
PROYECTO 1

Estuardo Sebastián Valle Bances
202001954

Resumen

El modelado de sistemas se basa en la abstracción de las ideas basadas en objetos del mundo real para construir, como su nombre lo dice, sistemas que relacionen sus objetos con las acciones específicas requeridas para la realización de una tarea. El lenguaje Unificado de Modelado permite representar estos sistemas en diagramas gráficos para facilitar la comprensión y complejidad de los mismos. Los diagramas se dividen en estructurales, que especifican la estructura computacional de los sistemas para los lenguajes de programación; y los de comportamiento, que grafican el estado de los objetos, sus acciones y consecuencias dentro del sistema sin llegar a la especificación.

Abstract

Systems modeling is based on the abstraction of object-based ideas from the real world to build, as the name implies, systems that relate their objects to the specific actions required to perform a task. The Unified Modeling language allows to represent these systems in graphic diagrams to facilitate their understanding and complexity. Diagrams are divided into structure diagrams, which specify the computational structure of systems for programming languages; behavior diagrams, that graph the state of the objects, their actions and consequences within the system without reaching the specification.

Palabras clave

Listas Enlazadas

Memoria

Heap

Stack

Keywords

Linked Lists

Memory

Heap

Stack

Introducción

En general, el presente proyecto busca la optimización en todos sentidos. La manera en la cual se ordenan las listas es una forma de optimización. La manera en la cual se guardan las listas es una forma de optimización. La manera en la cual se busca el cambio de un piso a otro es optimización de código en el mejor de todos sus sentidos.

Incluso, auxiliándose del concepto que plantea del enunciado en el siguiente párrafo:

“A la empresa “Pisos Artesanales, S.A.” le resulta en un costo que el robot realice cada una

de las operaciones antes mencionadas, de tal manera que, realizar una operación de volteo

de azulejo cuesta F Quetzales, mientras que realizar una operación de intercambio de

azulejos cuesta S Quetzales.”

Nos indica que esta optimización debe ser realizada para cualquier tipo de dato de entrada.

Desarrollo del tema

1. Superioridad en Estructuras de Datos

En referencia a las estructuras de datos, mayormente conocidas como TDAs, es importante reconocer que el alocamiento de memoria es dinámico y no estático. Esto quiere decir que cada vez que se realice un cambio dentro de ellas. La forma en la cual se adjuntará un nuevo nodo será a partir de copias en memoria.

Si preferimos los tecnicismos, dentro de muchas estructuras de datos, podemos ampliar lo siguiente:

1. La mayoría de algoritmos que poseen las listas enlazadas buscan la realización de distintas estructuras de datos son presentados

con un gasto de memoria de $O(n)$. Es decir, de manera lineal.

2. Al entender la mayoría de las listas y TDAs es posible notar que todas estas se basan en un algoritmo que crea un punto de partida y/o un punto de llegada.
3. Dentro de las listas enlazadas, las listas circulares y demás, esto puede ser visto como el HEAD o la CABEZA de la lista.
4. Todos los algoritmos de búsqueda poseen exactamente la misma estructura con cambios muy pequeños.

2. Diseño Orientado a Objetos

El diseño Orientado a objetos es esencial para el desarrollo de distintas actividades incluyendo la forma en la cual se crean las TDAs.

- Desarrollar gráficos de interacción de

objetos: Describen las intenciones del diseñador con respecto a cómo los objetos interactuarán en tiempo de ejecución para proporcionar la funcionalidad requerida, por lo que este constituye el primer paso en el desarrollo de un blanco modelo de caja del sistema.

- Desarrollar gráficos de visibilidad: Estos diagramas son utilizado para describir las otras clases a las que una clase determinada necesita hacer referencia, junto con las formas de referencia que se utilizarán y su duración.

- Desarrollar descripciones de clase (textuales): Deben especificar: los métodos y sus parámetros; los atributos de los datos; los atributos del objeto; las dependencias de

herencia.

