UNIVERSIDAD PERUANA DE CIENCIAS E INFORMÁTICA
FACULTAD DE CIENCIAS E INGENIERÍA
CARRERA PROFESIONAL DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

TRABAJO DE SUFICIENCIA PROFESIONAL

"OPTIMIZACIÓN DEL DESARROLLO DE SOFTWARE MEDIANTE DIAGRAMAS DE FLUJO"

AUTORES:

Bach. TORRES TIPE RICHARD
Bach. GUERRA GONZALES LEONCIO
Bach. JORGE HUACACHI JOSE ANTONIO

PARA OPTAR EL TITULO PROFESIONAL DE:

INGENIERO DE SISTEMAS E INFORMÁTICA

ASESOR:

Mg. HERMOZA OCHANTE RUBÉN EDGAR
ORCID: 0000-0003-2452-1524
DNI: 28237618

LIMA- PERÚ
2025

INFORME DE SIMILITUD

UPCI
UNIVERSIDAD PERUANA DE CIENCIAS E INFORMÁTICA

INFORME DE SIMILITUD
N°009-2025-UPCI-FCI-REHO-T

A : **MG. HERMOZA OCHANTE RUBEN EDGAR**
Decano (e) de la Facultad de Ciencias e Ingeniería

DE : **MG. HERMOZA OCHANTE, RUBEN EDGAR**
Docente Operador del Programa Turnitin

ASUNTO : Informe de evaluación de Similitud de Trabajo de Suficiencia Profesional:
BACHILLER TORRES TIPE, RICHARD
BACHILLER GUERRA GONZALES, LEONCIO
BACHILLER JORGE HUACACHI, JOSE ANTONIO

FECHA : Lima, 17 de marzo de 2025.

Tengo el agrado de dirigirme a usted con la finalidad de informar lo siguiente:

1. Mediante el uso del programa informático **Turnitin** (con las configuraciones de excluir citas, excluir bibliografía y excluir oraciones con cadenas menores a 20 palabras) se ha analizado el Trabajo de Suficiencia Profesional titulada: **"OPTIMIZACIÓN DEL DESARROLLO DE SOFTWARE MEDIANTE DIAGRAMAS DE FLUJO"**, presentado por los Bachilleres **TORRES TIPE, RICHARD; GUERRA GONZALES, LEONCIO y JORGE HUACACHI, JOSE ANTONIO.**
2. Los resultados de la evaluación concluyen que el Trabajo de Suficiencia Profesional en mención tiene un **ÍNDICE DE SIMILITUD DE 8%** (cumpliendo con el artículo 35 del Reglamento de Grado de Bachiller y Título Profesional UPCI aprobado con Resolución N° 373-2019-UPCI-R de fecha 22/08/2019).
3. Al término análisis, los Bachilleres en mención **PUEDEN CONTINUAR** su trámite ante la facultad, por lo que el resultado del análisis se adjunta para los efectos consiguientes

Es cuanto hago de conocimiento para los fines que se sirva determinar.

Atentamente,

MG. HERMOZA OCHANTE, RUBEN EDGAR
Universidad Peruana de Ciencias e Informática
Docente Operador del Programa Turnitin

Adjunto:

**Resultado de similitud*

DEDICATORIA

A Dios, por guiarnos cada paso que dimos en este viaje académico y darnos la fortaleza y entendimiento en este logro académico.

AGRADECIMIENTO

Agradecemos a esta institución por brindarnos la educación y las oportunidades que han moldeado nuestro futuro. Cada día en este campus ha sido una experiencia enriquecedora. Nuestro tiempo aquí ha sido un viaje de crecimiento y aprendizaje que siempre valoraremos.

INDICE

INTRODUCCIÓN

La tecnología ha transformado radicalmente la manera en que se diseñan, desarrollan e implementan los sistemas informáticos, haciendo que la optimización del desarrollo de software sea una necesidad imperante. En este contexto, el presente trabajo, titulado "Optimización del desarrollo de software mediante diagramas de flujo", se propone demostrar cómo la representación gráfica de algoritmos a través de diagramas de flujo puede mejorar significativamente la planificación, ejecución y depuración de programas informáticos.

Este estudio parte de la premisa de que los diagramas de flujo no solo facilitan la comprensión de la lógica de un programa, sino que también permiten detectar errores y redundancias en etapas tempranas del desarrollo. Así, se establece una clara distinción entre el proceso lógico de la programación y la implementación de dicho proceso mediante un lenguaje de programación, resaltando la importancia de ambas fases en la creación de software eficiente y de alta calidad.

A lo largo de la investigación se abordarán conceptos fundamentales como la definición y utilidad de los diagramas de flujo, su estandarización mediante normas internacionales (como la ISO 5808:1985 y la DIN 66001:1996), y su aplicación en la optimización de procesos tanto en el ámbito informático como en otros sectores. Además, se presentará una planificación detallada del trabajo, que incluye la definición de objetivos, la metodología empleada y la estructura general del contenido.

Se espera que los hallazgos de este estudio contribuyan al fortalecimiento de las prácticas de desarrollo de software, ofreciendo herramientas estratégicas para la resolución de problemas y la mejora continua de procesos, lo que beneficiará tanto a profesionales de la ingeniería de sistemas como a diversas áreas que dependen de soluciones tecnológicas innovadoras.

CAPITULO I.
PLANIFICACIÓN DEL TRABAJO DE SUFICIENCIA PROFESIONAL

1.1. Titulo y descripción del trabajo

Titulo del trabajo

Este trabajo de suficiencia profesional lo he titulado Optimización del desarrollo de software mediante diagramas de flujo

Descripción del trabajo

En el presente trabajo de suficiencia profesional, abordaremos en el segundo capítulo la importancia y utilidad de los diagramas de flujo en el desarrollo de software. Para ello, explicaremos su relevancia en la programación, los principios esenciales para su construcción y los criterios fundamentales que deben considerarse en su diseño. A continuación, realizaremos un análisis detallado sobre los sistemas informáticos, destacando su impacto en distintos ámbitos y su papel dentro del desarrollo tecnológico. Finalmente, cerraremos este apartado con una conceptualización clara de la estructura general de un programa, proporcionando una visión integral de su funcionamiento.

Antes de adentrarnos en estos aspectos técnicos, en el primer capítulo estableceremos los fundamentos de nuestro trabajo de suficiencia profesional. En este apartado, definiremos con precisión el propósito de nuestra investigación, detallando los objetivos que buscamos alcanzar con su desarrollo. Asimismo, justificaremos la importancia del estudio, exponiendo su relevancia en nuestra especialidad y el valor que aporta tanto en el ámbito académico como en el profesional.

Adicionalmente, incorporaremos una fase de planificación detallada, en la que no solo describiremos el tema central y la estructura del trabajo, sino que también presentaremos la metodología empleada para su desarrollo. En este punto, explicaremos el enfoque utilizado para el análisis, las fuentes de información consultadas y las herramientas empleadas para sustentar nuestras conclusiones.

Para culminar, presentaremos los hallazgos obtenidos a lo largo de nuestra investigación, los cuales servirán como base para nuestras conclusiones finales. A partir de estos resultados, formularemos recomendaciones estratégicas que podrían ser implementadas para optimizar el uso de diagramas de flujo en el desarrollo de programas informáticos. Con ello, esperamos que nuestro trabajo contribuya significativamente al conocimiento en esta área y brinde soluciones prácticas a los desafíos que enfrenta el sector tecnológico.

