Estenografía en contenido multimedia

Victor Adolfo Navarro Naranjo
ETIS

Antoni Martínez Ballesté

10/01/2007

RESUMEN DEL TRABAJO

En este proyecto se desea desarrollar una aplicación esteganográfica, la cual permita ocultar un texto en una imagen pasando este desapercibido ante los ojos de quien no lo espera, o le sea imposible recuperarlo a aquella persona que desee leer el mensaje sin la clave necesaria. Por el contrario la aplicación debe ser robusta, permitiendo a la persona que posee la clave recuperar el texto aunque la imagen haya sufrido pequeños cambios. Pero antes de continuar hagamos un pequeño estudio sobre la historia de esta arte.

La esteganografía es el arte o ciencia de comunicar de manera oculta un mensaje, camuflando la información entre otro conjunto de datos para que pase desapercibida. Hoy día suele utilizarse para esconder información en todo tipo en archivos tales como fotos, videos o audio.

El término esteganografía proviene del griego "steganos" (secreto) y "grafía" (escrito), y los primeros documentos que describen el uso de estas técnicas datan de los tiempos de Herodoto en la Grecia antigua. Una historia describe como enviaron un mensaje a Esparta para avisar de que Xerxes tenía la intención de invadir Grecia, de forma que pasara oculto ante cualquier inspección y que no levantara sospechas.

Por aquel entonces se escribía en tablones cubiertos de cera. Así, para camuflar el mensaje escribieron directamente sobre la madera, la cubrieron con cera y volvieron a escribir sobre ella. A simple vista sólo podía observarse el escrito sobre la cera pero, si se retiraba, podía leerse el mensaje oculto en la madera.

Durante la segunda guerra mundial el sistema más utilizado consistió en microfilmar un mensaje y reducirlo hasta el extremo de un pequeño punto, de forma que podía pasar como un signo de puntuación de un carácter dentro de otro texto. Por ejemplo, el punto de la vocal "i" podía ser en realidad un microfilm con un mensaje.

Con la llegada de los ordenadores se han ampliado y diversificado las técnicas esteganográficas. Una de las más comunes consiste en ocultar un mensaje dentro de contenidos multimedia, mezclando los bits del mensaje original entre los bits del archivo gráfico o de sonido. El archivo resultante será una imagen o archivo de audio totalmente funcional que, a primera vista, no levanta ninguna sospecha, pero con el software adecuado es posible extraer la información oculta.

Índice

1. INTRODUCCIÓN

Este Plan de Trabajo sentará las bases de planificación para la preparación, puesta en marcha y finalización de un proyecto sobre Es*teganografía*. El proyecto tiene un tiempo límite que abarca el segundo semestre del 2006 y en el que la investigación a partir de todos los medios y recursos disponibles será el motor impulsor del desarrollo del mismo.

El proyecto consiste en el desarrollo de un software *Esteganográfico*, el cual por su robustez deberá permitir ocultar textos en una imagen, así como recuperarlos de ser necesario. Para comprender el funcionamiento de la aplicación primero debemos tener claro que es la *Esteganografía* de la cual estaremos hablando durante toda la planificación, puesta en marcha y finalización del proyecto, y cuyo concepto principal se expone seguidamente.

Esteganografía: La **esteganografía** es la rama de la criptología que trata sobre la ocultación de mensajes, para evitar que se perciba la existencia del mismo.

Viene de un tratado de Johannes Trithemius llamado "Steganographia", del griego "escritura secreta", este tratado habla de la criptografía y de la esteganografía y está disfrazado como un libro de magia negra.

La palabra esteganografía proviene del griego:

- **Steganos: oculto, secreto**
- **Graphy: texto o dibujo**

Es el arte y ciencia de escribir mensajes secretos de tal forma que nadie fuera de quien lo envía y quien lo recibe sabe de su existencia; en contraste con la criptografía, en donde la existencia del mensaje es clara, pero el contenido del mensaje está oculto. Por lo general un mensaje de este tipo parece ser otra cosa, como una lista de compras, un artículo, una foto, etc.

Los mensajes en la esteganografía muchas veces son cifrados primero por medios tradicionales, para posteriormente ser ocultados por ejemplo en un texto que pueda contener dicho mensaje cifrado, resultando el mensaje esteganográfico. Un texto puede ser manipulado en el tamaño de letra, espaciado, tipo y otras características para ocultar un mensaje, sólo el que lo recibe, quien sabe la técnica usada, puede extraer el mensaje y luego descifrarlo.

Partiendo de la terminología, podemos definir a la esteganografía como la ciencia que estudia la ocultación. No son ciencias ocultas, de lo que se trata es de ocultar mensajes, pero no basándose en hacerlos irreconocibles como en la criptografía, sino en hacerlos pasar desapercibidos.

Para entrar en materia, se darán tres definiciones diferentes de la Esteganografía, desde el punto de vista etimológico, el informático y el punto de vista científico.

Etimológicamente hablando, esteganografía viene de la palabra griega stegos, que no significa otra cosa que "cubierta", por lo tanto, el significado de la palabra es "escritura cubierta".

Desde el punto de vista científico, esteganografía es el conjunto de métodos y técnicas para hacer pasar desapercibido o camuflar un mensaje.

Desde el punto de vista informático no dista mucho de la definición científica, pero desde su base técnica, esteganografía electrónica consiste en incluir un texto en un archivo de tal manera que este pase desapercibido para todo aquel que no espere el texto y aquel que lo espere, con las herramientas esteganográficas adecuadas y la clave, pueda recuperarlo.

Por otra parte ya se ha definido el lenguaje de programación que se utilizará para la programación de la aplicación esteganográfica y será Java. Su código se ejecutará a nivel de comandos sobre la consola de ejecución y se planificará en dos partes según los argumentos. Tratándose de una aplicación criptográfica no se ve necesario que el entorno de la aplicación tenga que ser amigable.

1.1. Limitaciones de la esteganografía

La esteganografía es una técnica que apenas esta desarrollada, estamos en la edad de piedra de la esteganografía.

Realmente, para utilizar de manera efectiva la esteganografía dependemos totalmente del tamaño del mensaje, de la imagen, de la paleta de color y lo que yo creo mas importante, el factor humano. De poco vale ocultar algo si luego resalta en su ambiente. Por ejemplo, de nada vale una foto de un paisaje con una calidad altísima en medio de un directorio lleno de archivos de otro tipo, el archivo contenedor no debe llamar la atención.

Personalmente, todas las aplicaciones de esteganografía están severamente limitadas, muchas solo trabajan con unos pocos tipos de archivo o exigen un tipo muy determinado de imagen, aparte que casi todas trabajan exclusivamente con imágenes, y no con otros tipos de archivos. En definitiva, en esteganografía no existe ningún tipo de programa referente y definitivo como puede existir en criptología -el PGP-, aunque entre todas las aplicaciones cubren un amplio abanico de posibilidades, todas ellas están bastante sesgadas y se necesita tener un montón de aplicaciones y tiempo para tener un poco de libertad a la hora de incluir mensajes.

También hay que tener en cuenta que muchos algoritmos de esteganografía no "sobreviven" a modificaciones de los archivos de portada que es otro dato muy importante a la hora de escoger un método u otro.

Otra limitación grave de la esteganografía, es sin duda los algoritmos, porque realmente son siempre los mismos, el programa "X" siempre incluye tal dato en la cabecera, y utiliza tantos bytes para una determinada función y siempre por poco que sea siempre dejan marcas de su paso por una imagen o archivo de portada. Esto hace frágil a la esteganografía cuando alguien esta buscando mensajes, por supuesto, la esteganografía es útil cuando no te están buscando.

1.2. Aplicaciones en la vida real

La esteganografía se utiliza en la vida real con mejores o peores resultados. Veremos algunos de sus usos reales, como pueden ser las marcas de agua, transmisión de mensajes terroristas, detección de copyright y algún que otro uso.

1.2.1. Marcas de agua

Sabiamente, corporaciones como la RIAA o la SGAE en España rebautizaron la fea palabra de "esteganografía" por "marcas de agua", evitando mancharse al utilizar una palabra muy fea que solo los chicos malos muy malos utilizan.

En la realidad, las marcas de agua y la esteganografía, aunque no son exactamente iguales (realmente las marcas de agua no son comunicación, son datos) todos sabemos que en la practica son lo mismo, y las intenciones pueden ser igual de "aviesas" tanto en el usuario de la esteganografía como en las sociedades de autores.

Las sociedades de autores pretenden incluir marcas de agua en todos sus archivos para luego soltar sus "spiders" (programas que buscan patrones) buscando mp3 con sus marcas de agua, cuando las encuentran, se quedan con la dirección de la pagina Web para ponerse en contacto con sus administradores y enviarles un mensaje muy poco amistoso sobre abogados y juicios.

La idea básica del Watermarking consiste en insertar información en la imagen mediante la realización de modificaciones sobre la misma, con el objetivo de proporcionar pruebas sobre quién es el propietario de la imagen o a quién ha sido vendida o enviada. La realización de este proceso sobre la imagen deberá ser imperceptible para el ojo humano, no afectando por tanto a su calidad. El objetivo es facilitar el control que se hace de materiales protegidos por derechos de autor, no sólo garantizar que el acceso a la información es el acordado, sino principalmente que no se va a hacer un uso ilegítimo de la misma, por ejemplo, comerciando con segundas copias. Es en este punto donde supone un importante valor añadido para los autores ya que con los sistemas criptográficos podemos autenticar al comprador y garantizar que el material ha sido entregado a la persona esperada y sin que nadie haya podido realizar copias ilícitas durante la transmisión. Sin embargo, una vez que el material multimedia está en poder del comprador, éste puede usarlo, copiarlo, revenderlo, etc... Sin control por parte del autor. Es aquí donde se detecta un grave problema.

1.2.2. Detección de copyright

Este caso es muy parecido al primero, las marcas de agua. Muchas empresas de software, por ejemplo Microsoft incluyen la licencia del producto, e incluso en algunos casos datos personales de la persona que creo el archivo.

1.2.3. Programas de ocultación

Hay que dejar clara la diferencia entre la criptografía y la esteganografía “estricta”:

Cuando únicamente utilizamos la criptografía, el dato puede ser ilegible, pero es obvio que allí existe un secreto.

Si el dato es sólo oculto y no encriptado, uno puede buscar todos los archivos sospechosos de contener información oculta y percatarse de que existe esa información que queremos ocultar.

La forma en que actúan juntos criptografía y esteganografía es que la criptografía hace el dato ilegible a quien no conozca la clave y la esteganografía, oculta además la existencia

de esos datos. Así los archivos siendo ocultados, hacen que lo oculto no sea ni leído ni detectado fácilmente.

Hoy en día, con la ayuda de los ordenadores ambas técnicas son perfectamente combinables, complementándose la una a la otra y consiguiéndose una seguridad aún mayor.

Cuando se oculta información en archivos de imágenes se aprovechan los bits menos significativos de los colores para introducir en ellos la información (con lo que se hace una reducción de colores respecto a la imagen original, si es necesario). Si la relación entre la información a ocultar, el tamaño de la imagen y el número de colores es buena, resulta prácticamente imposible diferenciar la imagen original de la imagen con información oculta.

Cuando se usan archivos de sonido, la información oculta aparece como ruido de fondo, pudiendo confundirse fácilmente con una simple grabación con algo de ruido.

- **Ejemplos de Software y su funcionamiento:**

La oferta de software esteganográfico disponible, gratuito o de pago, supera el centenar de programas. Obviamente, no todos los programas de esteganografía que circulan por Internet son eficaces al cien por cien, dado que la mayoría de estos programas de dominio público pertenecen a la 1ª o 2ª generación de esta tecnología, mientras que los investigadores están ya embarcados en la 4ª o 5ª generación.

El sistema más eficaz sería aquel que esconde un mensaje cifrado mediante un sistema como el PGP oculto en el interior de otro archivo. Se trata pues de una técnica mixta de ocultación y encriptación. Con respecto al tipo de mensaje en el que ocultar la información, dependerá de las costumbres de cada usuario, pero los formatos gráficos y musicales son los de intercambio más habitual en Internet.

En este apartado pretendemos realizar una pequeña comparativa entre programas que se utilizan para ocultar información en distintos tipos de ficheros. Normalmente, en los mecanismos que se utilizan para la ocultación de mensajes, los ficheros están muy relacionados con la naturaleza interna de estos ficheros están muy relacionados con la naturaleza interna de estos ficheros. Por ello, este tipo de programas suelen estar limitados a un tipo específico de ficheros(gráficos, de sonido o texto) e incluso dentro de un mismo tipo de ficheros puede tener limitaciones a sus propias características, por ejemplo: el formato, el tamaño, el número de colores utilizado, etcétera.

Digital Picture Envelope

Programa para Windows, gratuito y basado en las técnicas de Eiji Kawaguchi, uno de los grandes gurús de la esteganografía.

Utiliza para esconder los mensajes archivos BMP, y la técnica empleada es tan potente que un archivo de 100kb permite ocultar 50k de mensajes sin que el archivo original aumente de tamaño.

Hide and SEC

Conjunto de programas para MS-DOS. El programa oculta texto(encriptado o no) en ficheros tipo GIF.

El método utilizado por Hide and Seek se basa en el uso de los LSB's(bits menos significativos)de cada píxel para codificar los caracteres del texto que se pretende ocultar.

La dispersión utilizada es pseudo-aleatoria (como si fuera ruido). Cuanto mayor sea el fichero a ocultar mayor será el "ruido" y, por tanto, mayor será la evidencia del cambio. Por tanto, tendrá un límite en cuanto el tamaño del fichero a ocultar en 19kb.

Hide and Seek permite el uso de estego-clave con la cual encripta la cabecera del archivo de datos que se quiere guardar en la imagen. El algoritmo utilizado es IDEA.

Hide4PGP

Programa ejecutable para MS-DOS. Trabaja con ficheros BMP de 256 colores, WAV de 8,12 y 16 bits y VOC de 8bits.

Se basa en el hecho de que la mayoría de imágenes de 256 no tienen más de 100 colores distintos. Una rutina de duplicación ordena las entradas de la paleta en orden a la frecuencia de uso y duplica estas entradas. Esto le permite usar los LSB's sin que apenas exista alteración de la imagen.

Invisible Secrets

Programa de pago para plataformas Windows que cifra y oculta la información en archivos JPEG, PNG, BMP, HTML y WAV. Es capaz de interactuar con Windows Explorer.

MP3Stego

Este programa es gratuito y puede esconder la información en archivos musicales tipo MP3.

