UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

MÁSTER EN INGENIERÍA INDUSTRIAL


# "DISEÑO Y GUÍA DE USO DE UN SOFTWARE DE GESTIÓN EN EL ÁMBITO INDUSTRIAL"

TRABAJO FIN DE MÁSTER

2024

AUTOR: Jose Manuel Asensio Fuentes

DIRECTOR/ES: Abel Navarro Arcas

# ÍNDICE

ABSTRACT ..... 7

1. OBJETIVO ..... 8

1.1 ¿QUÉ ES UNA EMPRESA? ..... 9

1.2 ¿QUE ES UN ERP? ..... 12

1.3 EVOLUCIÓN DE LOS ERP ..... 14

2. ESTADO DEL ARTE ..... 18

2.1 MOTIVACIÓN ..... 19

3. LEAN ..... 20

3.1 HISTORIA DEL LEAN MANUFACTURING ..... 21

4. PILARES DEL LEAN MANUFACTURING APLICADOS EN ESTE SOFTWARE ..... 24

4.1 PRIMER PILAR ..... 24

4.1.1 JUST IN TIME (JIT) ..... 24

4.1.2 8 DESPERDECIOS ..... 26

4.1.3 KANBAN ..... 28

4.2 SEGUNDO PILAR: KAIZEN ..... 29

4.2.1 LAS 5 “S” ..... 30

4.3 TERCER PILAR: CONTROL DE CALIDAD ..... 31

5. HERRAMIENTAS SOFTWARE UTILIZADAS ..... 33

5.1 JAVA ..... 33

5.2 NETBEANS ..... 34

5.3 MYSQL ..... 35

6. SOFTWARE ..... 37

6.1 PRODUCTO. ..... 37

6.2 CONJUNTO ..... 44

6.3 CLIE/PROV ..... 50

6.4 PEDIDOS ..... 56

6.5 ENVIOS/DEV. (ver de llevar a última pestaña) ..... 67

6.6 ALMACÉN ..... 73

6.7 ÓRDENES ..... 78

6.8 GESTIÓN ..... 81

6.9 POSTVENTA ..... 90

7. CONCLUSIÓN Y TRABAJOS FUTUROS ..... 97

8. ANEXO – CÓDIGO FUNCIONES BASE DE DATOS ..... 99

9. ANEXO 2 – CÓDIGO BASE DE DATOS ..... 121

10. BIBLIOGRAFIA ..... 125

**TABLA DE ILUSTRACIONES**

Figura 1 – Posible división de una empresa en departamentos que pueden ser incluidos en un ERP. ..... 13

Figura 2 – En que se divide el BOM. ..... 15

Figura 3 – ERP comerciales. ..... 18

Figura 4 – Icono del lenguaje. ..... 33

Figura 5 – Icono de NETBEANS. ..... 34

Figura 6 – Icono MYSQL. ..... 35

Figura 7 – Pestaña PRODUCTO y campos para “crear producto”. ..... 40

Figura 8 – Buscar producto SIN ninguna condición. ..... 41

Figura 9 – Búsqueda por condición de familia de producto. ..... 41

Figura 10 – Búsqueda por criterio de descripción. ..... 42

Figura 11 – Se muestra el momento de antes de editar un producto. ..... 44

Figura 12 – Muestra el resultado después de haber editado un producto. ..... 44

Figura 13 – Crear nuevo conjunto. ..... 46

Figura 14 – Buscar en la tabla de conjuntos donde se muestra el conjunto creado. ..... 46

Figura 15 – Introducir productos en un escandallo. Completar conjunto. ..... 48

Figura 16 – Búsqueda del escandallo de un conjunto. ..... 48

Figura 17 – Crear un escandallo en Excel. ..... 50

Figura 18 – Crear un cliente. ..... 52

Figura 19 – Buscar cliente sin condición. ..... 53

Figura 20 – Buscar cliente según una condición. ..... 54

Figura 21 – Búsqueda por proveedor. ..... 55

Figura 22 – Generar un nuevo pedido. ..... 58

Biblioteca UNIVERSITAS Miguel Hernández

Figura 23 – Búsqueda de pedidos, sin criterio. Se puede ver como el pedido recién creado aparece con precio nulo. ..... 59
Figura 24 – Editar estado “TRAMITADO” de un pedido (antes). ..... 62
Figura 25 – Editar estado “TRAMITADO” de un pedido (después). ..... 62
Figura 26 – Añadir producto al pedido. ..... 65
Figura 27 – Introducir conjunto para crear KPI de garantías. ..... 66
Figura 28 – Resultado de crear la gráfica. ..... 67
Figura 29 – Apariencia de la pestaña. ..... 69
Figura 30 – Cuando se pulsa la acción “BUSCAR PEDIDOS TERMINADOS Y NO ENVIADOS” ..... 69
Figura 31 – Pasos para declarar un pedido enviado. ..... 70
Figura 32 – Resultado de haber enviado un pedido. ..... 70
Figura 33 – Información acerca de l ..... 77
Figura 34 – Aspecto de la pestaña órdenes. ..... 79
Figura 35 – Momento antes de declarar un pedido“TRAMITADO”. ..... 80
Figura 36 – Los pedidos “TRAMITADOS” ya no aparecen. ..... 81
Figura 37 – Registrar el inicio del trabajo diario de una orden por parte de un trabajador. ..... 83
Figura 38 – Registrar el fin del trabajo diario con esa orden. ..... 84
Figura 39 – Inicio de registro general de una orden. ..... 85
Figura 40 – Muestra los datos guardados del inicio de una etapa de la orden. ..... 86
Figura 41 – Muestra la información de trabajo de cada una de las etapas de una orden. 87
Figura 42 – Una vez terminada la orden, muestra el procedimiento para declararla terminada. ..... 88

Figura 43 – Muestra esa orden, con la información de que se encuentra "TERMINADA"..... 88

Figura 44 – Se trata de mostrar por cliente, justo una vez se ha creado la segunda interacción. ..... 93

Figura 45 – Calificar una interacción. ..... 94

Figura 46 – Gráfica que muestra todas las interacciones de un solo cliente. ..... 95

Figura 47 – Muestra todas las interacciones de todos los clientes. ..... 96

Biblioteca
UNIVERSITAS Miguel Hernández

# ABSTRACT

The aim of this project was to link two different knowledges as industrial and informatics to create a software with an intuitive interface for the user. This program will help the companies to increase the productivity reducing most of the wastes that the companies struggle. For this it is necessary to base in LEAN MANUFACTURING ideas.

All the information related to a manufacturer company can be stored in this software and later be easily accessible for the users. To sum up, the idea is to have everything what is happening in the company saved in a database. Thanks to this, as one department starts working after other ones finishes, it helps to improve the development of the company.

This software is combined with two different programming languages, one for the database, to store the information and another for the graphical interface. This will be used by the user, to find an easy access to the information stored. The point is to give a tool to the customer to deal with all the problems that a company face all days in an industrial world.



UNIVERSITAS Miguel Hernández

# 1. OBJETIVO

Una vez conocido este problema, el objetivo es evitar toda esta posible pérdida de información y centralizarla en un software al que toda la empresa pueda tener acceso. Así, independientemente de que persona haya estado a cargo de cada tarea, que de un simple vistazo, se pueda entender en que situación está ese departamento y cuales serían los siguientes pasos a seguir.

Y no solo sería el centralizar la información, también es muy importante el poder unificar el lugar donde se guarda la información, porque así se consigue que no se esté enviando en todo momento la información de un sistema a otro. Simplemente es una única base de datos, con diferentes tablas de información y un entorno visual en el que te permita comprender y acceder de forma rápida a toda esa información almacenada en la base de datos.

Pero no solo se trata de gestionar la información de la forma de trabajo de cada uno de los departamentos de la empresa. Sino también que permita obtener información del exterior, es decir, de los clientes. Que aparte de gestionarse de forma interna, un software que te permita registrar como es cada interacción con los clientes, es decir, que suelen necesitar cuando nos contactan y como es su grado de satisfacción. Ya que gracias a esto se pueden ir descubriendo patrones. Es decir, ver los clientes de una zona si siempre piden una información similar, para ver si la estrategia de mercado que se está siguiendo en esa zona no es la correcta. En cambio, también puede ocurrir que muchos clientes se estén quejando de lo mismo (y no exista ningún tipo de relación cercana entre estos) y así gracias a esto que permita tomar decisiones internas en la empresa para subsanar errores. En definitiva, se trata de añadirle una aplicación extra al software que permita obtener información del exterior para poder adaptar el interior (de la empresa) a las necesidades que aparezcan, para así poder mostrarse competitivo con respecto a los competidores y conseguir una rápida velocidad de adaptación que permita a la empresa seguir siendo competitiva.

## 1.1 ¿QUÉ ES UNA EMPRESA?

Para poder empezar definiendo que es una empresa, se usará la definición que aparece en la RAE (RAE, s.f.) es una actividad económica organizada para producir bienes o prestar servicios destinados al mercado.

Esta descripción se trata de una descripción genérica. Entonces, para este caso en concreto habrá que definir que es una empresa industrial, ya que en este trabajo está enfocado en la gestión de ese tipo de empresas.

Una empresa industrial es una organización que se encarga de transformar materias primas en productos finales que se introducirán posteriormente al mercado para su venta. Por lo tanto, esto implica que se realizará una transformación a este material mediante unos procesos de fabricación y/o producción.

Existe una secuencia que siguen este tipo de empresas. Es que un departamento se encarga de disponer de estas materias primas. Obtenerlas al mejor precio y que se siempre se encuentren a disposición cuando se deban utilizar (que no haya retrasos). Una vez está este material, conforme lleguen los pedidos, otro departamento se encargará de ver que se ha pedido y se deberá empezar los procesos para transformar esta materia prima que ahora mismo se dispone en el producto terminado que es el que se le enviará al cliente. Una vez fabricado este producto, se deberá enviar al cliente según la fecha que se haya acordado con él. Y, por último, cada vez que se entrega un producto, se le debe dar asistencia a ese cliente, siempre y cuando este lo requiera.

Esta sería una secuencia bastante resumida de cómo funcionan las empresas industriales. A continuación, se detallarán uno por uno, cuales serían los departamentos que componen este tipo de empresas:

- COMPRAS
- LOGÍSTICA

- ALMACÉN
- PRODUCCIÓN
- COMERCIAL
- SERVICIO TÉCNICO

Aquí se ha establecido que esos son los departamentos, pero no quiere decir que todas las empresas tienen que estar compuestos con ellos, ya que es posible que un departamento englobe a varios de los que se han detallado aquí, pero al final el proceso que sigue una empresa para su funcionamiento es el mismo.

El departamento de COMPRAS es el que está destinado a tratar con los proveedores. Este departamento se encarga de conseguir la materia prima al mejor precio posible para la empresa. Y no solo eso, también debe de disponer la materia prima siempre lista para cuando se necesita. Ya que no puede ser que se quede un pedido pendiente de tratar y empezar a producir, porque no se haya recibido la materia prima. Salvo que haya algún problema de suministro, este siempre debe encontrarse listo para cuando la producción lo necesite. También es el encargado de cuando se adquiere un material o producto a un proveedor, tiene que ser el encargado de tratar con ellos la posible garantía si es que fuera el caso.

El siguiente departamento para explicar es el departamento de logística. Este es el encargado de gestionar todos los presupuestos que tenga que realizar la empresa. Es decir, es el departamento encargado de preparar el documento que englobe todas las referencias que el cliente necesita y generándole un precio total para ver si este les cuadra a sus posibilidades de gasto o no. Después, si el cliente acepta este presupuesto, se transformará en un pedido y, por lo tanto, debe encargarse de que el pedido llegue al departamento encargado de fabricar y completar el pedido para que salga en la fecha prevista. Y una vez se encuentra todo preparado, debe de gestionar el envío. Ver donde se tiene que mandar y en qué condiciones lo debe hacer para que así le llegue el producto al cliente.

En estos dos primeros departamentos, ya se está comentando la necesidad de disponer la materia prima a punto para empezar a fabricar el producto, y ya se está hablando de cuando esté listo poder enviárselo al cliente en el tiempo establecido. Todas estas funciones deben ser realizadas por el departamento de ALMACÉN. Este departamento es el que se encarga de guardar todo el material que tenga en la empresa de una forma ordenada y controlada, para así en el momento que se necesite poder hacer uso de él. Es decir, este departamento debe de estar informado de que se va usar y en que momento, para que así justo cuando llegue el momento, todo se encuentre en su sitio y listo para ser usado. Cabe destacar, que se tiene que llevar un control férreo del stock. Cada vez que se tenga que sacar material para que sea transformado en producto final, o cada vez que el departamento de COMPRAS adquiera materia prima, siempre se debe de actualizar el inventario de la empresa, para que siempre sea real la cantidad que disponga la empresa.

Una vez comentado, que se debe disponer de la materia prima, el control de los pedidos, y que el material se saque para ser transformado en producto, sería el momento de explicar el departamento de PRODUCCIÓN. El departamento de producción es aquel que convierte las materias primas en el producto final que se mandará al cliente. Por lo tanto, este es el departamento encargado de gestionar al personal, para así destinar a un trabajador o grupo de trabajadores en cierto pedido, ya según sea el volumen y la complejidad de montaje, se necesitará más o menos mano de obra. No solo influye la complejidad de fabricación de ese producto, sino el tiempo necesario de fabricación. Es decir, dependiendo del tiempo que se haya acordado con el cliente la entrega de su producto final, será necesario destinar o no más recursos a la finalización de ese producto.

Cuando el producto se encuentra terminado, este se dejará preparado para que el departamento de ALMACÉN, pase a recogerlo, y lo deje preparado para enviar a falta de que el departamento de LOGÍSTICA prepare el envío con la compañía pertinente al destino acordado con el cliente.

El siguiente departamento es el departamento COMERCIAL, este departamento podría ser englobado con el departamento de LOGÍSTICA, pero no es exactamente el mismo.



UNIVERSITAS Miguel Hernández

Ya que el departamento COMERCIAL también tiene la capacidad de poder generar presupuestos y transformarlos en pedidos. Pero sobre todo, su principal objetivo es la de captar nuevos clientes, por lo tanto, ellos también deben de tener acceso dentro de la empresa a funcionalidades tales como la de crear un presupuesto para que el cliente o potencial cliente pueda ver cuando le costaría, y la e después de convertir este presupuesto en un pedido en firme.

Por último, falta por explicar el departamento de SERVICIO TÉCNICO. Ya que cuando una empresa desarrolla un producto, y lo introduce en el mercado para que los clientes lo puedan adquirir, puede ocurrir que los clientes desconozcan como hacer un uso correcto de este, por lo tanto, este es el departamento que deberá de generar toda la documentación necesaria y dar todo el soporte que se requiera al cliente para que el cliente sea capaz de poder hacer el uso correcto del producto desde el primer momento. Es más, no solo es para los casos en el que el cliente no sabe como usarlo, sino en casos que hay algún problema con el producto y no funcione correctamente, es cuando este departamento se debe de encargar de darle soporte a distancia para poder hacerlo funcionar otra vez e incluso organizar una asistencia, en la que un técnico de la empresa se deba acercar a las instalaciones del cliente para poder solucionarlo. Este departamento aparte de ser llamado como SERVICIO TÉCNICO, también se le conoce como departamento de POSTVENTA.

## 1.2 ¿QUE ES UN ERP?

Los software de gestión que se han estado comentando tiene como nombre genérico y reconocido de ERP (Enterprise Resource Planning). Pueden llegar a estar compuestos de todos los departamentos de la empresa (producción, compras, RRHH, contablidad, etc.).



UNIVERSITAS Miguel Hernández


*Figura 1 – Posible división de una empresa en departamentos que pueden ser incluidos en un ERP.*

*FUENTE: https://www.simla.com/blog/que-es-un-erp*

Los primeros software de gestión nacieron en la década de los 60, como software muy básico en el que se encargaban de controlar los inventarios. En cambio, en las dos épocas siguientes, empezaron a añadir a estos software módulos para que se pudieran empezar a controlar la producción junto el control de inventario que ya estaba anteriormente. Y por último, a finales de siglo se incorporaron el control de las finanzas de la empresa y su gestión de los recursos humanos.

Como se comenta en (APD, 2019), entre las ventajas de usar sistemas ERP destaca por encima de todo, la velocidad en la respuesta a la hora de tomar decisiones. Ya que, con estos sistemas, los datos se encuentran fácilmente al alcance del usuario y de una forma muy visual. Gracias a que se dispone de estos datos, se puede tomar rápidamente decisiones, y, es más, te ayuda a que la decisión que tomes sea la correcta, ya que en cada apartado se debe mostrar la información o los datos de más relevancia. Ya que no se trata

de mostrar toda la información posible, sino solo mostrar aquello que influye en esa etapa o módulo del sistema.

Gracias a esto, con una alta velocidad de repuesta aparte de venir por lo comentado en el párrafo anterior, también viene generado por una automatización en los procesos. Ya que este software debe ser lo más intuitivo posible para que se consiga un uso sencillo del mismo, hasta casi que se consiga que el uso directo y automático del mismo (que se utilice hasta casi sin pensar).

Al final, como consecuencia de estas ventajas, aparece una última ventaja que es la de reducción de costes, porque todas las ventajas anteriores generan una reducción de tiempos (reducción de tiempo para acceso a la información, reducción de tiempo para guardar información, reducción de tiempo para la toma de decisiones). Y por supuesto, una reducción de tiempos genera reducción de costes.

El problema aparece si no se hace un buen uso de esta herramienta, porque no se trata solo de generar un buen interfaz y trabajar con una base de datos potente. El problema aparece cuando no se escoge correctamente la información importante, y como destacarla. Es por eso que no solo se necesitan conocimientos de programación, se necesita también un conocimiento del funcionamiento de las empresas del sector industrial y en concreto del negocio a que se dedican para poder conocer que es lo importante y que no. Ya que no se puede mostrar toda la información para no confundir a los usuarios, y si se elimina información se corre el riesgo de eliminar información importante, es por eso que para esto es necesario poder distinguir lo importante de lo que no.