- **Desarrollar gráficos de herencia:** Este paso considera la herencia como una construcción de diseño. Observa las relaciones entre las clases, y para identificar abstracciones comunes. Identifica nuevas superclases abstractas que pueden generalizar conjuntos de las clases identificadas en el modelo. El objetivo era integrar todos los procesos más exitosos de los programas y que se acoplaran a los requerimientos del programa. Se considera un método de segunda generación. Una característica importante es la existencia del diccionario de datos que incluye el significado de los nombres de procesos, objetos, atributos, etc durante el proceso. Los pasos para completarlo se dividen en los pasos del análisis y los que se asignan para el diseño.

Pasos de Análisis

- **Desarrollar el modelado de objetos:** En este paso se identifican las clases piloto para el modelado, los atributos de las clases, las relaciones de ambas y también las invariantes (las variables que no cambian porque son requerimientos). Todos se incluyen en el diccionario de datos.
- **Determinar la interfaz del sistema:** Es una extensión del paso 1, pero con la

especificación de cómo se fusionarán los procesos del paso 1 en el software cuando se vea como un todo. Además de ello se crean escenarios de entrada y salida para agregar procesos al software y manejar todo tipo de problemas.

Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas,
Facultad de Ingeniería
Introducción a la programación y computación 2, 1er.
Semestre 2021.

- **Desarrollo del modelo de interfaz:** En este paso se estudian detalladamente los escenarios de entrada y salida propuestos en el anterior y se describen los pasos y procesos que requiere el software para enfrentarlos.
- **Evaluar los modelos de análisis:** En este paso se analizan los procesos y ciclos descritos en los pasos anteriores y se determina si existe coherencia y relación entre ellos, además se evalúa que esos mismos procesos sean consistentes con el problema inicial y cumplan con los requerimientos iniciales.

3. Resolución del Problema

- a. **Modelación Inicial:**
Anterior a la abstracción por tipos de datos, es necesario sostener una idea general sobre las implicaciones del proyecto y cómo se realizará. La manera en la cual se realizó el

proyecto fue a partir de la idea de listas dobles. Es importante mencionar que estos no existe un apuntador que se encuentre buscando cada una de las listas sino que estas están siendo contenidas una dentro de otra.

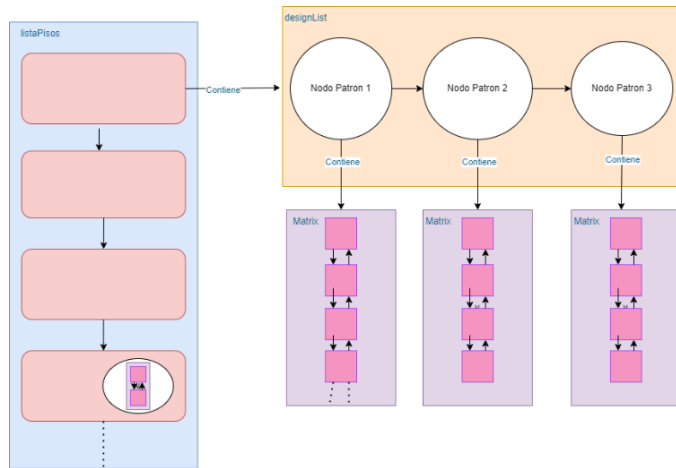


Figura 1. Manejo de Listas en el Proyecto.

Fuente: elaboración propia.

Además, el planteamiento de las matrices fue un diseño líneal que busca representar posiciones a partir de variables distintas

