VIVALDI software de acompañamiento industrial

Jahel Santiago León, Jordán Mauricio Escarraga Avila

No. de Equipo Trabajo: {*1*}

I. INTRODUCCIÓN

Este proyecto está enfocado en asistir a los supervisores del área de producción y/o manufacturación, debido a que a pesar de contar con la capacitación adecuada los errores humanos siguen estando presentes lo cual genera una menor productividad, por lo tanto el fin de este proyecto es ayudar a disminuir la cantidad de errores que se puedan generar debido al factor humano.

II. DESCRIPCIÓN DEL PROBLEMA A RESOLVER

Cuando una empresa empieza a crecer y expandirse, sus necesidades cambian y por lo tanto necesita adaptarse, para poder progresar, muchos administradores estarán de acuerdo cuando cuando se menciona que el área de planeación de la manufactura es de las más importantes y de las que más atención necesita, pues una inadecuada gestión puede terminar en el cierre de una empresa puesto que a medida que los pedidos en el área de producción y manufactura aumentan, se hace más complicado poder organizar los recursos disponibles, para dar abasto con la demanda y los tiempos de entrega en pos de poder cumplir con las órdenes, sin mencionar que solo cumplir en el mercado actual no es suficiente pues se debe eliminar totalmente la ineficiencia en el sistema para poder reducir los tiempos de fabricación, aumentando la productividad general y permitiendo devengar mejores ingresos.
Por lo tanto nace la necesidad de un software que facilite la gestión de estos procesos desde la priorización de los pedidos, pasando por el chek in del inventario y la asignación de los recursos disponibles, tareas que puede hacer un supervisor, pero que no están exentas de fallar debido a errores humanos en la planificación, por lo tanto se propone crear VIVALDI un software amigable con el usuario en donde no se necesita ser ingeniero industrial para poder operarlo como en la mayoría de software ERP del mercado, pues por el contrario proporciona herramientas intuitivas para facilitar el trabajo del usuario.

III. USUARIOS DEL PRODUCTO DE SOFTWARE

Vivaldi está pensado para ser usado por las plantas de producción de PIMES en el sector industrial, más específicamente para que sea manejada por un supervisor del area de produccion y manufactura de la empresa, el software no requiere conocimientos especializados por lo que cualquier persona lo puede operar, vale aclarar que es un software muy flexible que se adapta a las necesidades de cada empresa pero para el ejemplo este será enfocado a la simulación de productos relacionados al trabajo en alturas..

IV. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

- *Aclaraciones iniciales: Este proyecto está diseñado para solamente manipularse por lo que se consideran los usuarios supervisores, los cuales podrán realizar acciones de la naturaleza CRUD.*

- *Manejo de pedidos, empleados, órdenes y productos*

  Tipo de datos a ingresar por la interfaz; String

Más adelante estos se someterán a diversos castings según se requiera.

A través de las diferentes interfaces del programa se visualizarán y modificarán los diversos objetos cumpliendo con los requisitos mínimos de CRUD (Crear, leer, actualizar y borrar).

Extraer Datos de un archivo XML: En pos de mejorar la compatibilidad de la aplicación con grandes volúmenes de datos se creó la opción de poder leer archivos XML y poder incorporar sus registros en tablas de la interfaz.

Visualizar: Los objetos se leerán desde un archivo xml o desde un archivo serial para acceso rápido y se mostrarán en una matriz ubicada en la interfaz.

Crear objeto: Para crear un objeto se agregaran sus datos en la interfaz y después de hacer clic en el botón crear, se procederá a almacenarse en memorio xml (respectivo para cada tipo de objeto)automáticamente en la última posición, y además de esto se agregara al ArrayList que contiene a los demás objetos pertenecientes a la misma clase.

Eliminar y/o modificar un objeto: Desde la interfaz podemos seleccionar registros de las tablas correspondientes a cada objeto, por ejemplo podemos seleccionar un empleado de las tabla de empleados y modificarlo directamente, así mismo podemos hacer con las demás tablas correspondientes al inventario y órdenes.
Buscar registros: podemos buscar los registros dentro de la matrices de objetos, para poder acceder más rápido a la visualización, edicion, creacion y eliminacion.