La calidad con un ratio de compresión de 128kbps es bastante buena. Resulta complicado detectar que el archivo va "cargado".

Stegdetect

Programa de estegoanálisis. Se trata de una herramienta automática que puede procesar lotes de archivos. Sobre todo detecta mensajes en el interior de los archivos gráficos, aunque no funciona demasiado bien con la información escondida con los programas más avanzados.

STEGHIDE

Es un sencillo programa de uso público que esconde mensajes en ficheros de tipo BMP, WAV y AV.

Es bastante intuitivo de utilizar, aunque puede resultar un tanto primitivo, dado que carece de algunas funcionalidades avanzadas. Nos impone una restricción: sólo funciona adecuadamente con mensajes cortos.

Steganos 3 Security Suite

Este programa para plataformas Windows es muy sencillo de utilizar y cuenta con una versión en español.

Emplea técnicas mixtas de cifrado y de esteganografía para acabar ocultando la información dentro de archivos gráficos o de audio.

Aunque no es un programa gratuito, sí que cuenta con una versión de prueba de 30 días.

Stegodos

Es un grupo de programas para MS-DOS. Permite capturar una imagen, codificar un mensaje dentro de ella y mostrar la imagen. Dicha imagen puede ser capturada otra vez en

otro formato para después recapturarla y decodificar el mensaje previamente ocultado en ella. Trabaja sólo con ficheros gráficos GIF o PCX.

Requiere una serie de comandos bastante complicados para introducir o extraer los datos de la imagen, lo que provoca que resulte difícil comprender la secuencia de órdenes.

S-Tools4

Este programa proporciona un interfaz con el usuario por medio de ventanas. Es capaz de trabajar con ficheros de distinto tipo: BMP, GIF, WAV. No tiene limitaciones en cuanto a tamaño o cantidad de colores.

Además de ocultar la información, permite también encriptar la información usando un password y utilizando distintos algoritmos: IDEA, DES, triple DES y MDC.

El método que utiliza S-Tools4 se basa en la introducción de los datos según un número pseudoaleatorio generado a partir de la encriptación de un password introducido.

De esta manera a la hora de ocultar los datos se elegirán unos píxeles determinados dependiendo del password introducido.

Texto

Es un programa que transforma archivos encriptado con *PGP armor,* en archivos que contienen texto en inglés. Este programa aprovecha el hecho de que el inglés es un idioma relativamente simple y que tiene ciertas estructuras irregulares, lo que provoca que un texto pueda tener errores. Texto sustituye los caracteres ASCII por palabras inglesas que se obtienen de un archivo llamado WORDS. Después utiliza estas palabras según unas estructuras contenidas en otro archivo llamado STRUCTS, creando de esta manera el texto en inglés. Pero existe un problema, el tamaño del

fichero obtenido mediante esta técnica es 10 veces superior al que pretendíamos ocultar.

A modo de curiosidad, mencionar también la existencia de una página: http://www.spammimic.com, que resulta una de las opciones más sencillas y originales.

Permite esconder correo bajo la forma de mensajes comerciales corrientes. Es el correo que más inadvertido pasa en Internet y el que más posibilidades tiene de acabar en la papelera.

El uso de Sapm Mimic es muy sencillo, basta con escribir el mensaje y pulsar el botón que lo convierte en simple spam. Luego se puede enviar como un e-mail normal.

2. UN POCO DE HISTORIA

A través de la historia, la esteganografía ha sido empleada utilizando diversos métodos y variantes para ocultar la información.

Ya en el antigua Grecia, los textos eran escritos en tablas cubiertas de cera. Otro método utilizado también en la antigüedad e igualmente ingenioso consistía en afeitar la cabeza de los mensajeros, tatuar el mensaje y esperar a que le creciera el pelo para enviarle allí donde debía llegar la información.

Otros métodos esteganográficos más comunes y accesibles y a los que todos hemos tenido acceso sin tener ni idea de que era eso de la esteganografía, son la escritura con tintas invisibles (leche, jugo de frutas, vinagre), que al ser expuestas al calor se oscurecen, dejando entrever el mensaje oculto. Igualmente sencillo e inocente resulta enviar un mensaje escribiendo un texto, y luego, tomando la primera letra de cada palabra, se podía leer el mensaje que en realidad se quería transmitir.

Durante la II Guerra Mundial, comenzó a utilizarse el micro punto: un mensaje secreto era fotográficamente reducido a la medida de un punto y pegado como el de la letra i en un papel que contuviese un mensaje cualquiera escrito.

Al igual que ocurrió con la criptografía, la llegada de los ordenadores permitió grandes avances en las técnicas criptográficas, consiguiendo automatizar tareas que solían ser bastante engorrosas.

Ejemplo esteganografía en la historia

- *400 a.c. guerra de Persia contra Grecia,* Demaretus, griego, envía mensajes a través de un par de tablas unidas por una cera.
- *600 a.c. China,* Uso de tela muy fina, enrollado en forma de bola y cubierta de cera avalada por el mensajero.
- *Egipto,* El mensajero era rapado y el mensaje era escrito en su cabeza.
- *Siglo XVI, Giovanni Porta,* Usa una tinta a base de alumbre y vinagre se usa para escribir en la cáscara de un huevo duro. La tinta atraviesa el cascarón y se fija en el huevo.
- *2da Guerra Mundial, alemanes en Latinoamérica,* Puntos de 1 milímetro introducidos en una carta.
- *Tinta invisible.*
- *Hoy en día* Uso de software para ocultar imágenes en archivos de imágenes o documentos office.

3. ESTRUCTURA DE LA MEMORIA

En este tercer punto se explicarán las razones y motivaciones por las cuales se ha escogido este proyecto así como los objetivos principales que se desean conseguir a su finalización. Del mismo modo se describirá una planificación del proyecto tanto a nivel temporal como de costes.

3.1. Razón y oportunidad del proyecto

La razón por la que he escogido la esteganografía como proyecto, esta basada en el misterio que esta encierra. Detrás de este ambicioso arte milenario, se esconde todo un mundo de espionaje y secretismo muy elaborado y en el que han participado grandes científicos y sabios de todos los tiempos. Aprender a pensar como ellos, e intentar honrarlos con mis esfuerzos en crear una aplicación que se acerque a lo que ellos han diseñado y han dado nombre, es una oportunidad a no pasar por alto.

En nuestros días una herramienta como esta puede ser de mucha utilidad a todos los niveles, desde grandes empresas y multinacionales, hasta el ciudadano de a pie pasando por las medianas y pequeñas empresas. Por lo que mi afán de conocimiento, y mi interés por las prestaciones que una utilidad como esta puede brindar, me han hecho decantarme por este proyecto aunque mi desempeño y dedicación sean parciales.

3.2. Objetivos

El objetivo de este proyecto, esta en construir y evaluar la funcionalidad de un programa que ofrezca servicios de ocultación y recuperación de mensajes en un BitMap. La robustez de la aplicación será estudiada en profundidad.

A lo largo de la memoria se analizarán diferentes herramientas para el desarrollo de la aplicación, se hará un estudio minucioso de las técnicas de ocultación y se describirá el proceso de diseño seguido, previo a la implementación del mismo utilizando el lenguaje de programación JAVA.

A lo largo de este documento veremos que la esteganografía puede ser mucho mas que las definiciones que se pueden ver en muchos sitios en Internet de esconder un fichero en otro. Y detallaremos el diseño y programación de un sistema estenográfico, con sus particularidades, limitaciones y su funcionamiento en la vida real.

3.2.1. Objetivo general

El objetivo de este proyecto consiste en la programación de una aplicación esteganográfica. Esta aplicación deberá ser capaz de ocultar un texto en una imagen, y que permita que este texto sea recuperado nuevamente. Para esto la aplicación deberá ser robusta, y brindará un servicio a todas aquellas personas que reciban la imagen y conozcan la existencia del mensaje de poder recuperar este último. Pasando desapercibido para el resto de persona que tengan acceso a la imagen, pero que desconozcan la existencia del mensaje oculto.

Esto implica que dentro de los objetivos del proyecto entre un estudio minucioso sobre las cualidades y principios que esconde en si la Esteganografía. Así como los diferentes tipos y formas que encierra en su concepto principal, ya que aunque suene extraño, la

Esteganografía es un arte que existe desde hace mucho tiempo, incluso antes de que existiera la informática.

En general para un correcto diseño del proyecto, los objetivos estarán desglosados en pequeñas partes importantes que formarán un todo al culminar el proyecto, con una aplicación funcional y robusta, así como la documentación de la misma para el buen uso y aprovechamiento de quienes necesiten utilizarla.

3.2.2. Objetivo específico

El camino a seguir para llegar al objetivo final de este proyecto se divide en cuatro partes, *investigación, programación, prueba y documentación*.

Investigación: Este es el objetivo inicial, documentarse para apilar la mayor cantidad de información sobre el funcionamiento de las aplicaciones Esteganográficas, entre las cuales podemos tener como ejemplo “Camaleón”.

Programación: Será el objetivo más importante y al cual dedicaremos más tiempo, y consistirá en definir el diseño y el camino a tomar gracias a la investigación, así como programar el código con el lenguaje de programación ya determinado, en este caso Java.

Prueba: En esta parte de proyecto nos detendremos para asegurarnos que la aplicación funciona correctamente, así como que cumple que la robustez exigida en al diseño.

Documentación: Esta es la parte final e igual de importante que las anteriores, ya que en esta etapa se documenta todo el proceso de diseño, así como el funcionamiento y puesta en marcha de la aplicación. Este consistirá en una memoria del proyecto que acompañará a la aplicación como un manual de funcionamiento.

3.3. Planificación

Se describen a continuación las actividades de este Plan de trabajo, agrupados por tipos. Se hace una llamada de atención al hecho de que, dentro de un mismo temario previsto, la ficha técnica con el título concreto de cada actividad se ha adaptado al contenido previsto en cada caso.

Para comenzar definamos algunos conceptos en los que se basa el proyecto:

- Watermarking

Los esquemas de “Watermarking” son aquellos que se utilizan para proteger los contenidos digitales en cuanto a los derechos del autor. Su funcionamiento se basa en la inserción de marcas en los productos para su posterior identificación. De este modo, un producto marcado incorpora la identidad de su autor y la copia de dicho producto implica la copia de las marcas que identifican a su autor.

Existen distintas propiedades que deben cumplir los esquemas de protección del copyright, como la imperceptibilidad, la capacidad, o la robustez. De entre ellas, la más difícil de obtener es la robustez, que a su vez es la más importante dado que de ella depende la utilidad del esquema en cuestión.

- Esteganografía

La esteganografía es el **arte de transmitir información de modo que la presencia de la misma pase inadvertida**, típicamente escondida dentro un texto o una imagen. Proviene de las palabras griegas steganós (cubierto) y graptos (escrito), literalmente escrito cubierto, en el sentido de escondido.

A diferencia de la criptografía en la que el mensaje se ve que está cifrado e induce a sospecha, aquí **la información está inmersa sutilmente en un vehículo, sea texto, imagen o sonido, aparentemente normales**. La información que contiene sólo puede revelarse aplicando un procedimiento apropiado.

3.3.1. Búsqueda de información y recursos

Para comenzar será necesario conocer las APIs que se utilizarán en el desarrollo de la aplicación. Para ello se recopilará información de ambas estudiando sus JavaDoc, el código fuente y los ejemplos que se puedan encontrar en las diferentes Webs y artículos que hagan referencia a ellas.

Se prevé un tiempo de estudio e investigación en busca de información sobre el tema a tratar, es decir, La Esteganografía:

- Búsqueda de conceptos, utilización e historia, para esto se hará huso de Internet y archivos bibliotecarios.
- Búsqueda de aplicaciones parecidas que ya se encuentren funcionando en el mercado para aprender de ellas. Así como algoritmos y métodos criptográficos de ocultación.
- Estudio de código de aplicaciones así como del lenguaje de programación a utilizar.

Tiempo estimado: 80 horas

3.3.2. Diseño de la aplicación

Una vez se conozcan las APIs que se utilizarán para el desarrollo de la aplicación será necesario realizar el diseño de la misma previo a su implementación. Se realizará un diagrama genérico de las clases necesarias y diagramas de secuencia de las acciones principales, para todo ello se utilizarán diagramas UML.

El seguimiento será el siguiente:

- Búsqueda del lenguaje de programación para la construcción de la aplicación, en este caso como se programará en Java, se utilizará el Eclipse.
- Búsqueda de la Máquina Virtual de Java, así como de las diferentes APIs a utilizar que no vienen por defecto en la MVJ.
- Instalación y puesta en marcha de la aplicación y de la MVJ.
- Programación estructural de la aplicación a partir de la información obtenida en el apartado de Búsqueda de información.

Tiempo estimado: 70 horas

3.3.3. Implementación de la aplicación

Cuando tengamos listo el diseño de la aplicación se pasará a realizar la implementación de esta. Para ello se utilizará el IDE Eclipse. Durante la fase de implementación se irán realizando periódicamente pruebas para comprobar el correcto funcionamiento de cada una de las partes del proyecto por separado.

Esto facilitará mucho las labores de integración de las diferentes partes.

Tiempo estimado: 180 horas

3.3.4. Prueba de funcionamiento y de robustez de la aplicación

Una vez finalizada la aplicación se realizará una pequeña batería de pruebas que nos permita evaluar el funcionamiento de la misma en un entorno real.

Se ejecutará la aplicación con el juego de prueba, y se chequeará buscando posibles errores en ejecución, así como la actitud del programa ante el posible error de quien lo utiliza. Para de ser necesario ir depurando estos errores regresando al punto anterior de programación y reconstruyendo la aplicación aprendiendo del error.

Se irán guardando los resultados y se harán pruebas donde terceros utilizan la aplicación, de esta forma se definirá los mecanismos a utilizar para la enseñanza de la aplicación y los manuales de uso. Es increíble lo que se puede aprender de un usuario estándar a la hora de instalar o usar una determinada aplicación. Por lo que en las pruebas nos detendremos un poco, pero nos ayudará a comprender y mejorar el resultado final del proyecto.

Tiempo estimado: 20 horas

3.3.5. Redacción de la memoria

En este momento se procederá a recopilar toda la información obtenida durante las fases anteriores y se redactará el documento final que dará cuerpo a la memoria final del TFC.

