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 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 área de producción 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..

ACTUALIZACIONES:

Para esta segunda entrega se realizaron las siguientes correcciones y mejoras al proyecto:

Las estructuras no lineales implementadas (Árboles entre otras), fueron realizadas con las funciones CRUD como objetivo a ciumplir ademas de que fueron modificadas con el fin de optimizar la realización de funciones previamente implementadas o de suplir nuevas necesidades presentadas en el transcurso de la elaboración del proyecto..

1. Se implementó la estructura de árboles AVL en las clases de: Productos, empleados, máquinas y órdenes.

   Para la realización de estas actualizaciones se generó primero la estructura de árbol de búsqueda binario y luego a las funciones de insertar y eliminar se le agregaron funciones de balance, lo cual nos permitió implementar el árbol AVL.

   Además de esto se utilizó como medio principal de comparación entre objetos la sobrecarga del método compareTo debido que facilitaba mucho su implementación.

2. En la clase máquina se corrigió la asignación automática del serial para cada una de las máquinas nuevas, usando la estructura del árbol AVL con la función de findMax la cual nos ayuda a encontrar la máquina que posee el serial más grande por lo cual cuando se genera una nueva máquina está poseerá el serial mayor anterior sumado 1.

3. En el caso de los empleados se agregó un nuevo atributo booleano para diferenciar cuales están disponibles y cuales están despedidos, teniendo así 3 estados para los empleados:

   - Disponible – activo
   - No disponible – activo
   - Despedido

4. Los empleados ahora ya no se podrán buscar por el nombre o por parte de la id y debido a que el criterio de búsqueda basado en compareTo implementado requiere de la id exacta de los empleados.

5. Esta es una situación que sucede en todos los casos de búsqueda que involucran la implementación de los árboles AVL por lo cual se pierde versatilidad en los métodos de búsqueda, pero se gana eficiencia en el uso del tiempo debido a la implementación una estructura de datos más elaborada no lineal.

6. Se modificó la interfaz de procesos y se creó una funcionalidad para asignar las órdenes a las máquinas respectivas, contal de que estas se procesen, esto lo podemos realizar de dos maneras, de forma manual clickeando sobre a máquina o de forma automática dejando que el software tome la mejor decisión basado en las máquina disponibles y las órdenes en el sistema para aprovechar los recursos de la fábrica de la manera mas optima. (Para este caso se implementó la estructura no lineal de Hash y la estructura lineal de las colas)

7. En la clase de Productos de agregar un atributo booleano para tener la funcionalidad de descontinuar productos.

8. Correcciones en las funciones de la naturaleza CRUD de los diferentes objetos.

9. Se descontinuó la clase Pedidos debido a su inaplicabilidad en el proyecto.

10. Con los árboles AVL se soluciono el problema de crear productos repetidos (misma referencia), o de ingresar empleados repetidos.

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 memoria XML (respectivo para cada tipo de objeto) automáticamente en la última posición, y además de esto se agrega 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 la 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 las matrices de objetos, para poder acceder más rápido a la visualización, edición, creación y eliminación.

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 el programa de Excel y que genere las tablas de forma automática.

● *Procesos de maquina*

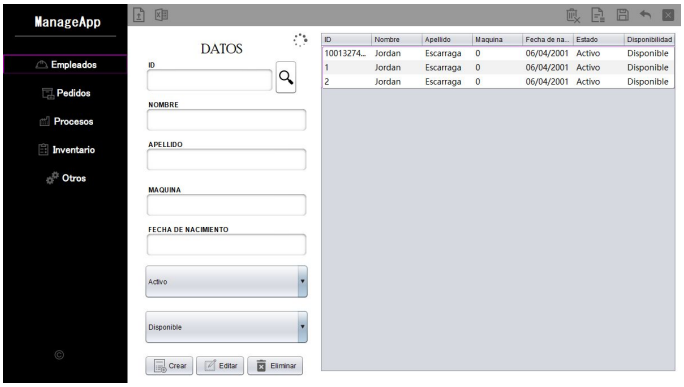
Este conjunto de procesos une si no todos, la mayoría de las implementaciones, procesos, clases y demás labores realizadas

a lo largo de todo el transcurso del proyecto. Por lo cual se espera que sea la piedra angular del proyecto.

