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

# DIGITAL INTELLIGENCE, S. A

**202004734 – Justin Josue Aguirre Román**

**Resumen**

Se realizó un software en el cual se resuelve el problema para encontrar el orden de armado optimo que el sistema de brazos de armado deberá seguir para emplear un tiempo mínimo en él armado de piezas de cualquier producto el cual las líneas de acción estén capacitadas. Empleando estructuras de datos dinámicas como: listas enlazadas y pilas especiales; y el paradigma de programación orientada a objetos (POO).

Consiste en la lectura de un archivo de entrada de tipo XML, archivo que contiene la información, que al ser ingresado se interpretará su información creando las estructuras necesarias para cada terreno inscrito en él, procesándolo y recorriendo su información para crear la ruta de armado optima. Ruta que será exportada en forma de un archivo XML de salida.

Las listas facilitaron el acceso a la información de los terrenos, así como la adaptabilidad a las dimensiones de la cantidad de líneas o acciones requeridas para el armado de productos.

**Palabras clave**
Estructura de Datos, Clases, Software, Memoria, Algoritmos

***Abstract***

*A software was created in which the problem is solved to find the optimal assembly order that the assembly arm system must follow in order to spend a minimum time assembling parts of any product which the lines of action are capable of. Using dynamic data structures such as: linked lists and special stacks; and the object-oriented programming (OOP) paradigm.*

*It consists of reading an input file of type XML, a file that contains the information, which when entered, its information will be interpreted creating the necessary structures for each land registered in it, processing it and going through its information to create the optimal assembly route . Path that will be exported in the form of an output XML file.*

*The lists facilitated access to information on the land, as well as the adaptability to the dimensions of the number of lines or actions required for the assembly of products.*

***Keywords***
*Data Structure, Classes, Software, Memory, Algorithms*

## Introducción

Se implementaron las estructuras de datos dinámicas para el desarrollo del software para asi contar con un almacenamiento que sea capaz de procesar una gran cantidad de terrenos.

El paradigma de programación orientada a objetos, permite la creación de objetos, cruciales para el desarrollo de este software, separando instrucción de armado con sus respectivas características, siendo cada interpretación de mayor facilidad.

## Explicación del programa

Al ejecutarse el software, mostrara una ventana con distintos componentes que se habilitaran y según se trabaje con el programa, permitiendo desarrollar diferentes acciones: Leer Archivo de Líneas de Armado, Leer Archivo de Simulación, Procesar Archivos, Imprimir Archivo XML, Ver Informacion del Producto, Generar Imagen de Secuencia de Armado y Salir. Cada una de estas opciones serán explicadas a continuación.

El funcionamiento principal del software consiste la lectura y análisis de archivos de tipo XML, guardando su información agrupada en simulaciones, líneas de armado y productos que se pueden realizar por medio de las líneas de armado.

XML es el acrónimo de Extensible Markup Language, es decir, es un lenguaje de marcado que define un conjunto de reglas para la codificación de documentos. Simplificando los conceptos expuestos podemos comenzar con que:

El lenguaje de marcado es un conjunto de códigos que se pueden aplicar en el análisis de datos o la lectura de textos creados por computadoras o personas. El lenguaje XML proporciona una plataforma para definir elementos para crear un formato y generar un lenguaje personalizado.

Los archivos XML que se ingresaran al software contienen información que será almacenada con el uso de distintos tipos de nodos y listas programadas de forma manual.

La programación informática considera que un nodo es cada uno de los elementos de una lista enlazada, un árbol o un grafo en una estructura de datos. Cada nodo tiene sus propias características y cuenta con varios campos; al menos uno de éstos debe funcionar como punto de referencia para otro nodo.

Una lista enlazada es una estructura de datos que puede utilizarse para la implementación de nuevas estructuras (tales como las colas, las pilas y sus derivados) y está formada por una serie de nodos que almacenan, además de la información deseada, un enlace, un puntero o una referencia al nodo que lo precede, al posterior, o bien uno a cada uno. La ventaja fundamental de una lista enlazada en comparación con un vector convencional es que sus elementos no presentan un orden rígido ni relacionado con el que tuvieron al momento de ser almacenados, sino que éste depende del enlace que posee cada nodo, y puede ser modificado cuando así se desee.