Aunque esta como último paso a realizar, realmente es algo que se deberá hacer desde el principio. De hecho ya lo estamos haciendo. Documentar un proyecto es como llevar todo un historial de todos los pasos que se dan en la realización del mismo. De aquí saldrá como resultado la memoria del proyecto, en el que estará muy detallado, el camino que se siguió en la realización, los diferentes caminos alternativos que se pudieron seguir y el porque no se escogieron, el regreso a un punto anterior para escoger otro camino ante la dificultad de seguir adelante por el camino escogido. La descripción de las pruebas, con sus resultados. El manual de funcionamiento y de uso. Y entre muchas otros detalles importantes el resumen final.

Una presentación Interactiva de la utilización y funcionamiento de la aplicación podrían completar el manual para la enseñanza del funcionamiento de la aplicación a terceros.

Tiempo estimado: 25 horas

3.3.6. Presentación

Finalmente se preparan las diapositivas necesarias para la defensa pública del TFC y el guión que se seguirá para la presentación final del TFC.

Tiempo estimado: 30

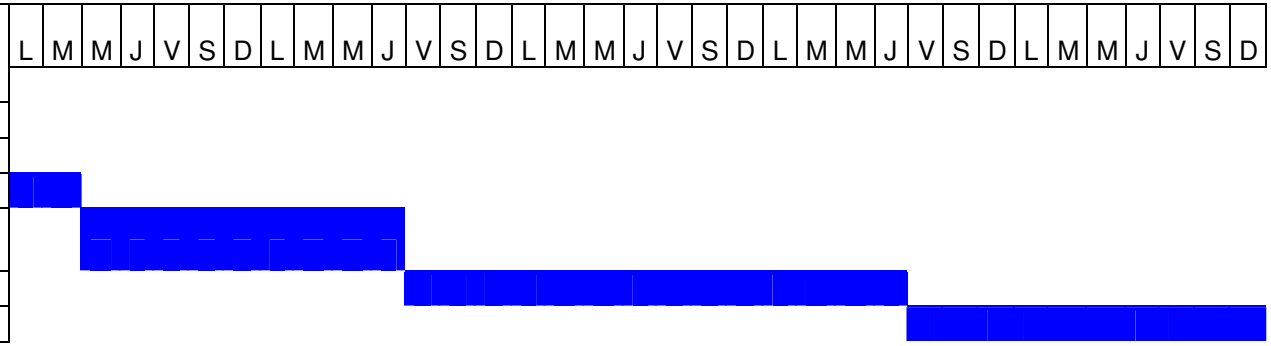
3.3.7. Diagrama de Gantt

TAREAS	DURACIÓN	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D
Busqueda de información	80 H																																			
Diseño de la aplicación	70 H																																			
Implementación de la aplicación	180 H																																			
Prueba de funcionamiento y robustez	20 H																																			
Redacción de memoria	25 H																																			
Preparación Presentación	30 H																																			

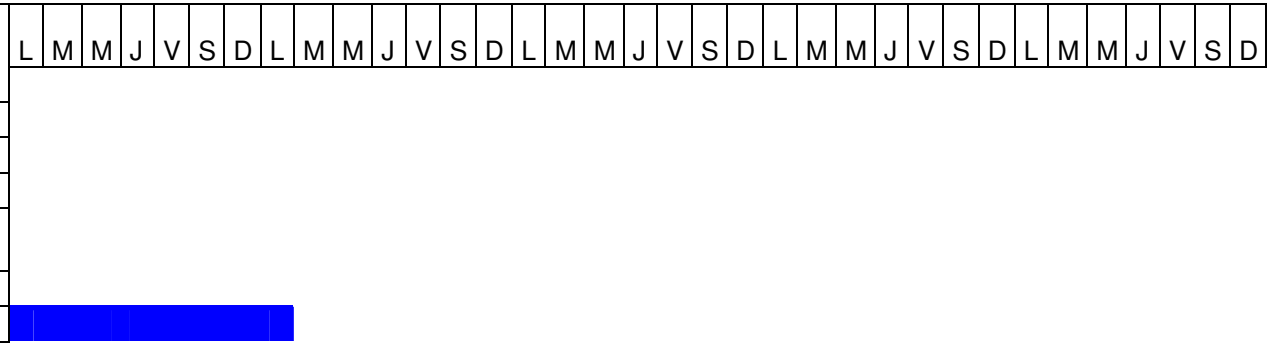
TAREAS	DURACIÓN	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D
Busqueda de información	80 H																																			
Diseño de la aplicación	70 H																																			
Implementación de la aplicación	180 H																																			
Prueba de funcionamiento y robustez	20 H																																			
Redacción de memoria	25 H																																			
Preparación Presentación	30 H																																			

TAREAS	DURACIÓN	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D
Busqueda de información	80 H																																			
Diseño de la aplicación	70 H																																			
Implementación de la aplicación	180 H																																			
Prueba de funcionamiento y robustez	20 H																																			
Redacción de memoria	25 H																																			
Preparación Presentación	30 H																																			

TAREAS	DURACIÓN
Busqueda de información	80 H
Diseño de la aplicación	70 H
Implementación de la aplicación	180 H
Prueba de funcionamiento y robustez	20 H
Redacción de memoria	25 H
Preparación Presentación	30 H



TAREAS	DURACIÓN
Busqueda de información	80 H
Diseño de la aplicación	70 H
Implementación de la aplicación	180 H
Prueba de funcionamiento y robustez	20 H
Redacción de memoria	25 H
Preparación Presentación	30 H



3.3.8. Planificación de Actividades

Téngase en cuenta que la planificación es una aproximación de los tiempos que se piensan serán suficientes para la realización del proyecto, pero no son exactos. Muchas pueden ser las causas que puedan alterar estos tiempos y sus resultados, pero de una buena planificación se pueden obtener buenos resultados. De lo contrario los resultados podrían ser inesperados.

Act	Fecha	Activitat	Esdeveniment
1	01 – 16 Octubre	Definir los objetivos del proyecto. Planificación temporal del trabajo, e idea inicial de diseño. Instalación del software a utilizar y puesta en marcha de los medios necesarios.	**Entrega de la planificación provisional.**
2	17 Octubre – 20 Noviembre	Realización del PAC2 y trabajo en la primera parte de la aplicación a entregar. Búsqueda de información, selección de algoritmos, y comienzo. Recogida de datos en la memoria. Debate	**Entrega de la planificación de la memoria hasta el momento y el PAC 2.**
3	21 Noviembre – 18 Diciembre	Realización del PAC3 y trabajo en la segunda parte de la aplicación a entregar. Programación y puesta en marcha de la aplicación. Documentación de seguimiento para la confección de la memoria. Primeras Pruebas y Debate.	**Entrega de la planificación de la memoria hasta el momento y el PAC 3.**
4	19 - 26 Diciembre	Finalización de la aplicación y pruebas finales. Documentación de seguimiento para la confección de la memoria final. Debate	**Entrega del borrador de la memoria.**
5	27 Diciembre - 10 Enero	Retoque final del código de la aplicación y de la memoria.	**Listo para ser entregado.**
6	10 - 15 Enero	Realización de la presentación y las conclusiones.	**Entrega final de todo el proyecto.**

Tabla 1 - P*lanificación de los tiempos estimados que se dedicaran a las diferentes partes en las que se dividirá la realización del proyecto.*

El proyecto quedará dividido en un número de tareas, las cuales también tienen asignadas tiempos de realización no exactos.

Id.	Tasca	Duración	Precedentes
A	Lectura del Proyecto	3	-
B	Redacción del plan de trabajo	7	A
C	Recaudación de información.	10	B
D	Estudio de las diferentes posibilidades.	4	B
E	Evaluación del hardware necesario	1	D
F	Evaluación del software a utilizar.	1	D
G	Instalación y puesta en marcha del HW y SW	2	E,F
H	Realización y pruebas de la aplicación	30	C,D,E,F
I	Preparación de la documentación	2	H
J	Preparación de una demostración	5	H
K	Montaje final y entrega.	2	H,I,J

Tabla 2 -*Tareas en las que se dividirá el proceso de trabajo a medida que se avanza en el proyecto.*

Otro punto de vital importancia es el lenguaje de programación a escoger para la construcción de la aplicación. Por las características iniciales del proyecto, la gran cantidad de librerías, y las avanzadas facilidades de los Lenguajes Orientados a Objetos; creo que el lenguaje más apropiado es Java.

Por lo que el software a instalar inicialmente para programar en este lenguaje será Eclipse, con sus librerías y la maquina virtual de java de SUN MICROSISTEM.

También será interesante hacer un estudio de características que debe cumplir la aplicación y la instalación de aplicaciones que simulen o realicen el proceso como muestra, para las posteriores pruebas y las comparaciones pertinentes. Por lo que aplicaciones como camouflage serían interesantes en el estudio y planificación del proyecto.

4. ENTORNO DEL PROYECTO

En este punto se describirá el funcionamiento básico de la estructura de una imagen BMP o mapa de Bits y la estructura de un texto partiendo de la distribución binaria de un carácter así como la conversión de un carácter a binario y de binario a carácter.

4.1. Características de una imagen BMP

Los archivos con extensión **.BMP**, en los sistemas operativos Windows, representan la sigla BitMaP, o sea mapa de bits. Los archivos de mapas de bits se componen de direcciones asociadas a códigos de color, uno para cada cuadro en una matríz de pixeles tal como se esquematizaría un dibujo de "colorea los cuadros" para niños pequeños. Normalmente, se caracterizan por ser muy poco eficientes en su uso de espacio en disco, pero pueden mostrar un buen nivel de calidad. A diferencia de los gráficos vectoriales, al ser reescalados a un tamaño mayor, pierden calidad. Otra desventaja de los archivos BMP es que no son utilizables en páginas web debido a su gran tamaño en relación a su resolución.

Dependiendo de la profundidad de color que tenga la imagen cada pixel puede ocupar 1 o varios bytes. Generalmente se suelen transformar en otros formatos, como JPEG (fotografías), GIF o PNG (dibujos y esquemas), los cuales utilizan otros algoritmos para conseguir una mayor compresión (menor tamaño del archivo).

Los archivos comienzan (cabecera o header) con las letras 'BM' (0x42 0x4D), que lo identifica con el programa de visualización o edición. En la cabecera también se indica el tamaño de la imagen y con cuántos bytes se representa el color de cada pixel.

4.2. Estructura de un texto y conversión a binario carácter a carácter

La **codificación de caracteres** es el código que relaciona cada **carácter** de un lenguaje natural (alfabeto o silabario) con un símbolo en otro sistema de representación, como un número o una secuencia de pulsos eléctricos en un sistema electrónico. Ejemplos de esto son el código Morse, la norma ASCII o la UTF-8, entre otros.

El problema es que el ASCII, por estar íntimamente ligado al octeto, y a los enteros que van del 0 al 127, no puede codificar más que 128 símbolos diferentes. 128 es el número total de diferentes configuraciones que se pueden conseguir con 7 dígitos binarios (*0000000, 0000001, ..., 1111111*), y el octavo dígito de cada octeto ("dígito de paridad") se usaba para detectar algún error de transmisión. Un cupo de 128 es suficiente para incluir mayúsculas y minúsculas del abecedario inglés, además de cifras, puntuación, y algunos "caracteres de control" (por ejemplo, uno que instruye a una impresora que pase a la hoja siguiente), pero el ASCII no incluye ni los caracteres acentuados ni el comienzo de interrogación que se usa en castellano, ni tantos otros símbolos (matemáticos, letras griegas) que son necesarios en muchos contextos. Por esta razón se vio que era insuficiente y se definieron varios códigos de caracteres de 8 bits.

Sin embargo, el problema de estos códigos de 8 bits es que cada uno de ellos se define para un conjunto de lenguas con escrituras semejantes y por tanto no dan una solución unificada a la codificación de todas las lenguas del mundo. Es decir, no son suficientes 8 bits para codificar todos los alfabetos y escrituras del mundo.

La solución que se ha venido en adoptar, internacionalmente, es *separar la norma de codificación* (elección de símbolos, y asignación de un código fijo a cada símbolo) *de la norma de transmisión,* usándose ambas en conjunto. La **norma de codificación** más universal en la actualidad, y desde 1991, se llama Unicode: es una gran tabla, que en la actualidad asigna un código a cada uno de los más de cincuenta mil símbolos, los cuales abarcan todos los alfabetos europeos, ideogramas chinos, japoneses, coreanos, muchas otras formas de escritura, y más de un millar de símbolos especiales. Normas de **transmisión** de Unicode hay varias, pero la más relevante en Internet es UTF-8.

Los mensajes de Internet se transmiten en paquetes que siempre constan de un número entero de octetos, y la detección de error ya no se hace con el octavo dígito de cada octeto, sino con octetos especiales que automáticamente se agregan a cada paquete. Aquello forma parte de los protocolos de transmisión de datos, que constituyen toda una jerarquía de normas; UTF-8 se limita a especificar una correspondencia, reversible, entre códigos (que representan caracteres), y secuencias de octetos (que han de ser transmitidos en calidad de datos).

Entonces, aquí está la razón de utilizar Unicode (norma de **codificación)** con UTF-8 (norma de **transmisión):** en primer lugar, Unicode asigna los enteros del 0 al 127 (un total de 128) a exactamente los mismos caracteres que ASCII; en segundo lugar, UTF-8 empaqueta cualquier entero del 0 al 127 en un octeto "a la antigua" pero con el octavo dígito siempre en cero. (Recordemos que ese octavo dígito, en ASCII llamado "dígito de paridad," modernamente ya no se usa para detección de errores, aunque sí se usa como parte de la codificación en ISO 8859.) Pero como la tabla de Unicode es tan grande, la mayoría de sus símbolos están asignados a enteros mayores que 127 (códigos que, en consecuencia, necesitan *más* que 7 dígitos para su representación binaria). En todos esos casos, UTF-8 envía, en un primer octeto con dígito de paridad = 1, el *comienzo* de la representación binaria del código en cuestión, y la máquina que recibe, al ver ese "1", no lo interpreta como indicación o no de error o como parte del carácter, sino como indicación de que, lo que está siendo transmitido es un código que no cabe en 7 dígitos binarios; y por tanto interpreta que el símbolo correspondiente no lo va a conocer mientras no lea el siguiente octeto, y tal vez el que sigue. En el peor de los casos, quizás se haga necesario leer ***seis*** octetos consecutivos para determinar un código alto.