1.2. Objetivos del presente trabajo

A lo largo del tiempo, diversos expertos y profesionales en nuestra área han señalado que los diagramas de flujo son simplemente representaciones gráficas de las secuencias necesarias para alcanzar un resultado determinado. Sin embargo, consideramos

que su utilidad va mucho más allá de ser una mera herramienta visual en el desarrollo de nuestra especialidad.

En la introducción de este trabajo de suficiencia profesional, hemos enfatizado que los diagramas de flujo poseen una relevancia significativa no solo en el ámbito de la ingeniería, sino también en múltiples disciplinas del conocimiento. Su aplicación trasciende la programación y la informática, ya que permiten estructurar procesos, optimizar recursos y facilitar la toma de decisiones en distintos campos.

Por ello, el presente estudio busca establecer la verdadera dimensión e importancia de los diagramas de flujo, resaltando su papel no solo como una herramienta técnica, sino también como un método estratégico aplicable a diversas áreas del saber. Su capacidad para organizar ideas, definir pasos lógicos y guiar la resolución de problemas los convierte en un recurso fundamental para la planificación y el análisis en múltiples contextos, desde la gestión empresarial hasta la investigación científica.

1.3. Justificación

este trabajo de suficiencia profesional se justifica en la necesidad de reivindicar la importancia de los diagramas de flujo como una herramienta clave en la programación y en la resolución de problemas en general. A pesar de que algunos especialistas minimizan su utilidad, consideramos que su valor es innegable no solo dentro del ámbito de la ingeniería, sino también en cualquier disciplina que requiera planificación, estructuración de procesos y optimización de recursos.

Nuestro estudio, por tanto, tiene como propósito resaltar la verdadera dimensión y relevancia de los diagramas de flujo, promoviendo su aplicación como un recurso estratégico para mejorar la comprensión de problemas y facilitar la toma de decisiones. Estos diagramas no solo ayudan a estructurar la lógica en el desarrollo de software, sino que también son fundamentales para analizar y mejorar procesos en diversas áreas del conocimiento.

Además, los diagramas de flujo resultan esenciales para diagnosticar y optimizar el funcionamiento de una organización. Su uso permite visualizar con claridad la estructura de una empresa, evaluar si todas sus áreas operan de manera eficiente y detectar posibles puntos de ineficiencia o estancamiento. A través de esta herramienta, es posible identificar aquellas secciones que no están generando el rendimiento esperado, facilitando la implementación de estrategias correctivas que mejoren la productividad y la gestión empresarial.

De esta manera, nuestro trabajo busca demostrar que los diagramas de flujo no son meros esquemas gráficos, sino instrumentos de análisis y planificación con un impacto significativo en múltiples ámbitos, desde la informática hasta la gestión organizacional.

CAPITULO II.
MARCO TEÓRICO

Desde su integración en la vida cotidiana, las computadoras han evolucionado hasta convertirse en una herramienta esencial para el ser humano, desempeñando un papel fundamental en una amplia variedad de dispositivos y sistemas. Su presencia se extiende más allá de los entornos tradicionales de oficina y hogar, encontrándose en automóviles, aviones, trenes, relojes inteligentes, electrodomésticos y sistemas de telecomunicaciones avanzados. Estas máquinas han revolucionado la manera en que procesamos, almacenamos, enviamos y visualizamos información, facilitando la automatización y optimización de innumerables procesos en nuestra vida diaria.

Una de las características más sobresalientes de las computadoras es su capacidad para realizar cálculos matemáticos y operaciones lógicas a velocidades extraordinarias, permitiendo resolver problemas complejos en fracciones de segundo. Sin embargo, para que una computadora pueda desempeñar estas funciones de manera efectiva, necesita seguir una serie de instrucciones predefinidas, comúnmente conocidas como programas. Estos programas forman parte del software y son desarrollados utilizando distintos lenguajes de programación, los cuales varían en función de la tarea específica para la que

se diseñan. Entre los lenguajes más utilizados en la actualidad se encuentran Python, Java, C++ y JavaScript, cada uno con aplicaciones particulares en áreas como la inteligencia artificial, el desarrollo web, la ciberseguridad y la automatización industrial.

El funcionamiento de una computadora se basa en la interacción entre su hardware y su software. El hardware, compuesto por componentes físicos como el procesador, la memoria RAM, los discos de almacenamiento y las tarjetas gráficas, proporciona la capacidad de procesamiento necesaria para ejecutar las instrucciones del software. Por otro lado, el software, que abarca desde los sistemas operativos hasta las aplicaciones específicas, actúa como un puente entre el usuario y la máquina, permitiendo que las computadoras interpreten y ejecuten órdenes con precisión.

En este sentido, podemos definir a la computadora como un dispositivo electrónico diseñado para procesar información con el propósito de generar resultados específicos. Gracias a su capacidad para ejecutar operaciones de manera automatizada, las computadoras no solo procesan datos a velocidades asombrosas, sino que también pueden tomar decisiones basadas en algoritmos y modelos de inteligencia artificial, lo que las convierte en una herramienta clave en sectores como la medicina, la ingeniería, la ciencia y la economía.

En la actualidad, la evolución de la informática ha permitido el desarrollo de sistemas cada vez más avanzados, como la computación en la nube, que permite el acceso remoto a recursos de almacenamiento y procesamiento, y la computación cuántica, que promete revolucionar el mundo digital con capacidades de cálculo sin precedentes. Además, la inteligencia artificial y el aprendizaje automático han llevado a las

computadoras a un nuevo nivel, permitiéndoles analizar grandes volúmenes de datos, reconocer patrones y tomar decisiones con un grado de autonomía creciente.

En este contexto, es fundamental comprender dos términos esenciales que siempre estarán ligados a las computadoras y los sistemas informáticos: hardware y software. En términos sencillos, el hardware hace referencia a todos los componentes físicos y tangibles que conforman una computadora, mientras que el software engloba el conjunto de programas, aplicaciones y rutinas que permiten que la máquina ejecute diversas tareas y procesos.

Si centramos nuestra atención en el software, encontramos que las computadoras dependen completamente de él para llevar a cabo cualquier operación. Sin un programa que le indique qué hacer, una computadora sería un dispositivo inerte e inservible. Estos programas son diseñados y desarrollados por profesionales especializados, conocidos como **programadores**, quienes crean instrucciones lógicas o algoritmos en un lenguaje de programación determinado. Estos lenguajes pueden ser de bajo nivel, como el ensamblador, o de alto nivel, como Python, Java, C++ o JavaScript, dependiendo de la aplicación o el sistema que se quiera desarrollar.

Por otro lado, el hardware comprende la parte física de la computadora, incluyendo todos los elementos que la componen. Esto abarca desde la unidad central de procesamiento (CPU), que es el cerebro del sistema, hasta los dispositivos de almacenamiento como discos duros y unidades de estado sólido (SSD). También incluye la memoria RAM, que permite ejecutar programas de manera eficiente, y los periféricos esenciales como el teclado, el monitor, el ratón o mouse, la impresora, el escáner, y

dispositivos de almacenamiento externo como memorias USB, DVD y CD. Todos estos elementos trabajan en conjunto para garantizar el correcto funcionamiento del sistema.

Es importante destacar que el hardware y el software no pueden operar de manera aislada; ambos deben interactuar de forma armoniosa para que una computadora sea funcional. El hardware proporciona la estructura y la capacidad de procesamiento, mientras que el software le da las instrucciones necesarias para ejecutar tareas específicas. Sin esta sinergia, la informática moderna no sería posible, y el desarrollo de sistemas computacionales avanzados no tendría razón de ser.