Los archivos XML de Líneas de Armado están conformados por la información tanto de las Líneas de Armado: número de línea, cantidad de componentes disponibles en la lista y el tiempo que tarda cada linea en ensamblar un componente; como de los productos que pueden armar dichas líneas: nombre del producto y las instrucciones para el orden de armado correcto de sus componentes, como se muestra en la Figura 1 y Figura 2, sección de Anexos.

La lectura de los archivos de entrada se utilizó la librería “xml.etree.ElementTree”, importándola internamente desde Python, que trabaja este tipo de archivos con la estructura de árbol de nodos, es decir, trabaja los archivos con ramificaciones y distintas propiedades, cada ramificación equivale a un nivel de profundidad, para acceder a niveles más profundos se agregaran un par de corchetes con el identificador de hijo que queramos acceder.

La biblioteca ElementTree incluye herramientas para analizar XML usando APIs basadas en eventos y documentos, buscando documentos analizados con expresiones XPath, y creando nuevos o modificando documentos existentes.

Para el análisis de los datos y su almacenamiento se optó por la creación de nodos, contando con cinco

tipos distintos de nodos, conectados entre sí por medio de los enlaces de sus respectivas listas enlazadas. La primera lista fue la de las lineas que por medio de una estructura repetitiva "while" se accedió a todos los registros de este tipo del primer archivo de entrada, por cada registro encontrado se creara un nodo de tipo "Línea de Armado" y se almacenara su información encontrada y mencionada anteriormente.

La siguiente lista creada durante el proceso de lectura será la lista de productos que sigue los mismos pasos de la lista anterior, con la diferencia del tipo de nodo e información que almacenara; los nodos serán de tipo Producto, los cuales están capacitados para armar generar por cada nodo una sub-lista dentro de ellos, estas listas almacenaran la información del orden de armado de sus piezas. Las líneas de armado poseen la siguiente sintaxis: "L1C2 L2C1 L2C2 L1C4". El proceso se lleva a cabo desglosando su línea de instrucción de modo que cada " " (espacio) delimita el fin de una instrucción y el inicio de otra. Cada instrucción tendrá información de a que Línea de Armado pertenece, un Id de instrucción para marcar su orden, el nombre de la instrucción completo y sus enlaces.

En este caso cada vez que la lectura de la estructura de árbol se encuentre con la etiqueta terreno creara un nuevo objeto con las características en el XML y unas extras que manejaran el combustible utilizado en dicho terreno, además de contar con una verificación de si dicho terreno ha sido analizado con anterioridad.

Para la lectura de simulaciones, el software creara un nodo de tipo simulación por cada uno de los productos solicitados en el archivo de entrada de cada archivo. Realizará el proceso de creación de estas estructuras hasta terminar de leer a plenitud el archivo XML.

Las enlazará y creará un enlace extra que dará acceso a las acciones realizadas luego de procesar la solicitud de simulación. Además de añadir el nodo producto como tal dentro del nodo Simulación que ha sido creado

Al leer los archivos mencionados la interfaz gráfica también sufrirá algunos cambios ligeros, al leer el archivo de lineas la tabla de información ubicada al costado derecho de la ventana principal tomara sus configuraciones, es decir, optara por crear el mismo número de columnas como líneas de producción ingresen en nuestra entrada. El segundo archivo al ser interpretado se añadirá a componentes tipo combobox los ítems correspondientes de las acciones, es decir, habrá un combobox encargado de las simulaciones mostrando el nombre de las simulaciones disponibles a procesar, mientras el segundo combobox se encarga de los productos en dicha simulación, este también cambiará de ítems dependiendo de cuál sea la selección en el primero.