## 1.3 EVOLUCIÓN DE LOS ERP

Tal como se explica en el artículo de Fernando Félix (VELNEO, 2020) los primeros ERP datan de la década de los años 40 del siglo pasado en el que lo empezó a usar el ejército de los Estados Unidos a finales de la segunda Guerra Mundial para cubrir sus necesidades de logística y producción. En aquel momento no se podía aplicar en un entorno industrial



UNIVERSITAS Miguel Hernández

debido a la infraestructura que se necesitaba para poder implantar un computador con un software específico.

El siguiente cambio se produjo en la década de los 60, cuando los ordenadores empezaron a llegar al entorno civil. En esta época, cuando se producían los ordenadores, estos venían incluidos con programas básicos para su uso y se empezaron a desarrollar los softwares conocidos como LISTA DE MATERIALES (BOM – BILLS OF MATERIALS) en el que simplemente se trataba de crear un escandallo de un producto. Es decir, incluir todas las piezas con sus respectivas cantidades que forman un producto. Gracias a este tipo de software permitía a las organizaciones poder realizar una planificación de la compra y gestión de las materias primas para así poder evitar roturas de stock.

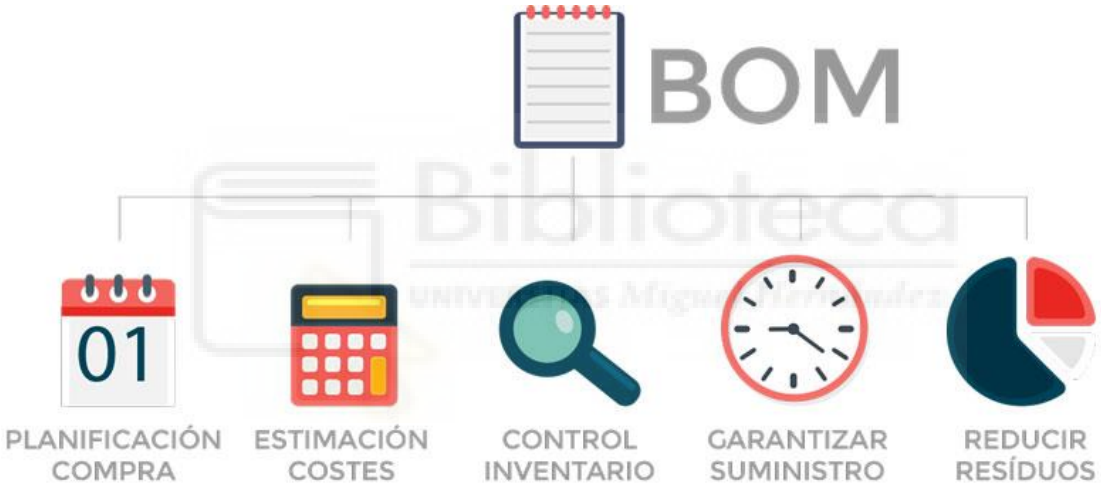


*Figura 2 – En que se divide el BOM.*

*FUENTE: https://development.grupogaratu.com/bom-bill-of-materials-componentes-fabricacion-producto/*

Una vez desarrollado el BOM se pasó a la creación del MRP, un MRP (PLANIFICACIÓN DE NECESIDADES DE MATERIALES) es un software que se encarga de evaluar los productos y materiales que se tienen, se compararán con los productos y materiales que se harán falta y además también determina en qué momento se harán falta. Esta es la diferencia del MRP con el BOM que permite calcular el cuándo de los productos.

Junto con el desarrollo de esta aplicación, se fue mejorando hasta que se llegó al punto de que se cerró el ciclo. A parte de desarrollar una aplicación que permitía ver el cuándo se podría disponer de ese producto terminado lo que se hizo fue intentar cerrar el lazo, en el que además se pudiera calcular cual era la capacidad de trabajo, es decir, cuantas máquina se disponen para realizar ese trabajo y cuánto tiempo implicaría realizarlo. Con esto, se podía ver si estos tiempos cuadraban o existía un desfase.

En la década de los 80, se evolucionó el MRP hasta alcanzar el MRP II. En este punto, donde solo intervenía la producción de una organización, se añadió el departamento de finanzas, en el que ahora las horas de trabajo de los operarios se evaluaban como un coste al que se le imputaba al material que se estaba tratando, y el producto terminado se consideraba como un activo más de la empresa y como una deuda que estaba pendiente de saldar con el cliente (esto considerado desde el departamento de contabilidad).

Para la década de los 90 ya aparece lo que conocemos como ERP. Consiste en ampliar el alcance a los diferentes departamentos de la empresa, para que así estos aporten su información y pueda ser accesible y usada para la toma de decisiones. Con esto se consigue un mayor control del proceso y una gestión de este, ya que con esto se define que departamento aporta cada información, y dentro de esto que persona debe aportar cada información, a que personas les debe llegar esta información y cuales serán las personas que decidan. Con esto permite mejorar los procesos existentes y por lo tanto obtener un mayor control.

En la actualidad, con la evolución de nuevos sistemas de gestión ha permitido desarrollar sistemas para poder integrarlos con el ERP que hasta se conocía, como son los softwares CRM (CUSTOMER RELATIONSHIP MANAGMENT) y los softwares PLM (PRODUCT LIFECYCLE MANAGMENT). Empezando por el primero, los softwares CRM se encargan de registrar y gestionar las relaciones que haga cada departamento de la empresa con los clientes y los potenciales clientes. Esto permite que se pueda realizar una atención personalizada a cada cliente ya que se conoce su forma de actuar y sus preferencias. En cambio, los softwares PLM hacen referencia a la gestión de toda la



UNIVERSITAS Miguel Hernández

información y documentación técnica generada por un producto en todas sus etapas, desde que es materia prima, y todos los procesos intermedios hasta que se alcanza el producto final. Gracias a este tipo de programas permite tener acceso de forma rápida y sencilla a toda esa documentación.

Biblioteca
UNIVERSITAS Miguel Hernández

# 2. ESTADO DEL ARTE

La situación actual es que existen diferentes ERP en el mercado (Outvio, 2020), tales como SAP, ORACLE, SAGE, ODOO, IFS… El principal inconveniente de estos problemas es el coste que tiene su adquisición y uso. Ya que se necesita adaptar completamente sus softwares, y que además convendría que se creará un equipo de personas para poder dar soporte a estos ERP desde dentro de la misma empresa.


*Figura 3 – ERP comerciales.*

*FUENTE: https://itspresso.com/sap-vs-odoo-vs-oracle-which-one-to-choose-for-your-enterprise/*

Todo esto implica un gran desembolso tanto de personal, como sobre todo económico. Por lo tanto, aparecía la oportunidad de enfocarse en un sector en concreto, como es el industrial, ya que concretamente, se dedique una empresa a lo que se dedique, siempre se regirá por unos estándares.

El punto que puede marcar la diferencia es la asistencia ante cualquier duda o problema, ya que el problema que puede haber al adquirir estos softwares ERP actuales, en no tener una asistencia personalizada, adaptándose a lo que el cliente necesita.

Y en el lado contrario a lo comentado con los ERP, hay muchas compañías que trabajan con softwares como Excel y Access. Si que permiten guardar información, pero la forma de visualizarla y gestionarla no es la óptima, ya que gestionarla se puede convertir en algo muy tedioso gestionarla.

## 2.1 MOTIVACIÓN

La motivación para el desarrollo de este proyecto nace de poder corregir muchos errores que tienen las empresas en su propio funcionamiento interno. Ya que, al ir enfocada a un entorno industrial, este software no está desarrollado para ver si los productos creados por la empresa cumplen las necesidades previstas o no. Sinó que está enfocado a que todas las relaciones de la empresa ya sean entre diferentes departamentos o entre la empresa y personas externes como proveedores, clientes sean las óptimos. Que siempre exista la mejor comunicación entre la propia empresa, y la empresa con el exterior.

El problema radica, en que, en una empresa pequeña, todas estas relaciones (tanto internas como externas) están centralizadas en muy pocas personas, por lo tanto, se conoce la información y puede ser fácil de recordar y acceder a ella. El problema aparece cuando la empresa crece, y esta empieza a dividirse en departamentos, en el que cada departamento gestiona sus tareas. Por lo tanto, cuanto más crece la empresa, está más se descentraliza y es muy posible que haya pérdida de información, esto puede llevar a generar malentendidos que provocarán pérdida de dinero directamente e incluso pérdida de tiempo (que a su vez provocaría pérdida de dinero).

# 3. LEAN

En este capítulo se definirá que es el concepto de LEAN MANUFACTURING, y como ha influido en la forma ver y entender a las empresas. Entonces, antes de comentar como han ido evolucionando las empresas gracias a técnicas LEAN, se deberá explicar estas que son.

Primero, el LEAN MANUFACTURING, traducido de forma literal podría significar como "producción rápida" o "sin grasa". De acuerdo a (Vizán, 2013) la definición de LEAN MANUFACTURING sería la de que se trata de una filosofía en la que se focaliza en las personas, que tiene como objetivo eliminar de los procesos de producción todo aquello que no genere valor. Es decir, todo aquel proceso que no aporte valor al producto debe ser eliminado. Pero no entender proceso como una etapa de la producción, sino es todo aquello que influye en obtener el producto final. Es decir, elevados tiempos de espera, exceso de niveles de stock, etc.

En definitiva, lo que el LEAN trata de conseguir una cultura en la empresa en la que se persiga la mejora continua, mejorando la comunicación interna en cada uno de los departamentos de la empresa. Pero no solo enfocada en una empresa del sector industrial, sigo a cualquier negocio, ya que la finalidad es conseguir una mejor comunicación y una mejor forma de trabajar, eliminando todo aquello que no aporta valor al objetivo final.

Ahora ya se está en posición de comentar que el LEAN no se trata de técnicas de reducir los costes como tal. No es conseguir materia prima más barata, o reducir los puestos de trabajo, sino el objetivo es identificar todo aquello que no aporte valor al producto final, y todo aquello eliminarlo. Es decir, todas estas acciones no son la causa de aplicar técnicas LEAN, sino todo lo contrario, representan la consecuencia de aplicar este tipo de técnicas. Siempre con el objetivo de aumentar la calidad del producto, en este caso la reducción de costes no forma parte del LEAN MANUFACTURING.



UNIVERSITAS Miguel Hernández

## 3.1 HISTORIA DEL LEAN MANUFACTURING

Tal y como se explica en el libro (Vizán, 2013) las primeras técnicas que aparecieron para organizar los procesos datan de finales del siglo XIX. En el que F.W. Taylor empezó a aplicar el método científico a personas, procesos y tiempos. También habría que destacar a Henry Ford, que se encargó de desarrollar la producción en cadena, en la que se utilizaban máquinas para la realización de tareas elementales y la estandarización de productos y procesos.

Posteriormente, a principio del siglo XX, Sahichi Toyoda, que unas décadas después se convertirá en el fundador de la compañía de coches Toyota, desarrolló un sistema que se integraba en las máquinas de telar, para que cuando hubiera algún tipo de fallo, lo tuviera y además indicara mediante una señal luminosa que ha habido algún tipo de fallo. Gracias a esta invención, permitía que el operador de la máquina, pudiera estar controlando a la vez más de una máquina en vez de una única máquina que podía controlar anteriormente. Es lo que se conoce como automatización con un toque humano.

En el año 1929, junto su hijo fundó la compañía Toyota. El problema surgió después de la Segunda Guerra Mundial, donde con una economía mermada, debían de intentar crear un sector competitivo. Entonces buscando técnicas para poder sobrevivir, dos empleados de Toyota viajaron a Estados Unidos para así poder aprender de los diferentes métodos productos que se aplicaban allí, el control estadístico y el control de la calidad. Se llegó a la conclusión, de que muchas de estas técnicas no eran viables para ser aplicadas en una sociedad como la japonesa, ya que no veían capaz de lograr economías de escala, ya que en ese momento, en Estados Unidos se fabricaban vehículos en gran cantidad, pero había una oferta limitada de diferentes modelos, siendo esta muy reducida, así lograban reducir los costes de su producción. En cambio, en Japón vieron que eso no sería posible, y se decantaron por una producción de modelos variada a bajo coste. Esto solo se podría realizar si se suprimieran stocks y una gran cantidad de despilfarros que se produce en el momento de la producción.

Por lo tanto, nació una nueva forma de gestión, que era la de producir solo que se demanda cuando el cliente lo solicite, para así poder reducir los stocks que tenían las empresas y reducir los costes que generan tener stocks. Por lo tanto, en la compañía Toyota, decidieron que lo que su filosofía de producción se basaría en producir solo lo que el cliente solicita, y empezar a producirlo en el momento que el cliente lo solicite. Para conseguir esto había que conseguir un flujo continuo en el sistema, en el que solo se produce lo que se solicita, y por lo tanto habría que reducir al máximo todos los tiempos que se pudieran, y el primer paso fue reducir los tiempos de cambio de herramientas. A raíz de esto empezaron a desarrollarse técnicas de producción tales como el JUST IN TIME (JIT) o el SMED. Estas técnicas se explicarán posteriormente.

Estos sistemas empezaron a convertirse en protagonistas, cuando en la crisis del petróleo en el año 1973, entraron en pérdida la mayoría de empresas japonesas y en cambio Toyota destacaba por su buen estado de salud. Entonces, el gobierno japonés fomentó que el sistema de producción que usaba Toyota fuera expandido por el resto del país para conseguir una mejor salud económica de las demás empresas.

Esta fue la época en la que se expandieron estas técnicas por Japón, en cambio por Europa y Estados Unidos hubo que esperar hasta la década de los 90, cuando se publicó el libro "LA MÁQUINA QUE CAMBIÓ EL MUNDO" en el que se trataba de comparar los diferentes sistemas de producción de automóviles de Japón, Estados Unidos y Europa. En esta obra detallaba que el sistema japonés era un sistema flexible y eficiente, capaz de ser usado en los otros lugares que se estudiaron. Conviene destacar, que en este libro, fue la primera vez que se usó el término LEAN MANUFACTURING para referirse a la forma/filosofía de trabajo de la compañía Toyota.

Por lo tanto, sabiendo por qué nació esta cultura, con el paso de los años se ha extendido de que la filosofía LEAN, según la información del libro de (Vizán, 2013)se basa en:

- Diseñar para fabricar.
- Reducir los tiempos de preparación de la máquina.
- Lograr la mejor distribución en planta que no implique un gran stock.

- Crear un sistema que permita detectar fallos lo antes posible.
- Formar al personal.

Biblioteca
UNIVERSITAS Miguel Hernández

# 4. PILARES DEL LEAN MANUFACTURING APLICADOS EN ESTE SOFTWARE

En este apartado se explicará como esta filosofía de trabajo ha influenciado este software, para que así todas las compañías que decidan implantarlo, les oriente hacia un modelo de gestión influenciado por estas técnicas. Porque la finalidad de este software no es solo organizar la gestión y el día a día de la compañía, sino además optimizar el trabajo diario de toda la organización, ya que está enfocado en trabajar en concreto en cada departamento para poder obtener el modelo óptimo de estos.

Entonces para explicar las influencias, al mismo tiempo que se detallan cada una de las técnicas LEAN, se explicarán como se reflejan en el software para que así, los clientes de este software puedan ir aprendiendo estas técnicas de una forma consciente ya que conociendo su utilidad, pueden ver de forma clara cuales son los beneficios que les aportan a su organización.

Por lo tanto, los pilares del LEAN MANUFACTURING comentados en (Vizán, 2013) que han influenciado este software serían:

## 4.1 PRIMER PILAR

### 4.1.1 JUST IN TIME (JIT)

Como ya se ha comentado, su traducción literal es la de “justo a tiempo”, y esta filosofía se puede aplicar a cualquier departamento de la empresa. Esto quiere decir que si estamos hablando del departamento de logística, el “justo a tiempo”, significa que el pedido una vez ya está terminado, estará listo para enviarse, y por lo tanto debe enviarse. Esto quiere decir que en el momento que el departamento de producción haya avisado de que todos los productos que componen un producto han sido fabricados, el departamento de logística deberá de estar preparando su envío. Es decir, ver que compañía de transporte interesa más según las características de todo lo que se envía y por supuesto el destino donde se mandarán.



UNIVERSITAS Miguel Hernández

Pero no solo esto afecta al departamento de logística, ya que "justo a tiempo" también significa que en el momento un cliente confirme un presupuesto que se les haya enviado, automáticamente este se debe convertir a pedido, y se debe informar al departamento de producción, para que según las características de los productos que tengan ese pedido, y el acuerdo que se haya quedado con el cliente para la fecha de envío, pues se deberá fijar una fecha para empezarlo a producir cuando toque.

Ahora, para el departamento del almacén, "justo a tiempo" significa que en el momento que el que son avisados, que un pedido ha sido terminado, deben de ir a prepararlo para su envío. Para que así este salga dentro de la fecha prevista y no haya retrasos. Pero no solo esto, sino, como se puede ver en el programa, cada uno de los productos que se fabrican y se montan en la empresa, están compuestos en etapas, y en cada etapa se realizan diferentes tipos de activades. Por lo tanto, lo que se trata es que el departamento de producción encuentre en cada etapa todo el material que necesita en el momento que sea necesario. Ya que ese material no puede estar mucho tiempo antes preparado en el taller a la espera de que sea montado, ya que esté estará ocupando un espacio, generando un despilfarro (además de suciedad).

Pero como todo el material tiene que estar preparado en el momento que el departamento de producción lo necesite, es donde entra el departamento de compras, ya que debe tener en cuenta los tiempos de cada proveedor, y el nivel de stock que haya, para poder elegir el momento idóneo en el que realizarle un pedido al proveedor, para disponer del material en el momento justo.