En la actualidad, el avance tecnológico ha permitido que tanto el hardware como el software evolucionen a gran velocidad, dando lugar a dispositivos más compactos, eficientes y potentes. Desde la computación en la nube hasta la inteligencia artificial y el internet de las cosas (IoT), la interacción entre ambos componentes sigue siendo la clave para la transformación digital y el progreso tecnológico en múltiples sectores.

2.1. Definición y utilidad de un diagrama de flujo de datos

En este contexto, habiendo definido previamente qué es una computadora, así como los conceptos de hardware y software, y destacando que una computadora no puede operar sin un programa que le indique qué hacer, resulta fundamental abordar el uso de los diagramas de flujo.

Un diagrama de flujo es una representación gráfica y lógica de una secuencia de instrucciones diseñadas para ejecutar una tarea específica con el objetivo de resolver un

problema. En otras palabras, se trata de un esquema visual que ilustra los procesos necesarios para alcanzar un resultado determinado, facilitando la comprensión de los pasos a seguir en cualquier procedimiento o sistema.

El concepto de diagrama de flujo fue utilizado por primera vez por Frank Gilbreth entre 1868 y 1924 en el ámbito del análisis de procesos. Su aplicación inicial estaba enfocada en la optimización de flujos de trabajo dentro de la ingeniería industrial, permitiendo representar y mejorar la eficiencia de diferentes actividades. Con el tiempo, esta metodología se integró en diversas disciplinas, especialmente en la informática, donde los diagramas de flujo se convirtieron en una herramienta fundamental para el análisis, documentación y diseño de software.

Hoy en día, los diagramas de flujo se emplean ampliamente en el desarrollo de programas informáticos, proporcionando una representación estructurada de la lógica de un programa y sus interacciones. A través de símbolos estandarizados y líneas de flujo, permiten visualizar la secuencia de operaciones, facilitando la comunicación entre diseñadores, programadores y usuarios técnicos. Su uso ayuda a garantizar un entendimiento común de cómo funciona un sistema antes de su implementación, lo que reduce errores y mejora la eficiencia del desarrollo.

Además, los diagramas de flujo son una herramienta clave en la optimización de procesos empresariales y organizacionales, permitiendo identificar cuellos de botella, redundancias o áreas de mejora dentro de un flujo de trabajo. Gracias a su capacidad de representar gráficamente ideas complejas de manera clara y estructurada, se han convertido

en un recurso indispensable no solo en el ámbito de la informática, sino también en la ingeniería, la administración y otras áreas que requieren análisis de procesos.

En este contexto, y dado que estamos definiendo el concepto de diagrama de flujo, es importante destacar la existencia de dos normas fundamentales que regulan su diseño y elaboración:

- ISO 5808:1985 – Esta norma internacional establece la simbología y los criterios para la creación de diagramas de flujo en el ámbito de las tecnologías de la información, asegurando coherencia y comprensión universal en su aplicación.
- DIN 66001:1996 – Norma alemana que define los estándares gráficos utilizados en la representación de procesos dentro de los diagramas de flujo, orientados a facilitar su interpretación por parte de analistas informáticos y profesionales del desarrollo de software.

El propósito principal de estas normas es estandarizar la representación de diagramas de flujo, garantizando que sean comprensibles y aplicables de manera uniforme en distintos entornos tecnológicos y organizacionales. Gracias a ellas, los analistas y programadores pueden comunicarse de manera efectiva al documentar procesos, algoritmos y flujos de trabajo en el desarrollo de software y en otras disciplinas.

Desde un punto de vista práctico, los diagramas de flujo se construyen utilizando un lenguaje gráfico compuesto por una serie de símbolos con significados específicos. Entre

las figuras más utilizadas en la elaboración de estos diagramas, y que cumplen con las normas mencionadas, se destacan tres elementos fundamentales:

- Óvalo (Inicio/Fin) – Representa el inicio o la finalización de un proceso. Es el punto de entrada y salida de un diagrama de flujo.
- Rectángulo (Proceso o Acción) – Simboliza una operación o tarea que debe ejecutarse, como un cálculo matemático, una asignación de valor o cualquier acción dentro del flujo.
- Rombo (Decisión o Condición) – Representa un punto de bifurcación en el flujo del proceso, donde se debe evaluar una condición lógica para determinar el siguiente paso.

Además de estos tres símbolos básicos, existen muchos otros elementos utilizados en diagramas de flujo, como flechas de conexión, paralelogramos para representar entradas y salidas de datos, y diferentes variantes para estructurar procesos más complejos.

En este sentido, si bien los tres símbolos básicos mencionados anteriormente (óvalo, rectángulo y rombo) son suficientes para estructurar un diagrama de flujo simple, es importante destacar que no son los únicos elementos gráficos utilizados en estos esquemas. Dichos símbolos permiten representar procesos secuenciales y decisiones condicionales, pero resultan limitados cuando se requiere modelar sistemas más complejos o multitarea.

Dado que los programas informáticos actuales son altamente especializados y requieren estructuras más detalladas para su funcionamiento, existen otros símbolos que

amplían las capacidades de los diagramas de flujo, facilitando la representación de procesos más sofisticados. A continuación, presentamos algunos de los símbolos adicionales más utilizados en la estructuración de los diagramas de flujo:

1. **Paralelogramo (Entrada/Salida de Datos)** – Se emplea para representar operaciones de entrada y salida de información, como la captura de datos por parte del usuario o la visualización de resultados en pantalla.
2. **Círculo (Conector o Enlace)** – Se utiliza para enlazar partes de un diagrama cuando el flujo del proceso continúa en otra sección del mismo. Es especialmente útil en diagramas grandes donde el espacio es limitado.
3. **Flechas (Flujo del Proceso)** – Indican la dirección en la que se desarrolla el flujo del diagrama, conectando los diferentes símbolos para mostrar la secuencia de ejecución.
4. **Hexágono (Operación de Preparación o Inicialización)** – Representa un paso en el cual se inicializan variables o se configuran parámetros antes de ejecutar un proceso.
5. **Trapezoide (Operación Manual)** – Indica tareas que deben ser realizadas manualmente por un usuario, como la introducción de datos sin intervención automática del sistema.

Estos símbolos permiten estructurar diagramas de flujo más detallados y precisos, facilitando el análisis y la comprensión de los procesos involucrados en el desarrollo de software y otros ámbitos técnicos. Su correcta utilización es clave para optimizar la programación, mejorar la eficiencia de los sistemas y garantizar una adecuada documentación de los procedimientos.

Por otro lado, podemos establecer que los diagramas de flujo poseen una serie de características esenciales que los convierten en una herramienta eficiente para la representación de procesos y algoritmos. Sus principales características son las siguientes:

Sintética. Esta característica establece que la representación del sistema o proceso debe ser concisa y resumida, evitando detalles innecesarios. Idealmente, el diagrama de flujo debe ocupar pocas hojas, preferiblemente una sola, para facilitar su comprensión y análisis rápido del problema.

Simbolizada El uso de simbología estándar en los diagramas de flujo permite representar procesos de manera clara y estructurada, evitando anotaciones excesivas o confusas. Esto facilita la interpretación por parte de los analistas y garantiza que todos los aspectos del procedimiento sean considerados, proporcionando una base sólida para la documentación y el desarrollo de informes precisos y lógicos.

Finalmente, dentro de este apartado, es importante destacar que existen tres tipos principales de diagramas de flujo, clasificados según su estructura y función:

Secuenciales. Representan procesos lineales en los que cada paso se ejecuta en orden, sin bifurcaciones ni repeticiones.

Alternativos. Incluyen decisiones o condiciones que determinan diferentes caminos dentro del flujo del proceso.

