Universidad de Guantánamo

UNIVERSIDAD DE GUANTÁNAMO
SEDE "REGINO E. BOTI"
Facultad de Ingeniería y Ciencias Técnicas
Departamento de Informática

Título: *"Herramienta Informática para la Seguridad de la Información aplicando Algoritmos Esteganográfico.(4ª FRACTAL)"*

Trabajo de Diploma para optar por el título de Ingeniero Informático

Autor: *Amauris Barzaga Favier.*

Tutor (es): *Lic. Alexeis Galano Compte.*

Ms.C Alen Pérez Labardí.

Guantánamo, julio 2020

"El ojo ve solo lo que la mente está preparada para comprender."

Henri Bergson

DEDICATORIA

Dedico este trabajo en primer lugar a Dios, por haberme acompañado en los momentos difíciles que sin él, hoy no podría haber hecho realidad mi sueño. Al amor de mi vida, mi mamá, por recordarme en cada momento que las cosas pasan cuando tienen que pasar, ni tarde ni temprano y que perseverar por lo que quieres te hace ser mejor persona. A mi amigo y hermano por darme fuerza y decirme que si fuera fácil cualquiera lo logra y a mi papá que este título es tuyo. A ellos que han estado a mi lado en lo bueno y en lo malo sin importarle nada, es mi tesis, mi futuro título de ingeniero y mi vida, porque les debo todo.

AGRADECIMIENTOS.

A mi mamá, por haberme dado fuerza cada vez que me caí y confiar en mí en todo monto, que gracias a su paciencia e incondicionalidad hoy estoy aquí haciendo realidad mi sueño, porque no podré dar un paso en la vida sin mencionar tu nombre, a ti mi amor te lo debo todo.

A mi hermanito a quien quiero con la vida y a pesar de las noches grises busco una estrella para mí y nunca dudo de que lo lograría por eso. Te quiero mucho.

A mi papá que de una u otra manera hacia mis metas imposibles para que me esforzara y eso me preparo para la vida, porque si tu dejas de creer en ti nadie podrá ayudarte.

A Yaima y Idamis que cuando creía que no volvería a empezar ellas me dieron esa oportunidad.

A mis abuelitos, a mi tío y a mis primos por sopórtame a su lado en estos últimos años.

A mis profesores por lograr sacar de mí, una mejor versión.

A mi aula que sobre paso mis expectativas y miren que he conocido compañeros a lo largo de mis estudios pero como ustedes ningunos creo que no podré olvidarlos jamás porque este donde este siempre me llegara a la mente una experiencia vivida con ustedes.

A todos aquellos que formaron parte de mi vida en algún momento y me enseñaron que los momentos son cortos y hay que disfrutarlos.

A mis tutores Alexeis y Alen por darme la oportunidad y la ayuda necesaria.

A mi tribunal y oponente por sus sugerencias y aconsejarme en cada momento.

A todos muchas gracias.

RESUMEN.

Con el desarrollo de las Tecnologías de la Informática y las Comunicaciones (TIC), el intercambio de información se ha incrementado significativamente a todos los niveles. Por lo general esta información suele ser privada, lo que implica que se usen métodos o mecanismo de diversas índoles para garantizar su seguridad. En particular los Sistemas Esteganográficos constituyen mecanismos de probada versatilidad para proteger información sensible y con una amplia aceptación en la comunidad científica en temas de seguridad. Es por ello que en este trabajo de investigación se desarrolló una herramienta informática para la Seguridad de la Información, aplicando el Algoritmo Esteganográfico “AA-Fractal”. Se hace uso de la Metodología ágil XP y lenguaje de programación Java, con el entorno de desarrollo Eclipse, para lograr la mayor integridad entre usuario y programador.

Palabras claves: Algoritmo , Esteganografía, Fractal.

ABSTRACT.

With the development of Information Technology and Communications, the exchange of information has increased significantly at all levels. In general, this information is usually private, which implies that methods or mechanisms of various kinds are used to guarantee its security. In particular, Steganographic Systems are mechanisms of proven versatility to protect sensitive information and with wide acceptance in the scientific community on security issues. That is why in this research work a computer tool for Information Security was developed, applying the Steganographic Algorithm "AA-Fractal". The Agile XP Methodology and the Java programming language are used, with the Eclipse development environment, to achieve greater integrity between the user and the programmer.

Key words: Algorithm, Steganography, Fractal.

TABLA DE CONTENIDO

Índice de tablas.

INTRODUCCIÓN.

Hoy en día con el uso de las TIC, puedes decidir cómo vestir al consultar en línea las condiciones actuales del clima, o su estado de cuenta bancario y pagar electrónicamente las facturas, recibir y enviar correos electrónicos o realizar una llamada telefónica a través de Internet, obtener información sobre la salud y consejos sobre nutrición de parte de expertos de todo el mundo y compartir en un foro esa información, al igual que las fotografías, videos hechos en casa y experiencias con amigos o con el mundo. Esto demuestra que las redes se han convertido en parte de nuestra vida diaria pero estos contenidos pueden verse en riesgo por personas maliciosas que violan nuestra seguridad y privacidad de la información.

Un informe del año 2018 desarrollado por la empresa Ellacoya Data recoge el tráfico a nivel global derivado del uso de redes sociales, colocando a Instagram como la primera plataforma social, englobando el 42,19%de las descargas y el 27,06% de las subidas en todo el mundo. En segunda posición se encuentra Facebook, que acumula el porcentaje más alto del tráfico a nivel mundial (un 48,77%, 54,15%). Cerrando el podio de las redes sociales se encuentra Snapchat (7,54% y 14.13%). Existen otras como Twitter, Skype, WhatsApp, FaceTime y Facebook Messenger por donde diariamente estamos compartiendo y recibiendo informaciones[21].

Como se ha mostrado gran parte del tráfico de Internet corresponde a contenidos multimedia, por lo que la protección de tales datos es un tema de vital importancia para las industrias y usuarios. Esto ha tenido como consecuencia el incremento de las investigaciones en lo que se refiere al ocultamiento de información para la protección del contenido, por ejemplo, en forma de marcas de agua para ocultar mensajes de derechos de autor y/o números de series o un conjunto de características que distingan un objeto de otro similar [8][7][19][20]. La idea es que después tal información pueda ser usada en contra de los infractores o para confirmar los derechos de autor. Hay además otras aplicaciones que han incrementado el interés de las comunidades de negocios y académicas, como son: protección de diseños de arquitecturas de hardware, detección de información oculta en archivos multimedia, entre otros.

La eficiencia en la protección de la información con el uso de la Esteganografía, depende de la aplicación de alguna de las diversas técnicas existentes y del uso de un algoritmo adecuado para la inserción y extracción de datos.

Existe una variada gama de técnicas esteganográficas de las cuales cada una tiene sus propias características. Específicamente en imágenes digitales se pueden encontrar distintas formas para ocultar información dentro de ellas, unas más populares que otras. Entre las técnicas más usadas se encuentran las correspondientes al dominio espacial, la cual consiste en la manipulación directa de los pixeles y en la inserción de la información secreta en los bits menos significativos (LSB) o bien de mayor redundancia. Esto minimiza la variación en los colores que crea el ocultamiento, de esta forma la distorsión de la imagen en general se mantiene al mínimo. También se encuentran las técnicas sobre el dominio de las frecuencias: la cubierta es transformada en un conjunto de coeficientes y es en estos donde se inserta la información secreta, proporcionando un nivel más alto de seguridad y robustez. Dentro de la variedad de métodos podemos citar: la Transformada Discreta de Wavelets (DWT), la Transformada Discreta de los Cosenos (DCT) y la Transformada Fractal (TF).

Uno de los primeros trabajos esteganográficos inspirados por la codificación fractal de imágenes es el desarrollado por Joan Puate y Fred Jordan en [24] donde se oculta información a partir del proceso de compresión de la imagen, o sea, se modificó el algoritmo de compresión fractal original propuesto por Jacquin en [17]. Los resultados reportados por este esquema no son muy buenos en lo que corresponde a la percepción visual, ya que la diferencia entre la imagen original y la estegoimagen es significativa. Tampoco la robustez del algoritmo es buena ya que se utiliza un nivel de redundancia sobre el 50% para poder extraer toda la información insertada. Otro esquema inspirado por la codificación fractal es propuesto por Patrick Bas, Jean Marq Chassery y Franck Davoine[5] en este esquema tiene un alto costo computacional y produce una alta percepción visual dependiente de la cantidad de bits que se inserten. También a mayor factor de compresión de la imagen menor es la precisión con que se recuperan los bits insertados.

La Transformada Fractal (TF) usa como principio básico el Teorema del Punto Fijo [12][6] generalizado a conjuntos. Esta propiedad fue utilizada por Zhen Yao [33] para ocultar información durante el proceso original de codificación fractal de imágenes. Los resultados en esta investigación reportan poca capacidad de inserción de información (8 bits) y una apreciable diferencia visual entre la imagen original y la cubierta. Kamal Gulati [14] propone otro algoritmo sobre imágenes con una cantidad de información insertada de 49 bits, mermando la robustez y la imperceptibilidad en dependencia de que tanto se comprima la imagen.