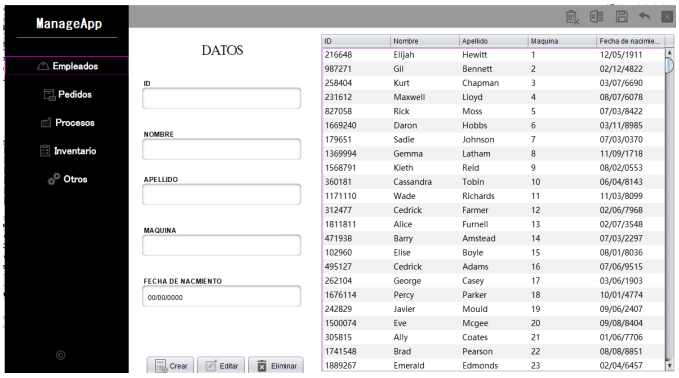
Guardar archivos en formato .obj: podemos guardar las tablas de objetos en un archivo serializable para tener un acceso rápido a estos.

Convertir las tablas a formato excel: para poder tener mejor compatibilidad de la aplicación con las principales plataformas de visualización de datos, la aplicación tiene la funcionalidad de guardar los datos en un formato XML para poderlo leer desde un excel y que genere las tablas de forma automática.

- *Procesos (NO PERTENECIENTE A ESTA ENTREGA)*

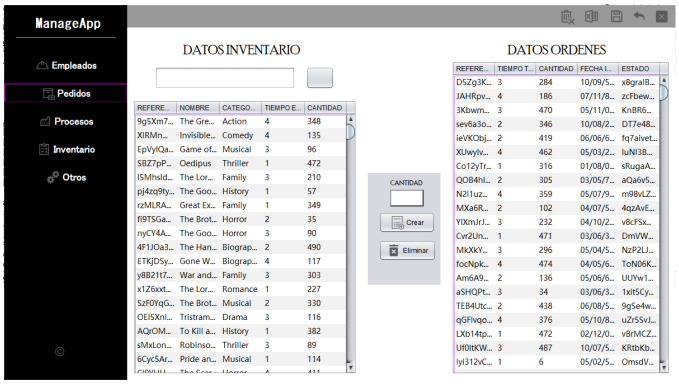
V. DESCRIPCIÓN DE LA INTERFAZ DE USUARIO PRELIMINAR

Pantalla Empleados:



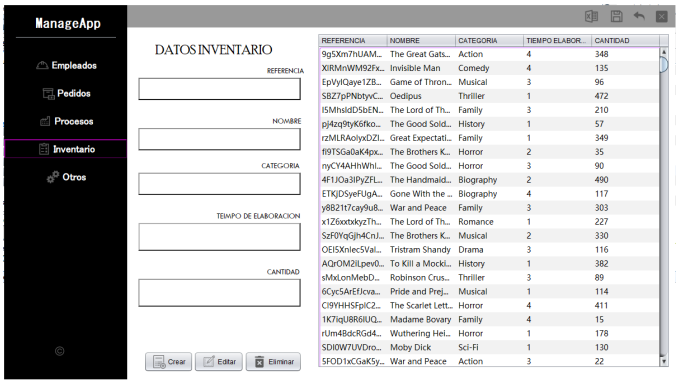
En esta primera pantalla podemos Manejar la administración de empleados, en donde podemos ver su estado, junto con todos los datos personales, además nos dice en cual maquina esta trabajando en este momento.

Pantalla Pedidos:



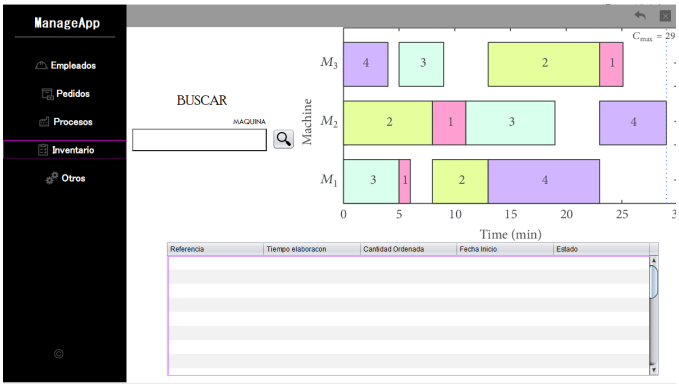
En esta segunda pantalla podemos gestionar y crear pedidos, seleccionando el artículo de la tabla inventario, asignando una cantidad, podemos crear pedidos de forma eficiente además el software hace un cálculo del tiempo promedio que debería tomar entregar ese pedido basado en la prioridad, el numero de maquina, los empleados disponibles y finalmente el tiempo de elaboración, de esta manera permitiendo una gestión más eficiente de la planta de producción.