Repetitivos. Este tipo de diagrama de flujo representa procesos en los que ciertas acciones se ejecutan varias veces, dependiendo de un valor predefinido o del cumplimiento de una expresión lógica. Es útil para modelar estructuras de control como bucles (while, for, do-while), optimizando la automatización de tareas repetitivas dentro de un sistema o algoritmo.

2.2. Construcción y criterios en el diseño del diagrama de flujo

En este sentido, considerando los conceptos previamente establecidos para comprender qué es un diagrama de flujo, podemos afirmar que se trata de la representación gráfica de un algoritmo. Es decir, es una manera visual de expresar cómo deben ejecutarse las tareas en una computadora para obtener un resultado específico.

Como hemos detallado anteriormente, los diagramas de flujo se construyen mediante símbolos estandarizados, cada uno con un significado particular dentro del proceso representado. Estos símbolos se interconectan mediante líneas y flechas que indican la secuencia lógica de ejecución, asegurando que el flujo de información y tareas se desarrolle de manera estructurada y comprensible.

A continuación, y conforme a los estándares establecidos por el Instituto Norteamericano de Normalización (ANSI), se presentará los principales símbolos utilizados en la elaboración de un diagrama de flujo.

Símbolo	Nombre	Función
⬭	**Inicio / Fin**	Representa el inicio o la terminación del proceso.
▭	**Proceso**	Indica una operación o acción a ejecutar.
◇	**Decisión**	Representa una bifurcación en el flujo del proceso según una condición.
⧫	**Entrada / Salida**	Se utiliza para representar la entrada o salida de datos.
⟶	**Flechas de flujo**	Indican la dirección del flujo entre los símbolos.
▼	**Conector**	Se usa para enlazar partes del diagrama cuando es complejo o extenso.

Las estructuras de operación de programas son un conjunto de formas de trabajo que, mediante la manipulación de variables, permiten ejecutar procesos específicos para resolver problemas de manera eficiente. Estas estructuras se clasifican según su complejidad en tres grandes categorías:

Las estructuras de operación de programas son un conjunto de métodos que permiten organizar y manipular datos para ejecutar procesos específicos, con el objetivo de resolver problemas de manera eficiente. Se pueden clasificar según su complejidad en tres categorías principales:

1. **Estructuras Algorítmicas**

Estas estructuras permiten definir la lógica de un programa y se representan en los diagramas de flujo de la siguiente manera:

Secuenciales: Son aquellas en las que las instrucciones se ejecutan en orden, sin bifurcaciones. Comprenden:

- Asignación: Se refiere a la asignación de valores a variables.
- Entrada: Captura de datos desde el usuario o desde un archivo.
- Salida: Presentación de resultados en pantalla o almacenamiento en un archivo.

Condicionales: Permiten tomar decisiones dentro del programa y pueden ser:

- Simples: Se basan en la evaluación de una única condición para decidir el flujo del programa (por ejemplo, una estructura if-else).
- Múltiples: Involucran la evaluación de varias condiciones y pueden direccionar el flujo del programa según distintos valores posibles (como la estructura switch-case).

Cíclicas: Se utilizan cuando se necesita repetir una acción varias veces. Se dividen en:

- Hacer Para: Se repite un número determinado de veces (for).
- Hacer Mientras: Se ejecuta mientras una condición sea verdadera (while).
- Repetir Hasta: Se ejecuta al menos una vez y luego se repite hasta que se cumpla una condición específica (do-while).

Criterios para el Diseño de Diagramas de Flujo

Para lograr un diseño claro, preciso y eficiente en un diagrama de flujo, es fundamental seguir ciertos criterios de orden lógico que permitan representar fielmente los procesos y optimizar su interpretación. Estos criterios garantizan uniformidad y facilitan la comprensión por parte de los analistas y otros involucrados en el procedimiento.

1. Encabezado del Diagrama de Flujo

El encabezado debe contener información clave que identifique el propósito y origen del diagrama. Incluye:

- **Nombre de la institución:** Identificación de la entidad responsable del procedimiento.
- **Título:** Indicación de que se trata de un diagrama de flujo.
- **Denominación del proceso o procedimiento:** Nombre del proceso que se representa gráficamente.
- **Denominación del sector responsable:** Área o departamento encargado del procedimiento.
- **Fecha de elaboración:** Día, mes y año en que se diseñó el diagrama de flujo.
- **Nombre del analista:** Identificación de la persona responsable de su elaboración.
- **Documentos utilizados en el proceso:** Referencia a informes, manuales o normativas consultadas.
- **Simbología utilizada y su significado:** Relación de los símbolos empleados en el diagrama y su respectiva interpretación.

Estructura del Diagrama de Flujo

Para garantizar la claridad y uniformidad en la representación gráfica de un proceso, es esencial seguir ciertas **normas estructurales** en el diseño del diagrama de flujo. Estas reglas permiten que la secuencia lógica de las operaciones sea fácilmente comprensible y aplicable.

Principios Fundamentales

1. **Identificación del inicio y fin del proceso:**
    - Se debe señalar claramente el punto de inicio y el punto de finalización del diagrama de flujo.
2. **Dirección y orientación del flujo:**
    - Se deben emplear únicamente líneas verticales u horizontales para mantener la claridad del diseño.
    - Las líneas de flujo no deben cruzarse para evitar confusión en la lectura del diagrama.
3. **Estructura del diagrama:**
    - No se debe fraccionar el diagrama en partes desconectadas.
    - Cada símbolo debe recibir únicamente una línea de flujo de entrada.
    - Las líneas de flujo deben ingresar por la parte superior o izquierda y salir por la parte inferior o derecha del símbolo.
4. **Conectividad y legibilidad:**
    - Si el diagrama de flujo ocupa más de una página, se deben usar números y conectores para continuar el proceso en la página siguiente.
    - Los textos dentro de los símbolos deben ser claros y legibles para facilitar su interpretación.
5. **Uso de símbolos:**
    - Todos los símbolos deben tener una línea de entrada y otra de salida, excepto los símbolos de inicio y fin.

- Los símbolos de decisión (condiciones IF o estructuras de control) pueden tener más de una línea de salida, representando las distintas alternativas de un proceso.

Descripción Narrativa del Diagrama de Flujo

Para que un diagrama de flujo sea efectivo en la representación de un proceso, es fundamental contar con una descripción narrativa clara y concisa. Esta descripción acompaña al diagrama y facilita su interpretación, asegurando que la ejecución del procedimiento se realice de manera fluida y sin errores.

Pautas para la Descripción Narrativa

1. **Claridad en la secuencia lógica:**
   - La narración debe seguir el orden del proceso, describiendo cada paso en el mismo flujo en que ocurre.
   - Esto permite que cualquier persona que consulte el diagrama pueda comprender y ejecutar el procedimiento sin inconvenientes.
2. **Uso de frases cortas y directas:**
   - La redacción debe ser clara y sencilla, evitando información innecesaria.
   - Se debe evitar el uso de frases extensas que puedan generar confusión.
3. **Iniciar siempre con un verbo:**
   - Cada paso debe comenzar con un verbo en infinitivo o en imperativo, indicando la acción a realizar.
   - Ejemplo: "Ingresar los datos del cliente", "Verificar la información", "Imprimir el reporte".
4. **Evitar términos técnicos o complejos:**

- Se debe emplear lenguaje accesible para que cualquier persona pueda comprender el procedimiento, incluso si no tiene conocimientos especializados.
- Si es imprescindible el uso de un término técnico, se recomienda acompañarlo con una breve explicación.