El procesar el archivo se refiere al cálculo del proceso óptimo de armado a recorrer para un ahorro de tiempo empleado. Al pulsar su respectivo botón dará inicio a la búsqueda de todos los nodos que contengan el nombre descrito en el combobox como uno de sus atributos, en cuanto encuentre uno de estos ingresara al producto dentro de este nodo y buscara el orden de armado para su construcción. Creando la ultima lista y tipo de nodo empleado para este software, estos serán de tipo Acción y poseen el numero de linea que lo realiza el tiempo en que se realiza y que es lo que se realiza (Movilizarse, Armar o simplemente esperar) la instrucción dependerá de que es lo que realicen los otros brazos y de la posición en la que se encuentre el brazo valuado.

Si la línea de trabajo aun no ha llegado a su destino, durante ese segundo realizara el movimiento ya sea derecha o izquierda según convenga, si la línea de trabajo está en su destino valuara si su pieza anterior ya fue colocada, en caso afirmativo procederá a colocar tardando tantos segundos tenga como especificación dicha línea de trabajo, en caso contrario tendrá que esperar hasta que la pieza anterior sea colocada en su respectivo lugar.

Así con todas sus instrucciones, al terminar contara las instrucciones para cada linea y aquellas que posean un numero de instrucciones menor a la cantidad de segundos trabajados se le añadirán instrucciones de espera hasta llegar a la misma cantidad en ambas partes.

Para cada producto en la simulación las lineas de tabajo empezaran en una posición 0, es decir, posición inicial siempre será fuera de los componentes por la izquierda del componente 1, teniendo que hacer por lo menos un movimiento para estar sobre este.

Al terminar el proceso de cualquier simulación se harán ciertos reajustes de los tiempos de cada una, además de reajustar las características de armado de los productos en la lista enlazada de productos a las características por defecto para futuras simulaciones requeridas.

Al seleccionar algún item del combobox de los productos por simulación este habilitara dos botones extra que en caso de haber procesado dicho producto, de lo contrario los mantendrá desactivados.

El primero de los botones nos muestra la información de dicho producto en la interfaz gráfica, tabulando los datos de armado y el proceso completo separado por columnas cada línea de trabajo y por filas cada tiempo t en el que se realizo n acción. Además de mostrarnos el tiempo total que tardo en el armado de este.

El proceso de armado de la tabla se da por medio de estructuras cíclicas que irán valuando que la información coincida con al solicitada, (el tiempo la línea, la simulación y el producto) para asi añadirlo a su posición correcta y pasar a mostrarse en la interfaz.

Dependiendo del armado que tome podrá valuar los 2 nodos más próximos a ese punto (Si se dirige al Sureste, realiza la comparación entre su nodo más próximo al Sur y al Este). Eligiendo aquel que consuma una menor cantidad de combustible y cambiando su atributo de camino de un estado falso a uno verdadero.

Este proceso se repetirá hasta llegar a una de las excepciones: Que se encuentre en la misma fila o columna que la posición de inicio, o que ya haya llegado. En el primer caso entonces el movimiento del robot se limitará a simplemente moverse en sentido al punto de meta. En el segundo únicamente habría terminado y cambiaria su característica de camino a verdadera. Posteriormente imprimirá un mapa compuesto por las posiciones del terreno simbolizadas como un 0 para aquellas posiciones

El segundo botón nos dará la opción de imprimir el orden de armado del producto seleccionado, ingresando directamente desde la información de la lista principal de productos y su sub-lista de armado de piezas. Las gráficas se realizaron generando primero un archivo de tipo dot el cual con ayuda de Graphviz se interpretará su contenido y devolverá una imagen con la terminación deseada (jpg o png).

Graphviz (Graph Visualization) es un conjunto de herramientas de software para el diseño de diagramas definido en el lenguaje descriptivo DOT.1 Fue desarrollado por “AT&T Labs” y liberado como software libre con licencie tipo Eclipse.

Graphviz consiste en un lenguaje de descripción de gráficos llamado DOT, un conjunto de herramientas y librerías que pueden generar o procesar archivos DOT.

Dot es una herramienta en línea de comandos para producir imágenes en capas de grafo dirigido en una variedad de formatos de salida (PostScript, pdf, svg, png, etc.).