Pero, para cartearse electrónicamente en mandarín (por ejemplo) falta un detalle importante:

La tabla que el Consorcio Unicode publica para ser leída por humanos, contiene una representación gráfica o descripción, de cada carácter incluido hasta ese momento; pero, los sistemas de visualización de documentos, para poder funcionar, requieren ***tablas de tipografía,*** que asocian un glifo (un dibujo) a cada carácter que abarcan, y sucede que hay muchísimas tablas de tipografía, con nombres como Arial o Times, que dibujan una misma letra a base de matrices diferentes y en diferentes estilos ("A" o "**A**"); sin embargo, la gran mayoría de las fuentes tipográficas abarcan sólo un pequeño subconjunto de todos los caracteres Unicode. Por eso, para leer páginas con caracteres asiáticos, por ejemplo, no basta que el visualizador usado "acepte" la codificación Unicode, sino que, además, el computador debe tener instalada una tabla tipográfica suficientemente extensa.

Para profundizar más podemos dirigirnos a las siguientes páginas:

http://www.jegsworks.com/Lessons-sp/lesson4/lesson4-2.htm
http://descartes.cnice.mecd.es/Documentacion_3/Doc/Simbolos.html
http://es.wikipedia.org/wiki/Plantilla:Tabla_Unicode

4.3. Basé de desarrollo de la aplicación

Partiendo del echo de que la aplicación ha desarrollar será programada en Java, vale destacar los paquetes que se van a utilizar y que forman parte de la gran API que es Java y su compilador de SUN MICROSYSTEM.

Para el tratamiento, creación y modificación de las imágenes se utilizara la clase java.awt, en especial, la clase java.awt.Image, toda la información de esta clase podemos encontrarla en: http://java.sun.com/j2se/1.5.0/docs/api/

Para el tratamiento de texto, conversión a sistema binario y viceversa se utilizará la clase java.io, toda la información de esta clase podemos encontrarla en: http://java.sun.com/j2se/1.5.0/docs/api/

5. PROPIEDADES DE UN SISTEMA ESTEGANOGRÁFICO ROBUSTO

Existe un gran número de publicaciones en las que se discuten los requisitos que debe cumplir la esteganografía. Es bueno destacar que la seguridad de estos sistemas no debe estar basada en la ocultación de los algoritmos utilizados, sino en la fortaleza de los mismos y en la seguridad de la clave.

Entre las propiedades deseables de un sistema esteganográfico se encuentran la robustez, la resistencia a las manipulaciones, imperceptibilidad, el costo computacional y la baja probabilidad de error.

5.1. Robustez

Los archivos digitales de imágenes, audio y video, están expuestos a muchos tipos de modificaciones (o distorsiones): las pérdidas por compresión, los cambios producidos por el mejoramiento de imágenes, la amplificación de las señales de audio, etc. Un sistema esteganográfico se considera robusto si perdura después de esas operaciones y, en imágenes y en video, también deben persistir después de las transformaciones geométricas (recortado, rotación, escalado). Esto quiere decir que el texto oculto presente en los archivos debe poder ser recuperado después de las distorsiones. Para consolidar su robustez, los sistemas esteganográficos, deben insertar el texto en regiones preceptúalmente significativas de los archivos multimedia.

Existen diversas opiniones a la hora de definir la robustez de estos sistemas, nuestro punto de vista coincide con las formuladas en los trabajos de Jiri Fridrich entre otros, por tanto aclaramos que la valoración de esta propiedad de los sistemas esteganográficos como las marcas de agua, no incluye los ataques basados en el conocimiento de los algoritmos de incrustado y detección de textos, la robustez significa resistencia a ciegas frente a aquellas modificaciones producidas por las operaciones comunes a las que estarán expuestos los archivos multimedia.

La robustez no debe exigirse incondicionalmente, ya que el sistema de algoritmos puede necesitar ser robusto respecto a determinados procesos y frágil respecto a otros. Si un sistema esteganográfico requiere que ciertas modificaciones de los archivos dañen la información oculta, se le denomina sistema de esteganográfico frágiles y es muy importante en determinadas aplicaciones como veremos más adelante.

5.2. Resistencia a manipulaciones

La resistencia a manipulaciones de un sistema esteganográfico es un aspecto que puede relacionarse con la seguridad del mismo; se refiere a su resistencia frente a los ataques hostiles basados en el total conocimiento de los algoritmos de incrustado y detección y de los archivos marcados, excepto de la clave utilizada. Se incluyen aquí los ataques a los protocolos y los ataques basados en la estimación del sistema. Según la aplicación de que se trate, unos ataques serán más importantes que otros; en general, un ataque efectivo deberá eliminar la posibilidad de recuperar el texto, sin cambiar la calidad perceptual del archivo en cuestión; sin embargo, existen varias aplicaciones en las que la resistencia a determinadas manipulaciones es un aspecto indeseable. A continuación se revisan una de las aplicaciones utilizadas para atacar y romper la seguridad de un sistema esteganográfico, intentando recuperar el texto oculto.

Stegdetect es una herramienta para detectar contenidos camuflados en imágenes es decir, detecta la posible información oculta en una simple fotografía. Una aplicación útil e interesante para contrarrestar la esteganografía.

5.3. Análisis estadístico

Las pruebas estadísticas pueden revelar si una imagen ha sido modificada por Esteganografía probando si las propiedades estadísticas de una imagen se desvían de una norma. Algunas pruebas son independientes del formato de los datos y simplemente miden la entropía de los datos redundantes.

Estas pruebas simples no pueden decidir automáticamente si una imagen contiene un mensaje oculto. Westfeld y Pfitzmann han observado que empotrando datos del encrypted en una imagen cambia el histograma de sus frecuencias coloridas. Una propiedad de los datos del encrypted es que el uno y el cero son pedazos igualmente probables. Al usar el método del pedazo menor-significante para empotrar datos del encrypted en una imagen que contiene el color dos más a menudo que el color tres, el color dos se cambia más a menudo al color tres que al revés. Como resultado, la diferencia en frecuencia colorida entre dos y tres ha estado reducida. Por lo que se asume que una imagen con datos ocultos empotrados tiene frecuencia similar por los coeficientes de DCT adyacentes.

Hay tres sistemas de la Esteganografía populares disponible en el Internet que esconde información en imágenes:

- JSteg, JSteg-cáscara
- JPHide
- OutGuess

Todos estos sistemas usan de alguna forma el BIT menos significativo para esconder el texto, seguidamente se presentan las características específicas de estos sistemas.

- **JPHIDE**

  JPHIDE y JPSEEK son programas que permiten que ocultes un archivo en una imagen. Hay muchos programas similares disponibles en el Internet pero JPHIDE y JPSEEK son algo especiales. El objetivo de diseño no era simplemente ocultar un archivo, sino también, hacerlo de manera tal que sea imposible probar que el archivo contiene un texto ocultado.

  Dado una imagen con un texto oculto, una tarifa baja de la inserción (debajo del 5%) y la ausencia de la imagen original, no es posible asegurar que la imagen retocada contiene datos insertados. Mientras que al aumentar el porcentaje de la inserción, aumenta la diferencia dejando de ser normal hasta el punto de levante la suspicacia. Sobre el 15% los efectos comienzan a llegar a ser visibles al ojo humano. Por supuesto algunas imágenes están mucho mejor que otras cuando están utilizadas un archivo original, la existencia de muchos detalles en la imagen hace que sea más difícil de explorar. Un cielo azul despejado sobre un paraíso cubierto de nieve no es la mejor imagen para ocultar un texto, por el contrario una cascada en un bosque es probablemente ideal.

- **OUTGUESS**

  Es una herramienta automatizada para detectar contenidos esteganográficos en imagines. Es capaz de detectar diferentes tipos de métodos esteganográficos para encajar/incluir información oculta en imágenes. OutGuess es una herramienta esteganográfica universal que permite la inserción de información oculta en los bits redundantes de fuentes de datos.

5.4. Armazón de descubrimiento

La esteganografía deja de cumplir su objetivo cuando se detecta que un fichero contiene información oculta. Esta detección se puede volver bastante laboriosa si se utiliza una estego clave (ver glosario). Por eso se utilizan diversos programas para la detección de posible información encubierta.

Por definición los cambios que realiza la esteganografía sobre las imágenes son imperceptibles para el ojo humano. Por eso la mayoría de los programas que ocultan información lo que hacen es introducir una ligera variación del color respecto al original.

Las imágenes poseen ciertas paletas de colores en las cuales aparecen todos los colores que se están usando. Los programas de detección lo que hacen es crear una lista de los colores que menos se utilizan. Si estos colores son muy numerosos es posible que esa imagen se trate de una estego imagen, y por tanto tenga oculta información.

El hecho de que haya muchos colores "poco usados" induce a pensar esto porque, como se ha dicho interiormente, los programas suelen introducir una ligera modificación en los píxeles de la imagen y en la realidad no se suelen dar muchos casos en que dos píxeles consecutivos difieran en un bit (y coincida en ser el siguiente de la gama de colores).

En la sección anterior, se ha hablado de aplicaciones que nos permiten encontrar mensajes ocultos y determinar qué sistema de Esteganografía fue usado para ocultarlos. Por lo que podría ser interesante nombrar a "Stegdetect," una utilidad automatizada para analizar imágenes y determinar si ocultan un texto y que sistema esteganográfico fue utilizado.

El funcionamiento de Stegdetect consiste en listar los sistemas del Esteganografía encontrados en cada imagen o "negativo" si no detecta la existe de información oculta en la imagen después de haber utilizado algún sistema de detección. Stegdetect expresa el nivel de confianza del descubrimiento con una, dos o tres estrellas.

5.5. Descripción de técnicas esteganográficas

Al igual que el criptoanálisis se usa con unos fines determinados, las técnicas esteganográficas también y si hacemos una analogía con dicho criptoanálisis se observa que un estegoanalista aplica el estegoanálisis para detectar la existencia de información oculta y el resultado obtenido es el estego medio (ver glosario), además el mensaje puede o no estar cifrado.

Una variedad de esta técnica son las *Marcas de agua* (*"Watermarks"*), éstas se utilizan para poner invisible a simple vista en algún fichero, texto plano, imagen, etc. Cualquier tipo de información. Bien conocidas son las marcas de agua de los billetes, visibles a trasluz.

A pesar de que las marcas de agua son un derivado de la esteganografía, la gran diferencia existente entre ellas es la intención con que se realizan cada una. La esteganografía oculta mensajes porque el mensaje oculto es un medio de comunicación. Y las marcas de agua introduce información considerada atributos de la cubierta, es decir, es una forma de hacer un copyright o establecer una información de licencia.

5.5.1 Estego medios

A pesar de que se pueden utilizar distintos estego medios para llevar a cabo la función de la esteganografía, como pueden ser sonidos, vídeo, texto, programas ejecutables..., el medio más utilizado es el fichero gráfico, por ello a continuación se van a ver algunos formatos de este tipo de ficheros.

A pesar de que cada tipo de formato se pueda utilizar para una cosa concreta todos ellos tienen características comunes. Éstas son las siguientes:

- Contienen una cabecera que identifica el tipo de fichero del que se trata, como puede ser el tamaño de la imagen o el número de colores, que contiene información para interpretar dicho fichero.

- Una vez descomprimidos los datos de un fichero mediante el algoritmo concreto de cada formato, los datos del fichero indican el color específico de cada píxel de la imagen.

- En función de los colores de la imagen se utilizarán más o menos bits que indicarán las cantidades de rojo, verde y azul que se utilizará para representar cada color en cada píxel. Debido a que la paleta de colores puede ser modificada según la imagen, es necesario almacenarla en un fichero. Un caso particular es utilizar los 16 millones de colores, en este caso no se utiliza la paleta de colores ya que la relación entre número de color y cantidad del mismo es implícita. De los 24 bits de cada pixel, se utilizan 8 bits por color, es decir, 8 bits para el rojo, 8 para el azul y 8 para el verde.

Así cada fichero contendrá una cabecera, los datos de los pixeles, que pueden estar comprimidos o no, y una paleta de colores, a excepción de que se usen los 16 millones de colores, es decir, 24 bits por píxel.

- **Los formatos más utilizados son los siguientes:**

BMP (Windows BitMaP)

Consiste en una cabecera y a continuación los valores de cada píxel siguiendo un orden de abajo hacia arriba, es decir, desde la última línea hasta la primera, y de izquierda a derecha. Permite compresión y una ventaja es la sencillez en contraposición al gran tamaño de los ficheros.

TIFF (Tagged-Image Format)

Es un formato flexible admitido por la mayoría de las aplicaciones de pintura, edición de imágenes, diseño de páginas y plataformas de ordenadores. Y a veces también permiten compresión.

PCX (PC Paintbrush)

Es un formato que utiliza el algoritmo RLE para la compresión de los datos. Suponiendo que usemos 1 byte por píxel, consiste en sustituir las secuencias de N pixeles consecutivos del mismo color por dos bytes (dependiendo del número de colores se utilizarán más o menos bytes), de estos dos bytes el primero indicará el número N de veces que se repite el color y el segundo indicará el color.

Así este algoritmo nos permite reducir el tamaño, de una forma considerable, del archivo siempre y cuando la imagen sea un dibujo donde muchos pixeles consecutivos sean del mismo color. Sin embargo cuando haya pixeles consecutivos de colores distintos, el tamaño del archivo puede aumentar de forma notable.

GIF (Graphic Interchange Format)

Utiliza el algoritmo de compresión LZW. Consiste en detectar, tanto las repeticiones de un color como las repeticiones de ciertas secuencias de forma consecutiva a partir de una especie de diccionario que se va construyendo. Como en el caso anterior funciona especialmente bien con dibujos, pero también es efectivo con fotografías escaneadas o cualquier otra imagen. Algo particular es que permite definir un color transparente, lo que lo hace muy útil en páginas web.

JPG (Joint Photographic Experts Group)

Este formato utiliza un complejo algoritmo de compresión de datos con pérdida de información, es decir, se modifican los datos ligeramente haciendo que la imagen se comprima mucho más sin que apenas se aprecien variaciones. Esto es verdad siempre y cuando la imagen sea una fotografía escaneada, ya que si es un dibujo aparecerán puntos visibles en la imagen.

5.5.2. Métodos utilizados

Debido a la existencia de una gran diversidad de formas de llevar a cabo este “arte”, los programas utilizados se pueden dividir en dos grandes grupos según el método utilizado:

- Basados en el dominio de la imagen (*Image domain*) Éstos abarcan métodos que aplican la inserción en los bits menos significativos de cada pixel o en la manipulación del ruido.