Los resultados mostrados hasta el momento son muy variados, pero todos tienen en común el deterioro de la imagen marcada y el elevado costo computacional de los procesos de inserción y extracción.

Actualmente en nuestro país no existe ninguna aplicación informática que satisfaga dicha problemática, en el mundo existen corporaciones como la RIAA o la SGAE en España que utilizan la palabra de "esteganografía" por "marcas de agua", aunque no son exactamente iguales, realmente las marcas de agua no son comunicación, son datos en programas que usan patrones ya sea en mp3, u otros, sin embargo, una vez que el material multimedia está en poder del comprador, éste puede usarlo, copiarlo, revenderlo, modificarlos, sin control por parte del autor. Es aquí donde se detecta el grave problema.

Otro de los programas parecido es el Detección de copyright, este caso es muy parecido al primero, de las marcas de agua. Muchas empresas de software, por ejemplo Microsoft incluyen la licencia del producto, e incluso en algunos casos los datos personales de la persona que creo el archivo y los más utilizados son programas de ocultación, está Digital Picture Envelope: programa para Windows, gratuito y basado en las técnicas de Eiji Kawaguchi, al igual que 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. Estos métodos de comprensión como se describió utilizan herramientas similares, pero no brindan la seguridad adecuada para el manejo y la protección de la información. [29]

Una vez analizada la situación existente para la protección de información dentro de imágenes, se define como **problema a resolver**: La ausencia de una herramienta informática que emplee algoritmos Esteganográfico.

Teniendo como **objeto de estudio**: el proceso de ocultamiento de información.

Para el desarrollo de la investigación el **objetivo general** trazado es desarrollar una herramienta informática para la Seguridad de la Información, aplicando el Algoritmo Esteganográfico “AA-Fractal”.

El **campo de acción** se centrará en el ocultamiento de la información utilizando algoritmo esteganográfico sobre imágenes.

Como motivación para la realización de esta investigación se tomó como **idea a defender:** el desarrollo de la herramienta informática para la seguridad de la información, haciendo uso del Algoritmo Esteganográfico “AA-Fractal”, que ocultara los datos en una imagen, y garantizará buenos índices de confiabilidad, seguridad e imperceptibilidad del mensaje, al transmitirse por un canal de comunicación.

Para dar cumplimiento de forma exitosa al objetivo planteado, se trazaron las siguientes tareas de investigación:

1. Caracterizar el proceso de ocultamiento de información para la transmisión de datos.
2. Realizar un estudio de las tendencias y tecnologías actuales para el desarrollo de software, así como la metodología a emplear.
3. Realizar el análisis y diseño del sistema informático.
4. Realizar pruebas con el objetivo de detectar errores.
5. Realizar el estudio de factibilidad.

Para el desarrollo de las tareas se utilizaron varios métodos de investigación:

Métodos Teóricos

Histórico-Lógico: se utilizó en el estudio de las tendencias del proceso de investigación

y de las herramientas utilizadas para el desarrollo de la aplicación informática de escritorio.

Análisis-Síntesis: se utilizó en el estudio del proceso para poder caracterizarlo y en la metodología utilizada para el desarrollo de la aplicación informática de escritorio.

Inductivo-Deductivo: permitió a partir de las insuficiencias detectadas determinar el problema a resolver, el objetivo y las conclusiones de la investigación.

Modelación: se utilizó durante la etapa de elaboración del sistema para modelar los componentes estructurales y funcionales la aplicación informática de escritorio propuesta.

Métodos Empíricos

Observación: se utilizó para observar los algoritmos Esteganográfico en Transformadas Fractal para la inserción y extracción de la información en imágenes.

Análisis de Documentos: se revisaron y analizaron documentos

El siguiente documento queda estructurado de la siguiente forma: resumen en español e inglés, introducción, dos capítulos, conclusiones, referencia bibliográfica y anexos.

En el **Capítulo 1. Fundamentos Teóricos:** Se ofrece una breve caracterización de los antecedentes. Se hace referencia a la descripción de las tendencias y tecnologías actuales inclinándonos por la metodología por las que guiará todo el proceso de desarrollo del software, así como el lenguaje utilizado y su entorno de desarrollo. Además se presenta la fundamentación de las herramientas utilizadas y el diagnóstico para su buen funcionamiento.

En el Capítulo 2. Análisis, Diseño e Implementación del Sistema: Este capítulo muestra la solución propuesta, a partir del desarrollo de la metodología empleada para el desarrollo de la herramienta informática. Se determinan los requerimientos funcionales y

no funcionales, roles que se juegan, así como el ciclo de vida de la metodología empleada que transita por cuatro fases fundamentales desde la Planificación con sus historias de usuario, el diseño y sus tarjetas CRC, explicando brevemente cómo se realiza la implementación del software y se describen los casos de prueba desde las unitarias hasta las de aceptación aplicados al mismo.

CAPÍTULO 1. Fundamentos Teóricos.

En este capítulo se caracteriza el objeto de estudio, la descripción de las tendencias y tecnologías actuales y su diagnóstico para crear nuestra herramienta informática que garantice la seguridad e imperceptibilidad del mensaje secreto al transmitirse por un canal de comunicación. Además, se describen las ideas generales de los principios y características fundamentales de la Esteganografía. Finalmente se describe el algoritmo AA-Fractal.

1.1 Antecedentes.

Durante la Segunda Guerra Mundial los avances en tecnología se vieron suspendidos ya que todo el dinero estaba destinado a sustentar el conflicto. Con el fin de la guerra se pudieron observar nuevamente avances en telecomunicaciones, tecnología, educación, entre otras áreas. La Esteganografía en consecuencia también se benefició, la información que se oculta actualmente está dirigida a un grupo de personas y el ambiente cambia de ser conflictivo (guerra) a un entorno donde el manejo de la información y la seguridad de los datos se convierten en el principal reto para los que emplean esta disciplina.

Uno de los estudios actuales más importantes que se han hecho sobre Esteganografía, Cachin propone un sistema de Esteganografía perfecto, incluye toda la formulación matemática de las condiciones que un estegosistema debería tener, plantea un problema de prueba de hipótesis donde compara un objeto inocente y un objeto con información oculta donde la discriminación entre los dos objetos es lo que se plantea como hipótesis para luego por medio de pruebas determinar su validez.

La Esteganografía moderna no depende en mantener en secreto un algoritmo de ocultación y tampoco del objeto que se utilizará para ocultar la misma, sino que se centra en mantener una cantidad mínima de información secreta entre los que intervienen en la comunicación como lo indica [2]. En este mismo sentido, La Esteganografía de llave privada es fruto de la combinación de Esteganografía con criptosistemas simétricos [18].

Se asume que un atacante podría conocer los algoritmos de ocultación y extracción de la información. Por este motivo el mensaje se cifra utilizando un cifrado simétrico antes de ocultarlo, de esta manera, si el atacante intercepta la transmisión y logra extraer la información a un tendrá que enfrentarse al criptoanálisis del criptosistema utilizado [4].

1.2 Algoritmos Esteganográficos con técnicas Fractales.

Los algoritmos Esteganográficos desarrollados en los últimos años que involucran técnicas Fractales no son abundantes debido a que el uso de los Sistema de Funciones Iteradas (SFI) [3] está patentado para uso comercial sobre imágenes, lo que limita que se inviertan recursos humanos con el fin de desarrollar productos esteganográficos y más precisamente de compresión de imágenes. No obstante, existen variedades de aplicaciones de orientación pedagógicas y con un acabado primitivo que permiten interactuar con la Teoría de Fractales. Dentro de estos trabajos se encuentran aplicaciones esteganográficas diseñadas con técnicas Fractales y que resumiremos como sigue:

- Kamal Gulati en [9] desarrolla técnicas de codificación fractal para identificar en que partes de la imagen se introducirá la información. La imagen es particionada en regiones dominio D (bloques dominio) y regiones rango R(bloques rango). Se construye una librería de dominios separada en dos conjuntos: D_0 y D_1. El esquema de inserción de la información es el siguiente: para cada R_i se busca el mejor bloque D_{i*} tal que:

  $$d(D_{i*}, R_i) \leq \min\{e: d(D_{i**}, R_i) = e, D_{i**} \in G\},$$

  Donde d es una métrica y $G = D_0$ si el bit que se va a insertar es 0 o $G = D_1$ de otra forma.