Pantalla Inventario:



En esta tercera pantalla podemos gestionar el inventario disponible de los productos que existen en la empres, se pueden consultar la cantidad disponible de cada producto, así como editarla y también permite hacer búsqueda por palabras clave.

Pantalla Procesos:



En esta cuarta pantalla podemos ver en tiempo real cómo están los procesos de producción basados en las órdenes creadas, podemos ver cómo están distribuidos en las distintas máquinas y el tiempo promedio que se tomara en completar las operaciones, además podemos ver las colas de los procesos en cada máquina.

VI. ENTORNOS DE DESARROLLO Y DE OPERACIÓN

En esta sección se debe describe el entorno en el que se desarrollará el software, así como el entorno en el que operará el software en tiempo de ejecución, específicamente el sistema operativo, recursos y el hardware sobre el que operará el software.

- Netbeans IDE 11.3, jdk 12.
- Windows 10 home basic, 12gb RAM (10.9 utilizables), AMD Ryzen 5 2500U (2 GHz), 64 bits.

VII. PROTOTIPO DE SOFTWARE INICIAL

El link del repositorio donde se alberga el primer prototipo de software es el siguiente:

Rama `primera entrega`

https://github.com/jahelsantiago/project-VIVALDI-.git

Para este proyecto se implementó una estructura de datos tipo ArrayList la cual se llama igual pero se ubica en la carpeta ED junto a otras estructuras de datos que planteamos en caso de necesitarlas(LinkedList, Nodos, pilas, colas), todas las estructuras que se almacenaron en la carpeta ED poseen las funcionalidades básicas requeridas que se nos indicaron a medida del transcurso de las clases, aunque a las clases que no utilizamos en este proyecto todavía falta mejorarlas en algunos aspectos.

- Almacenamiento de los datos

Se utilizaron archivos xml en los cuales se leen los diversos objetos almacenados a través de un sistema de etiquetas único para cada tipo de objeto. Se optó por utilizar este tipo de archivos debido a que también se pueden manipular a través del programas como excel.

Además de eso se implementó la clase serializable en algunos objetos para facilitar la lectura y manipulación de estos.

VIII. PRUEBAS DEL PROTOTIPO

A continuación se probaron las funcionalidad de manejo de elementos del software en general, más específicamente hablando el CRUD y casteo de datos a xml.

Guardar archivos en XML:

Guardar XML	
Numero de datos	tiempo (ms)
1000	5614
2000	14545
4.000	49990
6000	106783
8.000	179963
10000	227450

Tabla 1: tiempos de guardado de archivos XML

Gráfico 1: comparación número de datos con tiempo de ejecución en la creación de registros xml

- Como podemos ver en la gráfica nuestros tiempos de ejecución en consola se tornan cuadráticos algo que intentaremos mejorar en futuras entregas, pero de momento se quiere realizar no se recomienda hacerla con más de 10.000 datos que en promedio tarda 4 minutos.

Extraer Datos de un archivo XML:

Leer XML	
Numero de datos	tiempo (ms)
10000	558
50.000	1031
100.000	1997
500.000	8051
1.000.000	24878
10.000.000	*

Tabla 2: tiempos de lectura de archivos XML

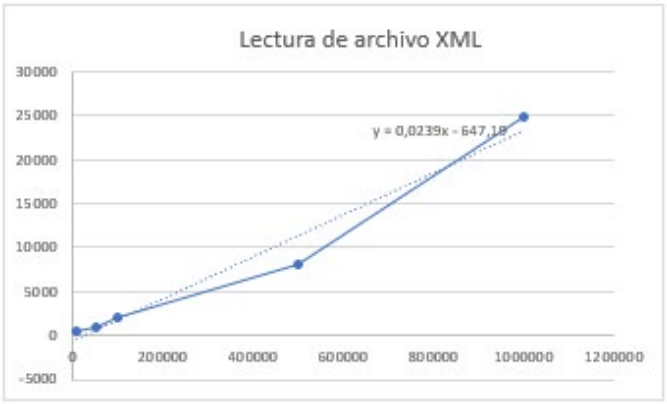


Gráfico 2: comparación número de datos con tiempo de ejecución en la lectura de registros xml

- Como podemos ver en el grafico, esta funcion se comporta de tal manera que asemeja un comportamiento constante O(1), pues al analizar la ecuación que genera su línea de tendencia, podemos observar que cuando n tiende al infinito, esta tiende a 0 el cual es el comportamiento más óptimo que se puede lograr con una función de este estilo.

