptions

- The scope of this project will not changed through the development process as everything that is stated in the deliverables will be 100% implemented into this tool.
- I have access to all resources in which I need to complete this project according exactly to the scope.

Risks

- Time may be a concern for the development of certain aspects of this project for me.
- My programming abilities may affect the development of the embedding function.

Risk Mitigation

- I am going to do a lot of programming over the Christmas period in order to complete the main aspects of the project such as the least significant embedding function ahead of the project deadline.
- I will follow the scope exactly in order and no additions will be made to it in order to prevent the risk of scope creep.

System Functions

The core functionalities of the application include the following:

1. Select a cover image from a file path in which the message will be embedded into.
2. Allow the user to input their secret message in which will be embedded.
3. Embed the plaintext of the secret message into the cover image.
4. Encrypt the secret message using AES 128-bit encryption then embed it within the cover image.
5. Saving the new steganographic image produced.
6. Embed the user's secret message into the cover image.
7. Extract the plaintext of the message from the cover image.
8. Decrypt and extract the message using the initial key in which it was encrypted with.

System Functionality Breakdown:

I aim create a windows-based tool that provides an easy-to-use graphical user interface (GUI) to the user. The purpose of creating the GUI as simple as possible is to make it user-friendly, allowing users to intuitively perform the basic operations of the tool in order to embed a message into a cover image and retrieve a message from a cover image without encountering any challenges. The main purpose of this tool is to allow a user to select a cover image of their choice in which they want to embed a secret message within.

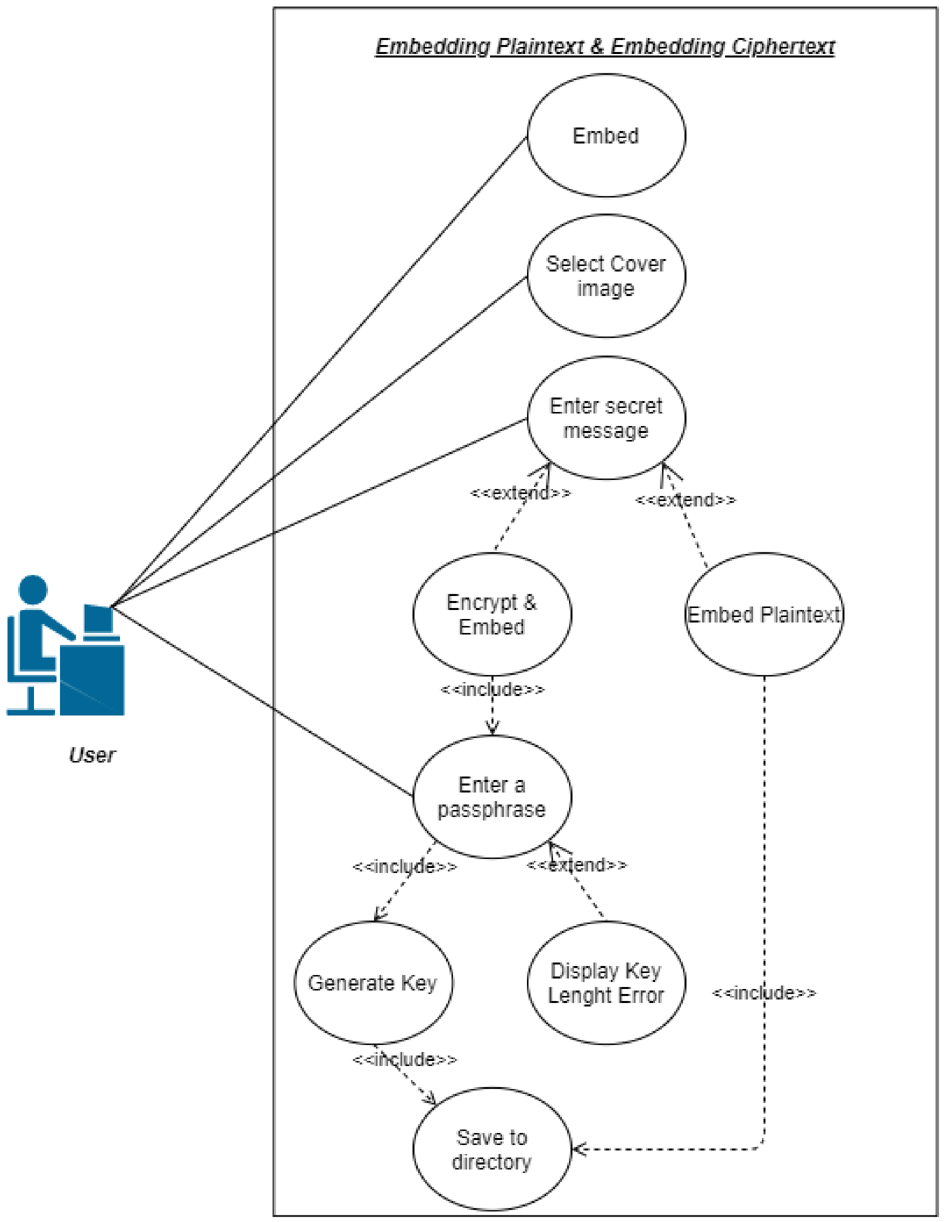
Firstly, the user will need to select a cover image in which they want to use as the transmission medium in order to embed a message into an image. The user will have the option to either embed message in plaintext into the cover image or to encrypt the message into ciphertext then embed it within the least significant bits of the chosen cover image using the LSB algorithm. The purpose of embedding the message into the least significant bits of the image is to ensure that data being substituted diminishes the amount of distortion caused to the quality and the attributes of the image.