- Dilip Vishwakarma en [31], usa técnicas de codificación fractal para identificar en que parte de la imagen se introducirá la información secreta. Se proponen dos algoritmos:
  1. La imagen es particionada en regiones dominio según los cuadrantes de la imagen y tomar solo D_I y D_{II} (bloques dominio) y regiones rango R (bloques rango). Al insertar la información: para cada R_i se busca el mejor bloque D_{i*}

tal que:

$$d(D_{i*}, R_i) \leq \min\{e: d(D_{i**}, R_i) = e, D_{i**} \in G\},$$

donde d es una métrica y $G = D_I$ si el bit que se va a insertar es 0 o $G = D_{II}$ de otra forma.

2. Similar al algoritmo anterior, además de que los bits menos significativos de R_i y D_i deben coincidir.

- Fadhil Salman en[11], la imagen es particionada en regiones dominios D (bloques dominios) y regiones rango R (bloques rango). Se calcula la Transformada Fractal (TF) para cada R_i, resultando una lista de coeficientes de funciones lineales $s_i * x + o_i$ almacenadas de la forma $[[s_1, o_1], \dots, [s_n, o_n]]$, donde n es la cantidad de elementos que posee R. Como $0 < s_i < 1$, lo que es equivalente a $s_i = n_1 n_2 n_3 n_4 n_5$ … tomando la primeras 5 cifras después del 0, la información que se desea ocultar es convertida a su valor decimal en código ASCII después de ser cifrada con RSA [18] e insertada en las últimas tres posiciones.
- Richa Gupta en [15] toma un mapa de píxeles RGB (.BMP) y lo convierte a escala de gris, localiza cuales son las regiones fractales en la imagen e introduce la información en dicha locación, pero en la imagen original .BMP de la siguiente forma: 1 bit en el Rojo, 1 bit en el Verde y 3 bits en el Azul.

Los algoritmos anteriormente citados carecen de una de las principales características que debe poseer un Estegosistema: la robustez. Dichos algoritmos insertan la información en los bits menos significativos de la imagen a través de técnicas fractales sin considerar que al variar unos pocos bits de la estego imagen el resultado de la TF al intentar recuperar la información puede ser gravemente alterado, esto es, un simple cambio entre formato de imagen, dígase de .BMP a .JPG y luego a .BMP, un procesamiento de la imagen para aumentarle/disminuirle el brillo y/o contraste o para realzar los bordes comprometería totalmente la extracción de la información secreta.

1.2.1 Características de la Esteganografía.

Durante los últimos años se han propuesto diferentes métodos de Esteganografía, la

mayoría de ellos se pueden ver como sistemas de sustitución, estos métodos tratan de sustituir las partes redundantes de una señal con un mensaje secreto, su principal desventaja es la relativa debilidad contra modificaciones de la cubierta. Existen además muchos métodos de escritura encubierta, desde microfilmes, tinta especial y arreglos de caracteres, permutaciones, sustituciones, firmas digitales, entre otros [23][7][19][20].

Existen varias formas de clasificar los sistemas Esteganográfico de acuerdo al tipo de cubierta utilizada para la comunicación secreta, o bien a las modificaciones aplicadas a la cubierta en el proceso de inserción, como se muestra en [14]. Específicamente sobre imágenes, se pueden clasificar como:

- **Esteganografía en el dominio espacial:** El sistema matricial de coordenadas de una imagen es lo que se denomina dominio espacial, sin embargo, la misma imagen puede ser considerada, como una función no periódica y definirse en otro espacio bidimensional, cuyos ejes vengan determinados por la amplitud y la frecuencia para cada dirección de la imagen. En la aplicación de la Esteganografía en el dominio espacial, los algoritmos son usados para la manipulación de los píxeles y la inserción de datos secretos en los bits menos significativos o bien de mayor redundancia.
- **Esteganografía en el dominio de la transformada**: Los métodos utilizados en este dominio 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 los elementos de la imagen como la luminosidad. Normalmente se crean mascaras indicando los lugares más idóneos para la ocultación de información.

Las características de un esquema Esteganográfico se pueden resumir en [9]:

- **Invisibilidad**: La invisibilidad corresponde al objetivo principal de la Esteganografía: ocultar la existencia del acto de comunicación. Está condicionada por la medida de la distorsión que se le produce al portador debido a la aplicación del algoritmo de inserción del mensaje. Para hablar de invisibilidad, solo debe tener conocimiento de la existencia de un mensaje embebido el que conoce el algoritmo para extraer. Ningún otro participante debe tener evidencias de su existencia [1].

- **Confiabilidad**: Se denomina confiabilidad a la probabilidad de que el algoritmo de extracción emita como salida el mensaje esteganográfico m correcto (es decir, el mismo que fue ocultado mediante el algoritmo de inserción) [4].
- **Robustez**: Se denomina robustez al grado de inmunidad del estegomensaje frente a alteraciones posteriores realizadas por un intruso. Esto es, la probabilidad de que el algoritmo de extracción emita como salida el mensaje esteganográfico m cuando el estegomensaje fue modificado por el atacante.
- **Capacidad de inserción:** Corresponde al máximo número de bits que pueden ser insertados en una cubierta [7].
- **Capacidad esteganográfica:** Corresponde al máximo número de bits que pueden ser insertados en una cubierta de modo que la probabilidad de detección por parte de un estegonalista sea despreciable [7]. La capacidad esteganográfica depende del tipo de cubierta y, en general, es mucho menor a la capacidad de inserción. Dependencia del portador: Los métodos esteganográficos están basados en el análisis de las características del portador utilizado y la posibilidad de explotar el uso del mismo para aplicar estas técnicas [27]. Es decir, existe una fuerte dependencia del portador utilizado.

1.2.2 Aplicaciones de la Esteganografía.

Los productos esteganográfico se enfocan en asegurar la privacidad de la información en computadoras, para protegerla de robo o el acceso a las mismas de personas no autorizadas. La industria médica y los sistemas de imágenes médicos se benefician de estas técnicas. En las imágenes de pacientes se adjunta información adicional que las describe, denominada metadata: nombre del paciente, fecha de la imagen, tipo de imagen, diagnóstico. La posibilidad de incorporar esa información dentro de la imagen evita la perdida de la vinculación entre la metadata y la imagen relacionada [19]. Existen aplicaciones de propósito general para agregar etiquetas a imágenes online para permitir indexación de las etiquetas y búsquedas [32].

Con respecto a las imágenes y a archivos multimedia en general, uno de los principales usos de las técnicas esteganográficas consiste en ocultar marcas de copyright en las

mismas, que permitan identificar al autor. Estas marcas pueden ser ocultas o visibles en los archivos multimedia. Estas prácticas se conocen con el nombre de Watermarking (marcas de agua). Además de las marcas de identificación del autor, se puede agregar marcas de identificación del destinatario (es decir, números de serie). Esto permite determinar quién fue el usuario que violo los derechos de copyright del archivo multimedia. Estas técnicas se conocen con el nombre de Fingerprinting (huellas dactilares) [26]. Una aplicación consiste en el monitoreo automático del material con copyright en Internet: un software robot descarga automáticamente imágenes de sitios web en busca de las marcas ocultas de copyright para identificar a quienes hayan violado la licencia de uso[19].

Ambas técnicas, Watermarking y Fingerprinting, no son estrictamente consideradas esteganográficas puesto que su resultado puede causar una modificación perceptible. El objetivo principal de las mismas es identificar los derechos de uso del portador y que los mismos no puedan ser quitados del portador sin dañarlo perceptiblemente. Aunque no es indispensable el disimulo de las marcas de copyright para este objetivo, bien se puede llevar a cabo a partir de técnicas esteganográficas.

1.3 Descripción de las tendencias y tecnologías actuales.

1.3.1 Metodología de Desarrollo de Software.

En un proyecto de desarrollo de software la metodología ayuda a definir: Quién debe hacer Qué, Cuándo y Cómo debe hacerlo. La metodología para el desarrollo de software es un modo sistemático de realizar, gestionar y administrar un proyecto para llevarlo a cabo con altas posibilidades de éxito. La mejor herramienta para el desarrollo de software, Xtreme Programmig pone el énfasis en la adaptabilidad del proceso antes que en la previsión de incidentes. Los partidarios de XP consideran que los accidentes, fallos o inconvenientes que surgen durante un proyecto son elementos naturales y que, por tanto, más vale saber adaptar el proceso antes que suspenderlo y poner en riesgo sus resultados. Esto se logra gracias a una integración de los elementos, humanos o técnicos, y a la retroalimentación entre ellos. Es idóneo para proyectos dinámicos y cambiantes.[22]

Una metodología para el desarrollo de software comprende actividades a seguir para

idear, implementar y mantener un producto de software desde que surge la necesidad del producto hasta que se cumple el objetivo por el cual fue creado, una Metodología ligera de desarrollo de aplicaciones que se basa en la simplicidad, la comunicación y la realimentación del código desarrollado:

OBEJTIVOS DE XP

- La Satisfacción del cliente.
- Potenciar el trabajo en grupo.
- Minimizar el riesgo actuando sobre las variables del proyecto: costo, tiempo, calidad, alcance.