Guardar Archivo .obj:

Guardar archivo .obj	
Numero de datos	tiempo (ms)
10000	1380
50.000	1968
100.000	2393
500.000	28612
1.000.000	29117
10.000.000	*

Tabla 3: tiempos de escritura de archivos .obj

Gráfico 3: comparación número de datos con tiempo de ejecución en la escritura de archivos obj

- Podemos ver que la función asemeja un comportamiento lineal O(n), el cual es el mejor tiempo posible que se puede obtener en una función de este tipo.

CRUD de objetos:

CRUD					
añadir una orden		editar una orden		eliminar una orden	
Numero de datos	tiempo (ms)	Numero de datos	tiempo (ms)	Numero de datos	tiempo (ms)
10000	20	10000	10	10000	10
50.000	29	50.000	49	50.000	10
100.000	50	100.000	32	100.000	30
500000	480	500000	320	500000	400
1.000.000	536	1.000.000	440	1.000.000	579
10.000.000	·	10.000.000	·	10.000.000	·

Tabla 4: tiempos de CRUD para la tabla de ordenes

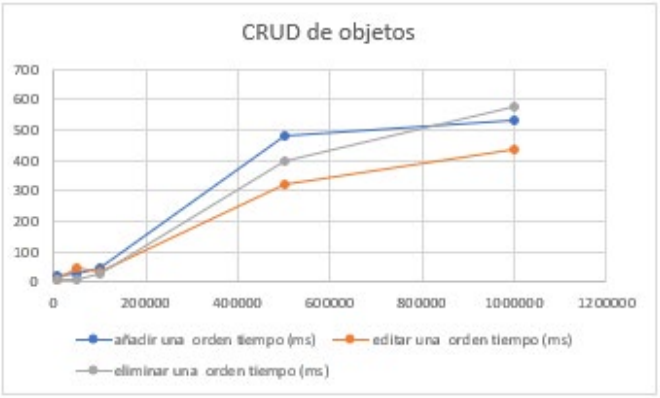


Gráfico 3: comparación de los métodos crud en la tabla ordenes

- Como podemos evidenciar en la gráfica podemos apreciar que estas operaciones tienen tiempo constante O(1), lo cual representa el mejor tiempo de ejecución que podemos obtener.

IX. ROLES Y ACTIVIDADES

Comunes: Es importante aclarar que debido a que el grupo solamente cuenta con dos integrantes, todas las tareas han sido compartidas, revisadas por los dos miembros y en caso de necesitar, diversas correcciones y mejoras se hicieron a lo largo del proyecto,

- Elaboración de estructuras de datos a utilizar.
- Refinamiento de los sistemas de guardados (XML y serialización de las clases).
- Creación del trabajo escrito.
- Control y revisión de cada actualización al proyecto.

Jahel

- Elaboración de la interfaz gráfica
- Pruebas de tiempos de ejecución de las funciones del proyecto.

Jordan

- Archivos xml para el guardado de los datos.
- Funciones de búsqueda.

X. DIFICULTADES Y LECCIONES APRENDIDAS

- Las estructuras utilizadas lineales no son óptimas para el manejo altos volúmenes de datos.

- Los tipos de archivos xml se escogieron por su facil manipulacion en excel, el problema es que el volumen de datos máximo son de 43250 en esta plataforma, para usos de volúmenes más extensos se recomienda archivos csv o de otro tipo. La forma en la que se modifican y eliminan objetos no es la mas optima debido al manejo de etiquetas en este tipo de archivos por lo cual no se recomienda para usar como almacenamiento de una base de datos en constante cambio.

- El cargar los archivos al disco cada vez que se modifica o elimina algún objeto era un paso que entorpece mucho el funcionamiento del proyecto, por lo cual según recomendaciones del profesor asignado se requiere realizar la carga de todos los archivos xml en memoria al comienzo y trabajar todas las modificaciones en la sección de memoria, para luego proceder con la escritura en disco justo al final de la ejecución de nuestro proyecto

- En próximas entregas se considerara cambiar el sistema de guardado de los datos.