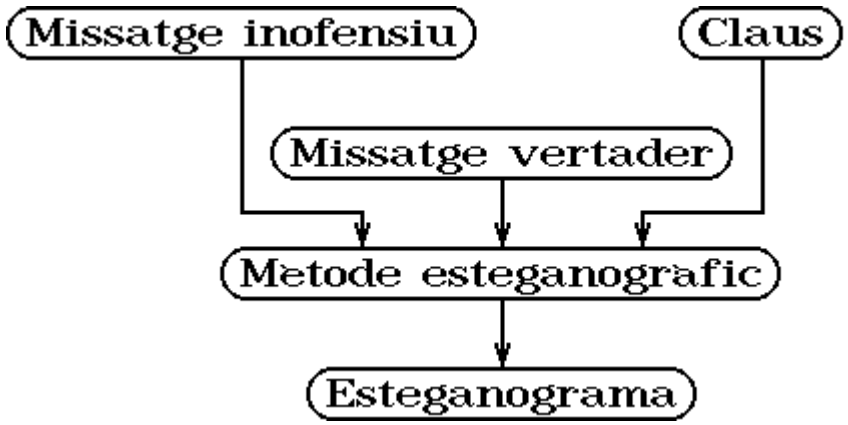
- Basados en el dominio de la transformación (*Transform Domain*) Éstos otros están relacionados con algoritmos de modificación y transformación de la imagen. Ocultan los mensajes en las áreas más significativas de la tapadera y pueden manipular las propiedades de la imagen como es la luminosidad.

Normalmente se crean máscaras indicando los lugares más idóneos para la ocultación de información. Estas técnicas son más complejas de realizar y más robustas que las anteriores.

5.5.3. Como se caracterizan y en qué se basan los métodos esteganográficos

Los métodos esteganográficos se caracterizan por la utilización de:

- Un mensaje que se quiere transmitir.
- Una información inofensiva donde esconder el mensaje a transmitir.
- Una o más claves secretas o públicas que son distribuidas esteganograficamente entre los que tengan acceso al mensaje.



Y como resultado devuelve una información parecida a la información inofensiva utilizada, pero que lleva escrito el mensaje que se quiere transmitir. Los métodos esteganográficos se aprovechan de la información desaprovechada que puede haber en un mensaje para ocultar la información, como por ejemplo utilizar los bits menos significativos de una imagen porque a la hora de variar estos, la imagen original varía muy poco.

El problema de esta característica de la esteganografía es que en consecuencia se encuentra muy influenciada por mecanismos externos que pueden modificar el mensaje original, manteniendo la información del paquete muy parecido al anterior, es como el ruido en las comunicaciones. Pero así y todo existen métodos que si al final de la transmisión no se ha perdido ninguna parte del mensaje se puede reconstruir el mismo. Esto se consigue introduciendo, estratégicamente, información de más en el mensaje final.

A la hora de diseñar un método de esteganografía, de la misma manera que cuando se quiere diseñar una de criptografía, se parte del conocimiento, por parte de una tercera persona no autorizado para obtener un mensaje secreto, de todos los elementos anteriormente citados menos de la clave secreta. Las claves pueden haber sido utilizadas en más de un mensaje, y por tanto las consecuencias de conocer las claves son más críticas que cualquier otro elemento.

Lo que se pretende con los métodos de esteganografía es que sea fácil ocultar un mensaje en un determinado fichero con una clave, y el proceso inverso, es decir, recuperarlo. Pero en cambio que sea muy difícil a partir de todos los elementos necesarios menos las claves, obtener el mensaje oculto.

Los métodos esteganográficos tienen su base, de la misma manera que los criptográficos, en la utilización de problemas matemáticos que permiten todas las características anteriormente especificadas de este método, como el caso de la teoría de nombres.

6. INICIO DE DISEÑO

La estructura a llevar a cabo para el diseño del programa esta pensada de la siguiente forma:

- El programa tendrá dos partes, la primera para ocultar un texto dentro de la imagen para proteger la misma, y la segunda parte para extraer el texto de la imagen.
- Ambas partes ejecutables del programa estarán protegidas con una clave, la cual se introducirá a la hora de proteger la imagen y deberá utilizarse para obtener la información oculta desde la segunda parte.
- La ejecución de ambas partes se efectuara a través de un comando en el que el primer parámetro pasado determinará la acción.
- Los tres parámetros que se le pasan para crear la protección serán, el identificador de protección en este caso –e, la dirección del fichero a ocultar en la imagen, y la imagen a proteger y en la que se ocultará el texto codificado en binario, la clave se pedirá en el momento de la ejecución.
- Para extraer el texto, al comando se le pasará dos parámetros, el identificador de decodificación –d en este caso, y la dirección de la imagen y preguntará la clave, la cual comprobará y de ser la correcta, mostrará el texto.
- La clave también estará oculta en formato binario dentro de la imagen y se tendrá que recuperar para comprobar que es la correcta.

El programa que oculta el texto codificado en la imagen, primero que todo pasará el texto a binario. Teniendo en cuenta la cantidad de caracteres posibles, cada carácter quedará oculto en cadenas de 8 bits. La clave también será ocultada dentro de la imagen de la misma forma. Por otra parte la cantidad de caracteres de la clave podrá ser variable, y existirá un separador entre la clave, los datos del fichero y dirección del mismo a codificar para poder encontrar la clave al descodificar la imagen. La descripción de los métodos de ocultación se verá más adelante.

Para extraer el texto se verificará que la clave que se pasa coincide con la que esta oculta en la imagen, y de coincidir, se pasa a la extracción del texto de la imagen, lo cual consiste en extraer todos los bits, formar la cadena de bits con el orden correcto y convertirlos en caracteres, imprimiendo el texto resultante en un fichero en la carpeta especificada, la clave y el texto pueden utilizar o no el mismo codificador y el mismo sistema de ocultación.

Este inicio de diseño es básico. Cumple con las normas de ocultación y recuperación de texto en una imagen, pero no tiene en cuenta la robustez de la aplicación. Más adelante se verá más detalladamente las diferentes cuestiones a tener en cuenta para la robustez y como lograrlo. También se podrá ver los fallos de la aplicación y las posibles soluciones.

6.1. Ocultación de bits

Existen miles de maneras de incluir un mensaje, sonido o imagen dentro de un fichero, pero los métodos cabían mucho en función del tipo de archivo que nos servirá de cubierta. Hoy en día existen algoritmos para hacer esteganografía en muy diversos formatos de gif, bmp, jpeg mp3, wav, mpeg...

Todas ellas utilizan algoritmos más o menos parecidos, pero en casi todos los casos bastante limitados, no se puede ocultar una Biblia entera en una foto de familia.

De forma básica para incluir un mensaje en un archivo BMP, se necesitarán una foto y un mensaje, el cual quedará oculto en la foto. Teniendo en cuenta que la foto es de 200 x 400 píxeles, y como el formato cada píxel es un byte, es decir una cadena de 8 Bits algo como (00110101), la imagen se construye con una matriz con todos estos píxeles (recordemos que este formato no contempla compresión de datos).

Con lo cual nos podemos imaginar que una porción del código podría ser algo así:

```
...
00010101 10100101 01010101 00110101 01110101 01000010 01010011 01101010
00001011 01010101 10100101 01010111 11010111 10000101 01010010 01010010
10101001 10101011 00001001 10100100 00010001 10100101 00010101 10100101
...
```

Cada byte es un color, y donde esta el truco... todos sabemos que si "retocamos" ligeramente los colores (o sea con colores parecidos), la imagen final no se vera afectada, y también sabemos que el bit menos significativo (el que esta mas a la derecha de cada byte) apenas alterara el resultado final de la imagen, o sea que cogeremos de cada byte el LSB (less significant bit), o sea el que esta mas a la derecha.

El mensaje que vamos a incluir es "SET", como mínimo para incluirlo, se necesitan tres octetos, uno por cada letra. Los caracteres que necesito para representarlo en binario son estos tres: 73 65 74 que en binario corresponden con estos tres bytes: 01001001 01000001 01001010

Por lo tanto, se retoca la imagen cambiando el último bit de cada byte):

```
00010100 10100101 01010100 00110100 01110101 01000010 01010010 01101011
      -      -      -      -      -      -      -      -
00001010 01010101 10100100 01010110 11010110 10000100 01010010 01010011
      -      -      -      -      -      -      -      -
10101000 10101011 00001000 10100100 00010001 10100100 00010101 10100100
      -      -      -      -      -      -      -      -
...
```

De esta forma en la imagen no se notarán cambios y en esta estará escondido el texto.

7. DISEÑO

Aunque en el punto anterior se ha explicado la base del mecanismo que se utiliza para la ocultación de texto en una imagen. Nuestro diseño haciendo uso de las técnicas ya expuestas, tanto para la ocultación como para la recuperación del texto, y basándose en una serie de directrices del formato de las imágenes BMP o BitMaP, desarrolla un diseño algo más complejo y estudiado.

7.1. Sistema de ocultación

El sistema de ocultación utiliza y verifica dentro del programa, que la imagen utilizada sea la adecuada. Para esto verifica en la cabecera del formato de la imagen, la existencia de unas determinadas marcas. El proceso en general es el siguiente:

- Primero se crea una matriz de bits con todos los pixels que tiene la imagen. Como formato de la imagen que se ha utilizado en el diseño es BMP, se deben tener en cuenta una serie de cuestiones:

    1. El formato BMP (Windows BitMaP) es probablemente el formato de fichero para imágenes más simple que existe. Aunque teóricamente permite compresión, en la práctica nunca se usa, y consiste simplemente en una cabecera y a continuación los valores de cada pixel, comenzando por la línea de más abajo y terminando por la superior, pixel a pixel de izquierda a derecha.

    2. Un mapa de bits es un formato de imagen digital que describe la imagen mediante una cuadrícula de valores de color. Cada celda de la cuadrícula representa un píxel. Cada píxel se dibuja mediante un procesador de representación con un valor de color determinado para formar una imagen. Los mapas de bits se almacenan mediante un número hexadecimal de 32 bits. Un número hexadecimal de 32 bits es una secuencia de cuatro pares de dígitos hexadecimales. Cada par hexadecimal define la intensidad de cada uno de los cuatro canales de color (rojo, verde, azul y alfa). La intensidad de un canal de color es una representación hexadecimal de un número decimal comprendido entre 0 y 255; FF es intensidad total (255), mientras que 00 equivale a ausencia de color en el canal (0). Para comprender el formato de los bits de un bitmap o imagen BMP puede dirigirse a la siguiente dirección Web: http://www.adobe.com/es/devnet/flash/articles/image_api_02.html

    3. El formato BMP al principio de su cabecera guarda los byte B y M de lo contrario no es un fichero BMP versión 3.0 que es para el que trabaja este programa. Tengase en cuenta que **su profundidad de color puede variar, puede ser por norma general de 1, 4, 8, 24 o 32 bits por pixel. Esto indica el número de bits necesarios para almacenar cada pixel en el archivo.** En este diseño nos referimos a los BitMaps de 32 bits.

- Una vez creada la matriz de bits, teniendo en cuenta el formato de la imagen y que cada píxel ocupa 32 bits de los que solo utilizaremos 24 bits y tendremos que acomodar los bits en la matriz invirtiéndolos para solo utilizar los bits pertenecientes a los colores básicos, RGB. Se procede a la codificación de la cadena de caracteres en la que se encuentra el texto que esta dentro del fichero, la clave y todo aquello que sea útil a la hora de recuperar el texto.

- Para el diseño se ha utilizado un formato largo de carácter, por lo que cada carácter ocupa dos byte, y los 16 bits que los conforman son extraídos de tres en tres y guardados de izquierda a derecha en el bit menos significativo de los tres colores básicos que conforman cada píxel del BitMaP. Es decir, los primeros 15 bits que conforman un carácter se guardan en los 5 primeros pixels de la imagen, y en el sexto píxel se guarda el último bit del carácter, en la posición más a la izquierda quedando los dos más a la derecha sin tocar.

La siguiente imagen muestra como se distribuyen los 16 bits que conforman un carácter en los 6 píxeles que utiliza, cada bit del carácter en el bit menos significativo de los 8 bits utilizados dentro del formato del píxeles para los tres colores bases.

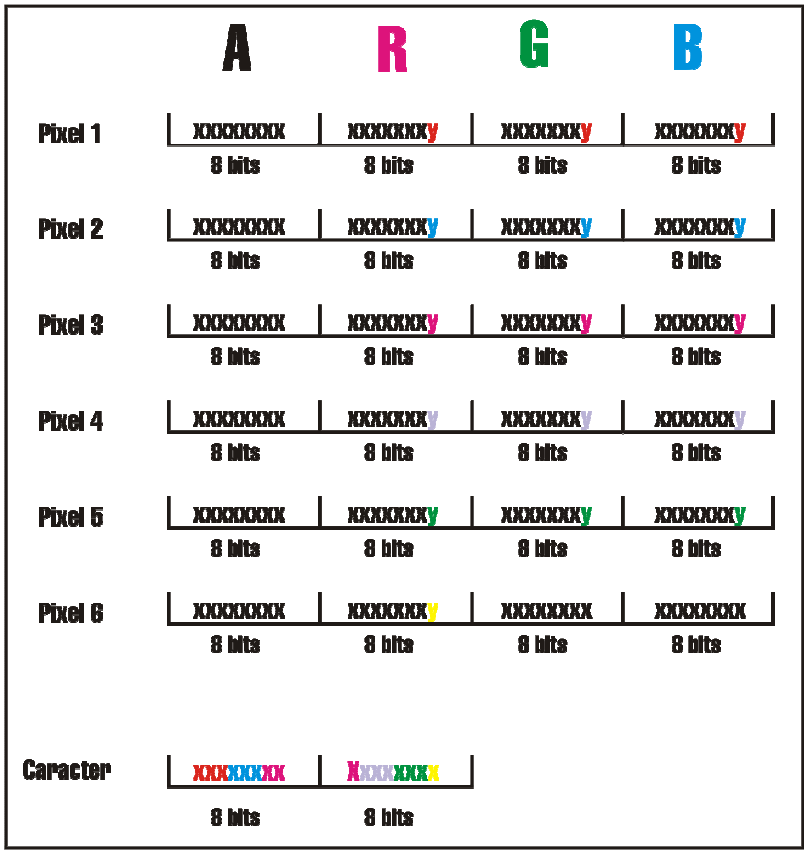


Imagen 1 – *Distribución de una carácter dentro de la imagen.*

Como se explico anteriormente y para explicar detalladamente la imagen anterior, la forma en que cada carácter del texto es guardado en la imagen es la siguiente.

- Se dividen de tres en tres los 16 bits del carácter, quedando 5 juegos de tres bits y un sexto juego de un solo bit. En la imagen se pueden ver por colores en la descripción del carácter.

