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.

ALGORITMO DE SIMULACIÓN DE PRODUCTOS PARA DIGITAL INTELLIGENCE S.A.

202003381 – Luisa María Ortíz Romero

Resumen

Se desarrolla un algoritmo que permita una simulación de ensamblaje, cargando una configuración de máquina para que el sistema conozca las líneas de producción disponibles y el tiempo que cada una se tarda en ensamblar, los productos y la cola de ensamblaje necesaria para ensamblarlos.

La aplicación propuesta funciona en memoria dinámica, a la máquina se le asigna una lista de líneas, cada una tiene el comportamiento de un brazo robótico y se mueve entre componentes según lo requiera la cola de producción del producto a ensamblar. La interfaz gráfica permite al usuario elegir un producto o cargar un archivo que contenga un conjunto de productos, además de visualizar los resultados en la aplicación y en un reporte con formato HTML.

La aplicación es capaz de indicar el tiempo óptimo de ensamblaje y los movimientos necesarios para lograrlo, además de generar una salida en formato XML, la cual podría ser útil para alimentar una máquina de ensamblado real.

Palabras clave
Simulación, TDA, Cola, POO, Componente

Abstract

An algorithm is developed that allows an assembly simulation, loading a machine configuration so that the system knows the available production lines and the time that each one takes to assemble, the products and the assembly line necessary to assemble them.

The proposed app works in dynamic memory, the machine is assigned a list of lines, each one has the behavior of a robotic arm and moves between components as required by the production queue of the product to be assembled. The graphical interface allows the user to choose a product or upload a file containing a set of products, as well as view the results in the application and in a HTML report.

The app can indicate the optimal assembly time and the movements necessary to achieve it, as well as generating an XML output which could be useful to feed a real assembly machine.

Keywords
Simulation, ADT, Line, OOP, Component

Introducción

Se realiza una abstracción de la información presentada por la empresa que solicitó la aplicación y se propone una solución realizada implementando Programación Orientada a Objetos. La máquina es el objeto principal, cada máquina tiene un listado de líneas de producción, un listado de productos y un listado de productos en espera, que son los que el usuario carga en un archivo de entrada para ser ensamblados uno tras otro.

Para cumplir con los requerimientos de aplicación en memoria dinámica se utilizan los TDA: Lista simple, Lista doble y Cola, según sea el caso. Cada elemento de una lista es un objeto nodo y está asociada a otro objeto lista, de esta manera se va utilizando solo la memoria requerida y se optimiza el rendimiento del equipo en el que se utiliza la aplicación.

Desarrollo del tema

a. Abstracción

La abstracción es el pilar de la Programación Orientada a Objetos que elimina detalles para simplificar y no considerar una o más propiedades de un objeto complejo a fin de atender las demás y centrar la atención en las que son imprescindibles para lo que se quiere lograr.

b. Tipos de Dato Abstracto (TDA)

Muchos de los lenguajes de programación de la actualidad implementan estructuras de datos como listas, tuplas y diccionarios, sin embargo, la mayoría de estos son ejecutados utilizando memoria estática, es decir, se le asigna un espacio en memoria fijo, lo que provocaría desperdicio o falta de memoria.

Para considerar el funcionamiento óptimo de la aplicación y el equipo en el que se está utilizando, se propone una solución que se ejecute en memoria dinámica, es decir, crezca su espacio mientras el programa así lo requiera. La estructura dinámica es lograda gracias a la abstracción de los elementos más importantes de los terrenos.

b.1. Listas enlazadas

Este tipo de estructura es de forma lineal y permite almacenar elementos comúnmente llamados nodos, en donde cada uno de ellos puede almacenar distintos datos y un apuntador hacia otros nodos.

Son estructuras dinámicas que pueden almacenar datos que cambian con frecuencia. Las listas enlazadas se propagan y se retraen para que sean más sencillas al añadir o eliminar información y permiten almacenar información en diferentes posiciones de memoria

b.1.1 Listas Enlazadas Simples

Es un conjunto de nodos con una sola dirección y forman una estructura de datos lineal. Cada nodo es un objeto que almacena un elemento (datos) y una dirección a otro nodo.