El siguiente departamento sería el departamento de producción, porque cuando logística recibe un pedido, siempre recibe la información del cliente de cuando es la fecha que le gustaría que se les realizara el envío. Con esta información, se debe empezar a ver que productos llevan ese pedido y en que fecha se debe empezar a fabricar para que así esté todo listo en las fechas más cercanas a las acordadas con el cliente sin pasarse. Pero lo que no puede ser posible es la de empezar a montar en el momento que se recibe el pedido,



UNIVERSITAS Miguel Hernández

ya que si el cliente no quiere que se le envíe pronto, estos productos terminados quedarán ocupando un espacio durante bastante, generando los sobrecostes ya mencionados.

Para el departamento de compras también se ha preparado el software para que se pueda seguir una nueva forma de trabajo en la que se aplique este primer pilar, ya que como se ha explicado en capítulos anteriores, el departamento de compras se encarga de que adquirir todo el material que se compra a un proveedor ajeno, es decir materia prima o productos ya fabricados. Por lo tanto, todo este material debe estar siempre disponible para que el departamento del almacén se lo saquen al de producción para que lo monten en producción.

Gracias al tipo de diseño de este software se intenta conseguir que, de una forma indirecta, se siga este pilar, en el capitulo de software donde se explica el programa se ve como te fuerza a seguir esta filosofía de trabajo.

### 4.1.2 8 DESPERDECIOS

Este apartado está incluido como un subcapítulo dentro del primer pilar, ya que va acompañando a la filosofía JIT, ja que al aplicar las técnicas JUST IN TIME, lo que se está buscando es reducir cada uno de los costes que se suelen generar en las empresas por una mala gestión.

El primero de los desperdicios sería los defectos. Aquí nace la necesidad, primeramente de controlar los tiempos del departamento producción, porque si un producto se está produciendo en la mitad de tiempo, posiblemente esa etapa no se haya hecho correctamente, y una vez el cliente tenga el producto, este podrá resultar defectuoso. Y también puede ocurrir lo contrario, que una etapa dure más tiempo del que debería durar. Esto puede ocurrir que haya habido problemas en montar algún tipo de pieza, y la causa puede ser que una sea defectuosa. Entonces si este tipo de piezas han escapado al control visual, puede ser una forma de encontrar un defecto de pieza.



UNIVERSITAS Miguel Hernández

Y por supuesto, lo que ya se ha comentado, control visual tanto del departamento de producción con lo que se está fabricando para ver si cumple con los estándares de calidad, y control visual del departamento de compras para ver si los productos que vienen directamente del proveedor, también cumplen con los estándares acordados.

El segundo desperdicio sería el de un exceso de producción. Con este software se trata de evitar la sobreproducción trabajando sobre pedido. Es decir, se empieza a fabricar en momento el cliente realiza un pedido al departamento de logística y logística le pasa la orden al departamento de producción para que lo empiece a tramitar. Así se evita el producir de más, reduciendo todos los gastos que esto conlleva.

Otro desperdicio sería el del tiempo de espera, por eso es muy importante la comunicación con el cliente, que aquí empieza tanto por el departamento comercial o por el departamento de logística, en el que le deben de consultar al cliente en que momento necesita ese producto. Para así poder ver si es algo urgente porque se le ha roto una pieza y necesita reemplazo, o por si en cambio necesita una nueva máquina para la próxima temporada de producción, y por lo tanto puede esperar un tiempo. Con toda esta información de cuando el cliente necesita el producto, hay que hacérsela saber al departamento de producción para que así pueda ver en que momento interesa empezar a producirlo. Es por eso que aparece en la tabla de pedidos, cual debería ser la fecha prevista en la que el producto debería de salir y a continuación ver cual ha sido la fecha en la que realmente ha salido. Con esto se puede ver si se producen costes derivados del tiempo de espera.

Y no solo en tiempos de espera con el cliente, sino en tiempos de espera internos. Como en el departamento de producción que no se pueda cambiar de etapa de producción por que haya un “tapón”. Con la gestión de tiempos de producción se puede ver si hay algún atasco en alguna etapa de producción. Así una vez detectado donde está el problema, se puede ver que es lo que lo está causando para así poder eliminarlo.



UNIVERSITAS Miguel Hernández

Ahora, los siguientes desperdicios a comentar, aunque sean 3, se pueden englobar en 1, ya que serán comentados conjuntamente. Estos desperdicios son los desperdicios generados por el transporte, por el inventario y por los movimientos. Para eliminar estos 3 desperdicios se podría usar lo comentado en unos párrafos anteriores, que se trata de mover el material y las materias primas lo justo y necesario, solo cuando haya que cambiar etapas de producción. Y las materias primas se deben de sacar del almacén solo cuando haya un pedido en el que sea necesario su uso. Porque siempre lo que hay que tratar de reducir es mover los productos lo mínimo posible y tener el mínimo stock posible.

Los dos últimos desperdicios serían los relacionados con el talento humano no aprovechado y los reprocesos. Para estos reprocesos no habría una aplicación que permita reducirlos de forma "directa", simplemente que el diseño de este software ha estado enfocado a reducir los errores/problemas relacionados con la falta de comunicación entre departamentos, que pueda provocar repetir alguna de las etapas.

### 4.1.3 KANBAN

Dentro del pilar en el que se busca una forma de trabajo en la que la empresa funcione de acuerdo con el ritmo de demanda de los clientes. Aparece el término KANBAN. Se trata de una palabra japonesa que significa signo. De lo que trata esta filosofía es que hayan señales o avisos entre las diferentes etapas de un mismo departamento, y que incluso mejore las comunicaciones entre los diferentes departamentos.

Esto significa de dejar una señal, lo más visible posible para que así no se pierda esa información. Pero esa señal que sirve para transmitir información podría ser entendida como una orden, en la que no se empiece la siguiente acción, hasta que no se reciba la señal/orden de que es posible empezarlo. Para explicar esto, se vuelve al inicio, en el que todo empieza con la entrada de un pedido que realiza un cliente. Con este pedido, empieza el ciclo, ya que el departamento de logística es el encargado de dejarle la "señal" al departamento de producción de que ha entrado un nuevo pedido que está pendiente de tratar. Entonces es cuando el departamento de producción, junto con la información



UNIVERSITAS Miguel Hernández

reunida por logística, sería el que se encargaría de ver cual sería le mejor momento para gestionarlo, según su contenido, y el momento que el cliente quiere que se le entregue.

Con toda esta información, una vez se ha estudiado cual sería el momento que se debe empezar a montar, por la aplicación, se debe de indicar al departamento de almacén esta información, así ellos para ese día deben de tener todo el material dispuesto para empezar directamente a fabricar y/o montar.

Y además, mientras se está fabricando el producto, se deben de ir señalizando en que etapa se encuentra, ya sea indicando que etapa ha empezado, y cual es la inmediatamente anterior que ha terminado. Todo esto indicando los tiempos que ha llevado para hacer cada cosa.

Una vez está terminado, hay que mandarle un aviso a logística, de que el producto está terminado, para así gestionar el envío y que almacén lo pueda preparar y embalar.

## 4.2 SEGUNDO PILAR: KAIZEN

El segundo pilar (Vizán, 2013) del que se basa el LEAN MANUFACTURING sería el "Kaizen", es otra palabra de origen japonesa que significa "cambio a mejor" o incluso "mejora continua". En el que se busca identificar todos los procesos que no generan un valor añadido y solo dejar aquello que incremente el valor y la calidad del producto.

Pero la mejora continua partiría de todos los niveles, es decir, que no solo aporte las soluciones la dirección de la organización, sino que se tenga en cuenta las ideas de los empleados pasando por los responsables de departamento.

### 4.2.1 LAS 5 “S”

Dentro del segundo pilar de mejora continua (Vizán, 2013), se encuentra la técnica de las 5 S, en la que se trata de buscar una mejora manteniendo siempre un orden y una estructura, las 5 S son:

- Seiri -> Clasificación
- Seiton -> Orden
- Seiso -> Limpieza
- Seiketsu -> Estandarización
- Shitsuke -> Mantener la disciplina

Aquí en este caso, no se hablará de como ha aplicado esta técnica a la empresa, sino de como se ha aplicado la técnica al software, porque así permite que la organización siga los principios que ha seguido en software en su diseño. Este programa ha sido diseñado basado en estos pilares, para que las personas que se encarguen de usarlo se lo encuentren en el que la información este clasificada según cada departamento, y que la secuencia de pestañas siga un orden concreto (como ya se ha explicado el software empieza desde que nace y se crea un producto nuevo, pasa desde que se presupuesta y se genera pedido, hasta que llega a terminar en el punto en el que se recibe información por parte del cliente de si está trabajando bien o no, como sería el departamento postventa).

El siguiente paso sería la limpieza, con la limpieza simplemente es que en el software solo se han introducido los campos y las tablas mínimas para poder añadir toda la información necesaria. Con la estandarización, se ha conseguido que siempre el proceso siga los mismos pasos, y sea quien sea el que en ese momento esté llevando una parte de la organización, sepa donde tiene que acceder para modificar cierta información, o donde tiene que acceder para poder consultarla.

Y por último, sería el de mantener la disciplina. Seguir siempre los canales de comunicación que aporta el software, y guardar ahí toda la información. Ya que en un principio puede parecer más rápido trasladar una información mediante una llamada telefónica, pero al final la información que aquí se comenta no queda recopilada en ningún

lado y no sería accesible por la empresa, ni ahora, y más aun, en años posteriores cuando se quiera ver como se realizó cierta actividad.

## 4.3 TERCER PILAR: CONTROL DE CALIDAD

Como tercer y último pilar aparece el control de calidad (Vizán, 2013). Es decir, que controles o mecanismos proporciona el propio software para garantizar la calidad en la forma de trabajar de todos los departamentos de una empresa. Y no solo eso, sino que además serían los mecanismos para garantizar que lo explicado en los dos primeros pilares se cumple.

Una vez comentado, como se intenta garantizar que la organización tenga una respuesta que sea justo a tiempo a cuando entra un pedido, como influye a los departamentos de compras, logística, almacén y producción. Y también se ha comentado como estos departamentos tienen que tener la mentalidad de mejorar cada día.

Pues aquí entra la importancia del último departamento del que todavía no se ha hablado en los anteriores pilares, y es el departamento de servicio técnico o también conocido como el departamento de postventa. Porque la función que tiene este departamento es la de recopilar todas las interacciones que se tienen con los clientes para así después ver porque se producen.

De ahí la importancia de lo que se comentaba en el capítulo en el que se explica cómo se utiliza el software, de porque clasificar cada interacción con los clientes, de porque se producen, ya que después se puede realizar una búsqueda por los diferentes motivos, así se puede ver cuántos hay de cada tipo. Esta información es la que se tiene que usar para ver si se está trabajando de una forma incorrecta, ya que, si en un periodo de tiempo esto se repiten los mismos motivos en diferentes clientes, ya no serían casualidades, sino que indicaría un patrón, y por lo tanto se debería ver que está ocurriendo en el departamento de producción, concretamente en la etapa donde se trabaje en la parte defectuosa que reporta el cliente.



UNIVERSITAS Miguel Hernández

Y no solo muestra los datos ordenados dentro de una tabla, sino que los puede organizar en gráficos, para que de una forma visual se pueda ver, de la cantidad total de incidencias que ha registrado el departamento de postventa, cuantas son de cada tipo.

Para este control de calidad, aparte del control de calidad la calidad interna, sobre todo va enfocado a la calidad que percibe el cliente del producto que está adquiriendo y del servicio que recibe. Por eso hay que estar pendiente de cuando un cliente adquiere un producto reciba toda la documentación, poder gestionar las garantías del cliente, además de poder canalizar las propuestas de presupuestos por parte de los clientes y sin olvidar el estudiar cada una de las quejas que recibe la empresa. Ya que se deben ver si las quejas que se reciben se tratan de quejas puntuales o si en cambio diferentes clientes (con diferentes situaciones, ya que pueden ser de sectores diferentes o de países diferentes) y por lo tanto, si son quejas similares en clientes que no tienen relación entre ellos es para darle una importancia considerable. Con esto permite detectar errores o fallos en el desarrollo de la organización para así poder cortarlos de raíz y poder mejorarlos. Siempre con el fin de mejorar la calidad.

# 5. HERRAMIENTAS SOFTWARE UTILIZADAS

## 5.1 JAVA

JAVA se trata de un lenguaje de programación, es decir, en el que se escriben líneas de código con comandos para así posteriormente compilarlo y así resolver un problema que se haya presentado.


*Figura 4 – Icono del lenguaje.*

*FUENTE:https://www.oracle.com/java/technologies/*

Para empezar a destacar sus características según el trabajo (Guevara), la primera sería la portabilidad. Esto se refiere a que una vez compilado, permite utilizarlo en diferentes sistemas operativos y es independiente de la plataforma. Todo esto es gracias a que el código escrito en JAVA se compila a un lenguaje de programación intermedio que sería bytecode, ahorra este código sería interpretado por la máquina virtual de JAVA del entorno de ejecución (conocido como JRE) y sería en este punto donde crearía un ejecutable que sería portable a diferentes sistemas operativos como WINDOWS, IOS, MAC OS, ANDROID…

Además, JAVA dispone de una amplia variedad de bibliotecas (Walton, 2018). Gracias a esta variedad de bibliotecas que contienen diferentes métodos y clases que permiten facilitar la programación del usuario sin necesidad de que las tenga que crear el mismo usuario, simplemente basta con importar estas bibliotecas. Con esto permite reducir el tiempo de programación, pudiendo aprovechar el código desarrollado por otros usuarios.

Por último, este lenguaje de programación permite trabajar con programación orientada a objetos. Ya que en los objetos se le pueden añadir atributos y poder trabajar con herencias para así relacionarlos con otros objetos. En cambio, en la programación de este software se ha seguido la idea de programación según métodos y funciones.

## 5.2 NETBEANS

Para poder programar todo este código se ha usado un entorno de desarrollo integrado (IDE). Un IDE se trata de una herramienta de software que permite desarrollar código, en el que ayuda a detectar errores de sintaxis y autocompletar código. Además de esto, aporta un simulador que permite comprobar el código programado para ver si cumple con la idea que se buscaba o si en cambio hay que volver a reprogramar. Por último, también permite depurar código, en el que indica que acciones realizar para poder trabajar con el código más fácil y corto posible.

Biblioteca
UNIVERSITAS Miguel Hernández


*Figura 5 – Icono de NETBEANS.*

*FUENTE:*
*https://cwiki.apache.org/confluence/display/NETBEANS/NetBeans+Logo?desktop=true&macroName=view-file*

El IDE utilizado es NETBEANS. El motivo de su elección (Fantino, 2021), ya que es de código abierto, así permite importar diferentes librerías y código desarrollado por otros programadores, para así poder aprovechar la ventaja en la velocidad de programación. También hay que destacar que es el IDE más usado para programar en JAVA.

## 5.3 MYSQL

En este programa lo que se busca es de aportar un entorno gráfico para poder almacenar toda la información de la empresa, clasificada según los diferentes departamentos. Por lo tanto, hasta ahora solo se ha explicado como crear ese entorno gráfico, y el siguiente paso sería el que toda la información que se escribe en ese software se almacene en algún lugar para así cuando sea necesaria poder mostrarla de nuevo.

*Figura 6 – Icono MYSQL.*

*FUENTE:https://aws.amazon.com/es/rds/mysql/*

Para poder realizar esto, lo que haría falta sería una base de datos. En concreto, la base de datos que se ha utilizado es MYSQL. Este software y lenguaje de código para base de datos, en la actualidad es propiedad de ORACLE.

MYSQL se trata de un gestor de bases de datos relaciones y además es de código abierto (Gustavo, 2022) con un modelo cliente-servidor. Por lo tanto, lo que caracteriza MYSQL sería:

- Base de datos relacional.
- Código abierto
- Modelo cliente-servidor.
- Sintaxis SQL.

Para explicar estas características, habría que empezar explicando que es una base de datos relacional. Esto quiere decir que todos los datos tienen una posición en concreto. Entonces esto significa que se está trabajando con tablas. Los datos de esta tabla ocupan

siempre una fila y una tabla en concreto, y este siempre será su posición. Al ser un software de código abierto también permite que pueda ser usado en diferentes plataformas.

Con lo de modelo cliente-servidor se refieren a que todo ordenador hace de cliente, por lo tanto, para poder tener acceso a los datos de esa base de datos se tienen que conectar a un servidor de MYSQL, que debe ser instalado en el computador. Así el servidor le transmitirá los datos al cliente y el cliente será él que los muestre.

Y por último, ya se ha comentado que MYSQL implementa un modelo cliente-servidor. Entonces, para que el cliente guarde datos en el servidor, o haga cualquier modificación o haga simplemente consulta de información necesita un lenguaje que los comunique entre ellos. Este lenguaje es el que se conoce por SQL. La traducción de estas siglas sería LENGUAJE DE CONSULTA ESTRUCTURADO. Este lenguaje de programación se usó para poder trabajar con las grandes cantidades de datos que se almacenan en las bases de datos.

Biblioteca
UNIVERSITAS Miguel Hernández

# 6. SOFTWARE

Como software de gestión del entorno industrial lo que debe incluir se trata de toda la gestión del producto antes, durante y después. Esto quiere decir que se debe empezar a recopilar información de la materia prima o pre-conjuntos que son los que se utilizarán para crear el producto, controlar todo el proceso de producción con sus diferentes etapas y por último recibir el feedback y procesarlo de cuando el producto ya ha abandonado la empresa.

Todo esto mencionado al párrafo anterior se debe incluir toda la logística que existe en cada etapa, ya que no se puede dejar de lado todo lo relacionado con los pedidos de los clientes, la organización del almacén y del stock para que el departamento de producción pueda disponer de todo el material para poder realizar sus actividades unido con toda la logística de los envíos.