- Los tres primeros bits se distribuyen en el bit menos significativo de cada uno de los tres bytes mas a la derecha de los cuatro que conforman el píxel. La posición en la que se ponen estos bits se pueden ver en la figura con el color rojo.
- Los siguientes tres bits del carácter y así hasta llegar al 5 juego de tres bits que conforman los primeros 15 bits de los 16 que tiene el carácter, quedan distribuidos como se puede ver en la imagen según los colores.

- Por último el ultimo bit del carácter quedará en el bit menos significativo del segundo byte de izquierda a derecha del píxel número 6, el tercer y cuarto bits del píxel quedan igual.

Este procedimiento se lleva a cabo carácter a carácter, de toda la cadena de caracteres que se debe esconder en la imagen.

7.2 Sistema de recuperación

El sistema de recuperación, además de comprobar que la imagen evaluada es la adecuada, verifica en la cabecera del formato de la imagen la existencia de unas marcas, no las que identifica a la imagen, sino la que identifica que se trata de una imagen que esconde un contenido y evitar el intento de extraer mensajes en imágenes que no ocultan texto alguno. Para esto ya anteriormente cuando se oculto el mensaje en la imagen, se escribió en la cabecera las marcas correspondientes.

Este proceso de recuperación del texto sigue los siguientes pasos:

- Verifica al igual que en el proceso de ocultación que se trata de una imagen con el formato adecuado. Esto lo hace buscando las marcas que identifican a la imagen en la cabecera del formato.

- Se crea una matriz de bits con todos los píxel que tiene la imagen. Recordar las características del formato BMP y que cada píxel esta formado por 32 bits, de los que utilizamos 24 bits para la ocultación y de estos debemos recuperar los bits adecuados para la extracción del texto.

- Antes se comprueba que la imagen a la que se le aplica el algoritmo esconde algún texto. Para esto se busca en la cabecera las marcas que se pusieron en esta cuando se efectuó el proceso de ocultación. Si estas marcas no se encuentran, es debido a que la imagen no posee ningún texto oculto.

- Una ves verificada la imagen y creada la matriz de bits de todos los píxeles de la imagen, girados y puestos en la posición correcta para la correcta extracción de los bits, se pasa al proceso de selección de los bits adecuado y la construcción de los caracteres siempre que se recuperen los 16 bits que los conforman.

- Los caracteres recuperados se guardan en la misma carpeta en la que se encuentra la imagen en un documento con el mismo nombre que la imagen con la diferencia de que el formato es TXT, el cual puede leerse con cualquier editor de texto. Para esto el programa utiliza la dirección que se pasa como argumento para saber donde y con que nombre debe guardar el documento en el que se inserta el texto.

- Terminado todo el proceso el programa nos avisa de la terminación de la extracción del texto y la dirección donde lo ha guardado.

8. DISEÑO DE SOFTWARE

La aplicación que ha sido diseñada sobre la plataforma java, esta compuesta por 5 clases. Entre estas clases existen relaciones, ya que se necesitan entre ellas para llevar a cabo el proceso de ocultación y recuperación del texto. La estructura y as dependencias de estas clases es la siguiente:

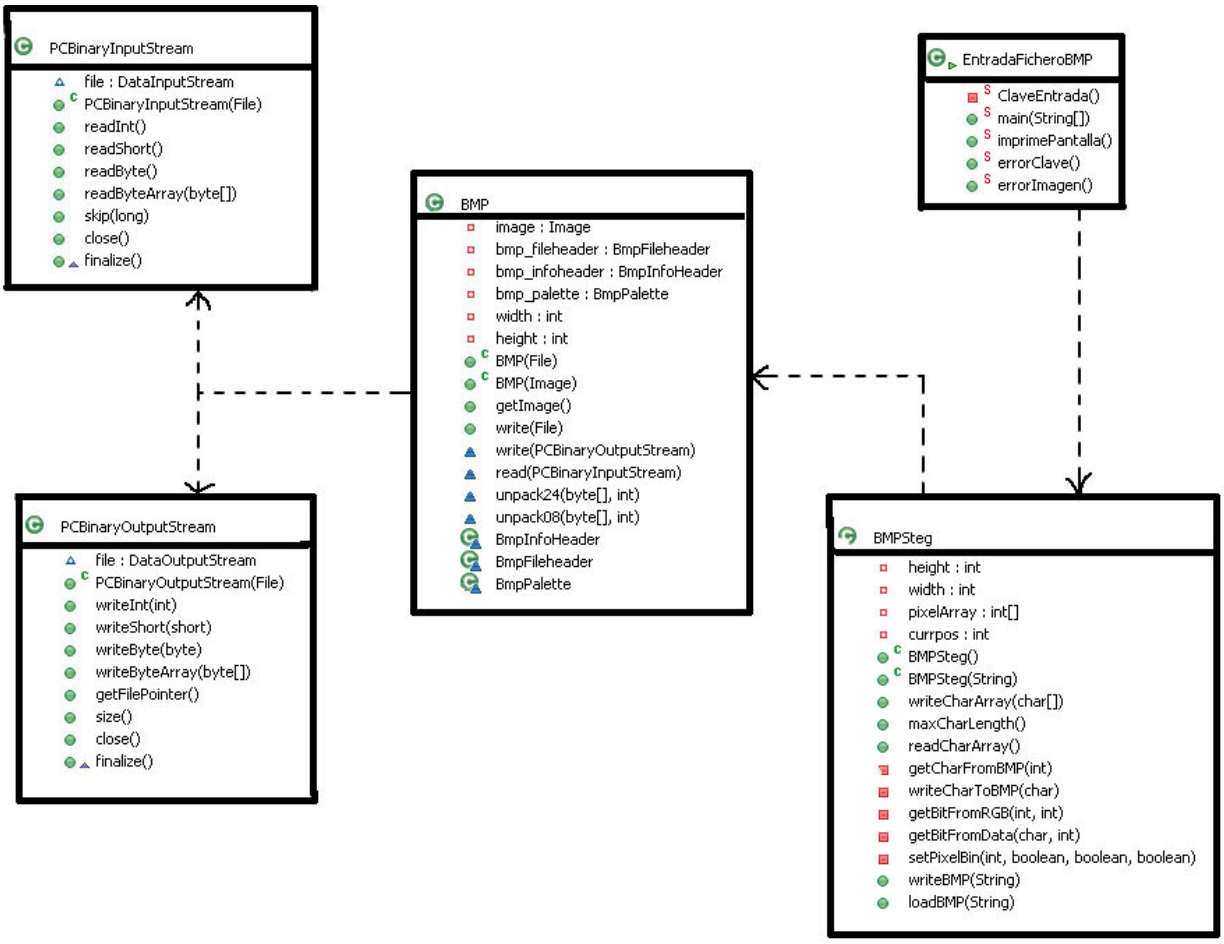


Imagen 2 –*Diseño de clases..*

En este diagrama de clases podemos ver las diferentes funciones propias de cada clase, las cuales se utilizan para la ejecución de la aplicación, y que no se explicarán detalladamente una a una, pero si es interesante conocer el desglose del caso de uso. Este puede ser muy amplio si lo hacemos siguiendo las clases, pero como las funciones principales de la aplicación son dos, nos detendremos en ellas ya que son la base para una correcta ejecución de la aplicación:

Nombre:	Ocultar un mensaje
Autor:	Victor Navarro Naranjo
Fecha:	01/05/2006

Descripción:
Permite ocultar un mensaje de texto en una imagen.

Actores:
Usuario en posesión del software.

Precondiciones:
La Imagen debe ser formato un BitMaP

Flujo Normal:

1. El actor ejecuta el siguiente comando desde DOS esteganografia –e [direcc del fichero de texto] [Direcc de la imagen donde ocultarlo].
2. El sistema pide una clave la cual será utilizada en la recuperación del texto.
3. El actor introduce la clave pactada con el receptor.
4. El sistema oculta el texto.

Flujo Alternativo:

4. El sistema comprueba la validez de los datos, si los datos no son correctos, se avisa al actor de ello.

Poscondiciones:
El mensaje ha sido escondido en la imagen.

Nombre:	Recuperar un mensaje
Autor:	Victor Navarro Naranjo
Fecha:	01/05/2006

Descripción:
Permite recuperar un mensaje de texto de una imagen.

Actores:
Usuario en posesión del software y de la clave pactada a la hora de la ocultación.

Precondiciones:
La Imagen debe ser formato un BitMaP y estar marcada como que oculta un mensaje.

Flujo Normal:

5. El actor ejecuta el siguiente comando desde DOS esteganografia –d [Direcc de la imagen donde esta oculto el texto].
6. El sistema pide la clave utilizada en la ocultación del texto.
7. El actor introduce la clave pactada con el usuario que oculto el texto.
8. El sistema recupera el texto dejándolo en un fichero .txt en la misma carpeta donde esta la

imagen.
Flujo Alternativo: 5. El sistema comprueba la validez de los datos, si los datos no son correctos, se avisa al actor de ello.
Poscondiciones: El mensaje se ha recuperado con exito.

Una forma de verlo con más claridad es a través del diagrama de caso de uso que tenemos en la imagen 3. En la imagen siguiente se puede observar como el usuario puede ocultar un texto o recuperarlo, o ambas cosas la vez. Lo más importante es que el emisor intercambie la clave con el receptor, de esta forma este podrá recuperar el texto.

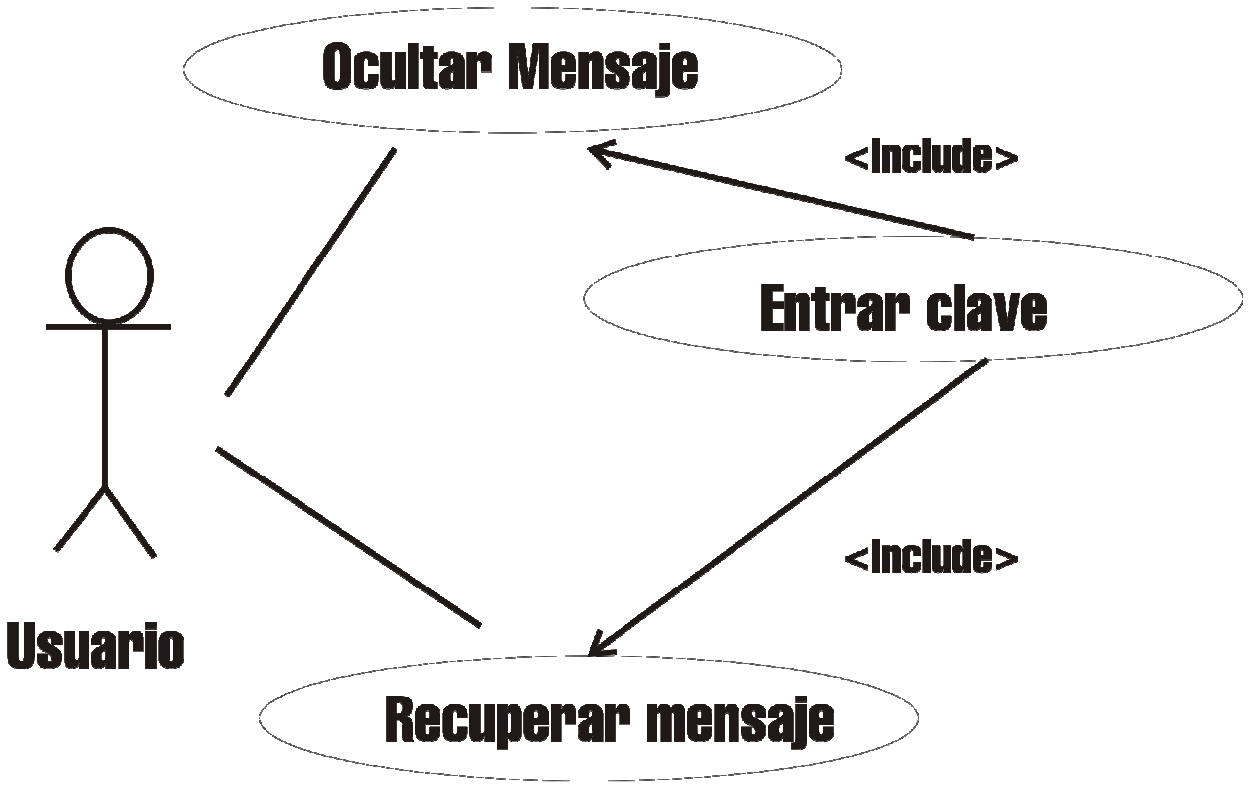


Imagen 3 –*Caso de uso.*

9. UN DISEÑO ROBUSTO

Partiendo de las cualidades y las necesidades antes descritas que debe cumplir la aplicación en cuanto al apartado de Robustez, es de destacar, que la aplicación que acompaña a este proyecto no las cumple. Lograr un sistema robusto es bastante complicado, y puede representar una perdida en la seguridad de la ocultación que en el fondo es lo que se desea.

Nuestra aplicación es capaz de ocultar un texto en una imagen y recuperarlo luego, pero siempre partiendo del punto en el que la imagen no sufre ningún cambio.

Se pueden dar muchos casos de modificación de la imagen, en el punto siguiente se hablará de las pruebas hechas con la aplicación y los diferentes errores que devuelve al intentar recuperar el texto. Pero antes detallare las diferentes posibles soluciones para acercarnos a las cualidades que debe tener una aplicación esteganográfica robusta.

9.1. Girar la imagen

En el siguiente ejemplo debemos tener en cuenta que sucede con la imagen cuando se escribe un texto en ella. La imagen en la que se escribe es marcada en la cabecera, lo cual, al necesitar recuperar el texto de la imagen, lo primero que se hace es un chequeo a ver si la imagen esta marcada. De esta forma si en la cabecera no existe la marca, se deduce que la imagen no oculta un texto, al menos con la aplicación esteganográfica diseñada.

Al girar la imagen, los patrones de la cabecera cambian, por lo que la imagen pasa a no estar marcada como imagen con un texto oculto. Por tal motivo, al intentar recuperar el texto, la aplicación de un error. Una posible solución a este problema, y quizás la más convincente, es que cuando la aplicación de recuperación de texto se ejecute, haga el chequeo en busca de la marca que indica que tiene un texto oculto, en los cuatro lados de la imagen. Para esto sería suficiente que la aplicación fuera rotando la imagen hasta llegar al punto inicial.

