**Sistema de decodificación**

**Documento de Especificación de Requerimientos de Software**

*Integrante:*

*Ángel Felipe Ramírez Molina*

*Luis Edwin Gutiérrez Acevedo*

**1. Introducción**

Este informe presenta el desarrollo de una solución computacional para el Desafío 1 del curso de Informática II, correspondiente al semestre 2025-1. El reto consiste en revertir una serie de transformaciones aplicadas a una imagen BMP de 24 bits, con el fin de recuperar su estado original a partir de una imagen alterada, una máscara de color y archivos de rastreo. Las transformaciones incluyen operaciones a nivel de bits como rotaciones, desplazamientos y XOR con imágenes generadas aleatoriamente, seguidas de procesos de enmascaramiento. El orden de las operaciones es desconocido, lo que convierte este desafío en un ejercicio de ingeniería inversa que pone a prueba las habilidades de análisis, deducción y programación en lenguaje C++ bajo el framework Qt.

**1.1. Propósito**

El propósito de este informe es describir el funcionamiento del código desarrollado, explicar la naturaleza del problema propuesto y presentar un análisis personal basado en la experiencia obtenida durante el proceso de solución. Asimismo, se detallarán los aspectos técnicos implementados, los algoritmos utilizados y las dificultades enfrentadas.

**1.2. Ámbito del sistema**

El sistema desarrollado está diseñado para recibir como entrada una imagen distorsionada (ID), una imagen de referencia (IM), una máscara (M) y un conjunto de archivos de enmascaramiento. A partir de esta información, el sistema debe identificar el tipo y orden de las transformaciones aplicadas, verificar los enmascaramientos realizados y reconstruir la imagen original mediante operaciones a nivel de bits.

**1.3. Definiciones, Acrónimos y Abreviaturas**

BMP: Bitmap, formato de imagen sin compresión.

RGB: Red, Green, Blue (colores primarios de la imagen).

ID: Imagen Distorsionada (resultado final de las transformaciones).

IM: Imagen Máscara o de referencia para operaciones XOR.

M: Máscara de enmascaramiento.

XOR: Operación lógica exclusiva OR.

Desplazamiento: Operación que desplaza bit

Rotación: Operación que mueve los bits como un circulo

**1.4. Referencias**

1. Enunciado del Desafío 1, Informática II – Universidad de Antioquia, 2025.
2. Documentación oficial de C++ y Qt framework.
3. Materiales de clase: estructuras de control, punteros, manejo de bits y arreglos dinámicos.

**2. Descripción general**

**2.1. Perspectiva del sistema**

El sistema desarrollado forma parte de una solución académica orientada a la ingeniería inversa de imágenes distorsionadas mediante transformaciones a nivel de bits. Se trata de un programa autónomo implementado en C++ bajo el framework Qt, que interactúa con imágenes BMP de 24 bits, archivos de rastreo en formato .txt, y estructuras de datos generadas dinámicamente en tiempo de ejecución. Su objetivo principal es revertir dichas transformaciones, deduciendo su orden y tipo a partir de la información suministrada, y reconstruir así la imagen original.

**2.2. Funciones del programa**

El programa desarrollado cumple con las siguientes funciones clave:

1. Leer y analizar imágenes BMP de entrada (ID e IM).
2. Leer máscaras de enmascaramiento (M) y archivos .txt de rastreo.
3. Ejecutar operaciones a nivel de bits (XOR, rotación, desplazamiento).
4. Verificar el resultado del enmascaramiento mediante comparación con los archivos proporcionados.
5. Probar diferentes combinaciones y secuencias de operaciones para reconstruir la imagen original.
6. Exportar el resultado final como una nueva imagen BMP reconstruida.

**2.3. Características de los usuarios**

El producto está dirigido principalmente a estudiantes y profesores del área de ingeniería informática o afines, con conocimientos básicos e intermedios en programación en C++, estructuras de datos, manipulación de imágenes y lógica computacional. Se espera que el usuario tenga familiaridad con el manejo de archivos, operaciones a nivel de bits y análisis algorítmico.

**2.4. Restricciones**

1. El desarrollo del sistema está sujeto a las siguientes restricciones:
2. El lenguaje de implementación debe ser C++ utilizando el framework Qt.
3. No se permite el uso de estructuras ni bibliotecas STL.
4. Se debe hacer uso de punteros, arreglos y memoria dinámica.
5. La solución debe operar sobre imágenes BMP de 24 bits, sin soporte para otros formatos.
6. Las transformaciones están limitadas a operaciones XOR, rotaciones y desplazamientos de hasta 8 bits.

**2.5. Suposiciones y dependencias**

1. Se asume que los archivos de entrada (ID, IM, M y los .txt) están bien formateados y son coherentes entre sí.
2. Se presupone que las transformaciones fueron realizadas exclusivamente mediante operaciones permitidas por el desafío.
3. La ejecución del sistema depende del entorno Qt, por lo que debe compilarse y ejecutarse en un entorno compatible.
4. La validez de la reconstrucción se basa en la comparación con los archivos de rastreo, por lo que cualquier inconsistencia en estos afectaría el resultado

**3. Requerimientos específicos**

**3.1. Interfaz**

La interfaz del sistema es de tipo consola, diseñada para facilitar la interacción del usuario mediante la ejecución de comandos básicos y la visualización de resultados en texto. Se incluyen mensajes claros para cargar archivos BMP, archivos de rastreo .txt y la máscara, así como para mostrar el avance del proceso de reconstrucción de la imagen

**3.2. Requisitos funcionales**

1. El sistema debe ser capaz de cargar correctamente imágenes BMP de 24 bits y extraer sus valores RGB.
2. Debe leer y procesar los archivos .txt de rastreo, interpretando los desplazamientos y las sumas de enmascaramiento.
3. Implementar operaciones de transformación a nivel de bits: XOR, rotación y desplazamiento de bits.
4. Verificar los resultados de las operaciones comparándolos con los datos de rastreo para deducir el tipo de transformación realizada.
5. Reconstruir de forma progresiva la imagen original aplicando las transformaciones en el orden correcto.
6. Exportar la imagen reconstruida en formato BMP.

**3.4. Otros requisitos**

1. Documentar de forma adecuada el código fuente, explicando las principales funciones y estructuras de control utilizadas.
2. Crear un video de presentación donde se explique el análisis del problema, la arquitectura de la solución, ejemplos de ejecución y el razonamiento detrás de las decisiones de programación.
3. Asegurar la disponibilidad del proyecto mediante la entrega de enlaces al repositorio Git y al video de sustentación antes de las fechas establecidas.
4. Asistir obligatoriamente a la sustentación oral para la validación final del desafío.

**4. Apéndice**

Diagrama

El contenido generado por IA puede ser incorrecto.