CARACTERÍSTICAS

- Metodología basada en prueba y error para obtener un software que funcione realmente.
- Fundamentada en principios.
- Está orientada hacia quien produce y usa software (el cliente participa muy activamente).
- Reduce el costo del cambio en todas las etapas del ciclo de vida del sistema.
- Combina las que han demostrado ser las mejores prácticas para desarrollar software, y las lleva al extremo.
- Cliente bien definido.
- Los requisitos pueden cambiar.
- Equipo con formación elevada y capacidad de aprender.

El ciclo de vida de XP consiste en cuatro fases:

1. **Planificación:**

En esta fase los clientes plantean a grandes rasgos las historias de usuario (artefacto de la metodología para especificar requisitos) y se establece la prioridad de cada una, y correspondientemente, el programador o los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Al mismo tiempo el desarrollador o el equipo de desarrollo estudia las herramientas, tecnologías y prácticas que se utilizarán en el proyecto.

2. **Diseño:**

En esta fase se detallan las tareas de ingenierías a implementar para satisfacer las historias de usuario que en cada iteración se describen. En la fase se diseñan, además, el plan de entrega de las versiones del sistema y las pruebas de aceptación para cada iteración.

3. **Codificación:**

En esta fase se codifican las tareas de ingeniería para cumplir los requisitos funcionales del sistema. Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El sistema en esta fase requiere de pruebas adicionales y revisiones de rendimiento antes de que sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase.

4. **Prueba:**

Tienen como objetivo verificar la funcionalidad del sistema a través de sus interfaces externas comprobando que dicha funcionalidad sea la esperada en función de los requisitos del sistema. Generalmente las pruebas del sistema son desarrolladas por los programadores para verificar que su sistema se comporta de la manera esperada, por lo que podrían encajar dentro de la definición de pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, que propone XP. Sin embargo, las pruebas del sistema tienen como objetivo verificar que el sistema cumple los requisitos establecidos por el usuario por lo que también pueden encajar dentro de la categoría de pruebas de aceptación, destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñadas por el cliente final.

Figura 1.1 Ciclo de vida de Xtreme Programmig (XP).

1.3.2 Lenguaje de programación

Los lenguajes de programación son idiomas artificiales diseñados para expresar cálculos y procesos que serán llevados a cabo por ordenadores. Un lenguaje de programación está formado por un conjunto de palabras reservadas, símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Donde el proceso de programación dará la escritura, compilación y verificación del código fuente del programa**,** para expresar algoritmos con precisión o como modo de comunicación humana. [23]

JAVA: es un lenguaje de programación desarrollado por Sun Microsystems. (Empresa que posteriormente fue comprada por Oracle) y fue presentado en la segunda mitad del año 1995 y desde entonces se ha convertido en un lenguaje de programación muy popular. E s un lenguaje muy valorado porque los programas que se pueden ejecutar, en diversas plataformas con sistemas operativos como Windows, Mac OS, Linux o Solaris. James Gosling, el director del equipo de trabajo encargado de desarrollar Java, hizo realidad la promesa de un lenguaje independiente de la plataforma. Se buscaba diseñar un lenguaje que permitiera programar una aplicación una sola vez que luego pudiera ejecutarse en distintas máquinas y sistemas operativos. Para conseguir la

portabilidad de los programas Java se utiliza un entorno de ejecución para los programas compilados. Este entorno se denomina Java Runtime Environment (JRE). Es gratuito y está disponible para los principales sistemas operativos. Esto asegura que el mismo programa Java pueda ejecutarse en Windows, Mac OS, Linux o Solaris. Su sintaxis es muy parecida a la de C o C++. [13]

Las características fundamentales que ofrece esta herramienta son:

- Basado en el diseño de clases de negocio.
- Alta Usabilidad.
- Facilidad de generación tanto para aplicaciones web como para aplicaciones de escritorio.
- Soporte de varios módulos propios de la herramienta.

La flexibilidad de la herramienta parte de la utilización de clases de negocio que encierran toda la complejidad de la interacción entre clases y objetos de nuestro sistema, es decir, en cada clase se definen los atributos, métodos y funciones con los que se va a interactuar desde la interface del aplicativo, así como las relaciones existentes entre clases. Aquí también se pueden definir reglas de validación, reglas de negocio, etc.

La usabilidad de esta herramienta radica en la facilidad que ofrece para desarrollar aplicaciones tanto para web como para escritorios, permitiendo que los desarrolladores se enfoquen única y exclusivamente en el diseño del sistema, debido a que partiendo de un diagrama de clases es factible el completo desarrollo de una herramienta o sistema totalmente funcional.

Existen distintos entornos de desarrollo de aplicaciones Java. Este tipo de productos ofrecen al programador un entorno de trabajo integrado para facilitar el proceso completo de desarrollo de aplicaciones, desde el diseño, la programación, la documentación y la verificación de los programas. Estos productos se denominan IDE (Integrated evelopment Environment). [13]

Existen entornos de distribución libre como: NetBeans, Eclipse o BlueJ. Entre los

productos comerciales están JBuilder o JCreatorPro. El que utilizaremos será el Eclipse.

Eclipse: no es más que la herramienta sobre la cual se puede montar cualquier lenguaje para el desarrollo, mediante la implementación de los plugins y librerías adecuados que facilitan la manipulación y utilización de la herramienta para el desarrollo de software. [25]

Conclusiones del capítulo.

En este capítulo se ha profundizado en las características más importantes de las técnicas de programación, la metodología de desarrollo, y las herramientas empleadas en el desarrollo de la herramienta.

CAPÍTULO 2. Propuesta e Implementación.

En el presente capítulo se describe e implementa la herramienta informática propuesta para brindar mayor seguridad a la información que es transitada por diferentes canales de comunicación, describiendo el algoritmo empleado, así como los requisitos funcionales y no funcionales de la herramienta, el proceso de diseño y los resultados obtenidos durante las fases de implementación y las pruebas realizadas empleando la metodología XP.

2.1 Propuesta de la herramienta.

La solución que se propone será una herramienta para el ocultamiento de información en imágenes que brindará una mayor seguridad a la información que se guarde en una imagen. La herramienta ofrecerá además que la inserción y la extracción de la información no tenga perdida al pasar por el objeto portador.

2.1.1 Descripción del Algoritmo AA-Fractal.

El algoritmo de ocultamiento tiene una estructura sencilla: se van recorriendo la imagen y bajo ciertas condiciones se va transformando partes de esta. Los elementos de la secuencia binaria K funcionará como llave secreta que será sumada XOR con la información que se insertará en los coeficientes de la Transformada Fractal (TF), aplicado a porciones de la imagen de manera imperceptible. Finalmente son aplicadas las transformaciones en orden inverso para construir una imagen muy similar a la original: la estegoimagen, ver figura 2.2.

2.1.2 Generación Fractal de Subimagen.

La función GenImage2x2 genera una imagen de tamaño 2x2 a partir de un código fractal que consta de 2 elementos: coeficiente de contraste s y de brillo o, los cuales caracterizan a una función lineal contractiva, lo que permite iterar la función y que tienda a su punto fijo bastando para ello 10 iteraciones. Tiene como entrada una matriz (subimagen) de

dimensiones 2x2 y con profundidad de color 256. Tiene como salida una matriz de 2x2. La figura 2.1 muestra el seudocódigo.

```
Algorithm 1 GenImage2x2
  INPUT: M, s, o.
  OUTPUT: M.
  for k = 1 to k = 10 do
    for i = 1 to i = 2 do
      for j = 1 to j = 2 do
        M[i][j] = s*( M[i][j] ) + o:
      end for
    end for
  end for
  return M:
```

Figura 2.1 función GenImage2x2 que genera una imagen 2x2.

2.1.3 Particionamiento de Imagen.

La función *PtcnImage* permite dividir la imagen en bloques de 2x2 pixeles, donde MxN son las dimensiones de la imagen. Tiene como entrada una imagen en escala de gris con una profundidad de color de 8 bits y como salida una lista de bloques. La función inversa *invPtcnImage* tiene como entrada una lista de bloques de tamaño 2x2 y retorna una imagen de dimensiones MxN construida a partir de dichos bloques.