Para empezar a explicar este software, hay que destacar que la filosofía que se ha seguido en su diseño y es la comentada unos párrafos arriba, y es seguir el proceso de producción, para eso se ha decidido dividir la aplicación en el siguiente conjunto de pestañas:

- Conjunto.
- Cliente/Proveedor.
- Pedidos.
- Envíos/Abonos. Producto.
- Almacén.
- Órdenes.
- Gestión.
- Postventa.

## 6.1 PRODUCTO.

En la primera pestaña, en la llamada PRODUCTO es donde se debe indicar cada producto “simple” o, mejor dicho, artículo. Con esto quiero decir que debe ser en individual. Por ejemplo, si en esta empresa se diseña una pieza de nylon, que se obtiene de un bloque de

400x300x100 mm, el bloque se obtendrá marcándolo como materia prima, y tendrá una referencia destinada a las materias primas, a continuación, esta pieza se le realizará algún proceso para transformarla en la pieza que se necesita para poder ensamblarla en la máquina/robot. Esta última pieza también tendrá una referencia, que será diferente a la de la de la materia prima.

También está el caso en el que se adquiere algún tipo de pieza/material/dispositivo de un proveedor y que se utilice directamente para el producto final sin necesidad de aplicarle ningún proceso de modificación o fabricación. Con estos productos me refiero a tornillos, arandela, tuercas, motores, variadores, servos, sensores, cilindros, racorería, cables, dispositivos de periferia, etc.

Estos tipos de productos son los que se añadirían en esta pestaña. Las diferencias entre ellas ya han sido comentadas arriba, pero la similitud que tienen entre ellos y que hace que se deban incluir en esta pestaña es que son piezas de una unidad indivisible, en el que no se encuentra compuesto por nada más.

Una vez comentado esto, la pestaña PRODUCTO del software tendría el formato que se puede ver en la imagen de debajo. Esta pestaña se divide en 3 partes:

- Crear el producto.
- Buscar el producto
- Editar el producto.

En la fase de crear el producto, aparecen unos campos que nos ayudarán a definirlo. Estos campos son:

- Referencia.
- Descripción.
- Precio.
- Propio.
- Conjunto.
- Nombre del proveedor.
- País.
- Coste.

En este apartado es donde se escribirá el código que identifique ese producto. Este código estará formado por 7 dígitos y tendrá el formato XXX-XXXX. El motivo de que tenga esa estructura, de tener 3 dígitos delante de un guion es para poder identificar a que familia pertenece. Por ejemplo, de esta forma, de un primer vistazo se puede saber a qué proveedor pertenece, de si se trata materia prima, de un producto "simple" o de un conjunto (en la siguiente pestaña se explicará que se refiere el término conjunto en este programa de gestión). Aquí se ve ya la influencia de las técnicas LEAN, como por ejemplo las 5S, en las que se muestra como seguir un orden, una limpieza y una estandarización en la forma del trabajo.

El siguiente campo es el de la descripción, aquí simplemente se trata de añadir una frase para poder definir el producto. Así ayudará a identificar el producto por si se desconociera la referencia. Seguido de este campo se encuentra el precio, ahí se debe añadir el precio de venta del producto. Este valor deberá ser previamente calculado por el departamento de I+D/ingeniería por ser aquellos que lo hayan diseñado junto con el departamento de compras, ya que son ellos los que lo adquieren del proveedor y saben lo que cuesta ese producto o materia prima.

A continuación, aparecen 2 campos booleanos, esto quieren decir que son de SI/No. Los dos campos que aparecen son los de propio y conjunto. Propio quiere decir que si es una pieza o conjunto diseñado y fabricado por nosotros. Y el campo de "conjunto" se refiere a si se trata de un conjunto o no. El significado de conjunto es cuando está formado por varios productos indivisibles y necesita tener carácter propio.

Los últimos 3 campos que aparecen en ese apartado están relacionados con si se ha indicado que se trata de un producto propio o no. Si no se tratara de un producto propio sería necesario rellenar los últimos campos. Estos hacen referencia a el "nombre del proveedor" en el que con la descripción deja claro a que hace referencia. A continuación, aparece "país" en el que se debe especificar el país del que proviene el producto. Y por último está "coste". Para que no se confunda con "precio", el primero hace referencia a la cantidad de dinero que se debe pagar al proveedor por adquirir ese producto, y en cambio, como se ha dicho dos párrafos anteriores a esto, "precio" es la cantidad de dinero

que cuesta la venta de ese producto. Es evidente, que productos que no sean propios tendrán un valor en el precio y otro diferente en el coste.

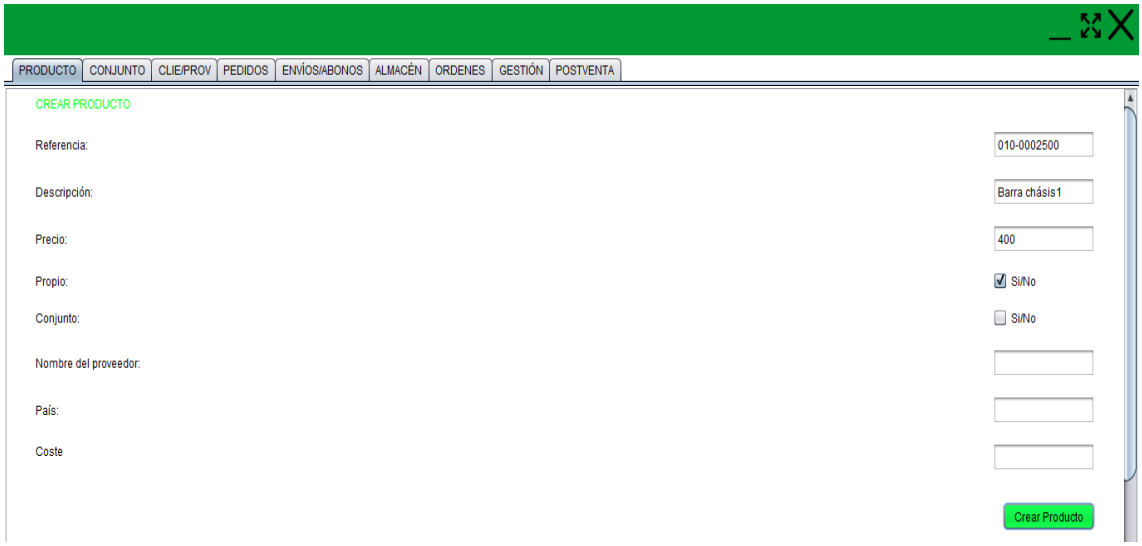


*Figura 7 – Pestaña PRODUCTO y campos para "crear producto".*

*FUENTE: Propia*

En el siguiente apartado de esta pestaña aparece lo de buscar producto, porque será necesario poder ir buscando todos los productos que se encuentren en la base de datos para poder usarlos en pestañas siguientes para poder introducirlos en pedidos y demás. Y no solo eso, porque puede ser que se tenga duda de si lo que se ha añadido en algún campo es correcto o no. La clave se encuentra en poder realizar la búsqueda tanto por:

- La referencia del producto, gracias a esto se podrá ir filtrando a través de la referencia. Aquí aparece una de las ventajas de poder relacionarlo con si se ha seguido o no una estructura y un orden a la hora de crear las referencias de los productos. Ya que, si se sigue una estructura de primero indicarla familia con un código, se puede escribir el código de esa familia, por ejemplo, si se quiere buscar productos (que no sean conjuntos) fabricados por la propia empresa a través de materia prima adquirida, se podría introducir el prefijo "010" y ver todos los productos que tiene esa empresa dentro de esa familia.

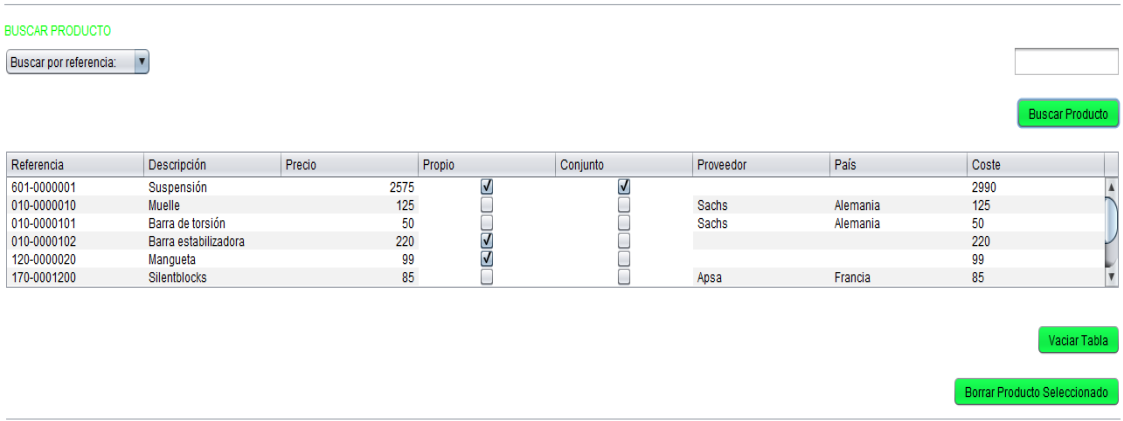


*Figura 8 – Buscar producto SIN ninguna condición.*

*FUENTE: Elaboración propia*

- También si se sabe que es uno de la familia 010, pero se sabe que es uno de los primeros que se diseñó, se podría acotar la búsqueda escribiendo 010-00, y ya finalmente te busca todos los productos que empiecen por esa referencia.

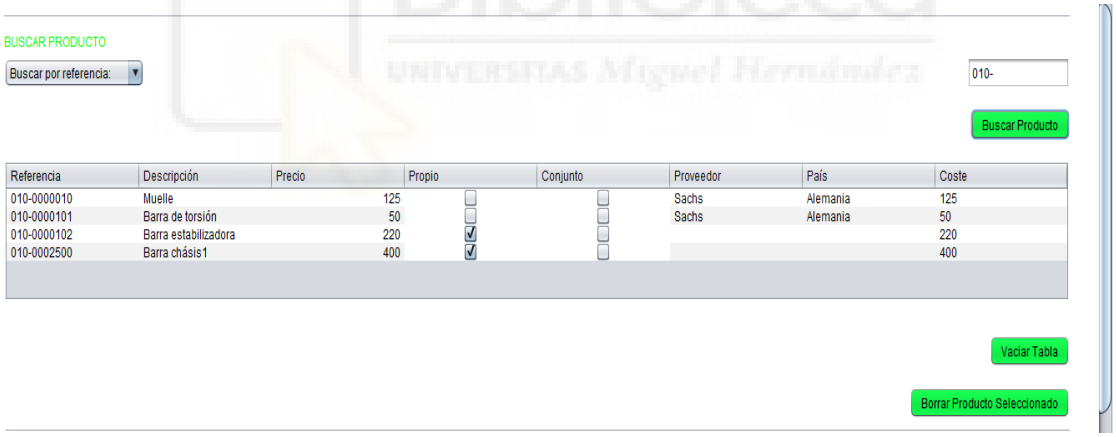


*Figura 9 – Búsqueda por condición de familia de producto.*

*FUENTE: Elaboración propia.*

- A través de la descripción del producto. Esto quiere decir que, si no se conoce a que familia pertenece, o que no se puede filtrar más a través de la referencia se pueden realizar búsqueda a través de palabras clave. Aquí recae la importancia de cuando se crea la descripción de un producto, que esta tenga las menores palabras posibles y que sirvan lo máximo posible para definir el producto para así poder identificarlo lo más rápido y fácil posible.

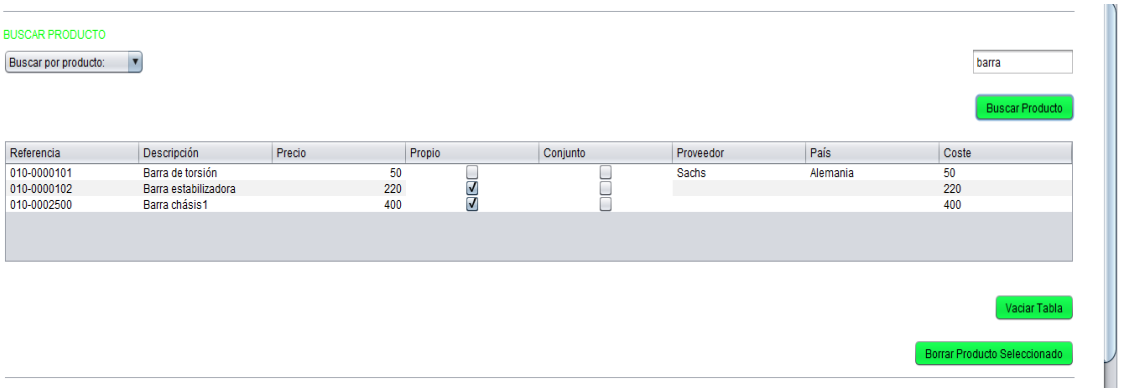


*Figura 10 – Búsqueda por criterio de descripción.*

*FUENTE: Elaboración propia*

Quizás, aunque no es óptimo desde el punto de gestión, sería posible realizar una búsqueda sin introducir ningún texto en ese campo y permitirá realizar una búsqueda total de todos los productos que se encuentran almacenados en la base de datos.

Como se ve en las imágenes de arriba, aparece un botón más, ya que en este apartado no solo permite realizar búsquedas de productos, puede ocurrir que un producto que se estaba intentado dar de alta en la base de datos para poder añadirlo a pedidos y demás se haya introducido de forma errónea. O quizás que un producto, con el tiempo haya quedado descatalogado o simplemente se ha decido de que no se vuelva a usar. Entonces, para cualquiera de estos casos se decide borrarlo para así evitar posibles confusiones por exceso de productos y además permite liberar espacio de la base de datos.

Otro botón que aparece en este apartado, que aparecerá siempre acompañado de una tabla para mostrar datos, se trata el de “VACIAR TABLA”. Ya que una vez se haya realizado una consulta, en ella aparecerán datos, y si posteriormente se crea un nuevo producto, estos no aparecerán, porque las tablas no estarán realizando búsquedas constantes, solo las harán en el momento que se pulsa sobre el botón, por eso interesa ir vaciando la tabla antes de realizar una nueva búsqueda.

Para terminar con esta pestaña, el último apartado está destinado para poder realizar una modificación sobre alguno de los productos ya creados. Porque no sería óptimo tener que borrar un producto y tener que crearlo de nuevo para simplemente cambiar un campo.

No todos los campos se pueden modificar todos los campos de la tabla productos, ya que este apartado no está hecho para corregir errores. Los motivos se explicarán a continuación de comentar que campos son los que sí se pueden editar:

- Descripción.
- Precio.
- Nombre del Proveedor.
- País.
- Coste.

El motivo por el que se puede modificar la descripción es que un producto ya creado y conforme ha ido pasando el tiempo, que se han creado más productos, puede darse la situación en que la descripción antes era la correcta, pero ahora no se adapta a la situación actual, ya que puede entrar en conflicto con otros productos con una descripción que si es la adecuada. Es por eso por lo que ahí lo más conveniente es actualizar la descripción.

En el caso del campo precio es similar, porque con el paso del tiempo puede darse diferentes factores que haga que se modifique el precio de un producto. Ya que la materia prima que interviene en la producción de ese producto puede ser más barata/cara, o simplemente de un año a otro varía la inflación y sería necesario realizar una modificación de los precios para adaptarlos a las nuevas condiciones. Así que simplemente se modificaría ese campo.

Los siguientes campos que quedan van relacionados con productos que se adquieren de un proveedor, es decir que no son propios. Lo que se pretende aquí es que puede ocurrir que un proveedor cierre, o simplemente, interese cambiar de proveedor por mejores condiciones. También puede pasar que el proveedor cambie su sede social y sea necesario cambiarlo, para así saber cuántos productos vienen de cierto país, ya que según el tipo de producto que sea y dependiendo del producto que sea, habría que declararlo de cierta forma, pagando unas tasas acordes a ello. Y por último aparece el campo de coste, ya que nos pueden variar el precio de compra de los productos y, por lo tanto, toda variación deberá quedar registrada en el sistema, para que dispongan de ella todos los departamentos para poder tomar las decisiones necesarias y hacer los cálculos que se deban de hacer.