Embedding and Encryption Operations:

In order for the user to embed a message in plaintext into the cover image, the user will need to enter their secret message into the provided text field then click the embed plaintext button. After the plaintext message has been embedded successfully, the user will need to click the save button to save the new steganographic image produced.

In order for a user to encrypt and embed a secret message into the chosen cover image using AES 128-bit encryption, the user will need to enter a secret message into the provided text field then create an appropriate passphrase. After the user has initialised the passphrase and clicked ok button, a hash of the passphrase will be used to generate the 128-bit encryption key. After the key has been generated the ciphertext of the message will be computed and embedded within the least significant bits of the cover image successfully. After the ciphertext of the encrypted message has been successfully embedded, the user will need to click the save button in order to save the new steganographic image produced.

Use Case Diagram to Embed Plaintext and to Embed Ciphertext into a cover image:

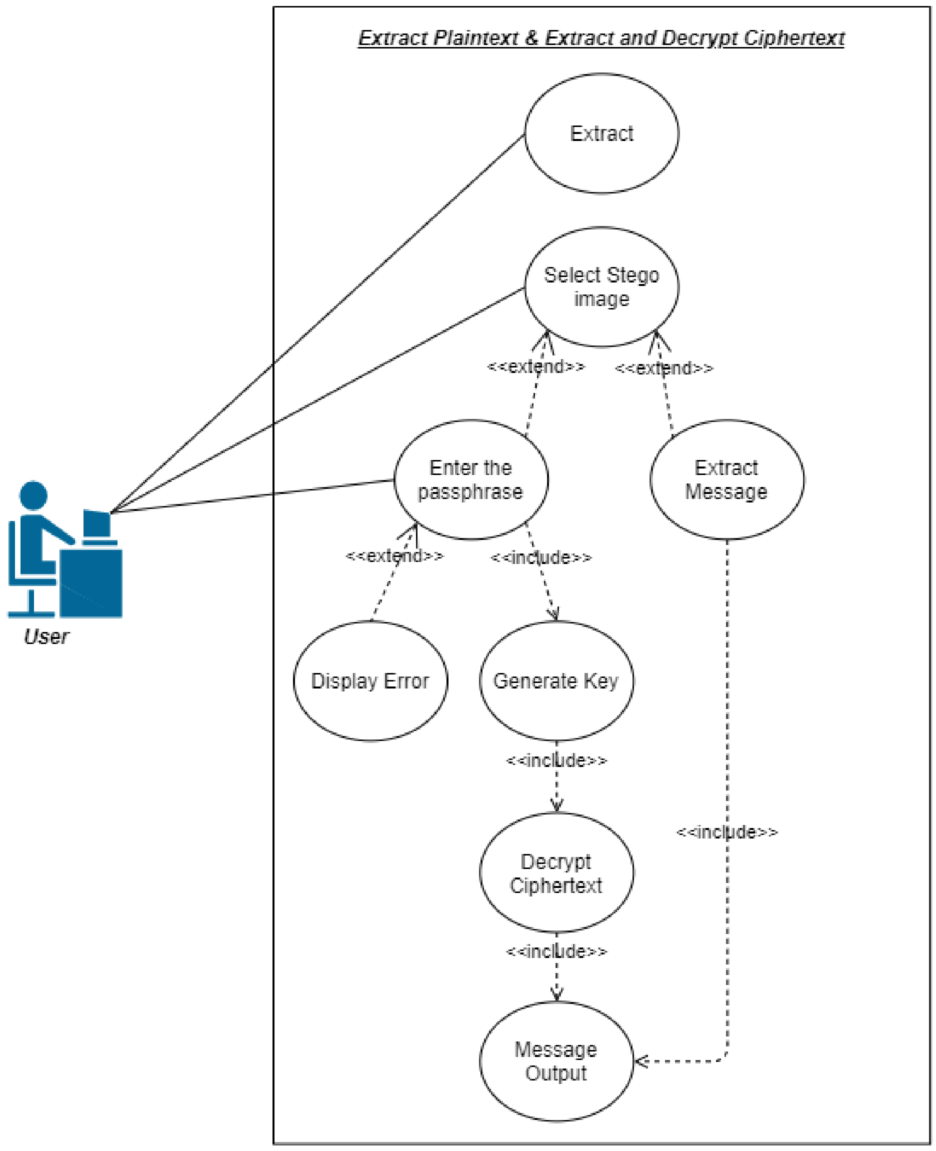


Extraction and Decryption Operations:

To extract the message from the cover image the user needs to select the extract button on the homepage of the application then select the stego image in which contains the steganographic secret message. After the user has selected the stego image if the message was embedded in plaintext the user can click the extract message button in order to retrieve the message.

If the secret message was encrypted before being embedded the user will need to provide the passphrase in which the encryption key derived from in order to be able to extract and decrypt the ciphertext of the concealed encrypted message from the cover image file into its corresponding plaintext.

Use Case Diagram to Extract Plaintext and Decrypt the Ciphertext from the cover image:



Furps

Furps is a model designed by Hewlett-Packard later developed further on by Robert Grady and Deborah Caswell. The primary principals of the furps model are to categorize modern software quality attributes and to inaugurate high grade metrics in software development phases based on the following factors – Functionality, usability, reliability, performance, and supportability. The main outputs of the furps model are the functional and non-functional requirements to develop a successful industry compliant software.

Functionality:

