**Explicación Descriptiva y Sustentación Conceptual**

**Rene Cardona**

**David Medina**

**Universidad de Cartagena**

**Programación orientada a objetos**

**Ingeniería de software**

**Profesor: John Carlos Arrieta**

**2022**

**Tabla de contenido**

[**Introducción**](#_jxroaavmiq4) **2**

[**Objetivos**](#_jtp8isl4ku4q) **3**

[Objetivo General](#_dabnstvsbvmo) 3

[Objetivo Específico](#_lrlbwq4jcsk0) 3

[**Justificación**](#_mjowxpju45cl) **3**

[**Desarrollo**](#_7ywso9pi5rua) **4**

[**Sistema para holding de empresas**](#_t6jw1iuuqdf) **4**

[Requerimientos funcionales:](#_g7m5vg15e251) 4

[Clases y propiedades:](#_vjkltf8kaz48) 5

[**Síntesis y Argumentación Individual**](#_lx3bu6s9ap4) **6**

[**Bibliografía**](#_4ggya67l5co2) **6**

# **Introducción**

Los sistemas de almacenamiento de datos e información siempre ha tenido una gran importancia a la hora de administrar un empresa, antes de la llegada del software moderno la empresas utilizaban métodos rústicos, poco efectivos, con bajos niveles de seguridad y altos costes de mantenimiento, apesar de que cumplian su función inherentemente sufrían de los problemas ya mencionados que dificultaban el manejo y análisis de datos para administrar una empresa de la mejor manera posible, las empresas tenían que andar con más precaución en caso de que sus sistemas de almacenamiento sufrieran de cualquier imprevisto como la pérdida o robo de datos lo cual hacía que el riesgo y los costos de montaje, mantenimiento y desplazamiento sean muy altos. Hoy en día gracias al software moderno las desventajas, riesgos y costos antes mencionado se reducen en grandes cantidades, los sistemas de almacenamiento digital actuales y bien estructurados poseen una alta seguridad, fácil mantenimiento y fácil manejo para las empresas que incrementa la facilidad de administración de la información y además de eso reduce los riesgos y costos de pérdida, robo, mantenimiento y almacenamiento de datos.

# **Objetivos**

## **Objetivo General**

Crear una sistema de almacenamiento de datos que cumpla la función de almacenar la información que el cliente pide por medio de los requerimientos funcionales.

## **Objetivo Específico**

Analizar los requerimientos funcionales que el cliente expone para poder construir el producto de forma efectiva.

Identificar las clases que conformarán el programa, sus atributos o propiedades y sus operaciones o métodos para tener de manera organizada los pilares que harán que el producto funcione.

Definir la relación y cardinalidad entre las clases previamente identificadas mediante un diagrama UML para que al conectarlas se tenga de forma visible y ordenada la estructura del programa.

# **Justificación**

Por medio de este trabajo damos evidencia de que obtuvimos los conocimientos necesarios para poner en práctica la abstracción para planificar un sistema de software orientado a objetos, el cual es la base para desempeñarnos laboralmente en el mundo real como ingenieros de software.

Poniendo en práctica estos conocimientos podemos crear la estructura con la cual se desarrollará un sistema de software para un holding de empresas en el que estas puedan guardar sus datos de forma organizada lo que los llevará a agilizar todos los procesos que se realizaban de forma manual.

# **Desarrollo**

## **Sistema para holding de empresas**

Un holding de empresas desea tener una base de datos referente a las empresas que posee, sus vendedores, así como los asesores que trabajan en el holding, para esto empezamos en este trabajo identificando los requerimientos funcionales que el sistema debe satisfacer.

### Requerimientos funcionales:

Los requerimientos funcionales de un sistema son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones.

Por lo general, estos deben incluir funciones desempeñadas por pantallas específicas, descripciones de los flujos de trabajo a ser desempeñados por el sistema y otros requerimientos de negocio, cumplimiento, seguridad u otra índole.

teniendo esto en cuenta nos fuimos a la información que nos daban e identificamos las funciones que debería tener el sistema y sacamos los requerimientos que debía satisfacer el sistema ejemplo:

· El sistema deberá tener un botón para desplegar un formulario donde se hará la captación de vendedores.

· Al captar un vendedor el sistema solo deberá permitir que este pertenezca a una empresa y se le pueda decir si fue captado por otro vendedor o no.

Luego de obtener los requerimientos fuimos a identificar todas las clases con sus propiedades, también se pueden obtener más requerimientos mientras se hace este proceso.

### Clases y propiedades:

Las clases son las plantillas para crear los objetos que podemos identificar en el texto como entidades separadas que deben estar en el sistema y las propiedades son las características que tienen estos objetos que pueden ser trasladadas al sistema.

Como por ejemplo para el caso de la misma entidad u objeto vendedor puede ser una clase independiente en el sistema con sus atributos los cuales serían por nombrar algunos; Nombre, código y dirección.

Ya al final de este proceso terminaríamos el proceso de abstracción identificando los métodos que ejecutará cada clase y estos no son más que las acciones que pueden ejecutar cada objeto en el sistema, estas van de la mano con los requerimientos funcionales un ejemplo de esto podria ser listar\_vendededores y agregar vendedor.

Ya al final con estos datos obtenidos procedemos a describir las clases con sus relaciones correspondientes, las relaciones de clases son lo que nos indica cómo se relacionan estas entre sí ejemplo holding con empresa su relación sería que Empresa pertenece a Holding y a su vez con una multiplicidad de 1 holding puede tener 0 o muchas empresas.

Luego de describir realizamos los correspondientes diagramas con los datos de las clases, propiedades y sus relaciones.

# **Síntesis y Argumentación Individual**

Como fue presentado anteriormente en los objetivos, la metodología usada para la modelación y realización de este producto de sistema de almacenamiento de datos fue la conocida como UML, primero realizando la abstracción, hallando las clases, reconociendo cada una de las propiedades o atributos de cada clase y sus métodos u operaciones también, después de identificar todas las clases las relacionamos entre ellas y adicionalmente agregamos la cardinalidad dependiendo de cada caso que se presentara entre las clases, después de tener el diagrama bien definido pasamos a la realización del código, en NetBeans IDE creamos un nuevo proyecto y reutilizamos el código que se usó previamente en la realización del diagrama UML en el programa PlantUML, lo modificamos agregando el nivel visibilidad, tanto de las propiedades o atributos de cada clase como de sus métodos u operaciones, y dependiendo del caso hacíamos al tributo y al método público o privado, después aplicamos la herencia donde se debía y por último introducimos los métodos constructores de cada una de las clases y los métodos getters y setters en las clases que se debía.

# **Bibliografía**

Arrieta, J. C. (2021). EJEMPLO DE ANALISIS Y DISEÑO ORIENTADO A OBJETOS (AYDOO) - PARTE 1: - DIAGRAMA DE CLASES - UML. Recuperado de <https://www.youtube.com/watch?v=q7qF5o5Og9I>

Arrieta, J. C. (2021). EJEMPLO DE ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS - PARTE 2 - DEL MODELO AL CÓDIGO JAVA -. Recuperado de <https://www.youtube.com/watch?v=aKeMmGLVQb4>

Arrieta, J. C. (2021). EJEMPLO DE ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS [PARTE 3] CONSTRUCTORES DE LOS OBJETOS O INSTANCIAS -. Recuperado de <https://www.youtube.com/watch?v=GhKiD3UUV60>

Lucid Software Español. (2019). Tutorial - Diagrama de Clases UML.Recuperado de <https://www.youtube.com/watch?v=Z0yLerU0g-Q&t=1s>