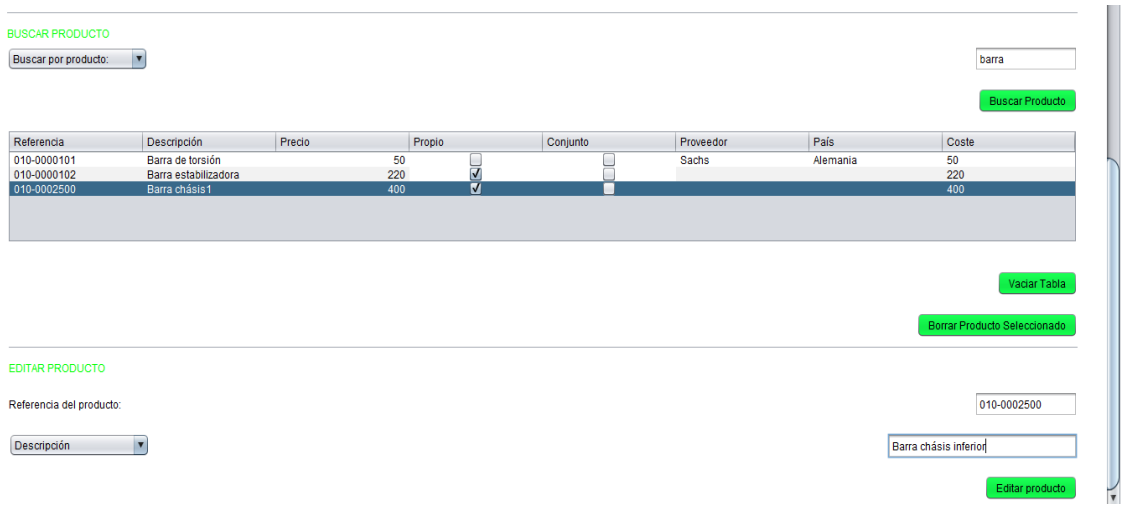


*Figura 11 – Se muestra el momento de antes de editar un producto.*

*FUENTE: Elaboración propia.*

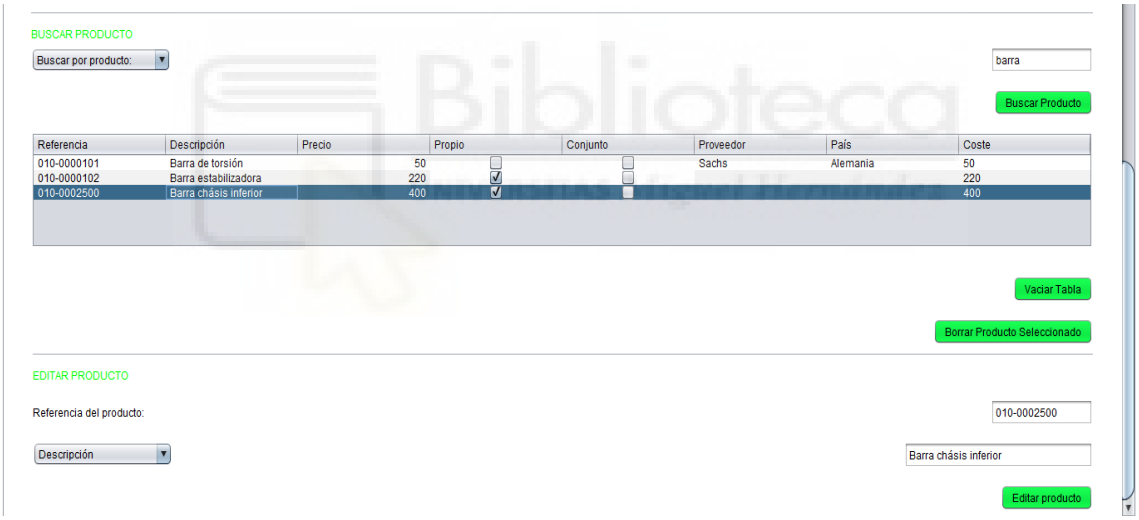


*Figura 12 – Muestra el resultado después de haber editado un producto.*

*FUENTE: Elaboración propia.*

## 6.2 CONJUNTO

La siguiente pestaña que aparece es la de "CONJUNTO", ya se ha comentado por encima en el capítulo anterior que significa este término, pero aquí se explicará un poco más en detalle. Ya su propio nombre hace indicar a que hace referencia, este engloba a todos los productos simples/indivisibles que ensamblados conjuntamente a través de tornillería

forman un nuevo producto (es por eso que, para poder identificarlos, estos productos tendrán unos 3 primeros dígitos de la referencia en concreto solo para ellos).

Debido a la singularidad o diferencia con los demás, es por eso que se ha decidido crear una pestaña para ellos. Porque no solo se trata del producto-conjunto que se crea, es por que como se ha explicado en el párrafo anterior, un conjunto lleva implícito que está formado por otras piezas. Es por eso por lo que se debe poder hacer que constar de que piezas está compuesto cada conjunto. Y no solo eso, sino en que cantidad. Con esto obtendremos el escandallo del producto. La importancia del escandallo de un producto es elevada en un entorno industrial, ya que, por ejemplo, cuando se ha de ensamblar ese conjunto se debe conocer de que piezas está compuesto para que lo puedan sacar al taller de montaje. Y las personas que se encargan en el taller de montaje también se deberán apoyar de este y de manuales o planos de despiece para poder guiarse para así formar la pieza conjunto.

En las 2 imágenes que vienen a continuación se puede ver el aspecto y la filosofía que sigue esta pestaña, que a diferencia de la anterior, no está diferenciada en apartados, ya que en esta se necesita de relacionar las diferentes acciones.

Para explicar estas acciones, se explicarán cada uno de los botones ya que estos son los que lanzan estas acciones para comunicar el entorno gráfico con la base de datos. Entonces, el primer botón que aparece (leyendo de arriba hacía debajo) sería el de “BUSCAR PRODUCTO”, en el que los datos que se obtengan de la búsqueda realizada se mostrarán en la tabla inmediatamente debajo. Esta parte es similar a la de la pestaña anterior (se explicará después el motivo de porque replicar la misma función en la siguiente pestaña) ya que se puede realizar búsquedas a través de 2 condiciones. Estas serían buscar por filtrando por la referencia y la otra sería la de buscar por la descripción. La elección del tipo de búsqueda se realizará pulsando en el desplegable, en el que se mostrará las 2 opciones de búsqueda ya comentadas. Y el criterio se añadirá justo en el campo de texto que se encuentra a la derecha a la misma altura que el elemento desplegable comentado.



UNIVERSITAS Miguel Hernández

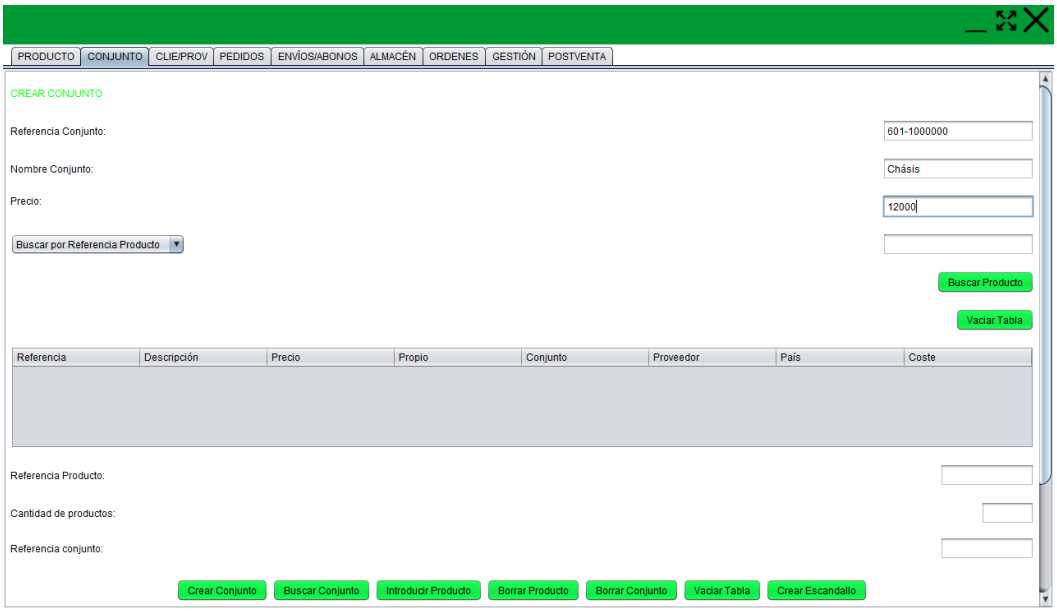


*Figura 13 – Crear nuevo conjunto.*

*FUENTE: Elaboración propia.*

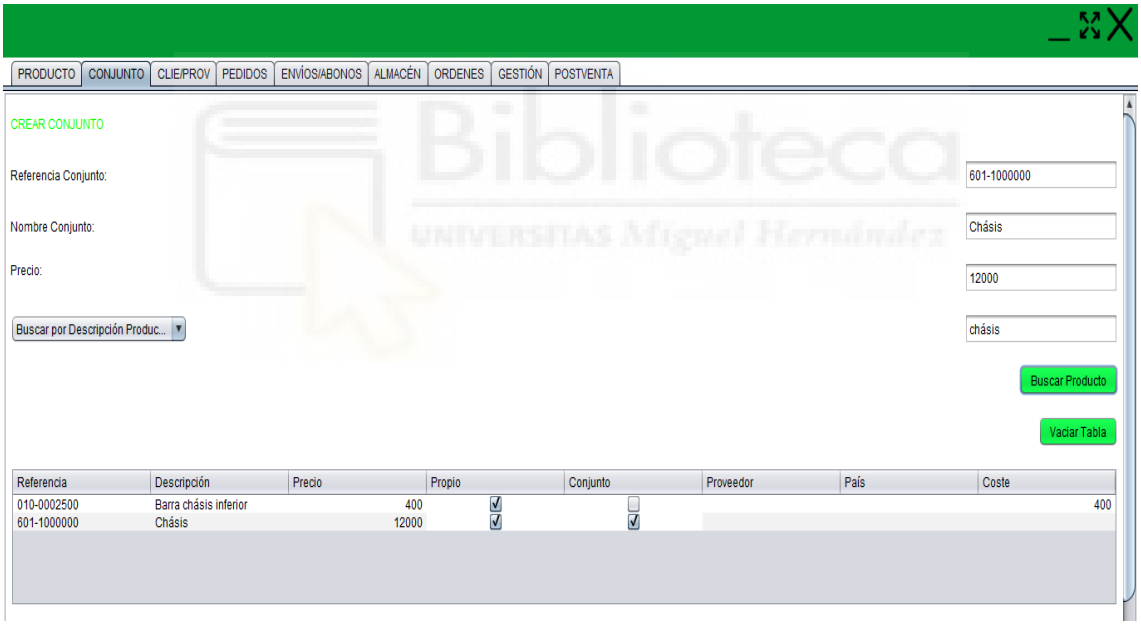


*Figura 14 – Buscar en la tabla de conjuntos donde se muestra el conjunto creado.*

*FUENTE: Elaboración propia.*

El siguiente botón el de "VACIAR TABLA", sería el de vaciar esa primera tabla, para así dejarla sin información para después poder realizar una nueva búsqueda de productos.

La siguiente acción que aparecería en esta pantalla sería la de "CREAR CONJUNTO". En este momento sería cuando entran en juego los campos que aparecen justo al principio de esta pestaña, que hasta el momento no se han llegado a comentar. Ya que para crear

ese conjunto lo que se trataría es de especificar que referencia llevará ese conjunto (a los conjuntos se les ha decido que lleven el identificador 600), a continuación, se indicará la descripción del producto y por último se fijará el precio de venta al consumidor.

En definitiva, lo que se está haciendo en esta aplicación, es lo mismo que se realizaba en la pestaña anterior, pero en ella podías marcar si se trataba de un conjunto o no. En cambio, en esta pestaña, todo lo que se cree se trata de un conjunto, no hay posibilidad de poder modificarlo. Pero evidentemente, toda la información se almacenará en la misma tabla de la base de datos, no es necesario duplicar esa tabla para después relacionarlo. Simplemente que al guardarlo, se marca directamente como conjunto y al crearlo la propia empresa no es necesario añadir ninguna información relacionada con el proveedor, ya que no existe.

Aunque parezca que se ha duplicado pantallas y funcionalidades, tiene un motivo de que la misma acción se realice por diferentes partes, ya que en los softwares de gestión puede tener diferentes niveles de acceso e incluso un acceso personalizado para cada usuario en función de cual sea el departamento o el cargo que ocupe el usuario en la empresa. Porque en los casos en el que el nivel de acceso sea el mínimo y si se quiere diferencia entre las etapas de fabricación y las etapas de montaje, que cada uno tenga su acceso propio.

Por lo tanto, para seguir la secuencia de cómo sería la creación de un conjunto, ahora para que se pueda definir, habría que añadirle el conjunto de piezas de las que estaría compuesto. Es por eso que ahora aparece la necesidad de usar la tabla que hay inmediatamente arriba. Como se ha explicado en la parte que se habla de la función del primer botón que aparece en esta pantalla. Y aquí se explica el motivo de porque incluir una tabla en la que se pueda realizar la misma búsqueda que en la pestaña anterior. Porque no es óptimo el tener que estar en este caso cambiando constantemente entre pestañas para ver la referencia exacta del producto que se quiere incluir como parte de ese conjunto. Teniendo los resultados de la búsqueda en la tabla, simplemente lo que habría que realizar es escribir la referencia de ese producto, junto con la cantidad del producto (que se refiere a las veces que ese producto aparece dentro del conjunto) y ya, por último, para guardarlo correctamente en la tabla de la base de datos habría que indicar la referencia del conjunto a la que pertenece.



UNIVERSITAS Miguel Hernández

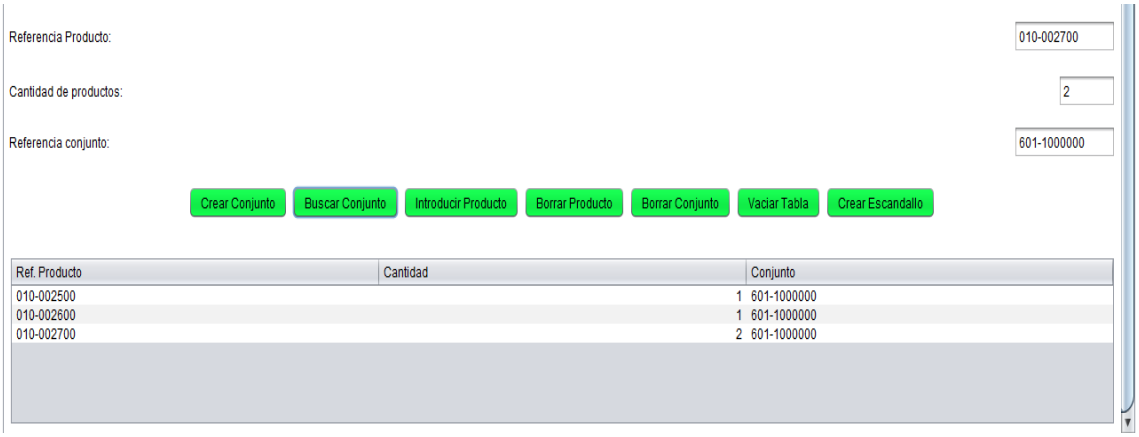


*Figura 15 – Introducir productos en un escandallo. Completar conjunto.*

*FUENTE: Elaboración propia.*

Como ya se conoce la forma de introducir productos al escandallo de un conjunto en concreto, aparece la necesidad de poder visualizar en el software de gestión, el escandallo de ese producto. Así que lo que sería necesario primero conocer es la referencia del conjunto, ya que se puede conocer la descripción pero no la referencia. Entonces habría que acudir a la tabla de arriba y habría que encontrarlo mediante los resultados de las búsquedas que se mostrará en la primera tabla de la pestaña. Una vez se conoce la referencia del conjunto, habría que ir al campo referencia del conjunto que hay en la parte final de la pestaña e introducir la referencia (en este caso debe ser una referencia completa, y no parcial como en las anteriores búsquedas que habíamos hecho hasta ahora, porque en estos casos solo quieres que haga una búsqueda de un conjunto en concreto, no de los máximos posibles.

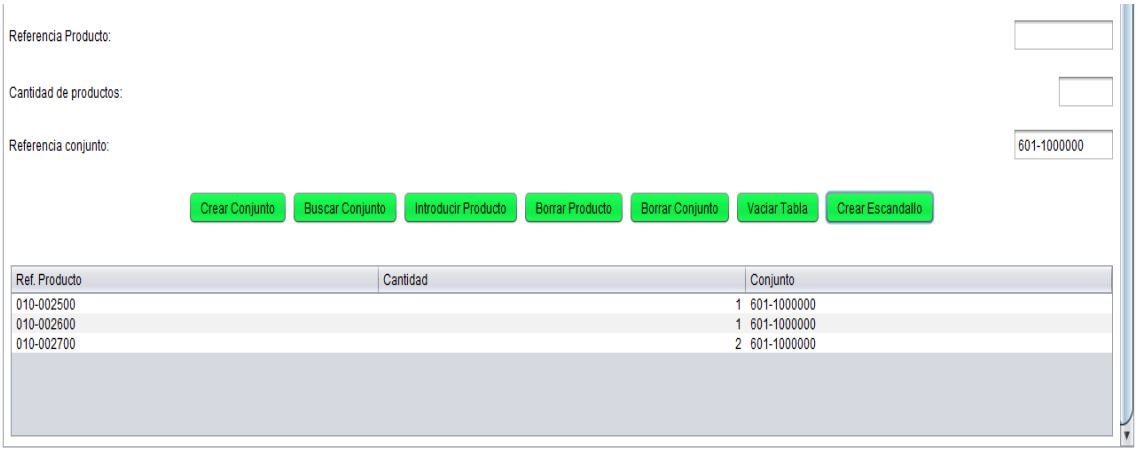


*Figura 16 – Búsqueda del escandallo de un conjunto.*

*FUENTE: Elaboración propia.*

Igual que una vez se ha comentado la acción anterior, otra acción que es necesaria es la contraria. Es decir, una vez se ha introducido un producto al escandallo de ese conjunto, se ha visto que ha sido un error al introducirlo, o que quizás, se ha actualizado el producto y se quiere sustituir por una pieza más moderna que tiene otra referencia. En esos casos, lo que se trataría de hacer es el de pulsar sobre la tabla de los conjuntos (donde aparece el escandallo del conjunto), pulsar sobre el producto que se quiere eliminar y justo después habría que pulsar sobre el botón de "BORRAR PRODUCTO".

Para el siguiente botón, el que aparece con título de "BORRAR CONJUNTO", ya como indica su nombre, permite al usuario borrar el conjunto que se haya seleccionador en la tabla después de haber realizado la búsqueda. Ya no que se puede borrar el conjunto escribiendo la referencia en un campo y luego pulsando en borrar. Ya que al pulsar directamente en la tabla hay menos posibilidades de equivocarse que escribiendo una referencia de 7 dígitos.