Pruebas echas con la aplicación han demostrado que la imagen puede girarse todas las veces que se desee, y que si al querer recuperar el texto esta en la posición en la que se oculto el texto, este se recupera sin ningún problema.

Otra posible solución y que podría servir para solucionar otras dificultades de robustez, pasaría por escribir el texto por los cuatro lados de la imagen, para lo cual al ejecutar la aplicación para ocultar un texto, esta debería una vez oculto el texto en una cara de la imagen, girar la imagen y volver al paso inicial de ocultación. Esto lo haría 4 veces, pero es bueno destacar que es un mecanismo mucho más engorroso.

9.2. Cortar la imagen

Hacer un corte en una imagen es algo común entre los aficionados al diseño, con el objetivo de crear montajes, acercar fotos y extraer objetos o personas para crear una imagen mucho más personalizada. Vale la pena destacar que de la misma forma que se hace un corte, también se pueden agregar nuevas figuras.

Existen muy pocos sistemas esteganográficos que funcionen al efectuar un corte en la imagen, y esto es debido a que utilizan, al igual que la aplicación aquí diseñada, el bit menos significativo de cada bytes que conforman un píxel, para ir ocultando los diferentes bits que forman parte de la definición en binario de cada carácter.

Al hacer un corte en la imagen, se pierden una cantidad de píxeles, con sus bytes correspondientes. Y al ejecutar la aplicación de recuperación, da un error debido a que detecta, al hacer el seguimiento bit a bit que falta información.

Las soluciones para lograr la robustez en la aplicación dado el caso anterior, podría pasar por insertar en la imagen todas las veces que se pueda el texto a ocultar. De esta forma aunque se haga un corte a la imagen, este tendría que ser capaz de recuperar el texto.

Otra posible solución podría estar en escribir en la imagen en los cuatro lados, igual que en el caso anterior. Por lo cual aunque hagan un corte a un lado de la imagen, con un sistema de rotación de imagen se chequean los cuatro lados y

9.3. Cambio de colores en la imagen

El cambio de colores, es otro de los cambios que rompe con la posibilidad de recuperar el texto. Y es que una alteración en el color, puede afectar al bit menos significativo de los bytes que forman la imagen. Por lo que el texto se perdería.

Una posible solución pasaría por ocultar el texto en la cabecera de la imagen, pero solo podrían ser textos muy pequeños como el encabezado

9.4.

10. PRUEBAS Y FUNCIONAMIENTO

Se ha decidido realizar 2 tipos de pruebas básicas a la aplicación, pruebas de funcionalidad, para comprobar que la aplicación cumple con los objetivos iniciales, y pruebas de robustez, para probar la reacción de la aplicación ante los ataques y las alteraciones en la imagen.

10.1.Prueba funcionalidad

Al inicio del proyecto se determinó que la aplicación debería ofrecer los siguientes servicios:

- Ocultación de un texto dentro de una imagen, sin que la imagen sufra cambios detectables por el ojo humano.
- Recuperación del texto extrayéndolo de la imagen sin que se pierda información.
- Buscar que el texto pueda recuperarse aunque la imagen sufra alteraciones (Robustez).

A continuación pasamos a valorar si la aplicación ofrece, de forma satisfactoria todos los servicios.

El funcionamiento de la aplicación es muy sencillo. Para su ejecución basta con abrir una consola con el comando CMD haciendo clic en Inicio/Ejecutar, y según sea el caso, es decir se trate de una ocultación o de la extracción del texto oculto se utilizan el siguiente comando:

Ocultación del texto en la imagen .BMP:

- esteganografia –e [direcc del fichero de texto] [Direcc de la imagen donde ocultarlo]

Donde en el primer conjunto de corchetes esta la dirección del fichero de texto que guarda el texto a esconder en la imagen, y en el segundo par de corchetes se encuentra la dirección de la imagen que se va a utilizar para ocultar el texto.

Recuperación del texto de la imagen .BMP:

- esteganografia –d [Direcc de la imagen donde esta oculto el texto]

Donde en el conjunto de corchetes esta la dirección de la imagen que esconde el texto a recuperar.

Después de hacer las comprobaciones pertinentes como que la imagen que se utiliza es un BitMaP y en el caso que el comando que se utilice sea el de recuperación de texto, que la imagen de la que se quiere extraer el texto es la correcta, el programa pide una clave, esta clave que se utiliza tanto para ocultar el texto, como para extraerlo puede ser cualquiera que se defina cuando se oculta el texto, pero debe ser la misma para recuperarlo.

Como esta definido, la aplicación se ha creado en Eclipse, por lo que las pruebas de compilación se harán sobre esta plataforma. Primero compilamos el programa y ejecutamos el comando de ocultación, esto se puede ver en la imagen 4 donde el nombre de la clase que se ejecuta es EntradaFicheroBMP y en la que existe una función especial que es la función main a la que se le pasa como argumento el comando de ocultación. Esto podemos verlo en la imagen 4.

Luego nos pide la clave, para la cual utilizaremos “esconder”, se puede ver en la imagen 5. Y seguidamente el programa oculta el texto en la imagen, y devuelve un mensaje donde avisa de la correcta inserción y ocultación del texto en el BitMap. Este resultado se puede observar en la imagen 6.

Para la recuperación del texto, solo se necesita la imagen. Al ejecutar el comando de extracción antes mencionado, el programa comprobará que se trata de una imagen de tipo BitMap, y que esta marcada en la cabecera, lo cual significa que oculta un texto. Seguidamente, pide la clave que permitiría recuperar el texto a aquellas personas que la posean. La calve esta guardada también en la imagen, por lo que una vez se teclea la clave, esta es comprobada con la que se extrae de la imagen y si es la correcta, se genera un documento en la misma carpeta donde esta la imagen, en el cual se podrá leer el mensaje oculto.

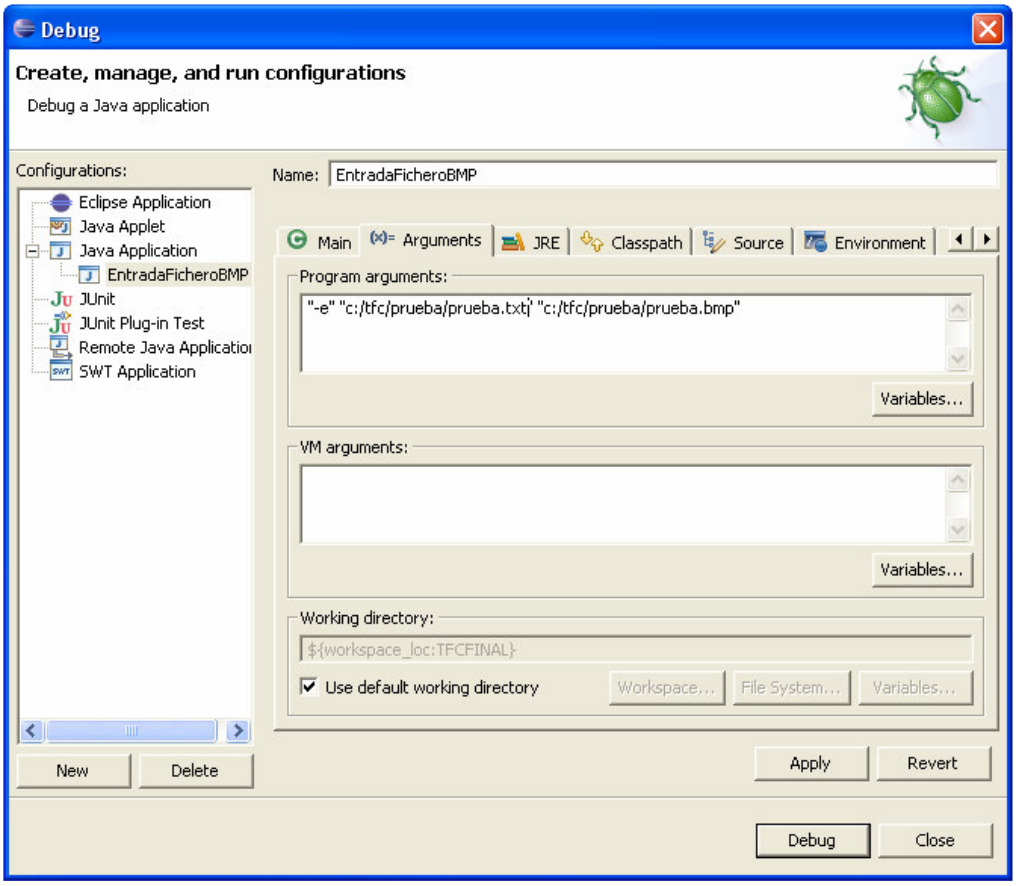


Imagen 4 –*Compilación de ocultación.*

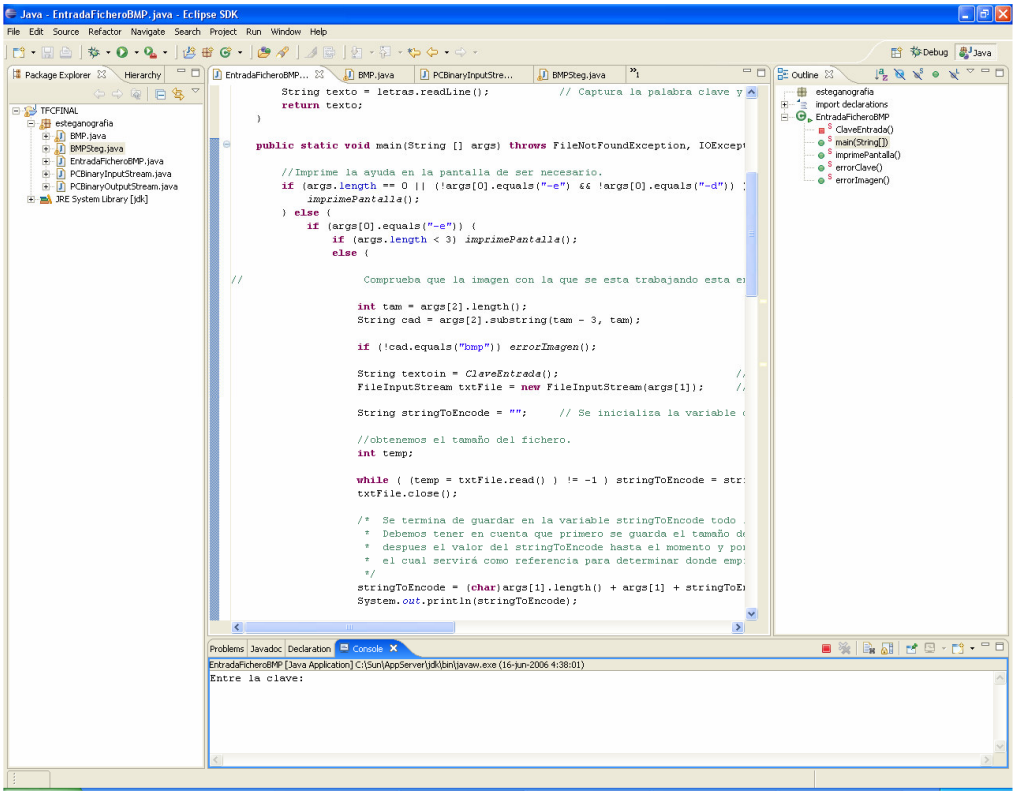


Imagen 5 –*Compilación de ocultación pedido de clave.*

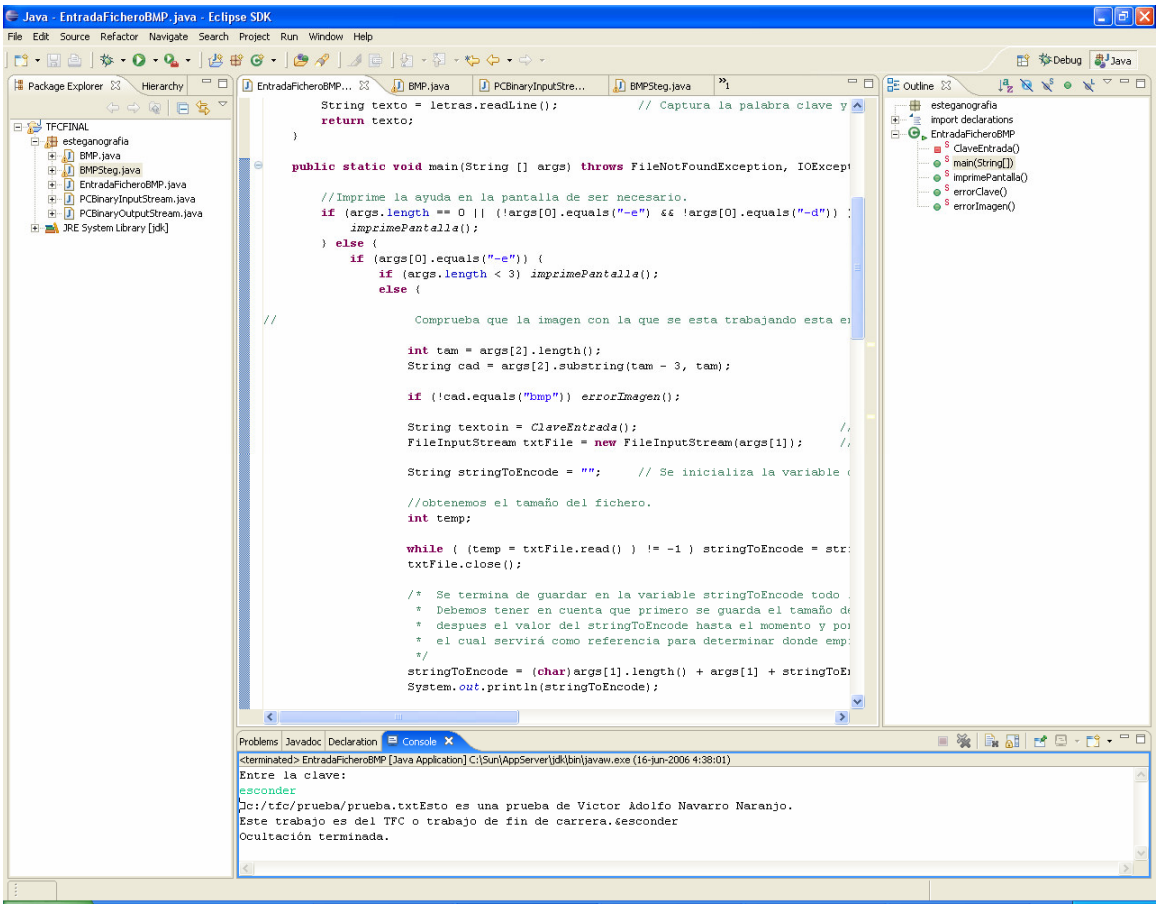


Imagen 6 –*Compilación de ocultación fin.*

10.2.Prueba de Robustez

Como comentaba en un punto anterior, la robustez en la estenografía no siempre es deseada. De hecho muchos sistemas esteganográficos prefieren la perdida de información, antes de la Robustez de la aplicación. Ya que la destrucción del mensaje cuando se quiere recuperar este a la fuerza es más seguro. A medida que se intenta ocultar el texto de tal forma que se pueda recuperar a pesar de los cambios que puede haber sufrido el mensaje, da lugar a una menor seguridad en la aplicación.