Los archivos tipo DOT que esta extensión interpretara se generaron a través de la implementación de una cadena de texto la cual llevara todo el contenido requerido, nodos y sus relaciones. Los nodos son inicializados con ayuda de una estructura cíclica de tipo “while” que hará el recorrido completo valuando.

Una vez llevados los nodos a la cadena de texto se asignarán las conexiones necesarias para que cada nodo este con su siguiente pieza nodo. Para ello se utilizaron estructuras “while”, valuando si este tenia un nodo siguiente y añadirlo, de lo contrario para el funcionamiento.

Se declararía con una última instrucción el tipo de imagen al cual se exportará toda la información, se indicará el archivo en el cual esta guardada la información y el nombre que se le asignara a la imagen generada.

Al finalizar el proceso de generación mostrara el mensaje de que el proceso se ha completado con satisfacción, de lo contrario mostrara un mensaje de error.

Por ultimo nos encontramos con la opción de impresión de XML Para imprimir el archivo de salida se hará una impresión de todos los productos que han sido ingresados en la simulación el nombre introducido por el combobox será buscado por medio de esta entrada en la lista enlazada principal. Sin embargo, existen restricciones para realizar esta tarea, se verificara que dicha simulación haya sido

procesada con anterioridad.

Habiendo verificado que el terreno puede ser exportado a un XML, se buscara la información básica del terreno y junto con las librerías "ElementThree" con sus propiedades ElementThree se hará la conversión de esta información de un formato de texto a una estructura XML adaptado y lista para la generación del documento final.

Solo la información principal acerca del recorrido será la introducida al archivo de salida. La información requerida será el nombre de la simulacion, nombre del producto n dentro de la simulación, el tiempo de armado del producto n y procedera a desglosar las acciones realizadas en la simulación del producto para su armado separándolo en linea y segundo que se realizo como se muestra en le Figura 2, sección de Anexos

Esta información será ordenada por medio de las librerías ya mencionadas, asignándolas como sub elementos de un nodo padre el cual será el nombre del mismo terreno. Por último, en el proceso nos solicitara la ruta exacta en donde queremos que se guarde nuestro archivo XML de salida. El software nos indicara si ocurre algún tipo de error en este proceso, de lo contrario nos notificara imprimiendo un mensaje que el archivo XML se ha generado satisfactoriamente.

Las soluciones expuestas en el texto anterior fueron a través del método de análisis extenso y arduo con la información y experiencia obtenida mediante investigación del uso de herramientas para Python y librerías y herramientas externas. Para lograr hacer la búsqueda y guardado de información del software.

El análisis y creación de nodos y estructuras de datos dinámicas se ha utilizado por ser un recurso que haga relaciones de distintos tipos entre la información y poder acceder entre ellos de una manera práctica.

**Conclusiones:**

La solución presenta el algoritmo en el cualfuncionan convencionalmente las matrices, lo que permite al usuario una familiarizaciónadecuada para del uso del software.

El análisis y creación de nodos y estructuras de datos dinámicas se ha utilizado por ser un recurso que haga relaciones de distintos tipos entre la información y poder acceder entre ellos de una manera práctica.

Es importante modelar un problema antes de empezar a realizarlo, por medio de diagramas de flujo y clases para comprender, analizar y encontrar la solución de mejor manera.

**Anexos:**
Archivo de entrada solicitado

```
<Maquina>
    <CantidadLineasProduccion>2</CantidadLineasProduccion>
    <ListadoLineasProduccion>
        <LineaProduccion>
            <Numero>1</Numero>
            <CantidadComponentes>4</CantidadComponentes>
            <TiempoEnsamblaje>1</TiempoEnsamblaje>
        </LineaProduccion>
        <LineaProduccion>
            <Numero>2</Numero>
            <CantidadComponentes>4</CantidadComponentes>
            <TiempoEnsamblaje>1</TiempoEnsamblaje>
        </LineaProduccion>
    </ListadoLineasProduccion>
    <ListadoProductos>
        <Producto>
            <nombre>Smartwatch</nombre>
            <elaboracion>L1C2 L2C1 L2C2 L1C4</elaboracion>
        </Producto>
        <Producto>
            <nombre>Camara</nombre>
            <elaboracion>L1C2 L2C3 L1C4 L2C2</elaboracion>
        </Producto>
        <Producto>
            <nombre>USBStick</nombre>
            <elaboracion>L1C1 L2C3</elaboracion>
        </Producto>
    </ListadoProductos>
</Maquina>
```