La siguiente acción que aparece se trata la de "CREAR ESCANDALLO", aquí no se trata del significado literal de esa acción, porque no se está creando ningún escandallo, ya que el escandallo ya está creado en el momento que se crea la referencia del conjunto y se añade el primer producto con sus cantidades. En ese momento esa información ya se encuentra en la base de datos, y si en algún momento se quiere visualizar en el software ese escandallo, se escribiría la referencia del conjunto y se pulsaría en buscar conjunto como ya se ha comentado. La diferencia radica en que con este botón te permite visualizar el escandallo de una forma diferente, y es porque te crea un archivo Excel con toda la información que aparece en la tabla del sistema pero en un Excel que se guardará en una carpeta en el que la compañía que adquiera este software haya indicado y se guardará con un el mismo nombre cada vez.



UNIVERSITAS Miguel Hernández

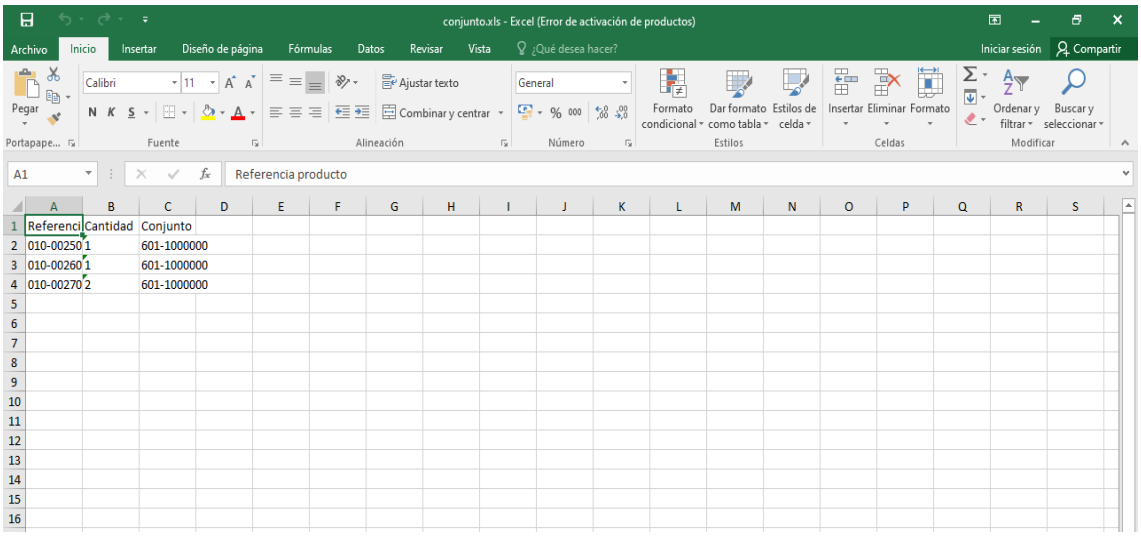


*Figura 17 – Crear un escandallo en Excel.*

*FUENTE: Elaboración propia.*

Y, por último, lo que aparece es lo de "VACIAR TABLA" que siempre aparece acompañando una tabla y es para vaciar su contenido antes de realizar una nueva búsqueda.

## 6.3 CLIE/PROV

En esta pestaña tiene por título 2 abreviaturas de las palabras CLIENTES y PROVEEDORES. Por lo tanto, en esta pestaña lo que se hará es de registrar e introducir en la base de datos la información tanto de los proveedores como de los clientes de la empresa. Esta acción siempre se hará cada vez que se empiece una nueva relación con ellos.

Aquí, las acciones que se pueden realizar en esta pestaña serían las siguientes:

- Crear.
- Editar.
- Borrar.
- Buscar.
- Vaciar Tabla.

En la primera acción, lo que se trataría sería de introducir en la base de datos al cliente o al proveedor según sea el caso. Por lo tanto, habría que rellenar los 5 primeros campos de



Biblioteca UNIVERSITAS Miguel Hernández

texto que hay en esa pestaña más el campo “booleano”. Aquí lo que se trata es de introducir:

- El nombre del cliente/proveedor.

- Su CIF, posiblemente este sea el campo más importante de todos para su identificación.

- La dirección. No se debe de confundir con la dirección de envío del material, ya que ese campo se rellenará para cada pedido (por que un cliente puede querer que cierto pedido en una ocasión vaya a cierto lugar, pero en otra ocasión quiere que vaya a otro). En cambio, en esta pestaña lo que se indica es la dirección fiscal de una empresa. La dirección fiscal es necesaria para según el país en donde resida ese domicilio fiscal, tenerlo en cuenta al a hora de realizar las declaraciones necesarias para la AGENCIA TRIBUTARIA. Por que como software de gestión industrial debe permitir el acceso o envío de la información a otros ámbitos como pueda ser el de gestión contable. Ya que según el tipo de negocio y según la localización, está sujeto a una normativa tributaria que se debe cumplir, por lo tanto, toda esta información debe ser almacenada y utilizada para que el correcto funcionamiento del otro ámbito.

- El campo de proveedor. Concretamente, no se trata de un campo de texto, sino que se trata de un booleano, es decir, como ya se ha comentado, se trata de indicar SI o NO, si SI es un proveedor, o en su defecto si se deja sin marcar, quiere decir que no es un proveedor y por lo tanto, se trata de un cliente.

- El campo de correo electrónico, para así disponer de una canal de contacto con en el cliente, para que si no se indica lo contrario en el momento que se realiza el pedido, sería donde mandar todos los pedidos o cualquier comunicación que se quiera mantener con el cliente.

- El teléfono. Aquí ocurre lo mismo que en el campo de arriba, es un campo que se utiliza para almacenar un dato para poder contactar con el cliente o el proveedor en caso que fuera necesario. Ya que, si ha habido algún problema con el envío de

un pedido, o por algún tema de impago, que haya un número de teléfono para el contacto directo.

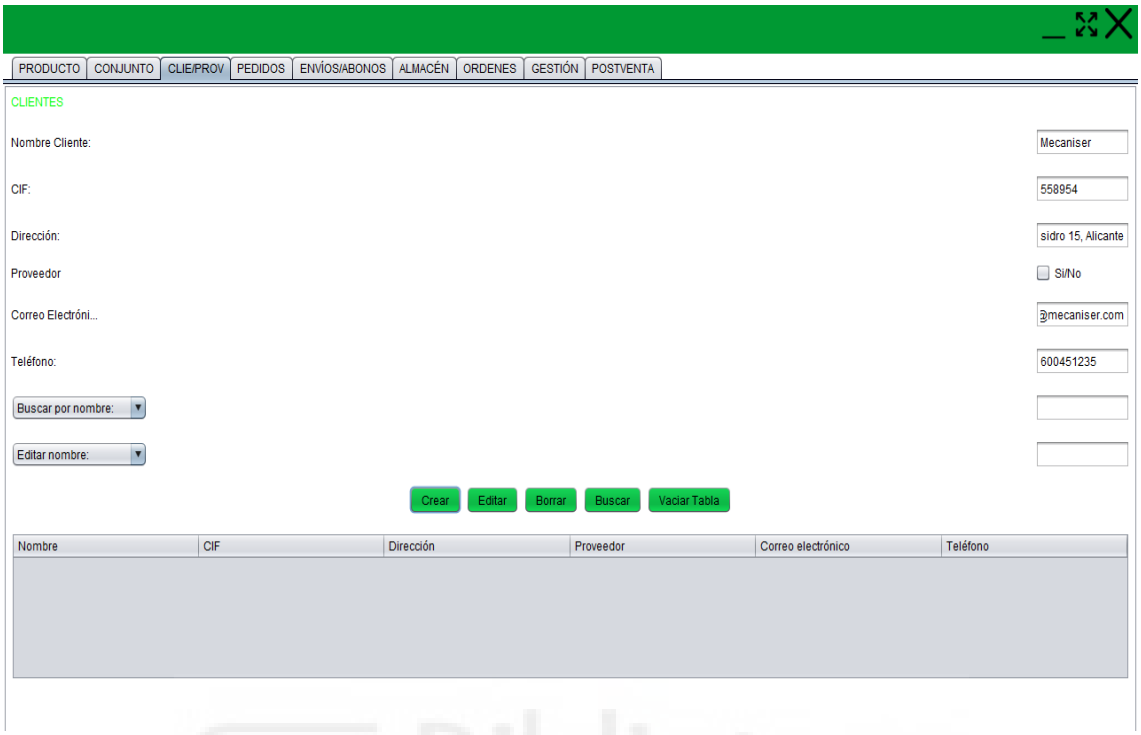


*Figura 18 – Crear un cliente.*

*FUENTE: Elaboración propia.*

La siguiente acción que comentar en esta pestaña se trata la de "EDITAR". Con esta acción te permita modificar cualquier registro en esta base de datos. Para modificar un registro habría que irse al último campo de texto que hay en esa pestaña, y ahí escribir el nuevo valor del campo que se quiera cambiar. Ahora sería importante de destacar que no se pueden modificar todos los campos de un registro. En este caso, como se puede ver en el desplegable, los únicos campos que te permite seleccionar se trata del "nombre" y del "cif". Son los que se ha considerado que son los valores más importantes que ante cualquier cambio o error se pueda subsanar de forma rápida sin necesidad de borrarlo y empezar de nuevo. Aunque sería necesario volver a destacar, que es un software orientado a las necesidades del cliente, por lo tanto, se podría adaptar y añadir la opción de modificar algún campo más que ahora mismo no aparece (por ejemplo, el caso del campo teléfono, ya que en la época actual es recurrente realizar cambios de número de teléfono).

Continuando con el procedimiento de como editar un registro, aparte de elegir que campo se quiere modificar y al lado escribir el nuevo. Lo que habría que hacer es seleccionar en la tabla de esa pestaña la fila o línea de texto de la base de datos que se quiere modificar

(por lo tanto, para editar un registro, primero se ha tenido que haber realizado una búsqueda para así tener datos en la tabla). Entonces con la fila seleccionada de la tabla, seleccionando el campo a modificar y escribiendo en el último campo de texto de la derecha el nuevo dato a guarda, ya simplemente habría que pulsar sobre el botón de "EDITAR".

Como ya se ha comentado que para poder editar un cliente o un proveedor habría que realizar primero una búsqueda para guardar datos en la tabla. Entonces necesitamos una acción que nos permita hacer eso, y esta acción es la "BUSCAR". Ya se ha explicado una de las necesidades para hacer búsquedas, pero no solo permite realizar simples búsquedas. Sino que se puede elegir que tipo de búsqueda se quiere hacer, o mejor dicho, que condiciones se incluyen para realizar esas búsquedas. En concreto se puede realizar 3 tipos de búsquedas, y estas son:

- Buscar por nombre -> Cuando no se conoce el nombre completo de un cliente, ya que suele ir acompañado de "S.L.", "Cooperativa", etc. Pues escribiendo las palabras que se recuerden o que se consideren clave para así poder filtrar según este campo.

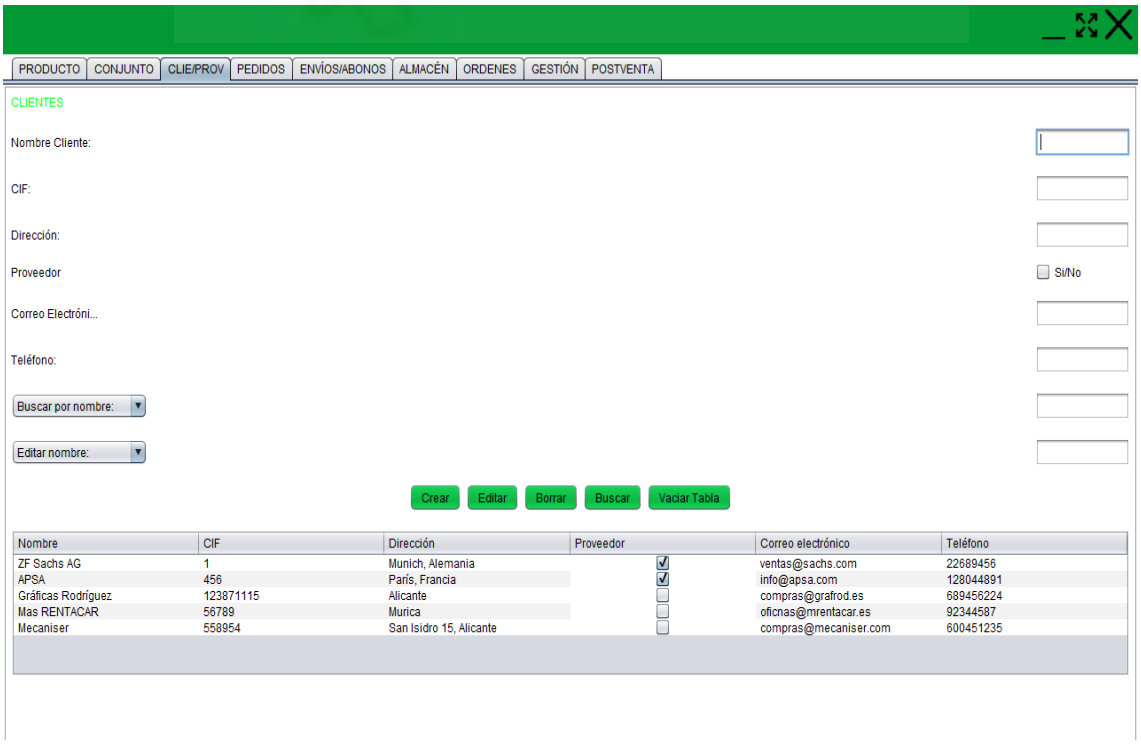


*Figura 19 – Buscar cliente sin condición.*

*FUENTE: Elaboración propia.*

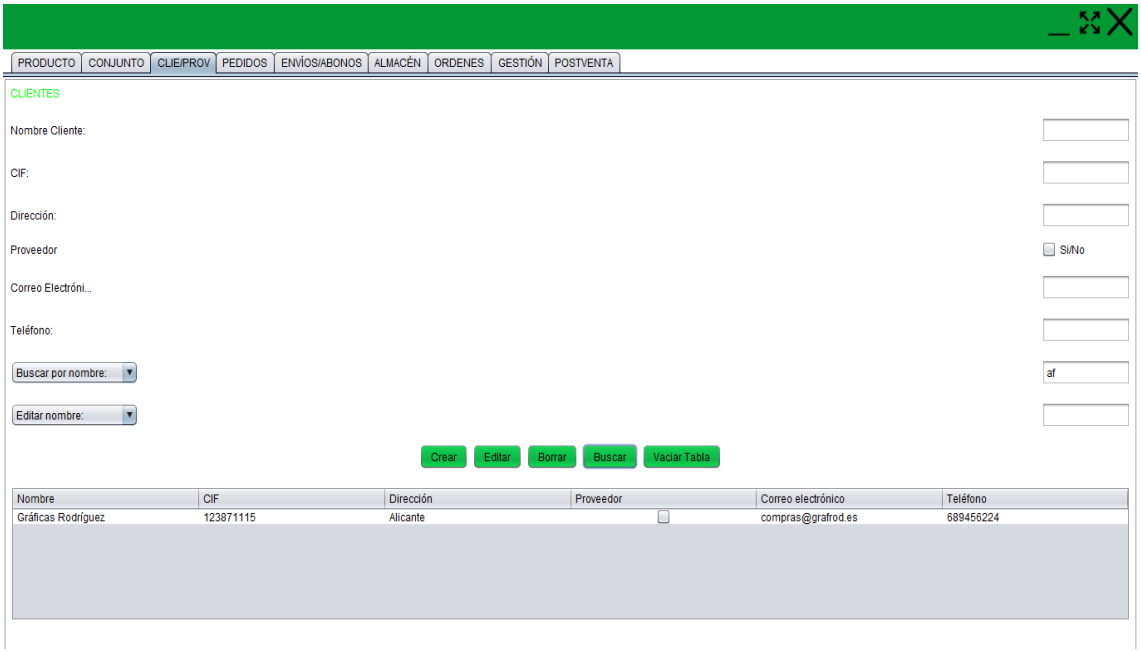


*Figura 20 – Buscar cliente según una condición.*

*FUENTE: Elaboración propia.*

- Buscar por CIF -> Ya que en todos los países no es obligatorio incluir el CIF de la empresa en las facturas (como es el caso de Estados Unidos), pero como en la Unión Europea, ese conjunto de códigos, si se quiere ir pudiendo filtrar, ya que como es un campo clave, se ha considerado que se puede buscar como condición.

- Buscar por Proveedor -> Como en esta tabla de la base de datos se almacenan tanto los proveedores como los clientes, y la diferencia entre ellos se trata de que valor tienen en un campo booleano de la misma; se otorga la posibilidad de poder buscar solo los proveedores.