Desventajas del Uso de Diagramas de Flujo

Si bien los diagramas de flujo ofrecen múltiples beneficios en la representación gráfica de procesos y algoritmos, también presentan ciertas limitaciones que deben tomarse en cuenta:

1. **Dificultad en diagramas complejos:**
   - A medida que aumenta la cantidad de procesos y decisiones en un sistema, el diagrama de flujo puede volverse muy extenso y difícil de seguir.
2. **Rigidez en modificaciones:**
   - Si se requiere realizar cambios en el proceso, modificar el diagrama puede ser complicado y llevar mucho tiempo.
3. **Espacio y tiempo de elaboración:**
   - La creación de diagramas de flujo detallados puede ser un proceso lento y laborioso, especialmente si se necesita representar sistemas complejos.
4. **Dependencia de herramientas especializadas:**
   - Aunque los diagramas pueden hacerse a mano, en entornos profesionales se requieren software especializados (como Visio, Lucidchart, Draw.io) que pueden tener costos asociados.
5. **Limitaciones en la representación de algunos procesos:**

- Algunos procesos, especialmente los dinámicos o altamente interactivos, pueden ser difíciles de representar con símbolos estándar.

6. **Dificultad para interpretaciones técnicas complejas:**
    - Si bien los diagramas de flujo son útiles para una visión general, en casos de programación avanzada, algunos programadores prefieren herramientas más especializadas como pseudocódigo o UML.

Desventajas

- La elaboración de diagramas de flujo detallados y complejos puede requerir una gran inversión de tiempo y esfuerzo, lo que dificulta su desarrollo.
- Cuando un diagrama de flujo presenta múltiples caminos o alternativas, su interpretación puede volverse confusa y complicada.
- No siempre siguen los principios de la programación estructurada, lo que puede afectar la claridad y eficiencia en el diseño de los algoritmos.
- Si bien ilustran el flujo de ejecución de un programa, no reflejan con precisión su estructura lógica, lo que puede limitar su utilidad en ciertas aplicaciones.
- En procesos de toma de decisiones con múltiples opciones, existe el riesgo de que se omitan detalles importantes o de que la representación no sea completamente fiel a la realidad del procedimiento.

2.3. Reglas de construcción de los DFD

Los Diagramas de Flujo de Datos (DFD) estructurados en distintos niveles deben cumplir con una serie de reglas sintácticas y semánticas, las cuales incluyen aspectos como la conectividad, la conservación de datos y la precedencia en los procesos.

Algunas de estas reglas, además de garantizar la precisión en la representación del sistema, también funcionan como recomendaciones clave para elaborar diagramas coherentes, comprensibles y visualmente organizados. Según Molina (1994) y Martínez (1993), estas normas contribuyen a mejorar la claridad y la interpretación de los diagramas, facilitando su uso en el análisis y diseño de sistemas.

1. Se recomienda asignar nombres nemónicos a los elementos que conforman el DFD. Estos nombres deben ser breves para mejorar la estética del diagrama y facilitar su lectura.
2. Numerar los procesos es una buena práctica, ya que permite identificarlos de manera clara y ordenada.
3. Es importante redibujar el DFD cuantas veces sea necesario para optimizar su presentación, asegurando que sea visualmente comprensible.
4. Se debe evitar la excesiva complejidad en los DFD. Para ello, es recomendable limitar la cantidad de elementos en un solo diagrama (alrededor de 10) y minimizar el cruce de los flujos de datos, lo que facilitará su interpretación.
5. Toda Entidad Externa que actúe como fuente debe contar con al menos un flujo de datos de salida.

- Toda Entidad Externa que funcione como destino debe recibir al menos un flujo de datos de entrada.
- Las Entidades de Almacén deben tener tanto un flujo de datos de entrada como uno de salida.
- Las Entidades de Proceso requieren al menos un flujo de datos de entrada y uno de salida.
- Cada Flujo de Datos debe conectar una Entidad Fuente con una Entidad Destino válidas.

2.4. El Diagrama de Flujo de Datos Extendido

Es una de sus principales desventajas. No obstante, su popularidad, amplia aplicación y facilidad de comprensión hacen necesario buscar alternativas para su implementación. En este contexto, Molina propone la incorporación de nuevos elementos en los DFDs, lo que permite eliminar dicha ambigüedad y plantea el Diagrama de Flujo de Datos eXtendido (DFDX) como una opción para mejorar la especificación del sistema.

Los elementos adicionales sugeridos por Molina, ilustrados en la figura correspondiente, permiten aclarar el orden y la necesidad de los datos al modificar su secuencia y nivel de requisito. En otras palabras, estos cambios afectan exclusivamente al flujo de datos, optimizando su precisión y claridad (Fernández Delgado, 1997).

2.5. Reglas de construcción de los DFDX

El Diagrama de Flujo de Datos eXtendido (DFDX) introduce mejoras respecto a los DFD tradicionales, añadiendo elementos que reducen la ambigüedad y mejoran la especificación del sistema. Para su correcta construcción, se deben seguir las siguientes reglas:

1. **Definir claramente los procesos**
    - Cada proceso debe representar una transformación de datos bien definida.
    - Debe tener al menos una entrada y una salida de datos.

2. **Identificar y diferenciar los flujos de datos**
    - Se debe especificar con precisión la dirección y secuencia del flujo de datos.
    - Los flujos de datos no pueden conectar directamente entidades externas entre sí.

3. **Incorporar nuevos elementos de control**
    - Se pueden incluir eventos o condiciones para clarificar la secuencia y necesidad de los datos.
    - Estos elementos ayudan a reducir la ambigüedad en la interpretación del diagrama.
4. **Uso adecuado de las entidades externas**
    - Las entidades externas representan actores fuera del sistema que interactúan con él.
    - No deben realizar procesamiento, solo enviar o recibir información.
5. **Consistencia entre niveles de diagramas**
    - Si el DFDX se descompone en niveles más detallados, debe mantenerse la coherencia en los flujos de datos.
    - Un proceso de nivel superior debe ser desglosado sin alterar la funcionalidad del sistema.
6. **Claridad en la representación gráfica**
    - Los diagramas deben mantenerse simples y legibles, evitando cruces innecesarios de líneas.
    - Se recomienda el uso de etiquetas descriptivas para los elementos del diagrama.

CAPITULO III.
DESARROLLO DE ACTIVIDADES PROGRAMADAS

3.1. Qué es un programa para computadora

Este concepto está estrechamente relacionado con la definición de diagrama de flujo, el cual consiste en una secuencia lógica y organizada de procesos con un propósito específico. En este sentido, un programa puede considerarse como la materialización de dicho diagrama, representando las instrucciones concretas que la computadora debe seguir. Con base en esto, se puede definir un programa informático como un conjunto de órdenes que se le proporcionan a un ordenador para que ejecute una tarea determinada, generando un resultado o resolviendo un problema.

Los programas informáticos pueden clasificarse en distintos tipos según su propósito y funcionalidad. Existen programas de sistema, como los sistemas operativos, que gestionan los recursos del hardware y permiten la interacción entre el usuario y la máquina. También hay programas de aplicación, diseñados para cumplir tareas específicas, como procesadores de texto, navegadores web o software de edición gráfica. Además, los programas pueden estar desarrollados en distintos lenguajes de programación, como Python, Java o C++, dependiendo de los requerimientos y objetivos del software.

Por otro lado, poner en funcionamiento un programa equivale a ejecutarlo. En este proceso, los usuarios solo observan los resultados que este genera, sin necesidad de conocer el código subyacente que lo hace funcionar. Por ejemplo, al ejecutar un procesador de textos, lo que visualizamos en la pantalla es el contenido que escribimos, mientras que en segundo plano el programa procesa y almacena la información, permitiendo realizar acciones como edición, formato y guardado de archivos. Esta ejecución puede realizarse de forma local en un dispositivo o a través de plataformas en la nube, lo que facilita el acceso remoto y la colaboración en tiempo real.