*Figura 1.* Archivos de Entrada

Fuente: Elaboración propia, 2021

```
<Simulacion>
    <Nombre>prueba2</Nombre>
    <ListadoProductos>
        <Producto>USBStick</Producto>
    </ListadoProductos>
</Simulacion>
```

*Figura II.* Archivo de Simulación

Fuente: Elaboración propia, 2021

## Archivo de salida solicitado por simulacion

*Figura III.* XML de Salida

```
<SalidaSimulacion>
  <Nombre>prueba2</Nombre>
  <ListadoProductos>
    <Nombre>USBStick</Nombre>
    <TiempoTotal>4</TiempoTotal>
    <ElaboracionOptima>
      <Tiempo NoSegundo="1">
        <LineaEnsamblaje NoLinea="1">Movimiento hacia componente 1</LineaEnsamblaje>
        <LineaEnsamblaje NoLinea="2">Movimiento hacia componente 1</LineaEnsamblaje>
      </Tiempo>
      <Tiempo NoSegundo="2">
        <LineaEnsamblaje NoLinea="1">Armando componente 1</LineaEnsamblaje>
        <LineaEnsamblaje NoLinea="2">Movimiento hacia componente 2</LineaEnsamblaje>
      </Tiempo>
      <Tiempo NoSegundo="3">
        <LineaEnsamblaje NoLinea="2">Movimiento hacia componente 3</LineaEnsamblaje>
        <LineaEnsamblaje NoLinea="1">No hacer nada</LineaEnsamblaje>
      </Tiempo>
      <Tiempo NoSegundo="4">
        <LineaEnsamblaje NoLinea="2">Armando componente 3</LineaEnsamblaje>
        <LineaEnsamblaje NoLinea="1">No hacer nada</LineaEnsamblaje>
      </Tiempo>
    </ElaboracionOptima>
  </ListadoProductos>
</SalidaSimulacion>
```

Fuente: Elaboración propia, 2021

## Diagrama de Clases del software

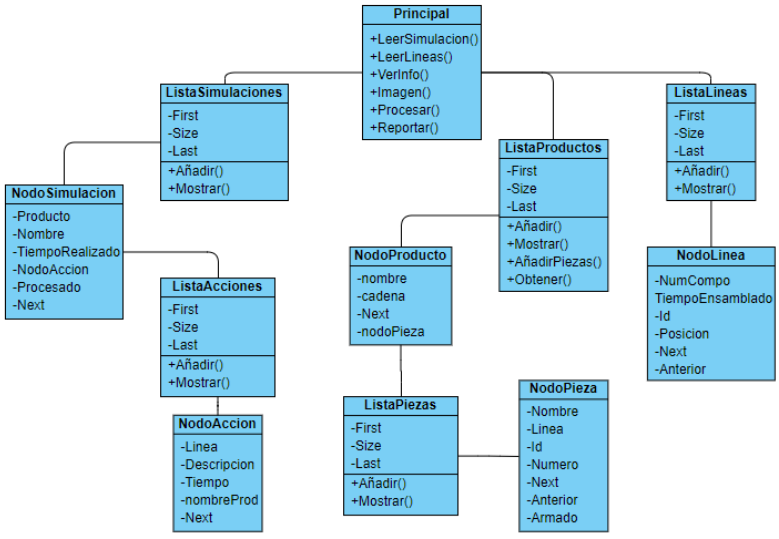


*Figura IV.* Diagrama de Clases

Fuente: Elaboración propia, 2021