2.1.4 Transformada Fractal

El cálculo de la Transformada Fractal se hará sobre una imagen P de dimensiones 2x2. A partir de estos cuatro píxeles se calculará los coeficientes de la función lineal:

$$\boldsymbol{w(x) = sx + o.}$$

Esta función debe ser contractiva, lo que significa que 0 < s < 1. Aplicaremos entonces el Método del Escalado Constante por lo que consideraremos el valor de s = 3/4 y calcularemos a o como:

$$\boldsymbol{s = \sum_i \frac{r_i}{4}.}$$

2.1.5 Proceso de Inserción.

El mensaje secreto es insertado en la imagen cubierta mediante el siguiente **procedimiento**: El algoritmo AA-Fractal divide la imagen cubierta en bloques no solapados de 2×2 pixeles, resultando el conjunto B, y luego aplica la TF para cada elemento que posea todos sus componentes mayores que un rango mínimo y menor que un rango máximo, lo que llamaremos factores de calidad. La secuencia binaria K = {k 1, ..., k 128} es sumada XOR con el mensaje secreto S = {s 1, ..., s 128}. Si el correspondiente Bi cumple con los factores de calidad, entonces el bloque seleccionado se le aplica la TF, resultando los coeficientes s y o. La información se introducirá en el bit menos significativo de o en dependencia del bit s_i, si ambos son cero o uno no hay cambios, sino se suma uno a o. Después de insertar el mensaje secreto se realiza la transformación en orden inverso: es reconstruida la matriz de orden 2 a partir de los respectivos coeficientes de la TF, obteniendo un bloque de la estegoimagen. La figura 2.2 muestra el seudocódigo.

Algorithm 2 Embedding Algorithm

INPUT: $\mathcal{S}$, $\mathcal{K}$, $\mathcal{E}$.

OUTPUT: $\mathcal{E}^*$.

▷ Se adicionan XOR la clave $\mathcal{K}$ y la secuencia $\mathcal{S}$ para producir una secuencia cifrada $\mathcal{C}$ de 128 bits.

$B :=$ PtcnImage($\mathcal{E}$).

for $B_i \in \mathcal{B}$ **do**

if $Min \leq B_i[0][0], B_i[0][1], B_i[1][0], B_i[1][1] \leq Max$ **then**

$f :=$ TransformFractal (B_i) :

$I_{2\times2} :=$ GenImage2x2 $(I_{2\times2}, f)$:

if $Min_{2\times2} \leq I_{2\times2} \leq Max_{2\times2}$ **then**

if $c_i = 0$ **then**

if $f[2] \bmod 2 = 1$ **then**

$f[2] := f[2] + 1$:

end if

else

if $f[2] \bmod 2 = 0$ **then**

$f[2] := f[2] + 1$:

end if

end if

$I_{2\times2}; :=$ GenImage2x2 $(I_{2\times2}, f)$:

if $Min_{2\times2} \leq I_{2\times2} \leq Max_{2\times2}$ **then**

$B_i := I_{2\times2}$:

end if

end if

end if

end for

$\mathcal{E} :=$ invPtcnImage(B_i):

return $\mathcal{E}$:

Figura 2.2 Algoritmo AA-Fractal.

2.1.6 Proceso de Extracción.

El proceso de extracción sigue un procedimiento similar que el ejecutado en el proceso de inserción, luego se extraen los bits correspondientes del mensaje. Para ambos la extracción se ingresa la clave privada de 128 bits y la estegoimagen, retornando una secuencia de bits.

Algorithm 3 Extraction Algorithm

INPUT: $\mathcal{K}, \mathcal{E}$.

OUTPUT: $\mathcal{S}$.

$B :=$ PicnImage$(\mathcal{E})$.

for $B_i \in \mathcal{B}$ **do**

  if $Min \leq B_i[0][0], B_i[0][1], B_i[1][0], B_i[1][1] \leq Max$ **then**

    $f :=$ Transform Fractal (B_i) :

    $s_i := f[2] \bmod 2$:

    $s_i := s_i \oplus k_i$:

  end if

end for

return $\mathcal{S}$:

Figura 2.3 seudocódigo del procedimiento para la extracción.

2.2 Especificación de los requisitos de software

Un requisito de software es una condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, un estándar u otro documento impuesto formalmente, que le permita al usuario resolver un problema o lograr un objetivo.

Requerimientos funcionales.

Son aquellos que el usuario solicita que efectúe la herramienta. Para el sistema se definieron las siguientes funcionalidades:

RF 1 Gestionar Imagen.

RF 1.1 Cargar Imagen.

RF 1.2 Codificar Imagen[1].

RF 1.3 Guardar Estegoimagen.

RF 2 Gestionar Estegoimagen.

RF 2.1 Extraer mensaje.

[1] Procedimiento de inserción del mensaje a ocultar, además de la clave.

RF 3 Validar calidad de Imagen.

Requerimientos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener, además son aspectos importantes que el producto debe cumplir para lograr un producto atractivo, usable, rápido o confiable.

La herramienta de escritorio propuesta posee los siguientes requerimientos no funcionales:

RNF1 Seguridad: La herramienta de escritorio garantiza la seguridad e integridad de la información basándose en su algoritmo AA-Fractal, asegurando que no exista la perdida de la información al ser guardada en su medio portador.

RNF2 Soporte: La herramienta de escritorio debe permitir posteriores modificaciones y actualizaciones a fin de alcanzar mayor funcionalidad o dado que cambien algunos elementos o se descarte su uso en algún negocio.

RNF3 Apariencia o interfaz externa: La herramienta de escritorio tiene una interfaz sencilla, agradable y completamente flexible a las dimensiones de la pantalla del usuario. El contenido se muestra de manera comprensible y fácil de leer, teniendo en cuenta algunos elementos de diseño como la combinación de los colores, el tipo y tamaño de las letras y los íconos, los cuales están en correspondencia con la información que se presenta. Además, el diseño de la interfaz está encaminado a facilitar el uso del sistema, logrando que el usuario tenga un buen entendimiento y mejor desempeño de lo que se muestre en la herramienta.

RNF5 Rendimiento: La herramienta de escritorio permite más tiempo de imperceptibilidad ante el ataque de personas maliciosas o programas, siendo capaz de procesar con rapidez y eficiencia los datos, con tiempos de inserción y extracción mínimos.

RNF6 Portabilidad: La herramienta de escritorio está diseñada para ser ejecutada sobre

Windows y a su vez soportará la inserción de todo tipo de imágenes, pero en dependencia de su formato será la cantidad de pixeles utilizables.

2.3 Análisis del sistema

Durante la fase de planificación se realiza una estimación del esfuerzo que costará implementar cada historia de usuario. En XP las métricas son libres, pudiendo utilizarse cualquier criterio para medir el desarrollo del proyecto.

Historia de Usuario.

En la metodología XP, es la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente, quien es el mayor responsable de realizar esta tarea, describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario debe ser lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en tiempo delimitado y generalmente poco prolongado.

A continuación se presenta una pequeña muestra de las Historias de Usuario que fueron definidas en este proceso para el desarrollo de la solución propuesta y que reflejan la información brindada por el Usuario.

Historia de Usuario	
Número: HU1	**Nombre Historia de Usuario:** Gestionar imagen.
Modificación de Historia de Usuario Número: ninguna	
Usuario: todos	**Iteración Asignada:** 1ra Iteración
Programador responsable: Amauris Barzaga Favier	
Prioridad en Negocio: Alta	**Puntos Estimados:** 1 Semana
Riesgo en Desarrollo: Alto	**Puntos Reales:** 2 Semana
Descripción: La ventana principal muestra un formulario que solicita al usuario la opción de cargar una imagen, en el caso de que sea imagen original, se procede a insertar el texto (mensaje) a codificar y la clave (es obligatorio). Una vez llenado los	

campos necesarios, se le da a la opción codificar que genera la estegoimagen. Si se realiza la operación exitosamente, se procede a guardar la estegoimagen, en caso contrario se muestra un error en el llenado de los datos.
Observaciones: El sistema verifica que el usuario halla llenado los formularios correctamente, de lo contrario muestra un mensaje de error.

Tabla 2.1 Historia de Usuario Gestionar Imagen.

Historia de Usuario	
Número: HU2	**Nombre Historia de Usuario:** Gestionar Estegoimagen.
Modificación de Historia de Usuario Número: ninguna	
Usuario: todos	**Iteración Asignada:** 1ra Iteración
Programador responsable: Amauris Barzaga Favier	
Prioridad en Negocio: Alta	**Puntos Estimados:** 1 Semana
Riesgo en Desarrollo: Alto	**Puntos Reales:** 2 Semana
Descripción: La ventana principal muestra un formulario que solicita al usuario la opción de cargar una imagen, en el caso de que sea estegoimagen, se procede a descodificar la imagen, solicitando al usuario que ingrese la clave. Luego de ingresada la clave se genera el mensaje codificado. Si se realiza la operación exitosamente, se procede a mostrar el mensaje, en caso contrario se muestra un error en el llenado de los datos.	
Observaciones: El sistema verifica que el formulario esté llenado correctamente para poder mostrar el mensaje.	

Tabla 2.2 Historia de Usuario Gestionar Estegoimagen.