All core functionalities of this steganography messaging tool have been specified in system requirements and a high-level breakdown of each functionality was stated.

Usability:

This tool will allow an end user to attain the goal of embedding a steganographic message within a cover image and the ability to extract the message regardless or not the message be in plaintext or in an encrypted format without encountering any challenges. The GUI for this tool will be a simple as possible in order to ensure that the user can intuitively perform the basic behaviours of the system.

Reliability:

This system will be very straight forward meaning that there are not many points of failure within this system. This standalone tool will be developed in the simplest form possible with a restricted number of functionalities leaving minimal room for error. To ensure the reliability of this application I will conduct multiple reliability tests such as load testing, feature testing manually and automated to ensure that this application can perform at its optimum failure free.

Performance:

The performance of the embedding and extraction functions may vary with regards to computational speed depending on the length of the secret message in which the user is embedding. Overall, this application should operate swiftly enough as this application is not going to be resource intensive and not a lot of computational power will be needed to perform the basic operations that are present in the tool.

Supportability:

This application will be developed in Java meaning that it will not be platform dependent and therefore it will have the ability to operate across all operating systems. No internet connection will be required to run this program.

Metrics

Project metrics are considered to be the key factors in which will determine and scale the overall success of the software product. When the following strict criteria is met, the steganography messaging tool is considered to be a success.

Criteria	Metric Description	✓/ ✗
C1	A fully functionable steganographic tool with a GUI is created.	✓
C2	Users can select a cover image in which the message will be embedded into.	✓
C3	Uses can enter a secret message in which will be embedded into the cover image.	✓
C4	The plaintext of the secret message can be embedded into the cover image.	✓
C5	The encrypted secret message can be embedded into the cover image.	✓
C6	Users can save the new steganographic image produced.	✓
C7	The plaintext of the message can be extracted from the cover image.	✓
C8	The ciphertext of the message can be decrypted using the key it was encrypted with.	✓