La dirección que guarda un nodo de otro nodo se llama puntero y el salto que los une, se llama salto de puntero. El primer nodo de una lista es la cabecera y el último se denomina final. El nodo final apunta a null, haciendo referencia a un nodo nulo, es decir, inexistente.

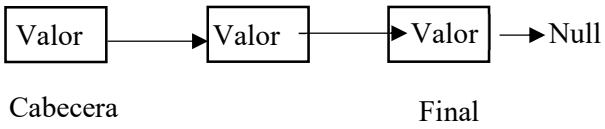

Figura 1. Representación de una Lista enlazada simple

Fuente: Elaboración propia

b.1.2 Listas Enlazadas Dobles

Cuando los nodos tienen un doble enlace, es decir, un enlace al siguiente (dirección sucesora) y otro enlace al nodo anterior (dirección predecesora), entonces se conocen como lista doblemente enlazada.

La lista doblemente enlazada aventaja a la lista simplemente enlazada en su recorrido en ambos sentidos, de izquierda a derecha como de derecha a izquierda, los nodos primero y último apuntan a null, que simboliza el final de la lista. El puntero anterior permitirá el acceso hacia el elemento anterior de la lista, mientras que el puntero siguiente permitirá el acceso hacia el próximo elemento. Su desventaja con relación a las listas simplemente enlazadas es que ocupan más memoria por nodo.

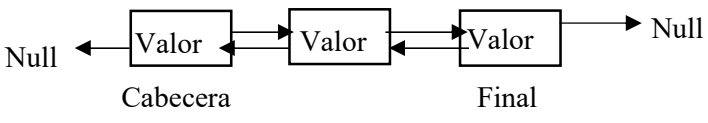

Figura 2. Representación de una Lista enlazada doble

b.1.3 Colas

A partir de una dirección de memoria, los datos se almacenan sucesivamente como si fueran una colección ordenada de elementos (cartas, clientes frente a una ventanilla, libros en un estante, mensajes a una casilla de correo, etc), en este caso, los elementos de las colección son las instrucciones de ensamblaje de un producto.

En cualquier momento se puede recuperar el objeto que se encuentra primero en la estructura (es decir, el primero que fue guardado), basado en esto, se dice que su acceso es de tipo FIFO (First In First Out).

Una cola permite sacar el elemento que entró primero, es decir, que está en la cabecera, permite saber cuál es el elemento que se encuentra al final y la longitud de la cola.

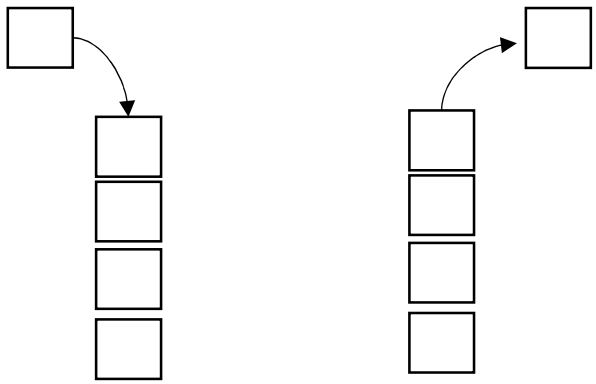

Figura 3. Representación del acceso FIFO de una cola

Fuente: Elaboración propia, 2021

Conclusiones

La máquina puede ser abstraída en sus características más importantes para simular su funcionamiento en una aplicación, en la que la máquina es el objeto principal a la cual se le son asignadas listas simples o listas dobles, dependiendo en qué direcciones se deseen recorrer.

Cada línea de producción es un nodo que posee, además de su información abstraída, el componente actual y el componente al que se desea llegar, logrando la simulación de los movimientos de un brazo robótico.

Las instrucciones de elaboración de un producto se comportan como una cola, en la que se debe ensamblar cada componente en el orden indicado, ensamblando (o desencolando) el primero en llegar, avanzando en orden hasta ensamblar el componente del fondo.

Referencias bibliográficas

Álvarez Morales, A. M. (2015). Implementación de un TAD que gestione una lista enlazada de nodos de tipo Árbol AVL y aplicación práctica.

Cristina, H. L. E. ED Lineales.

González, A. H. (2013). TAD-Pilas y Colas.