Historia de Usuario	
Número: HU3	**Nombre Historia de Usuario:** Validar Calidad de Imagen
Modificación de Historia de Usuario Número: Ninguna	
Usuario: todos	**Iteración Asignada:** 2 Iteración
Programador responsable: Amauris Barzaga Favier	
Prioridad en Negocio: Alta	**Puntos Estimados:** 4 Semana
Riesgo en Desarrollo: Alto	**Puntos Reales:** 6 Semana
Descripción: Cuando el usuario hace clic sobre la opción de codificar , el software	

realiza las pruebas de calidad de la imagen y muestra los resultados de las pruebas. En caso de que una prueba no cumpla los parámetros de calidad, se le muestra al usuario un mensaje.
Observaciones:

Tabla 2.3 Historia de Usuario Validar calidad de imagen.

Tareas de Ingeniería

Las tareas de ingeniería son un artefacto generado en la metodología XP, de cierta forma un complemento de las historias de usuarios, pero desde el punto de vista de los desarrolladores, ya que se precisa cuáles son las tareas que las componen y que son las que marcarán el sistema en general. Las tareas pueden ser: desarrollo, corrección, mejora, entre otras. Estas tareas tienen relación con una historia de usuario; se especifica la fecha de inicio y fin de la tarea, se nombra al programador responsable de cumplirla y se describe qué se tratará de hacer en la tarea. A continuación se describen las tareas de ingeniería.

Tarea de Ingeniería	
Número de tarea: TI1	**Número de historia:** HU1
Nombre de tarea: Cargar imagen	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 10/2/2020	**Fecha de fin:** 28/2/2020
Programador responsable: Amauris Barzaga Favier	
Descripción: Haciendo clic sobre el botón Cargar imagen, aparece una ventana para seleccionar la imagen con la que se desea trabajar, es decir insertar el mensaje.	
Prototipo Interfaz: Anexo 1	

Tabla 2.4 Tarea Ingeniería TI1 de HU1.

Tarea de Ingeniería	
Número de tarea: TI2	**Número de historia:** HU2
Nombre de tarea: Extraer mensaje	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1

Fecha de inicio: 2/3/2020	**Fecha de fin:** 16/3/2020
Programador responsable: Amauris Barzaga Favier	
Descripción: Haciendo clic sobre el botón Cargar imagen, aparece una ventana para seleccionar la estegoimagen, una vez que se muestra la estegoimagen se procede a ingresar la clave para extraer el mensaje.	
Prototipo Interfaz: Anexo 2	

Tabla 2.5 Tarea Ingeniería TI2 de HU2.

Tarea de Ingeniería	
Número de tarea: TI3	**Número de historia:** HU3
Nombre de tarea: Calidad de imagen	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha de inicio: 23/3/2020	**Fecha de fin:** 16/5/2020
Programador responsable: Amauris Barzaga Favier	
Descripción: Cuando se tiene la imagen en la que se va introducir el mensaje, se da clic en el botón codificar, se muestra la estegoimagen para la comparación visual con la imagen original, además se muestra un panel con los resultados con los parámetros de calidad de imagen que se requieren.	
Prototipo Interfaz: Anexo 3	

Tabla 2.6 Tarea Ingeniería TI3 de HU3.

Estimación de esfuerzo por Historias de Usuarios.

Con el objetivo de alcanzar el buen desarrollo del sistema propuesto, se realizó una estimación de esfuerzo para cada una de las historias de usuario identificadas y se muestran en la tabla a continuación.

Historia de usuario	Puntos de estimación
Gestionar imagen	1
Gestionar Estegoimagen.	1
Validar Calidad de Imagen	4

Tabla 2.7 Estimación de esfuerzo por Historias de Usuario

Plan de iteraciones

Luego de haberse identificado las historias de usuario del sistema y haberse estimado el esfuerzo que debe ser empleado en la realización de cada una de estas, se procede a la planificación de la etapa de implementación del proyecto. Sobre la base de lo planteado anteriormente se decide realizar tres iteraciones para alcanzar la solución propuesta. Éstas se detallan a continuación:

Iteración 1:

En esta iteración se desarrollará la historia de usuario que más peso tiene sobre el sistema que se desea desarrollar, el punto de partida para la realización de cada una de las restantes historias de usuario que deben ser realizadas. Al concluir el desarrollo de esta iteración se obtendrá una primera y aún incompleta versión del sistema, que ya puede ser probada.

Iteración 2:

Esta iteración tiene como principal objetivo realizar las historias de usuario de prioridad media que son necesarias para la realización de otras historias de usuario de mayor prioridad que serán realizadas en la misma iteración. Al concluir se obtendrá una nueva versión que será el cliente el responsable de revisar y comunicar al equipo de desarrollo los cambios que son necesarios realizarle a la misma.

Iteración 3:

Durante esta iteración se realizarán las restantes historias de usuario de prioridad baja que aún no se han implementado y que son prescindibles para una versión 1.0 del sistema. Esta versión ya puede ser considerada un componente como tal y por tanto debe ser puesto en funcionamiento por un periodo determinado de tiempo y de esta forma ser probado por varios usuarios.

Plan de duración de las iteraciones.

Iteración	**Orden de las HU**	**Duración de las iteración**
Iteración 1	Gestionar imagen	1

	Gestionar Estegoimagen.	1
Iteración 2	Validar Calidad de Imagen	2

Tabla 2.8 Plan de duración de las iteraciones

Plan de entrega del sistema propuesto

No. Iteración	Duración total	Fecha inicio	Fecha fin
Iteración 1	2	10/2/2020	28/2/2020
Iteración 2	2	2/3/2020	16/3/2020
Iteración 3	6	23/3/2020	16/5/2020

Tabla 2.9 Plan de entregas del sistema propuesto

Planificación de la entrega

No	Historia de Usuario	Prioridad	Riesgo	Esfuerzo	Iteración
1	Gestionar imagen	Alta	Alto	2.0	1
2	Gestionar Estegoimagen.	Alta	Alto	2.0	1
3	Validar Calidad de Imagen	Alta	Alto	6.0	2

Tabla 2.10 Planificación de la entrega

Para el diseño de las aplicaciones, se utilizó la metodología XP, esta no requiere la representación del sistema mediante diagramas de clases utilizando notación UML. En su lugar se usan otras técnicas como las tarjetas CRC como una extensión informal a UML. La técnica de representación mediante tarjetas CRC se puede usar para guiar el sistema a través de análisis basados en la responsabilidad. Las clases se examinan, se filtran y se refinan sobre la base de sus responsabilidades con respecto al sistema, y las clases con las que necesitan colaborar para completar las mismas.

Las tarjeta CRC (Clases, Responsabilidades y Colaboración), pero debemos decir que estas son realizadas con el objetivo de facilitar la comunicación y documentar los resultados. Estas permiten que el equipo completo contribuya en la tarea que se diseñó. Una tarjeta CRC representa un objeto, por lo tanto es una clase, cuyo nombre se pone en forma de título en la tarjeta, los atributos y las responsabilidades más significativas se

colocan a la izquierda y las clases que están implicadas con cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente. Para una mejor comprensión de las mismas decidimos agruparlas por historias de usuarios. A continuación se muestran las mismas:

Clase ImageRGB (Clase que permite manejar la imagen a la hora de insertar la información)
Superclases: Esteganografia (Clase que permite insertar y extraer textos en imágenes basado en el algoritmo AA-Fractal)
Subclases: PixelRGB,
Responsabilidades
private BufferedImage img public ImageRGB(String f) private ImageRGB (int ancho, int altura) public static ImageRGB unir(int ancho, int altura, PixelRGB pix) public static ImageRGB blanco(int ancho, int altura) public static ImageRGB negro(int ancho, int altura) private static int leeEntero32bits(DataInputStream in) private static int leePixel24bits(DataInputStream in) public boolean guardaBitMap(String f) private static void escribeEntero32bits(DataOutputStream salida, int n) private static void escribePixel24bits(DataOutputStream salida, int p) public int altura() public int ancho() public PixelRGB getPixelRGB(int i, int j) public void setPixelRGB(int i, int j, PixelRGB pix)

Tabla 2.11 Tarjeta CRC: Clase ImageRGB

PixelRGB (Clase para crear y manejar pixeles a la hora de la extracción de la información)
Superclases: Esteganografia (Clase que permite insertar y extraer textos en imágenes basado en el algoritmo AA-Fractal)
Subclases: ImageRGB,
Responsabilidades
private int r private int g private int b public PixelRGB(int r, int g, int b) public static PixelRGB blanco() public static PixelRGB negro() public static PixelRGB gris(int v) public int getR() public int getG() public int getB() public String toString()

Tabla 2.12 Tarjeta CRC: Clase PixelRGB

Prueba (Clase para comparar la estegoimagen con la imagen cubierta)
Superclases:
Subclases:
Responsabilidades
package pruebas; import java.awt.image.BufferedImage; import java.awt.image.Raster; public class Pruebas()

public static double logbase10(double x)
public static double PSNR(BufferedImage image1, BufferedImage image2)
public static double CQ(BufferedImage image1, BufferedImage image2)
public static double SC(BufferedImage image1, BufferedImage image2)
public static double IF(BufferedImage image1, BufferedImage image2)
public static double AD(BufferedImage image1, BufferedImage image2)

Tabla 2.13 Tarjeta CRC: Clase Prueba

En esta fase de la metodología se explicarán los estándares para la codificación utilizados durante el desarrollo del software. También se explicará cómo se planificó cada una de las tareas necesarias para darle solución a los objetivos propuestos en la introducción a la investigación.