Para una implementación más efectiva se optó por utilizar la estructura de datos no lineal hash, la cual no era necesaria incluirla en esta entrega del proyecto, pero debido a que recientemente en clase se ha podido apreciar su efectividad y capacidad de adaptación ante diferentes situaciones, se reconoció su valor para el proyecto.

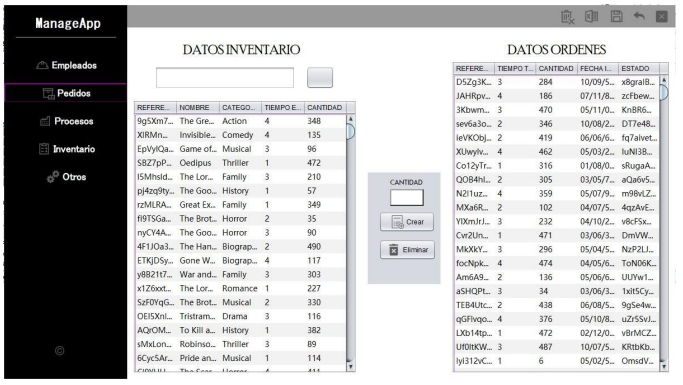
V. DESCRIPCIÓN DE LA INTERFAZ DE USUARIO

Pantalla Empleados:



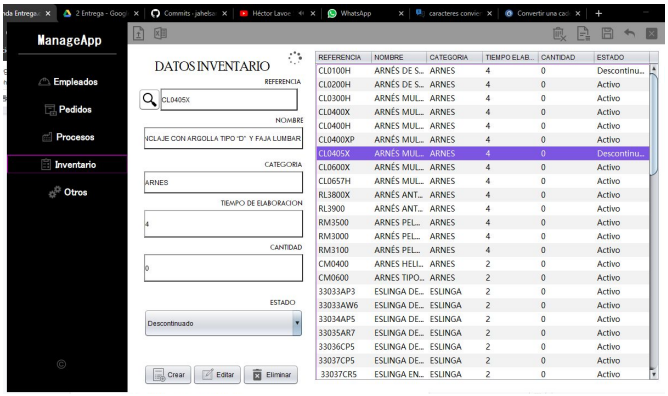
En esta primera pantalla podemos Manejar la administración de empleados, en donde podemos ver su estado, junto con todos los datos personales, además nos dice en cual maquina esta trabajando en este momento.

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 número de máquina, los empleados disponibles y finalmente el tiempo de elaboración, de esta manera permitiendo una gestión más eficiente de la planta de producción.

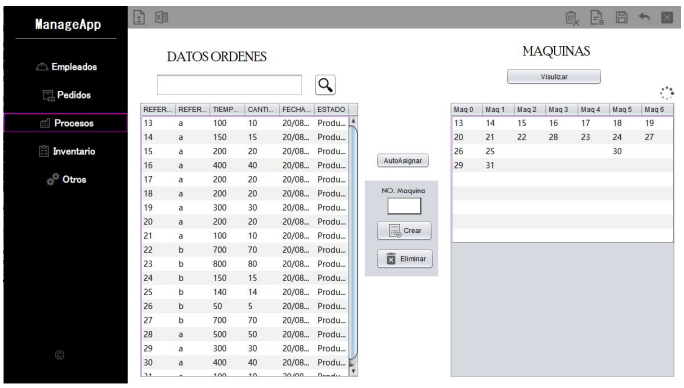
Pantalla Inventario:



En esta 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.