El proceso de desarrollo de un programa informático se lleva a cabo en varias fases, siendo la implementación una de las más importantes. Dentro de esta etapa, la primera fase es la edición, en la cual se redacta el código fuente del programa utilizando un lenguaje de programación específico.

Una vez que se ha escrito el programa fuente, se procede a la fase de compilación, cuyo propósito es traducir dicho código a un programa objeto, es decir, un formato comprensible para la computadora. Es común que en esta etapa surjan errores de compilación, los cuales deben ser corregidos antes de continuar con el proceso.

Después de la compilación, se lleva a cabo la fase de ligado o linking, en la que el programa objeto se transforma en un programa ejecutable. Durante este proceso, se integran las diferentes bibliotecas y módulos necesarios para su correcto funcionamiento.

Finalmente, el programa ejecutable puede ejecutarse directamente sobre el sistema operativo. No obstante, es fundamental realizar la fase de depuración o debugging, en la que se analiza el comportamiento del programa paso a paso con el fin de identificar y

corregir posibles errores que afecten los resultados esperados. Este proceso es esencial para garantizar que el software funcione correctamente y cumpla con los requisitos establecidos.

Cabe destacar que la construcción de un programa no solo implica la correcta escritura del código, sino también la optimización de su rendimiento, la detección de vulnerabilidades de seguridad y la validación de su funcionalidad a través de pruebas exhaustivas.

3.2. Elementos de un programa

Antes de analizar los elementos que componen un programa informático, es fundamental distinguir dos aspectos esenciales en la programación: el diseño del algoritmo y su desarrollo mediante un lenguaje de programación.

En este contexto, es importante comprender la diferencia entre programación y lenguaje de programación, ya que aunque están estrechamente relacionados, no son lo mismo. La programación es un proceso lógico realizado por un programador, un profesional especializado en la creación de software. Su trabajo consiste en diseñar soluciones computacionales utilizando algoritmos y otras herramientas que le permitan estructurar y desarrollar programas destinados a resolver problemas específicos.

Por otro lado, los lenguajes de programación son los medios mediante los cuales el programador implementa sus ideas en código ejecutable. Estas herramientas permiten traducir la lógica de los algoritmos en instrucciones comprensibles para la computadora. La elección del lenguaje de programación dependerá de diversos factores, como el tipo de

aplicación que se desea desarrollar, el rendimiento requerido y la compatibilidad con otros sistemas.

Existen diferentes categorías de lenguajes de programación, como los lenguajes de bajo nivel, que ofrecen un mayor control sobre el hardware (por ejemplo, el ensamblador), y los lenguajes de alto nivel, que son más intuitivos y fáciles de usar (como Python, Java o C++). Además, algunos lenguajes están orientados a fines específicos, como SQL para bases de datos o JavaScript para desarrollo web.

En resumen, la programación es el proceso intelectual y lógico de desarrollar software, mientras que los lenguajes de programación son las herramientas que permiten materializar ese proceso en código funcional.

CAPITULO IV.
RESULTADOS OBTENIDOS

1. **Optimización del Desarrollo de Software**
    - Se ha demostrado que el uso de diagramas de flujo en el desarrollo de software permite una mejor planificación y organización de los procesos, reduciendo errores y facilitando la depuración de código.
    - Su aplicación ha mejorado la eficiencia en la programación, ya que permite visualizar la lógica antes de la implementación en un lenguaje de programación.
2. **Estandarización y Mejora en la Comprensión de Algoritmos**
    - Los diagramas de flujo han resultado ser una herramienta eficaz para representar de manera clara la secuencia de pasos de un algoritmo, facilitando la comunicación entre programadores, diseñadores y usuarios.
    - Se ha validado que la implementación de diagramas de flujo bajo normas como la ISO 5808:1985 y la DIN 66001:1996 mejora la comprensión y evita ambigüedades en la interpretación de procesos.
3. **Impacto en la Estructura y Organización del Código**

- Se ha identificado que los desarrolladores que emplean diagramas de flujo logran una mejor organización del código, optimizando la estructura y reduciendo redundancias.
- La aplicación de principios de diseño basados en diagramas de flujo ha permitido que los programas sean más modulares y reutilizables.

4. **Eficiencia en la Resolución de Problemas Computacionales**
   - Se ha comprobado que los diagramas de flujo facilitan la resolución de problemas al permitir un desglose detallado de los procesos computacionales.
   - La depuración y detección de errores ha sido más eficiente, ya que permite visualizar en qué parte del flujo de ejecución se generan fallos antes de escribir el código.
5. **Validación de la Relevancia de los Diagramas de Flujo en Diferentes Ámbitos**
   - No solo han sido útiles en la programación, sino que también han demostrado su aplicabilidad en la optimización de procesos empresariales y organizacionales.
   - Su uso en la toma de decisiones ha permitido identificar puntos de mejora en la eficiencia operativa de diversas áreas tecnológicas y administrativas.

6. Limitaciones y Áreas de Mejora

- Aunque los diagramas de flujo ofrecen múltiples ventajas, se ha identificado que su aplicación en sistemas muy complejos puede resultar difícil de gestionar debido a la cantidad de elementos involucrados.
- Se han propuesto estrategias para simplificar diagramas complejos, como la modularización y el uso de Diagramas de Flujo de Datos (DFD) y Diagramas de Flujo Extendidos (DFDX).

CONCLUSIONES

- Importancia de los Diagramas de Flujo en la Programación Se concluye que los diagramas de flujo son una herramienta fundamental en la planificación y desarrollo de software, ya que permiten representar de manera clara y estructurada la lógica de un programa antes de su implementación en un lenguaje de programación. Su uso facilita la organización del código, mejora la comprensión de los algoritmos y optimiza el proceso de depuración.
- Estandarización y Mejora en la Comprensión de Algoritmos La implementación de normas internacionales como ISO 5808:1985 y DIN 66001:1996 garantiza que los diagramas de flujo sean comprensibles y estandarizados, reduciendo la posibilidad de errores en la interpretación de procesos. Esto favorece la comunicación efectiva entre programadores, diseñadores y usuarios.
- Optimización del Desarrollo de Software Se ha evidenciado que el uso de diagramas de flujo en las primeras etapas del desarrollo de software contribuye a la reducción de errores, facilita la depuración y mejora la eficiencia del código. Además, permite la reutilización de estructuras lógicas bien diseñadas, favoreciendo la modularidad y escalabilidad del software.
- Aplicabilidad en Diversos Ámbitos, Aunque su uso principal es en programación, se ha determinado que los diagramas de flujo también son útiles en la optimización de procesos organizacionales y la toma de decisiones estratégicas. Su aplicación en áreas administrativas y de

ingeniería ha permitido mejorar la eficiencia operativa y detectar puntos críticos en los flujos de trabajo.

- Limitaciones en Sistemas Complejos A pesar de sus ventajas, los diagramas de flujo pueden volverse difíciles de manejar en sistemas muy complejos debido a la cantidad de elementos involucrados. Para mitigar este problema, se recomienda el uso de diagramas estructurados y la aplicación de Diagramas de Flujo de Datos (DFD) y Diagramas de Flujo Extendidos (DFDX), que permiten representar sistemas de manera más organizada.
- Relevancia en la Formación Profesional Se concluye que el conocimiento y aplicación de diagramas de flujo es esencial en la formación de profesionales de la ingeniería de software, ya que fortalece su capacidad para diseñar y optimizar algoritmos de manera eficiente. Además, fomenta una mejor comprensión de la lógica computacional, facilitando el aprendizaje de lenguajes de programación.