Serna, E. (2011). La abstracción como componente crítico de la formación en ciencias computacionales. *Avances en Sistemas e Informática*, *8*(3), 79-84.

Anexos

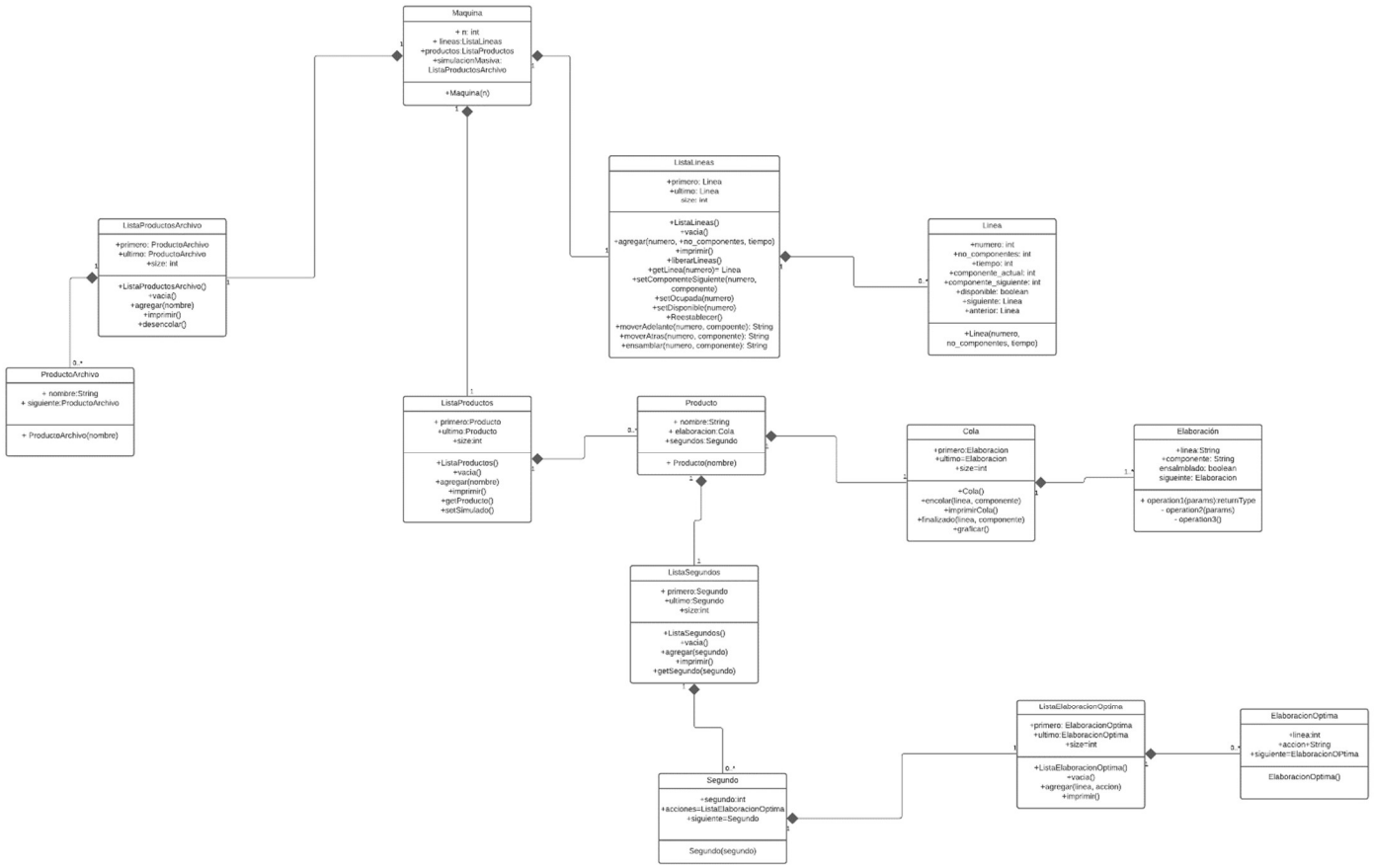

Figura 4. Diagrama de clases que modela la solución.

Fuente: Elaboración propia, 2021