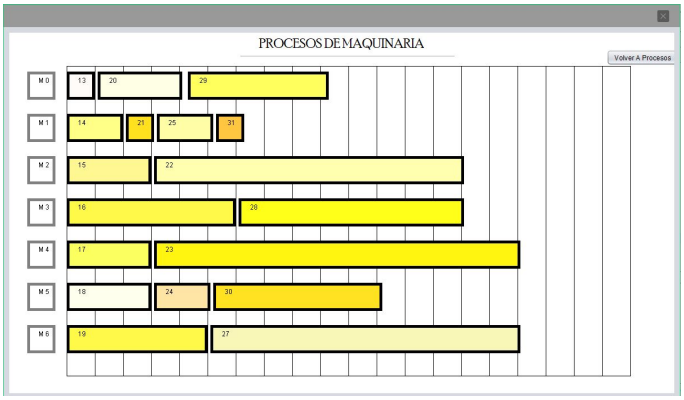
Los productos utilizados en esta entrega son los definitivos, debido a que son productos de venta oficiales en el mercado de el trabajo en alturas.

Pantalla Procesos:



En esta quinta pantalla podemos ver las órdenes que se encuentran en cola para asignar, y las maquinas disponibles en el area de produccion, en donde con un simple click podemos asignarlas manualmente seleccionando la orden y la máquina donde queremos que esta sea procesada, también tenemos la opción de asignar las órdenes de forma automática en donde el software encontrará la manera más adecuada y eficiente para organizarlas.

Pantalla visualización de Procesos de Maquina (Pruebas):



En esta interfaz se permitirá la visualización a escala basado en el tiempo de realización del proceso en ejecución,

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 segunda_entrega

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

Para este proyecto se implementaron tanto Estructuras lineales como no lineales, las cuales están ubicadas en el paquete ED (Estructuras de Datos) . Está estructuras puede que cambien en algunos aspectos a lo largo de la realización de este proyecto dependiendo de las necesidades que se presenten.

- 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 de 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 la 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

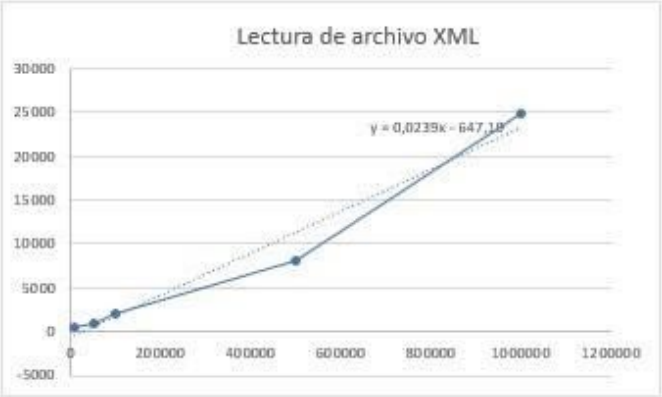


Gráfico 2: comparación número de datos con tiempo de ejecución en la lectura de registros xml

- 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

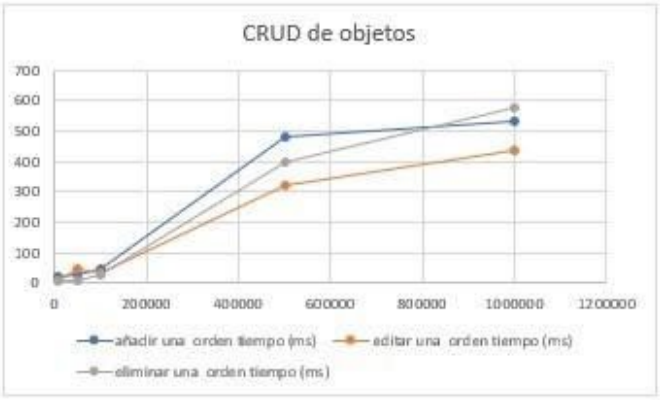


Gráfico 3: comparación de los métodos crud en la tabla ordenes

- 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.