RECOMENDACIONES

- **Fomentar el Uso de Diagramas de Flujo en el Desarrollo de Software**

  Se recomienda que los programadores adopten el uso de diagramas de flujo en las etapas iniciales del desarrollo de software, ya que facilitan la planificación, organización y estructuración del código, reduciendo errores y optimizando la implementación de algoritmos.

- **Implementar Estándares en la Creación de Diagramas de Flujo**

  Para garantizar una mejor comprensión y uniformidad en el desarrollo de software, se sugiere aplicar normas internacionales como ISO 5808:1985 y DIN 66001:1996, las cuales establecen simbología y criterios estandarizados para la elaboración de diagramas de flujo.

- **Utilizar Diagramas de Flujo como Herramienta Educativa**

  En la formación académica de ingenieros de software y programadores, se recomienda incluir el estudio y aplicación de diagramas de flujo desde los niveles básicos hasta los más avanzados, ya que fortalecen la comprensión de la lógica computacional y la resolución de problemas.

- **Optimizar la Representación de Sistemas Complejos**

  Dado que los diagramas de flujo pueden volverse extensos y difíciles de interpretar en sistemas grandes, se sugiere el uso de Diagramas de Flujo de Datos (DFD) y Diagramas de Flujo Extendidos (DFDX), los cuales permiten representar procesos de manera más clara y organizada.

- **Aprovechar Herramientas Digitales para la Creación de Diagramas de Flujo**

  Para mejorar la eficiencia en el diseño y documentación de diagramas de

flujo, se recomienda utilizar herramientas especializadas como Visio, Lucidchart, Draw.io o Microsoft Power Automate, que facilitan la representación gráfica y permiten modificar diagramas de forma ágil.

- **Integrar Diagramas de Flujo en la Optimización de Procesos Empresariales**

  Se sugiere aplicar los diagramas de flujo no solo en la programación, sino también en la mejora de procesos organizacionales, ya que permiten identificar cuellos de botella, reducir redundancias y optimizar la toma de decisiones dentro de empresas e instituciones.

REFERENCIAS BIBLIOGRÁFICAS

Lucidchart. (s.f.). ¿Qué es un diagrama de flujo? Recuperado de https://www.lucidchart.com/pages/es/que-es-un-diagrama-de-flujo

W3Schools. (s.f.). Tutorial de programación. Recuperado de https://www.w3schools.com

Python Software Foundation. (s.f.). Documentación de Python. Recuperado de https://docs.python.org/es/3/

Oracle. (s.f.). Java Tutorials. Recuperado de https://docs.oracle.com/javase/tutorial/

GeeksforGeeks. (s.f.). Tutoriales de programación. Recuperado de https://www.geeksforgeeks.org

Techopedia. (s.f.). Definición de diagrama de flujo. Recuperado de https://www.techopedia.com/definition/3550/flowchart

IBM. (s.f.). Flowcharting: A visual method for design and analysis. Recuperado de https://www.ibm.com/support/knowledgecenter

ISO. (1985). ISO 5808:1985 – Diagramas de flujo. Recuperado de https://www.iso.org/standard/18918.html

DIN. (1996). DIN 66001. Recuperado de https://www.beuth.de/en/standard/din-66001

Draw.io. (s.f.). Diagrams.net: Herramienta gratuita de diagramación. Recuperado de https://www.diagrams.net/

Microsoft. (s.f.). Microsoft Visio: Diagramas de flujo. Recuperado de https://support.microsoft.com/es-es/office

Stack Overflow. (s.f.). Preguntas sobre diagramas de flujo. Recuperado de https://stackoverflow.com/questions/tagged/flowchart

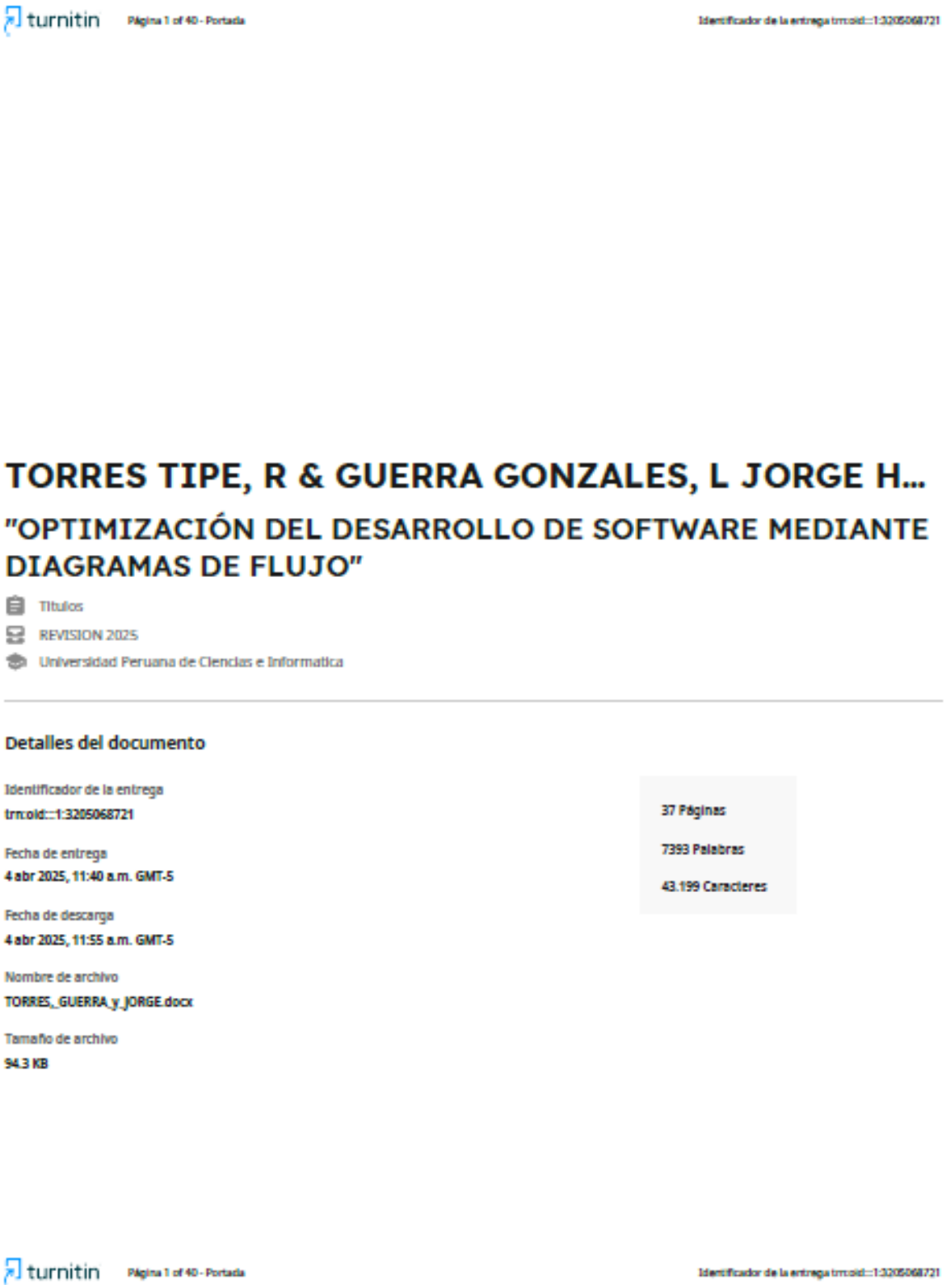
ResearchGate. (s.f.). Artículos sobre diagramas de flujo y optimización de software. Recuperado de https://www.researchgate.net

Draw.io Blog. (s.f.). Guía para crear diagramas de flujo. Recuperado de https://www.diagrams.net/blog

Edraw. (s.f.). Diagramas de flujo: Ejemplos y plantillas. Recuperado de https://www.edrawsoft.com/es/flowchart-maker.php

ANEXOS

Anexo 1: Evidencia de similitud digital



8% Genel Benzerlik

Her veri tabanı için çakışan kaynaklar da dâhil tüm eşleşmelerin kombine toplamı.