2.4 Arquitectura.

Patrones de diseño.

Para organizar el desarrollo del componente se aplicaron diversos patrones de diseño, los cuales tiene como objetivo mantener guiado el desarrollo de un sistema de software. A continuación, se describen los mismos:

Patrones GRASP:

- Experto: Este patrón, tiene como objetivo que la responsabilidad de realizar una labor sea de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. Si se realiza bien la aplicación de este, los sistemas tienden a ser más fáciles de entender, mantener y ampliar, y existen más oportunidades para reutilizar componentes en futuras aplicaciones. Este patrón es utilizado en la aplicación cuando se está dando todas las funcionalidades a las clases que se crean.

- **Creador:** Guía la asignación de responsabilidades relacionadas con la creación de objetos, una tarea muy común. La intención básica del patrón Creador es encontrar un creador que necesite conectarse al objeto creado en alguna situación. Eligiéndolo como el creador se favorece el bajo acoplamiento. Este se utiliza cuando se crea algún objeto que sea o no de la clase en la que se quiere crear, pues va a contar con todos los valores posibles para crear dicho objeto, así como para poder utilizarlo.

- **Bajo Acoplamiento**: Debe haber pocas dependencias entre las clases. Si todas las clases dependen de todas no se puede extraer software de forma independiente y reutilizarlo. Este patrón es un principio que asigna la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Esto facilita la centralización de las actividades.

Se concluye que XP como metodología de desarrollo ágil se adapta casi perfectamente a las condiciones existentes para el desarrollo del componente ya que no es un sistema de software de gran tamaño, el equipo de desarrollo estaba compuesto únicamente por una persona y había además constante comunicación con el Usuario. Se determinó que la utilización de Java como lenguaje de programación en el entorno de desarrollo Eclipse facilitaría la integración con la versión actual del sistema 4ª-Fractal que se ha desarrollado en su totalidad utilizando estas dos herramientas.

2.5 Pruebas al sistema.

Las pruebas no son más que procesos que permiten verificar y revelar la calidad de un producto de software. Son utilizadas para identificar posibles fallos de implementación y calidad de un sistema.

Uno de los pilares de la metodología XP es el proceso de pruebas. XP promueve a probar tanto como sea posible reduciendo así el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. Esta metodología divide las pruebas del sistema en pruebas unitarias, pruebas funcionalidad y pruebas de aceptación.

Las pruebas de funcionalidad y de aceptación son creadas en base a las historias de usuarios, en cada ciclo de la iteración del desarrollo y permiten confirmar que la historia ha sido implementada correctamente. El cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada y es el responsable de verificar que los resultados de estas pruebas sean correctos. Asimismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de funcionalidad y de aceptación.

Durante la implementación de esta aplicación se diseñaron un conjunto de casos de prueba para comprobar su funcionamiento de acuerdo a los requerimientos descritos en las historias de usuarios (HU), que fueron definidas en el capítulo anterior. A continuación, se exponen algunas de las pruebas de funcionalidad y de aceptación realizadas.

Caso de Prueba	
Código: CP_HU1	Historia de Usuario: HU1
Descripción: Esta prueba consiste en verificar la correcta ejecución de codificación de la imagen, verificando que se inserte un mensaje y la clave.	
Condiciones de ejecución: Cualquier usuario puede llenar cada espacio de los formularios para dar paso a la ejecución del programa.	
Entrada/Pasos de ejecución: Se carga la imagen y luego se introduce la información correcta en los espacios de la ventana en dependencia a la operación a realizar.	
Resultado esperado: Si se llenan todos los campos correctamente se genera la estegoimagen. En caso contrario muestra un mensaje de error.	
Evaluación de la prueba: Bien	

Tabla 2.14 Caso de prueba CP_HU1

Caso de Prueba	
Código: CP_HU2	Historia de Usuario: HU2
Descripción: Prueba para verificar la correcta ejecución de extracción del mensaje.	
Condiciones de ejecución: Cualquier usuario puede cargar la imagen que contenga	

el mensaje, para obtener dicho mensaje se debe ingresar la clave correcta.
Entrada/Pasos de ejecución: Se carga la imagen se introduce la clave y luego se decodifica la imagen para mostrar el mensaje.
Resultado esperado: Si se introduce la clave correcta, se muestra el mensaje codificado. De lo contrario muestra un mensaje de error.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 2.15 Caso de prueba CP_HU2.

Caso de Prueba de Aceptación	
Código: 5	Historia de Usuario: 5
Nombre: Prueba	
Descripción: Prueba para verificar que la imagen y la estegoimagen sean idénticas a un 100% antes de guardar.	
Condiciones de ejecución: Cualquier imagen modificada antes de guardar se compara con su imagen original, además con pruebas de calidad y perceptibilidad.	
Entrada/Pasos de ejecución: Una vez modificada la imagen e incorporada la información su nueva cadenada de pixeles deberán ser idénticas a la imagen original.	
Resultado esperado: al generarse la estegoimagen se verifica que sean idénticas a los ojos humanos, además que los valores de calidad sean correctos, mostrando los resultados de cada una de las pruebas. En caso de que una prueba no cumpla con los parámetros de calidad se le mostrara una alerta al usuario.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 2.16 Caso de prueba CP_HU3

Con la realización de los casos de pruebas que permitieron comprobar la integridad y funcionamiento del sistema. Se realizaron pruebas de funcionalidad que determinaron si se estaba en condiciones de continuar avanzando y al final se logró una herramienta robusta y menos proclive a errores en su funcionamiento, además del control de la calidad del software utilizando la metodología de desarrollo ágil XP, logrando una satisfacción del producto desarrollado.

CONCLUSIONES GENERALES

Con la realización del presente trabajo se ha dado cumplimiento al objetivo general propuesto para esta investigación, se concluye además que:

- Luego del estudio de las diferentes vías para dar solución a nuestro objetivo utilizando bibliotecas libres de código abierto para la programación orientada a objeto (POO), la implementación del algoritmo AA-Fractal sustenta la seguridad a la información que viaja por un canal de comunicación a través de un medio portador, lo que hace más factible para dar solución a la problemática planteada y brindar las funcionalidades necesarias para la herramienta 4ª-Fractal y utilizarla para satisfacer cada una de las historias de usuario planteadas.
- Se determinó que XP como metodología de desarrollo ágil se adapta perfectamente a las condiciones necesarias para el desarrollo del componente, ya que no se iba a desarrollar un sistema de software de gran tamaño, el equipo de desarrollo estaba compuesto únicamente por una persona y había, además, constante comunicación con el usuario.
- Se determinó que la utilización de Java como lenguaje de programación en el entorno de desarrollo Eclipse, lo que facilitó la integración con la versión actual del sistema 4ªFractal que ha sido desarrollado en su totalidad utilizando estas dos herramientas.
- Fueron realizadas exitosamente las tareas definidas para llevar a cabo cada una de las historias de usuario, evidenciándose la efectividad de cada una de ellas en las pruebas de aceptación que le fueron aplicadas al sistema.

Referencias Bibliográficas

[1] Aura, T. (1996). Practical invisibility in digital communication. *Springer*.

[2] B, H. C. (2001). Steganalysis and A. Westfeld. F5 a steganographic algorithm. *Springer*.

[3] Barnsley, M. F., & S, D. (1985). Iterated function systems and the global construction of fractals.

[4] C, C. (2011). *Digital steganography. Encyclopedia of Cryptography and Security.*

[5] C, J. M., & F D, P. B. (1998). Self-similarity based image watermarking. *EUSIPCO*.

[6] Choudhury, B. S., Z , K., N , M., & S, R. (2019). A survey of fixed point theorems under pata-type conditions. *Bulletin of the Malaysian Mathematical Sciences*.

[7] Cox, I., Miller, M., & Bloom, J. (2007). Digital watermarking and steganography.

[8] Daraee, F., & S , M. (2014). Watermarking in binary document images using fractal codes.

[9] Deymonnaz, P. A. (2012). Analisis de vulnerabilidades esteganográficas en protocolos de comunicación IP y HTTP.

[10] E, J. A. (1992). Image coding based on a fractal theory of iterated contractive image transformations. *IEEE transactions on image processing*.

[11] F S, A. (2012). A proposed encoding and hiding text in an image by using fractal image compression. *International Journalon Computer Scienceand Engineering*.

[12] Granas, A., & Dugundji, J. (2013). Fixed point theory. *Springer Science & Business Media*.

[13] Guevara, J. M. (s.f.). *Fundamento de la programacion en Java.* Madrid: EME.