CRUD nueva versión:

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	63	10000	48	10000	53
50.000	252	50.000	240	50.000	248
100.000	484	100.000	479	100.000	489
500000	2569	500000	2396	500000	2474
1.000.000	4809	1.000.000	4792	1.000.000	4799
10.000.000	*	10.000.000	*	10.000.000	*

Tabla 5 tiempos de CRUD nueva version**:**

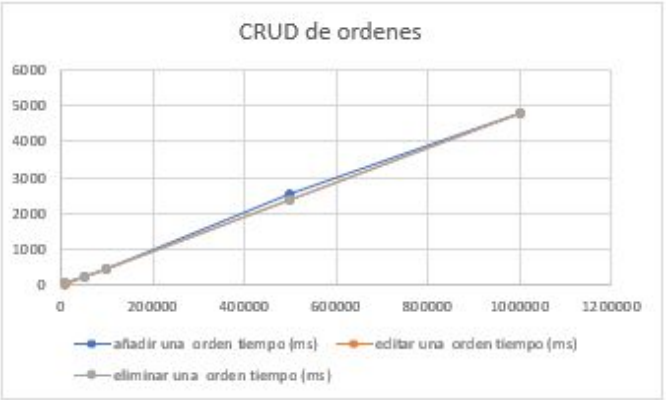


Gráfico 5: comparación número de datos con tiempo de ejecución en los CRUD

Para esta nueva versión los tiempos de CRUD aumentaron en comparación con la versión pasada, algo que tiene total sentido y se explica pues utilizamos árboles para almacenar los índices de la los objetos, pero esto sopesa el hecho de reducir el tiempo de las búsquedas a logaritmico.

Asignar órdenes Automáticamente:

Auto-asignar Ordenes	
10000	153
50.000	223
100.000	388
500000	1773
1.000.000	4612

Tabla 6 tiempos de asignar órdenes.

Gráfico 6: comparación número de datos con tiempo de ejecución en la organización de forma automática de las órdenes.

Para la implementación de esta funcionalidad, usamos una combinación de distintas estructuras de datos como heaps, matrices dinámicas, linked list y hash, para este ejemplo el tiempo de ejecución es el mas optimo que se puede conseguir.

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, lo más importante siempre ha sido la comunicación pronta y oportuna:

- 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

- Los tipos de archivos XML se escogieron por su fácil manipulación en Excel, el problema es que el volumen de datos máximo es 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 más óptima 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.

- Debido a que para esta entrega se implementó la estructura de Arboles AVL, se generaron algunos percances en la implementación de esta. Debido a que se buscaba implementarla de la manera más eficiente, uno de los aspectos más importantes para lograr este objetivo es utilizando la sobrecarga del método compareTo entre otras cosas, por lo cual se invirtió un tiempo considerable en la búsqueda de información y documentación respectiva para la construcción correcta de esta estructura. Una de las complicaciones de hacer esto es que solamente podemos buscar eficientemente utilizando un solo criterio de búsqueda.

- El implementar una GUI en un proyecto en el cual buscamos mucho la optimización y manejamos diversas clases e implementaciones, genera una gran inversión de tiempo en lo respectivo a pruebas, correcciones y mantenimiento del proyecto.

- Como el tema de los hash se a visto recientemente en clase se decidió también por implementar esta estructura de datos en el proyecto, se espera que con el avance que se tenga próximamente en clase se pueda implementar de manera más eficiente.

- A pesar de las diversas situaciones de fuerza mayor presentadas durante el transcurso de la realización de esta segunda entrega las cuales no serán mencionadas en el documento por respeto a la privacidad de los

integrantes del equipo, se ha aprendido a siempre anteponer los objetivos más importantes (creacion e implementacion de las estructuras requeridas con las debidas pruebas y correcciones), siempre con el fin de optimizar y nutrir este proyecto.

- La búsqueda versátil con los Árboles AVL resulta un poco complicada debido a la manera de implementar método compareTo, por lo cual se espera optimizar en un futuro este proceso implementando de una manera más eficaz la estructura no lineal del hash.

- Se espera ampliar los conocimientos relacionados a la estructura no lineal del hash con el fin de mejorar su implementación en el proyecto, no solamente en la sección de procesos de maquinas.

- Se plantea la posibilidad de informes en PDF generados por el proyecto.

- Queremos refinar la los algoritmos de creación de máquinas y de su relación con los empleados.