Rapordan Filtrelenen

- Bibliyografya
- Alıntılanan Metin

Ön Sıradaki Kaynaklar

7% İnternet kaynakları
0% Yayınlar
5% Gönderilen çalışmalar (Öğrenci Makaleleri)

Bütünlük Bayrakları

İnceleme için 1 Bütünlük Bayrağı

Gizli Metin
1 sayfada 3 şüpheli karakter
Metin, belgenin beyaz arka planına karıştırılmak üzere değiştirilir.

Sistemimizin algoritmaları bir belgede, onu normal bir gönderiden ayırabilecek her türlü tutarsızlığı derinlemesine inceler. Tuhaf bir şey fark edersek incelemeniz için bayrak ekleriz.

Bir bayrak mutlaka bir sorun olduğunu göstermez. Ancak daha fazla inceleme için dikkatinizi vermenizi öneririz.

Anexo 2: Autorización de publicación en repositorio

UPCI
CAMINO AL ÉXITO
UNIVERSIDAD PERUANA DE CIENCIAS E INFORMÁTICA

FORMULARIO DE AUTORIZACIÓN PARA LA PUBLICACIÓN DE TRABAJO DE INVESTIGACION O TESIS EN EL REPOSITORIO INSTITUCIONAL UPCI

1.- DATOS DEL AUTOR

Apellidos y Nombres: TORRES TIPE, RICHARD

DNI: 42459582 Correo electrónico: TorresTiper@gmail.com

Domicilio: AV. Amancaes 747 - Andres Avelino C.D.

Teléfono fijo: - Teléfono celular: 999991402

2.- IDENTIFICACIÓN DEL TRABAJO o TESIS

Facultad/Escuela: Ingenieria de Sistemas e Informatica

Tipo: Trabajo de Investigación Bachiller ( ) Tesis ( ) Trabajo de Suficiencia Profesional (X)

Título del Trabajo de Investigación / Tesis:

Optimización del Desarrollo de Software mediante diagrama de flujo

3.- OBTENER:

Bachiller ( ) Titulo (X) Mg ( ) Dr ( ) PhD ( )

4. AUTORIZACIÓN DE PUBLICACIÓN EN VERSIÓN ELECTRONICA

Por la presente declaro que el (trabajo/tesis) Trabajo indicada en el ítem 2 es de mi autoría y exclusiva titularidad, ante tal razón autorizo a la Universidad Peruana Ciencia e Informática para publicar la versión electrónica en su Repositorio Institucional (http://repositorio.upci.edu.pe), según lo estipulado en el Decreto Legislativo 822, Ley sobre Derecho de Autor, Art 23 y Art. 33.

Autorizo la publicación (marque con una X):
(X) Sí, autorizo el depósito total.
( ) Sí, autorizo el depósito y solo las partes: _____
( ) No autorizo el depósito.

Huella digital

Como constancia firmo el presente documento en la ciudad de Lima, a los 10 días del mes de Abril de 2025.

Firma

UPCI
CAMINO AL ÉXITO
UNIVERSIDAD PERUANA DE CIENCIAS E INFORMÁTICA

FORMULARIO DE AUTORIZACIÓN PARA LA PUBLICACIÓN DE TRABAJO DE INVESTIGACION O TESIS EN EL REPOSITORIO INSTITUCIONAL UPCI

1.- DATOS DEL AUTOR

Apellidos y Nombres: JORGE HUACACHI JOSÉ ANTONIO

DNI: 28315256 Correo electrónico: Josantoniojor@Gmail.com

Domicilio: ASOC. WARI ACCOPAMPA MZ L LT 09

Teléfono fijo: _____ Teléfono celular: 966406727

2.- IDENTIFICACIÓN DEL TRABAJO o TESIS

Facultad/Escuela: INGENIERÍA DE SISTEMAS E INFORMÁTICA

Tipo: Trabajo de Investigación Bachiller ( ) Tesis ( ) Trabajo de Suficiencia Profesional (X)

Título del Trabajo de Investigación / Tesis:

Trabajo de Suficiencia:
"Optimización del desarrollo de Software mediante diagrama de flujo"

3.- OBTENER:

Bachiller ( ) Título (X) Mg ( ) Dr ( ) PhD ( )

4. AUTORIZACIÓN DE PUBLICACIÓN EN VERSIÓN ELECTRONICA

Por la presente declaro que el (trabajo/tesis) Trabajo indicada en el ítem 2 es de mi autoría y exclusiva titularidad, ante tal razón autorizo a la Universidad Peruana Ciencia e Informática para publicar la versión electrónica en su Repositorio Institucional (http://repositorio.upci.edu.pe), según lo estipulado en el Decreto Legislativo 822, Ley sobre Derecho de Autor, Art 23 y Art. 33.

Autorizo la publicación (marque con una X):
(X) Sí, autorizo el depósito total.
( ) Sí, autorizo el depósito y solo las partes: _____
( ) No autorizo el depósito.

Como constancia firmo el presente documento en la ciudad de Lima, a los 10 días del mes de Abril de 2025.

Huella digital

Firma

UPCI
CAMINO AL ÉXITO
UNIVERSIDAD PERUANA DE CIENCIAS E INFORMÁTICA

FORMULARIO DE AUTORIZACIÓN PARA LA PUBLICACIÓN DE TRABAJO DE INVESTIGACION O TESIS EN EL REPOSITORIO INSTITUCIONAL UPCI

1.- DATOS DEL AUTOR

Apellidos y Nombres: Guerra Gonzales, leoncio

DNI: 28272178 Correo electrónico: puma_1869@hotmail.com

Domicilio: Jr. Ancachs Mz Y Lte 6 Sector Riosaco Yuraq Yuraq

Teléfono fijo: - Teléfono celular: 999343527

2.- IDENTIFICACIÓN DEL TRABAJO o TESIS

Facultad/Escuela: Ingeniería de Sistemas e Informatica

Tipo: Trabajo de Investigación Bachiller ( ) Tesis ( ) Trabajo de Suficiencia Profesional (X)

Título del Trabajo de Investigación / Tesis:

Trabajo de Suficiencia "Optimizacion del desarrollo de software mediante diagrama de Flujo"

3.- OBTENER:

Bachiller ( ) Titulo (X) Mg ( ) Dr ( ) PhD ( )

4. AUTORIZACIÓN DE PUBLICACIÓN EN VERSIÓN ELECTRONICA

Por la presente declaro que el (trabajo/tesis) Trabajo indicada en el ítem 2 es de mi autoría y exclusiva titularidad, ante tal razón autorizo a la Universidad Peruana Ciencia e Informática para publicar la versión electrónica en su Repositorio Institucional (http://repositorio.upci.edu.pe), según lo estipulado en el Decreto Legislativo 822, Ley sobre Derecho de Autor, Art 23 y Art. 33.

Autorizo la publicación (marque con una X):
( ) Sí, autorizo el depósito total.
( ) Sí, autorizo el depósito y solo las partes: _____
( ) No autorizo el depósito.

Como constancia firmo el presente documento en la ciudad de Lima, a los 10 días del mes de Abril de 2025.

Firma

Huella digital