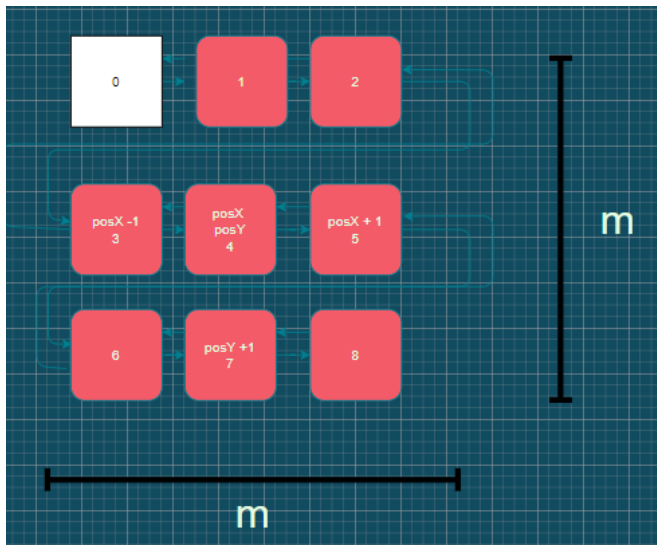


Figura 2. Manejo de Listas en el Proyecto.

Fuente: elaboración propia.

En este caso es sumamente notoria la realización de algunas de las listas y TDAs utilizadas.

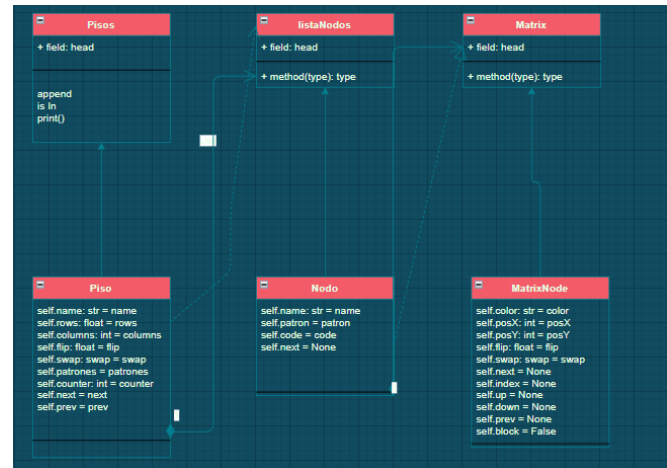


Figura III. Manejo de Listas en el Proyecto.

Fuente: elaboración propia.

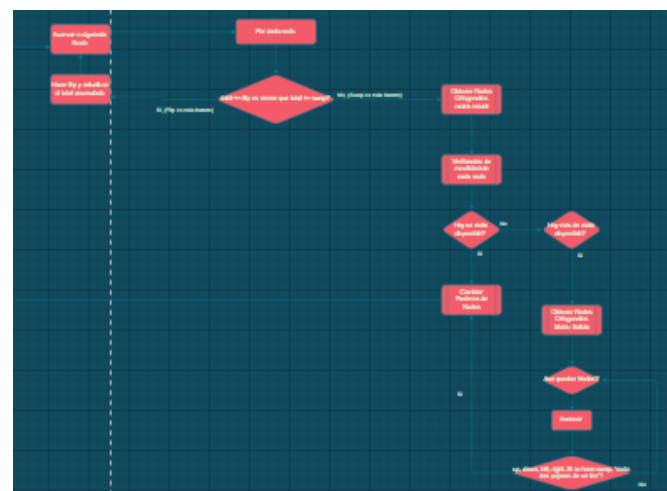


Figura 41. Manejo de Listas en el Proyecto.

Fuente: elaboración propia.

Conclusiones

El diseño orientado a objetos busca una visualización clara de instancias, clases y la manera en la que estos se relacionan en su ambiente.

2. Los procesos para el desarrollo de software más comunes dentro de la orientación a objetos son el “Unified Process” y el “Fusion

Process”

3. El “Fusion Process” busca la implementación secuencial de un proyecto, este crea una manera en la cual se debe orientar a eventos cada situación.

Referencias bibliográficas

Máximo 5 referencias en orden alfabético.

C. J. Date, (1991). *An introduction to Database Systems*. Addison-Wesley Publishing Company, Inc.

Cuatro páginas de extensión máxima.

Adicionalmente, se pueden agregar apéndices con modelos, tablas, etc. Que complementan el contenido del trabajo.