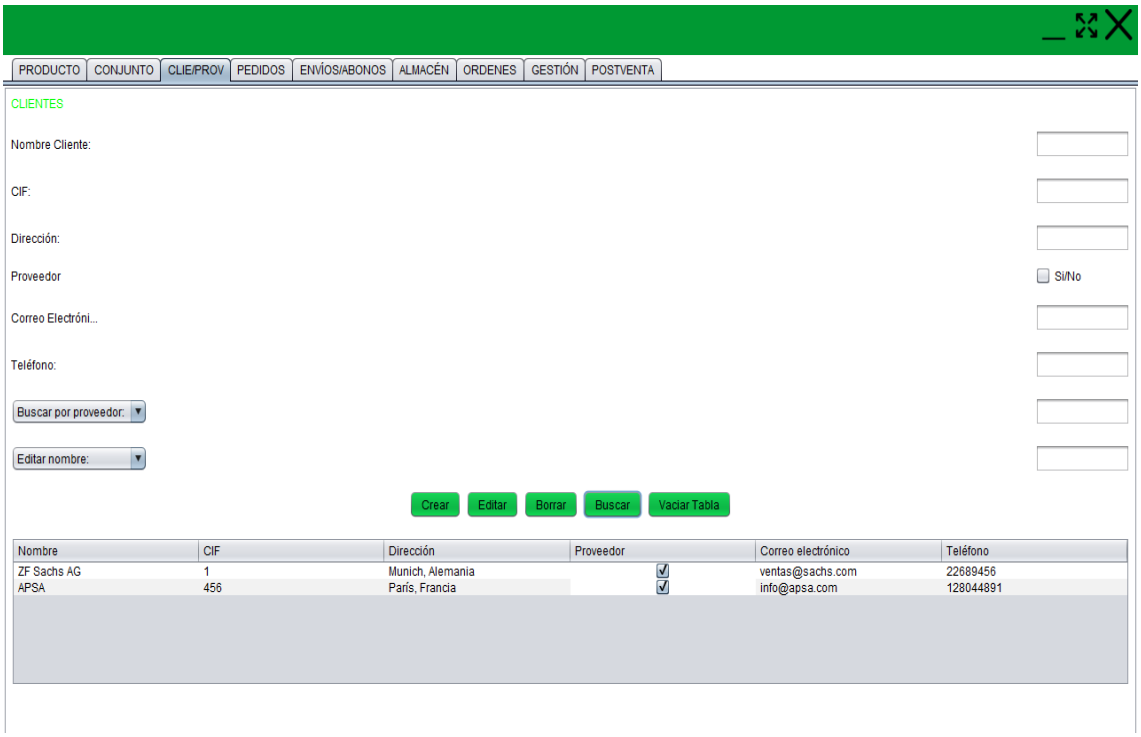


*Figura 21 – Búsqueda por proveedor.*

*FUENTE: Elaboración propia.*

Por lo tanto, para hacer la búsqueda, el proceso sería pulsar sobre el campo desplegable donde se puede leer "Buscar por nombre" y ahí dentro se podrá elegir entre las 3 opciones comentadas arriba, después escribir en el campo de testo de la derecha la palabra clave y por último pulsar en el botón/acción "BUSCAR".

También hay que destacar (esto ocurre en cualquier botón que permite realizar una búsqueda de una tabla), que siempre se puede hacer una búsqueda para mostrar todos los datos de esa tabla si en cualquiera de esas 3 condiciones de búsqueda, se pulsa sobre el botón de "BUSCAR" pero no se escribe nada en ese campo de texto.

La siguiente acción que aparece se trata la de "BORRAR", ya que puede ser que un cliente cierre, o se decide que a cierto proveedor no se le va a comprar nunca más su producto, o incluso, que se ha cometido algún error en la introducción de algún campo para ese cliente o proveedor y por lo tanto haya que borrarlo. Pues para estos casos, lo que se trata es de tener claro que fila de datos es la que se quiere borrar, con esto quiero decir, que anteriormente se habrá tenido que realizar una búsqueda para que hayan datos en la tabla del software. Una vez se ve que fila es la que se quiere borrar, se pulsará una vez sobre

ella, y en ese momento se pulsará el botón de la acción “BORRAR” y esa fila quedará eliminada de la tabla del software y también de la tabla de la base de datos de forma definitiva.

Para terminar, como en todas las pestañas que aparece alguna tabla, está la acción de “VACIAR TABLA” para eliminar todo el contenido que pueda haber en esta.

## 6.4 PEDIDOS

En esta nueva pestaña, como se puede ver está divida en 3 apartado como marcan esos separadores. En el primer apartado que el usuario se encontraría es el apartado que permite consultar información de los pedidos/ofertas. En el segundo sería para calcular el precio de esa oferta. Y, por último, nos encontramos un apartado donde ya por primera en el software, se llega al punto de poder crear una gráfica para así poder enfrentar datos y realizar un estudio de la situación.

Como se ha podido leer ya en el anterior párrafo, ya se han introducido dos términos, como es el caso de ofertas y el caso de pedidos. Que en un caso se han utilizado los otros términos, y en el otro caso solo se ha usado uno.

La diferencia entre estos dos términos es que el término de oferta le sirve al cliente para conocer cuál sería el precio de todo aquello que quiere adquirir. Esto quiere decir, que si un cliente quiere comprar 4 productos de la compañía y cada uno con sus respectivas cantidades ya sea para adquirir de más para tener stock o por lo que sea, entonces ería para saber cuánto le costaría ese todo eso. Esto sería una oferta, y en cambio el pedido sería la confirmación de la oferta. Esto significa que cuando el cliente una vez recibe la oferta, y es estudiada por los responsables de la administración para aprobar ese gasto, entonces, una vez ese gasto es aprobado, se le pasa orden de confirmación de esa oferta y ya en este software se marcaría que esa orden se ha aceptado y pasa a ser pedido.

Profundizando en el contenido de esta pestaña, como ya se ha dicho primero se empieza definiendo y cambiando las propiedades de las ofertas/pedidos, por lo tanto, las acciones que se pueden encontrar en este apartado serían:

- Crear.



UNIVERSITAS Miguel Hernández

- Buscar.
- Editar.
- Borrar
- Vaciar.

Empezando, siguiendo el proceso del funcionamiento de una empresa con un entorno industrial, ahora tocaría crear la oferta, para eso habría que empezar añadiendo un identificador a la oferta, esto serviría para que sea algo similar cuando se añade una referencia a un producto, en este caso como identificador se opta por el siguiente formado xx-xxxxx. En el que los 2 primeros dígitos hagan referencia a los últimos dígitos del año en el que se encuentre en el momento de crear la oferta (ahora mismo es 2022, y por lo tanto, la primera oferta de este año sería 22-00001, la siguiente oferta sería 22-00002 y así sucesivamente).

A continuación, habría que indicar quien es el cliente que solicita la oferta y la cuál aparecerá escrito su nombre para que se sepa después quien debe abonar la cantidad solicitada. Además de esto, el siguiente paso sería indicar la dirección de envío, que se la facilitará a la empresa de transporte, para cuando ya todo esté terminado que le llegue al cliente. Aquí ya se ve claro la diferencia entre la dirección de envío y el campo de dirección (fiscal) que aparece en la pestaña de Cliente/Proveedor.

La información que faltaría por añadir para definir por primera vez una oferta es si se trata de una garantía o no. Quizás aquí podría confundir, ya que, si en todo momento se está hablando de oferta, para presentar un precio y que después el cliente lo acepte, para pagarlo, parece contraproducente que ahora se hable de garantía. Pero como cada vez que se fabrica un producto puede ocurrir que haya algún tipo de problema en alguna de las piezas o pase cualquier problema de funcionamiento, sería necesario sustituir esas piezas en garantía, y por lo tanto, pasaría directamente a fase de espera para producir esa piezas en garantía, ya que se consideran directamente aceptadas.

Definidos estos campos, que serían 3 campos de texto y un campo booleano, ya se podría pulsar sobre “CREAR” y esa oferta se añadiría.

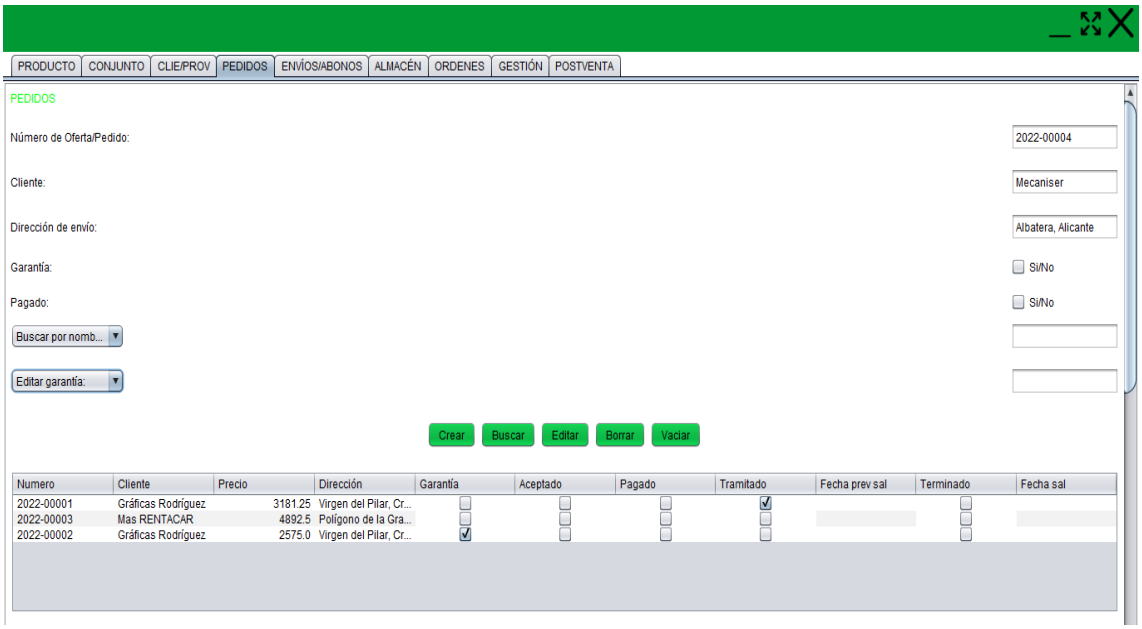


*Figura 22 – Generar un nuevo pedido.*

*FUENTE: Elaboración propia.*

La siguiente acción sería la de "BUSCAR", y una vez se pulsa sobre este botón, la información que aparecerá en la tabla será la siguiente:

- Número de oferta/pedido.
- Cliente.
- Precio.
- Dirección
- Garantía (booleano).
- Aceptado (booleano).
- Pagado (booleano).
- Tramitado (booleano).
- Fecha prevista salida.
- Terminado (booleano).
- Fecha salida.

Simplemente añadir como nota, que mucha información de la que aparece en esta tabla como se ha explicado hasta ahora no se añade arriba, se trata de una información que se añadirá o editará posteriormente. El motivo de esto (que ya se ha comentado anteriormente) es que en diferentes etapas del funcionamiento de una empresa industrial

se añadirá una información, pero quizás es necesario poder acceder desde otras partes, ya que aunque el software se muestre como un completo, una de las características es poder crear niveles de acceso (según las necesidades del cliente). Entonces muchos de esos campos que aparecen comentaos arriba se explicarán más adelante.

Como ocurre con otros casos, se pueden realizar búsquedas según 2 criterios o según dos condiciones. Estas dos condiciones son:

- Buscar por número. Lo de buscar por número se refiere a filtrar por el número de pedido. Ya que, si por ejemplo no se conoce el número de un pedido, pero sabe que se ha hecho a principios del año 2022, se podría buscar los 100 primeros de ese año, y la búsqueda sería tal que así "22-001", con esto ya busca todos aquellos que empiecen con esa condición, y el resto lo completa la búsqueda, con 100 posibilidades.

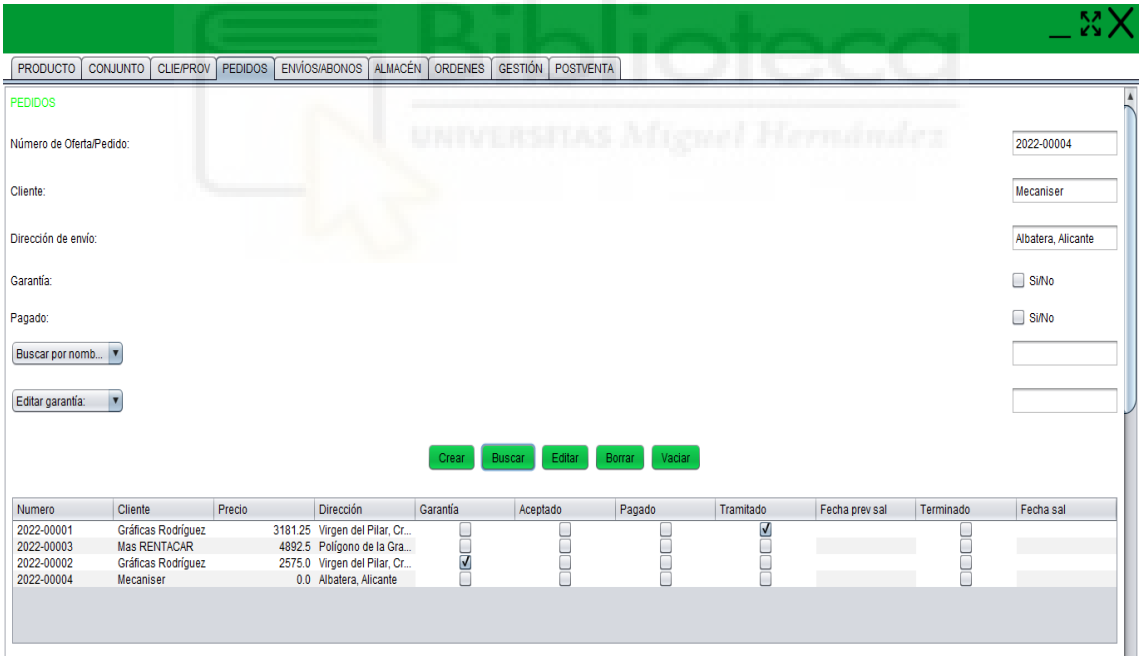


*Figura 23 – Búsqueda de pedidos, sin criterio. Se puede ver como el pedido recién creado aparece con precio nulo.*

*FUENTE: Elaboración propia.*

Búsqueda de pedidos (sin criterio) (Se puede ver como el pedido que se acaba de crear, no tiene precio todavía)

- Buscar por cliente. Para esta búsqueda lo que se intenta, es como indica su nombre filtrar para que aparezcan todas las ofertas realizadas por un cliente para así poder realizar estudios de cuantas ofertas se le ha pasado a un cliente, cuantas de estas ofertas se han convertido en pedidos, y las fecha de cuando estas se producen. Ya que toda esta información puede ser muy útil ya que por ejemplo, se puede ver si un cliente pide gran cantidad de ofertas, pero realmente estas no llegan a transformarse en pedidos, así conociendo esto se puede actuar y ver si es que los precios están disparados, o si quizás el cliente solo quiere tener información para después acudir a la competencia. También filtrar por cliente se puede ver, si se han llegado a frenar los pedidos para un cliente junto con las ofertas. Esta información sería útil para así conocer si es que el cliente se ha pasado a la competencia, o por si esa empresa ha entrado en crisis y así ver por que, para ver si es una crisis del país en general y así poder ver como actuar ante clientes que vivan esa misma situación.

La siguiente acción a comentar se trata la del botón “EDITAR”, para que así como ocurre siempre, ser capaz de editar algún valor erróneo introducido, pero en este caso no es solo por eso, ya que aparecen campos que no se habían indicado antes y habrá que indicarlos más adelante, pero esto se explicará en el párrafo posterior. Como ha ocurrido hasta ahora, no es posible editar todos los campos que hay en la base de datos. Los campos que se pueden modificar son:

- Garantía.
- Aceptado.
- Pagado.
- Tramitado.

Lo que se tiene que explicar que los campos que aparecen para editar, hay campos que en ningún momento se ha declarado cuando se crea un pedido nuevo por primera vez, salvo el de la garantía. Esto quiere decir que estos campos estarán vacíos. Por lo tanto por eso que se debe pulsar este botón de “EDITAR”, porque esa línea de información de la tabla de la base de datos ya está creada, entonces se debe editar un campo, aunque ya esté vacío.

El motivo de que se pueda modificar la columna de garantía, es porque haya habido una equivocación al crear la oferta, y se ha puesto que es una garantía o viceversa. O también puede darse el caso de que primero se lance la oferta, (siendo correcto el procedimiento), pero después de haber estudiado con detenimiento el caso en concreto, se puede ver que esa oferta venía para sustituir una pieza que ha fallado cuando la cubría el periodo de garantía. Se ve que lo correcto es pasarlo en garantía, pues se puede elegir editar esa garantía.

El siguiente campo que se puede editar es el de aceptado. Este campo, lo que indica es si se ha aceptado la oferta que se le ha enviado al cliente. La importancia de indicar si una oferta se ha convertido o no, es para que otros departamentos puedan empezar a gestionar ese pedido. Esto quiere decir que se empiece a ver cuando se debe preparar ese pedido (de acuerdo a una fecha que se especificará más adelante) o empezar a ver quien se va encargar de producir el contenido de ese pedido, etc.

Otro campo por editar es el de pagado. Simplemente, como se puede intuir, esta opción sirve para indicar si un pedido ha sido pagado o no. Esta opción sirve para la hora de hacer una búsqueda según los pedidos que ha sido pagados o no, para así después pudiendo ordenar según orden alfabético. Con esta información ya permite a departamentos de contabilidad, marketing que puedan hacer sus estudios para poder mejorar el funcionamiento de la empresa. Ya que, si una empresa siempre deja sus pedidos sin pagar, estos estudios servirán para así poder decidir si se deja de enviar ofertas o dejar de enfocarse en un cliente, hasta que este no haya pagado sus deudas con la empresa.

El último campo que se puede editar es el de tramitado. Aquí lo que se refiere es lo que se ha empezado a comentar 2 párrafos anteriores. Con este campo se indica si un pedido se ha empezado a tratar o a gestionar por el departamento encargado de ello, que en este caso en un entorno industrial se trata del departamento de producción. Desde esta opción de editar, no puedes marcarlo como tramitado, ya que esto se realizará en algunas pestañas posteriores, que por función estén más destinadas a un departamento como el de producción.



UNIVERSITAS Miguel Hernández

Dicho lo anterior, lo único que se puede hacer es deshacer que un pedido aparezca como tramitado, y esta opción es solo por si acaso ha ocurrido un error, que se pueda deshacer.