Esta aplicación no es lo suficientemente robusta, pero veremos los diferentes mensajes de error que se producen al probarla:

10.2.1. Al girar una imagen que posee un texto oculto:

El comportamiento de la aplicación al recuperar el texto de una imagen en la que se oculto un texto anteriormente y luego se giró cambiando la posición, se puede ver en la imagen 7. Sencillamente no encuentra la marca, ya que la cabecera también cambia al girar la imagen y devuelve un mensaje donde dice que no existe ningún texto oculto en la imagen.

Esto puede ser interesante, ya que se puede voltear la imagen antes de ocultar el texto, y ocultar el texto estando la imagen de cabeza. Luego se pone en la posición original, y aunque se tenga la aplicación y la clave, si no se sabe que se debe voltear la imagen, no se tendrá acceso al texto oculto.

La solución a este problema de robustez ya fue expuesta anteriormente en el punto **9.1**.

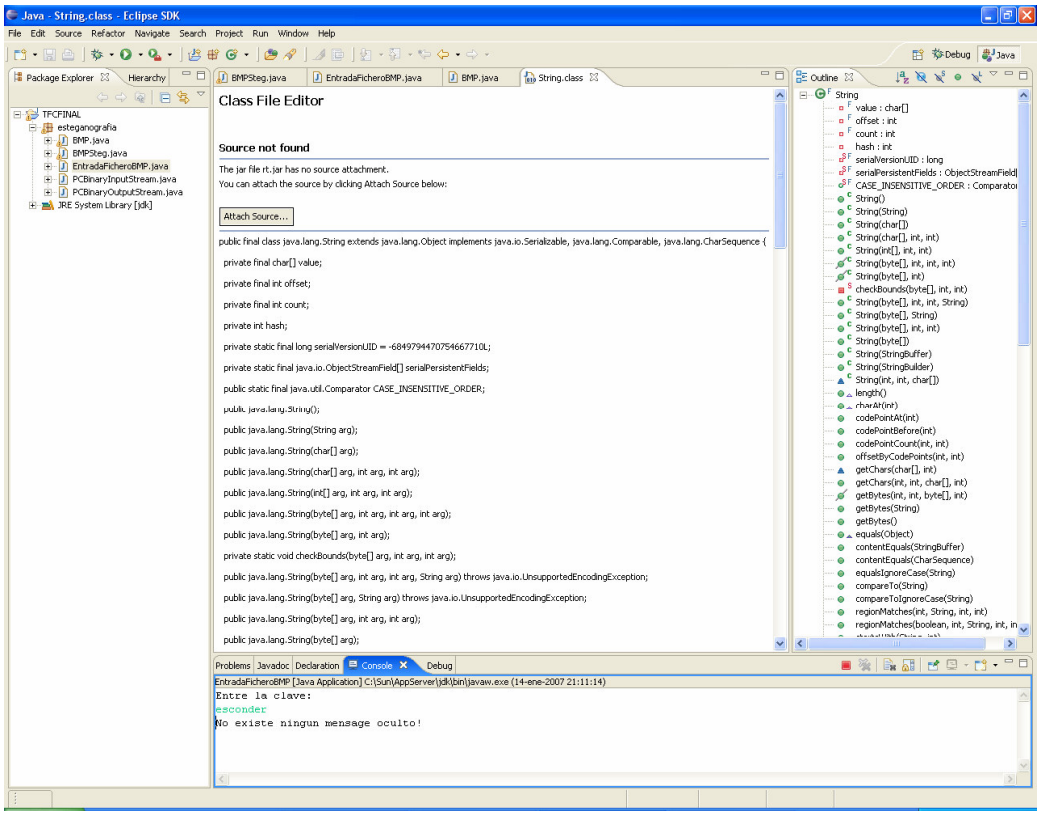


Imagen 7 –*Error al girar la imagen.*

10.2.2. Al cortar un trozo de la imagen:

Para esta prueba tomaremos la imagen y haremos una copia. Luego en un programa de edición de imagen le cortaremos un pequeño trozo y la guardaremos.

Luego ejecutamos el comando de recuperación y este pide la clave, lo cual quiere decir que ha detectado que la imagen posee un texto oculto. Una vez tecleada la clave, si el corte no ha afectado a los bits donde se oculto la imagen, esta es comprobada con la que se oculta en la imagen, y continua con el proceso de recuperación. Pero falla inesperadamente, ya que no puede recuperar todo el texto. El resultado de esta prueba se puede ver en la imagen 8. Igual que en el punto anterior, de las posibles soluciones a este problema se ha hablado en el punto 9.2.

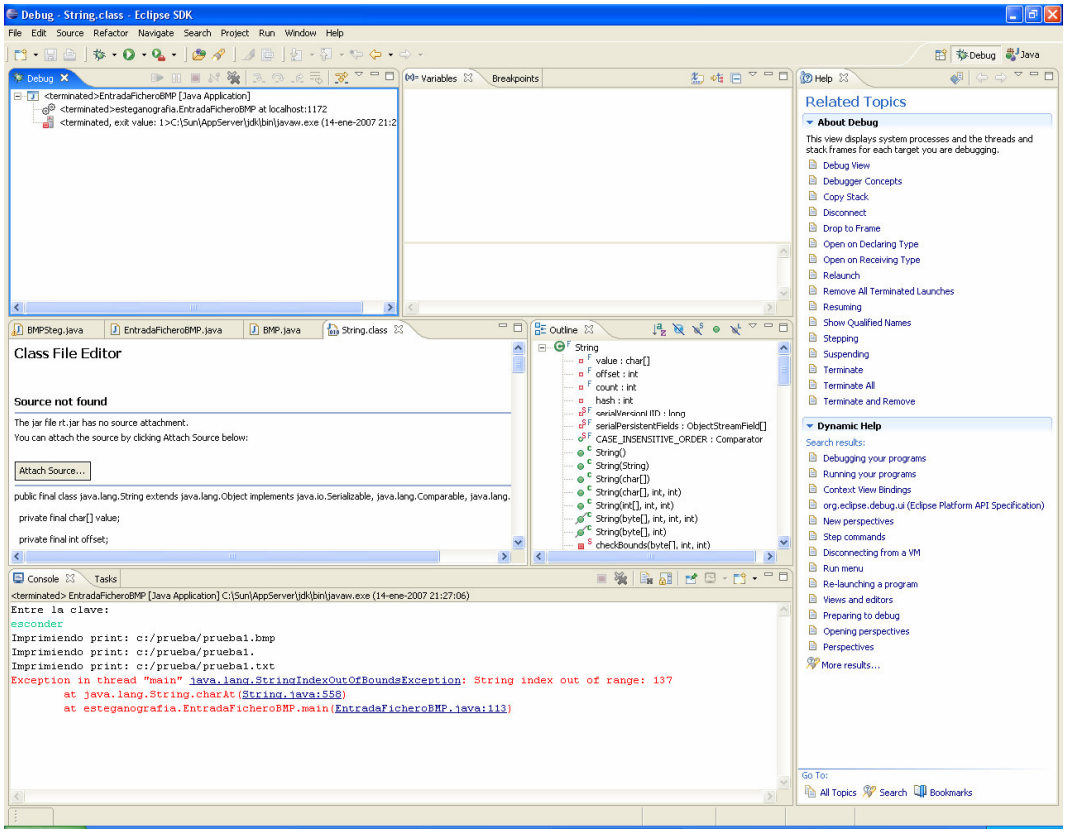


Imagen 8 –*Error al cortar la imagen.*

Para terminar, me gustaría destacar que no se ha podido cumplimentar una aplicación robusta por un problema de tiempo. No le he dedicado el tiempo suficiente al trabajo, y mucho menos a la programación de la aplicación, pero el diseño y los posibles algoritmos han sido bien estudiados y revisados a partir de las diferentes aplicaciones esteganográficas existentes.

Existen muchas otras pruebas a realizar, como el cambio de color, y el conjunto de modificaciones. Pero es evidente, que la aplicación por el algoritmo que utiliza, no podrá dar solución a no ser que este sea retocado considerablemente.

El funcionamiento de la aplicación se puede ver en el vídeo de muestra que se puede encontrar en la siguiente dirección:

www.edutel.es/TFC/TFC.html

11. GLOSARIO

Estego clave (“stego-key”): password requerido en algunos casos, a partir del cual se incluye el mensaje de una forma u otra en el medio tapadera.

Estego medio (“stego-medium”): fichero que resulta después de ocultar un mensaje en un fichero utilizado como medio. En el caso de archivos gráficos se puede hablar de *estego imagen* o *“stego-image”*.

Gurú: maestro, visionario, guía.

Media tapadera (“cover medium”): medio a través del cual se pretende ocultar el mensaje. Algunos de estos medios son los ficheros de imagen, sonido, texto plano...

Mensaje (“embedded message”): información que se trata de ocultar a través de imágenes, sonidos, texto plano...

Píxel: acrónimo de la palabra inglesa "pictorial element" o "picture element". Un pixel es cada uno de los puntos que son asignables o direccionables en una pantalla de video. Normalmente está gobernado por una matriz en memoria que contiene la información necesaria para iluminar cada píxel con un color, brillo e intensidad determinados.

Plug-In: Pequeño programa que se adiciona a la aplicación principal.

Spam: *C*ualquier mensaje, destinado a una audiencia en general (o sea, un comunicado de masas) que se envía a la gente por e-mail sin que lo soliciten. Es el equivalente de los anuncios impresos que le llegan a uno a la casa. Generalmente el SPAM se trabaja con listas de direcciones "robadas" (o sea, de direcciones que sacan de la gente que envía mensajes por USENET u otras áreas públicas de discusión), por lo cual representa una violación de la privacidad del usuario.

Watermarking: técnicas de marcas de agua.

12. BIBLIOGRAFÍA

LINKS:

http://www.infovis.net/printMag.php?num=101&lang=1
http://www.criptored.upm.es/guiateoria/gt_m073b.htm
http://im3.rediris.es/IM3_files/watermarking_IM3.pdf
http://www.preemptive.com/obfuscation-faq/es/
http://www.tvyvideo.com/pragma/documenta/tv/formas/8487/fig1.htm
http://www.apprendre-en-ligne.net/crypto/tritheme/bio_tritheme.html
http://www.jjtc.com/Steganography/ --> Un sitio bastante bueno.
http://www.StegoArchive.com --> Bastante completa.
http://steganography.com/ --> Para ir de compras.
http://www.privacyexposed.com/resources/steganog.htm --> Muchos links.

Referencias

Bender, W. Gruhl, D., Morimoto, N and Lu, A., 1996. "Techniques for data hiding". IBM Systems Journal, 35, 3&4, 313-336

Lemmens, M. , 2003. "Free of Charge or Revenue Generation", GIM International, 17, 2, 40-49

López, C., 2003a. "Digital Rights Managements of Geo-Datasets: Protection against Map Piracy in the Digital Era", GIM International, 17, 2, 51-53

López, C., 2003b. "Protección de propiedad intelectual mediante Marcas de Agua", visitado mayo 2003, http://www.thedigitalmap.com/curso/propiedadIntelectual.pps

López, C., 2002. "Watermarking of Digital Geospatial datasets: a review of Technical, Legal and Copyright issues", International Journal of Geographic Information Science, 16, 6, 589-607.

M. Barni, F. Bartolini, R. Caldelli, A. Piva, Geometric-Invariant Robust Watermarking through Constellation Matching in the Frequency Domain, Proceedings of 7th IEEE International Conference on Image Processing ICIP 2000, Vancouver, Canada, September 10-13, 2000, Vol. II, pp. 65 -68.

I. J. Cox, J. Kilian, T. Leighton, T. Shamoon. Secure Spread Spectrum Watermarking for Multimedia. IEEE Transactions on Image Processing, vol. 6, no. 12, pp. 1673-1687, 1997.

J.J. Eggers, J.K. Su and B. Girod, Robustness of a Blind Image Watermarking Scheme, International Conference on Image Processing (ICIP 2000), Vancouver, Canada, September 2000.

J. J. Eggers, J. K. Su and B. Girod, Performance of a Practical Blind Watermarking Scheme, Electronic Imaging 2001, San Jose, CA, USA, January 2001.

J. J. Eggers, and B. Girod, Blind Watermarking Applied to Image Authentication, submitted to ICASSP 2001, Salt Lake City, Utah, USA, May 7-11, 2001.

L. Friedman, The trustworthy digital camera: restoring credibility to the photographic image, IEEE Trans. On Consumer Electronics, vol.39, pp 905-910, November 1993.

J.Fridrich, M. Goljan, Comparing Robustness of Watermarking Techniques. Proc. of SPIE Vol. 3657 (Security and Watermarking of Multimedia Content), San Jose, Jan 25-27, 1999.

J.K. Su, J.J. Eggers, and B. Girod, Capacity of Digital Watermarks Subjected to an Optimal Collusion Attack, European Signal Processing Conference (EUSIPCO 2000), Tampere, Finland, September 2000

F. A. P. Petitcolas, R. J. Anderson, M. Kuhn. Attacks on copyright marking systems. II Int. Workshop on Information Hiding, 1998.

A. Piva, M. Barni, F. Bartolini. Copyright protection of digital images by means of frequency domain watermarking. Mathematics of Data/Image Coding, Compression, and Encryption, Proceedings of SPIE Vol. 3456, pp. 25-35, 1998.

S. Voloshynovskiy, S Pereira, T. Pun, JJ. Eggers and J. K. Su. Attacks on Digital Watermarks: Classification, Estimation-based attacks and Benchmarks submitted to IEEE Communication Magazin, 2001.

S. Voloshynovskiy, S Pereira, V. Iquise, T. Pun. Towards a second generation watermarking benchmark Signal Processing, Special Issue on Information Theoretic Issues in Digital Watermarking, 2001.