[14] Gulati, K., & V , G. (2003). Information hiding using fractal encoding.

[15] Gupta, R., D, M., R K , T., & R , K. (2005). Digital image encoding scheme using fractal approach. *IEEE*.

[16] H Al, R. (2014). Steganography using fractal images technique. *IOSRJournal of Engineering*.

[17] Jacquin, A. E. (1992). Image coding based on a fractal theory of iterated contractive image transformations. *IEEE transactions on image processing*.

[18] Katz, J., & J Menezes, A. (1996). Handbook of applied cryptography. *CRC*.

[19] Katzenbeisse, S., & F Petitcolas, P. (2000). *Digital watermarking.* London: Artech House.

[20] Kundur, D., & K, A. (2003). Practical internet steganography.

[21] Labardí, A. P. (2019). *Algoritmo Esteganográfico en el Dominio de las Transformada Fractal.* La Habana.

[22] *Metodologías de Desarrollo*. (20 de enero de 2020). Obtenido de http://www.marblestation.com/?p=644.

[23] *MK@Marketing & Web*. (10 de mayo de 2019). Obtenido de 10 Lenguajes de Programación más usados en el 2018: https://www.marketingandweb.es/marketing/lenguajes-de-programacion-mas-usados/

[24] Paute, J. J. (1996). Using fractal compression scheme to embed a digital signature into an image.

[25] Peinado, F. (8 de febrero de 2020). *Programación Orientada a Objeto con Java en Eclipse*. Obtenido de LPS: http://www.federicopeinado.es

[26] Petitcolas, F. A., R J , A., & M G , K. (1999). Information hiding a survey. *IEEE*.

[27] R, A. (1996). Stretching the limits of steganography. *International Workshop on Information Hiding*.

[28] Ramírez, L. A. (2012). *Componente para la reproducción avanzada de video IP.* Habana.

[29] V, N. A. (2007). Esteganografía en contenido multimedia.

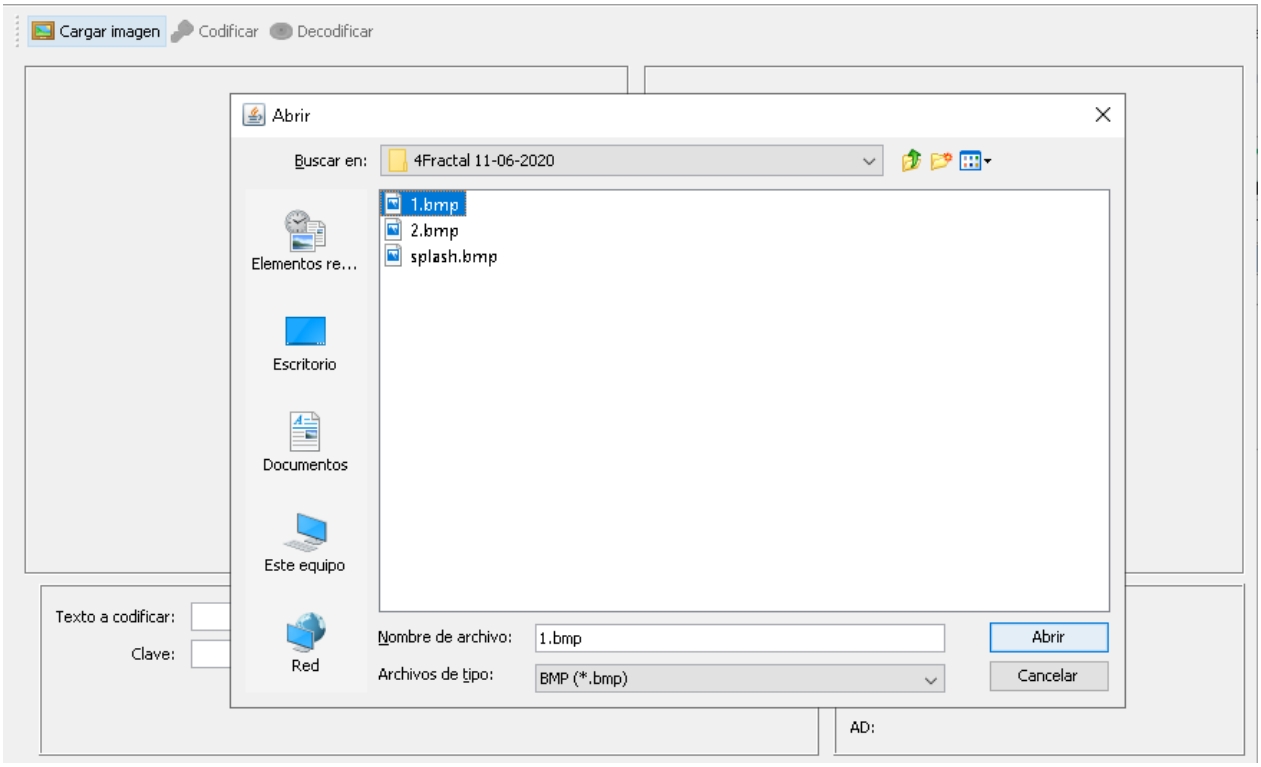
[30] Valladarez, S. M., Gaitan, M. E., & Pérez Reyes, N. N. (s.f.). METODOLOGIA ÁGIL DE DESARROLLO DE SOFTWARE PROGRAMACION EXTREMA.

[31] Vishwakarma, D., S, M., & S, J. (2012). Efficient information hiding technique using steganography. *IJETAE*.

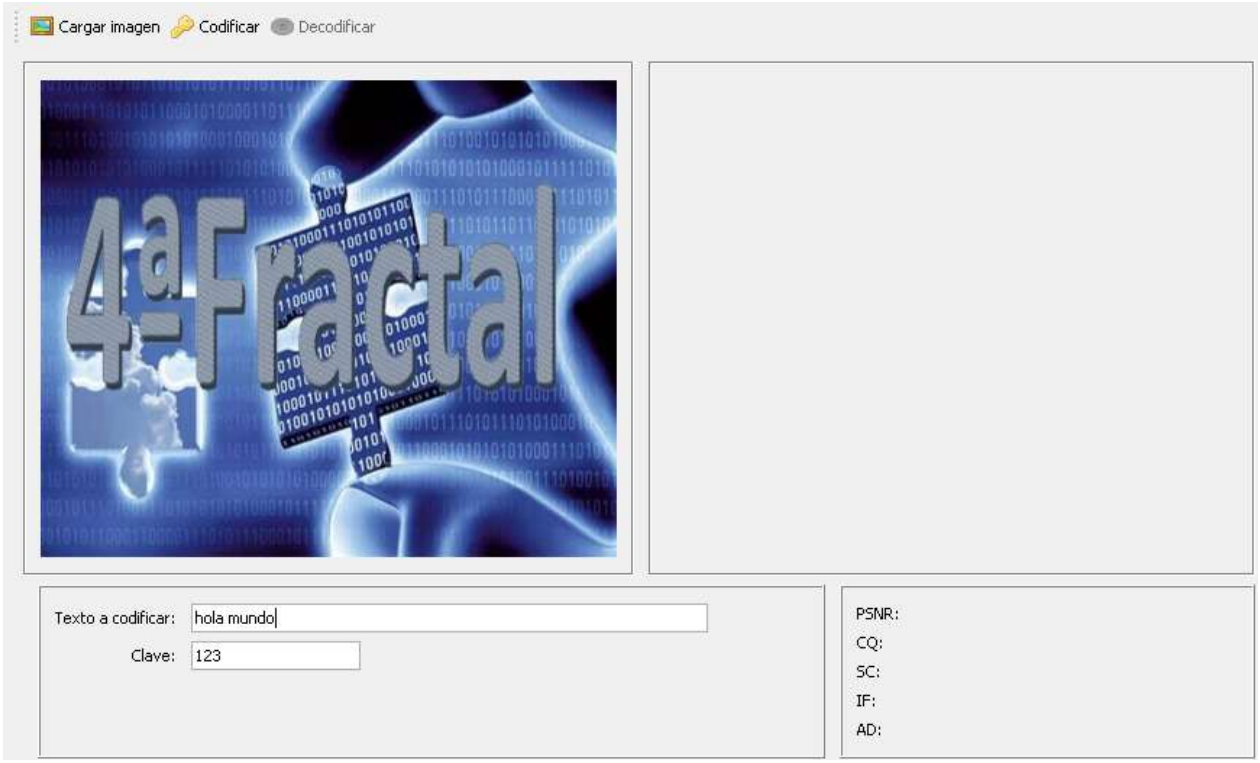
[32] Westphal, K. (2003). Steganography revealed. SecurityFocus. *Retrieved January*.

[33] Yao, Z. (2003). Fixed point in fractal image compression as watermarking. *IEEE*.

ANEXO 1 Cargar Imagen.



Anexo 2 Extraer mensaje.



Anexo 3 Calidad de imagen