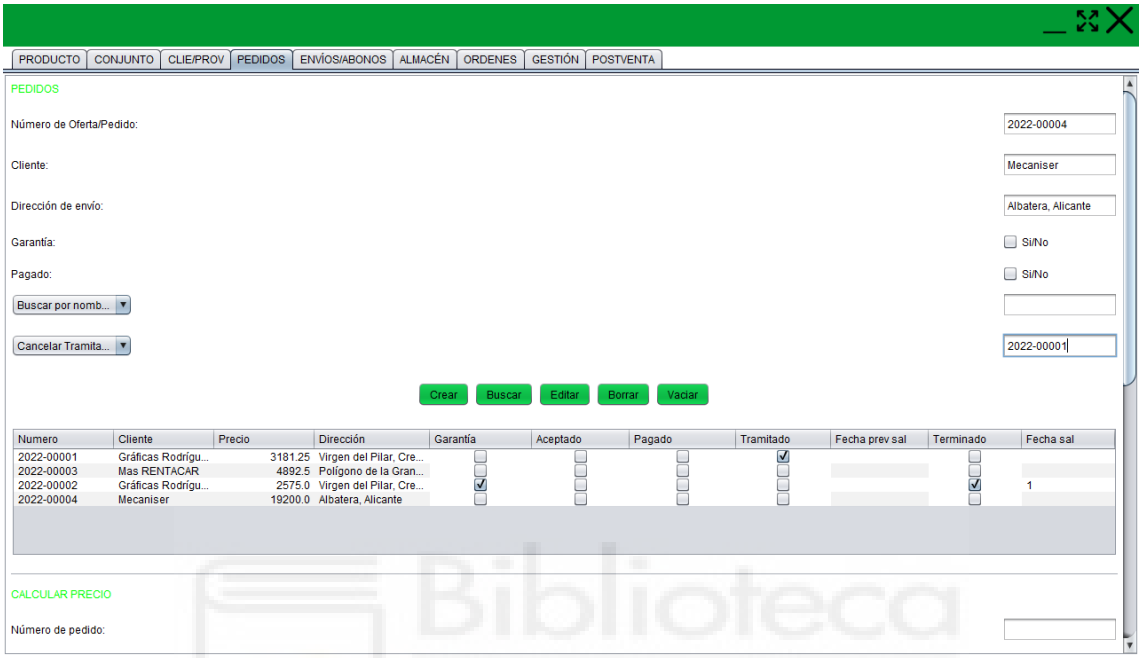


*Figura 24 – Editar estado "TRAMITADO" de un pedido (antes).*

*FUENTE: Elaboración propia*

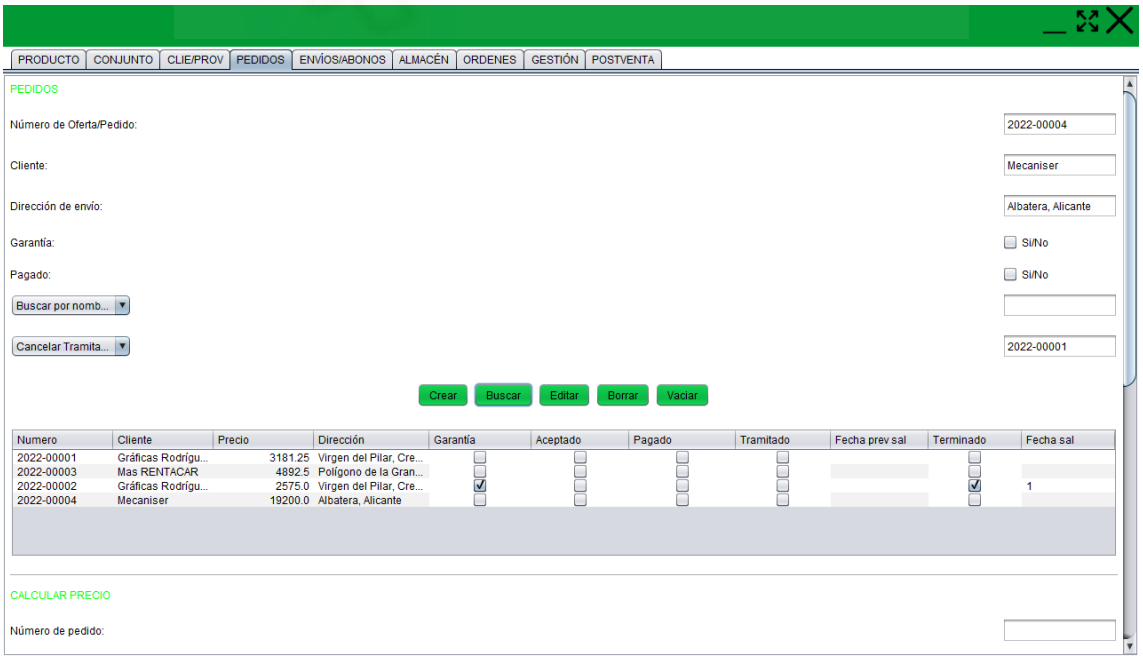


*Figura 25 – Editar estado "TRAMITADO" de un pedido (después).*

*FUENTE: Elaboración propia*

Aquí ya se empieza a notar la relación que existe entre varios departamentos, ya que se puede visualizar información en la que aquí en ningún momento se ha declarado, pero como va relacionada con los pedidos debe aparecer, aunque sea como consulta, o en su defecto ofrecer la posibilidad de deshacer errores.

La siguiente acción se trata la de "BORRAR", en esta es evidente que se refiere a borrar un pedido, concretamente el pedido que se haya seleccionado en la tabla que hay inmediatamente debajo. Por último, queda la acción de "VACIAR" que como se usa siempre, es para eliminar la información que tenga una tabla para poder añadir una nueva.

En la misma pestaña, la siguiente etapa es la CALCULAR PRECIO. Ya que realmente hasta aquí se han creado ofertas y después pedidos y se han editado, pero en ningún momento se ha hablado del precio. En este apartado es donde se añadirá el precio a una oferta. Pero el precio no se añade por que sí. El precio será un cálculo de todos los productos que se añadan, en función del precio que tenga y de las cantidades que se añadan de cada producto.

Las acciones que aparecen en este apartado son:

- Añadir Producto.
- Buscar Pedido.
- Eliminar Producto.
- Generar Pedido.
- Vaciar.

Para la acción de "AÑADIR PRODUCTO" lo que permite hacer es añadir un producto al pedido especificado. Para eso lo que hay que hacer es primero indicar el número de pedido en el que se guardará un producto. Entonces a continuación, se debe añadir el producto en concreto que se añadirá y por lo tanto en el siguiente campo de texto se debe escribir la referencia en concreto del producto y justo debajo la cantidad de ese producto. El siguiente campo a rellenar sería el de descuento. Este campo está enfocado a decisiones comerciales. Es decir, cuando por cortesía comercial se quiere realizar cierto descuento a la oferta que se le realiza al cliente por el motivo que sea. Y no solo eso, el descuento no



UNIVERSITAS Miguel Hernández

está aplicado a toda la oferta, sino que se aplica al producto en concreto que se está añadiendo a la oferta. Para así poder llevar diferentes estrategias de marketing.

El último campo que habría que indicar es el de si es una garantía. Que, aunque se haya indicado que es una garantía en el momento que se crea la oferta, es necesario indicarlo aquí también para así poder usarlo en una acción que se explicará más adelante.

La siguiente acción que aparece es la de "BUSCAR PEDIDO". Esta acción no es la misma que aparece en el apartado anterior. Ya que en la anterior buscabas entre varios pedidos y podías filtrar por año, número y demás. En este caso, se debe conocer el pedido en concreto, porque así lo que muestra en la tabla es las líneas de producto que tiene ese pedido. Si no se buscara el pedido en concreto, lo que ocurriría es que aparecerían líneas de productos de diferentes pedidos.

Los campos que aparecen en la tabla que muestra los productos que hay en un pedido en concreto es:

- Pedido.
- Referencia.
- Cantidad.
- Precio x cantidad.
- Descuento.
- Precio final.
- Garantía.

Cuando se realiza una búsqueda, lo que se puede ver es el número del pedido al que se refiere el producto, la referencia del producto y la cantidad. Todo esto ha sido especificado en el momento de añadir esa línea de producto. El siguiente campo que aparece, no se ha especificado nada. En este campo lo que hace es multiplicar la cantidad de veces que se ha añadido ese producto, por el precio unitario de ese producto. Para eso el software realiza una búsqueda interna y "coge" ese precio unitario y lo multiplica por la cantidad y lo guarda en esa casilla. El siguiente campo muestra el descuento que se le aplica. Entonces te mostrará en el siguiente campo el precio final, que simplemente es, que a ese precio por la cantidad de producto, le quita el porcentaje del descuento aplicado,



Biblioteca UNIVERSITAS Miguel Hernández

que sería el precio que se usará para mostrarlo como parte del pedido. Y, por último, el campo que aparece es el de garantía, que es un booleano que dice si es o no.

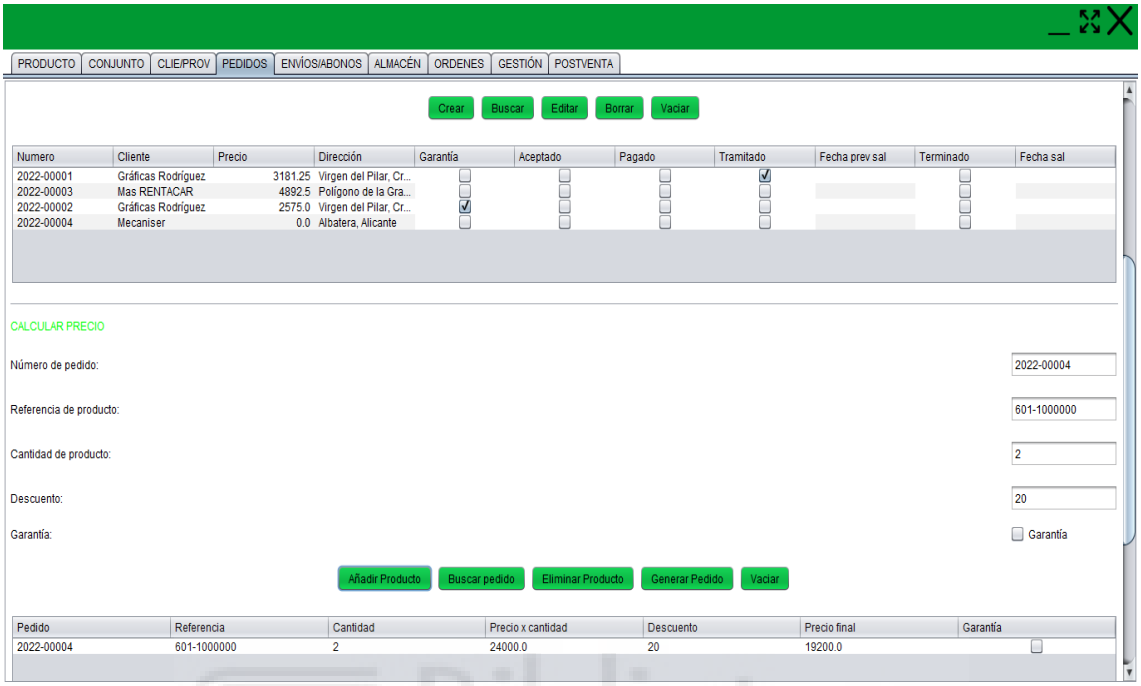


*Figura 26 – Añadir producto al pedido.*

*FUENTE: Elaboración propia.*

La siguiente acción es la "ELIMINAR PRODUCTO", que se trata de eliminar una línea de producto de un pedido. Por supuesto esta acción está pensada para los casos, en el que se produce un error y se introduce un producto equivocado. Pero además de esto, también se debe usar en los casos que se cree una oferta, y posteriormente el cliente no ve necesario un producto por su precio o por el motivo que sea, o quizás quiere reducir la cantidad de ese producto.

Ya se ha comentado dos párrafos arriba, que cuando se busca la oferta o el pedido en concreto, muestra el contenido en una tabla. Pues en la acción de "GENERAR PEDIDO" sirve para mostrar el contenido de la oferta/pedido en una tabla de Excel. Así este contenido podrá ser utilizado para incorporarlo en plantillas que te puedan generar un documento en "papel" para así poder hacer copias y mandarlas, y posteriormente poder tener copias de las facturas, para poder mandarlas y archivarlas según sea la necesidad.
La siguiente acción es la que se ha comentado que aparecen cuando hay una tabla, y es la de poder eliminar toda la información que aparece sobre ella.

Por último, el apartado que aparece es para poder generar KPI. Es decir, en toda esta pestaña se ha generado, almacenado, editado y gestionado gran cantidad de información. Entonces en este apartado, lo que te permite es poder clasificar esta información, o mejor dicho, cuantificarla para que pueda ser estudiada que permita mejorar el enfoque y las decisiones de la empresa.

Además de cuantificar, lo que permite es mostrar estos datos de una forma visual, no solo mostrando el número, sino generando una gráfica, para así conseguir el objetivo perseguido de ser lo más visual posible.

Aquí la gráfica que se puede crear es la de mostrar las garantías. Es decir, se debe escribir el en el primer campo de texto que aparece, la referencia del producto que se quiera conocer cuántas garantías se han generado y lo muestra en una gráfica circular en el que aparecen todas las garantías generadas.

También lo que se puede es mostrar un gráfico para ver de todas las ofertas que han sido generadas se han convertido a pedido, y además pueda crear una relación entre estos datos calculados para así ver un porcentaje que te permita entender si hay una gran cantidad de ofertas que no llegan a pedidos. Y ya gracias a eso ver si puede ser el precio o algún otro factor.

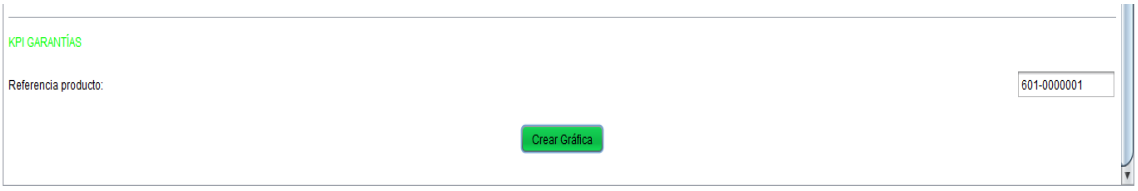


*Figura 27 – Introducir conjunto para crear KPI de garantías.*

*FUENTE: Elaboración propia.*

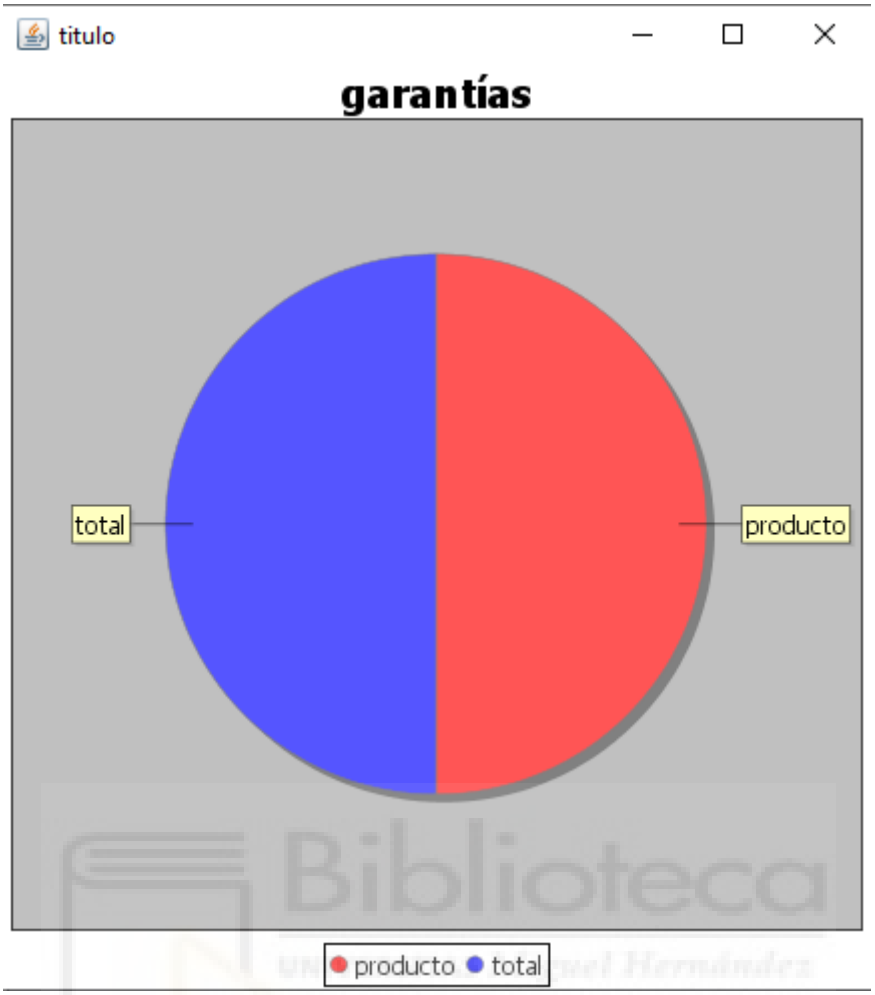


*Figura 28 – Resultado de crear la gráfica.*

*FUENTE: Elaboración propia*

## 6.5 ENVIOS/DEV. (ver de llevar a última pestaña)

En esta pestaña, se tratará toda la logística de los envíos, de ver que pedidos se encuentran terminados para así prepararlos dentro de un contenedor o una caja según sean sus dimensiones, embalarlos y entregarlos a la compañía transportista para llevarlos.

Las acciones que aparecen en este apartado para los envíos serían:

- Buscar pedidos terminados.
- Enviado/ Deshacer
- Vaciar.

La primera acción como aparece en el nombre es la que te permite realizar una búsqueda de los pedidos que ya aparece que se hayan terminado de producir, es decir, que estén listo para preparar para el transporte y en la búsqueda se añade la condición de que muestre los pedidos terminado, pero que no se hayan enviado ya.

Pero no se trata de ver todos los pedidos que ya estén listos y empezar a mandarlos para enviarlos ya. Ya que todo depende de cuando se haya establecido con el cliente de una fecha prevista para el envío. Porque, lo importante es no acumular stock como se explica en el capítulo de LEAN MANUFACURING, pero pueden aparecer ciertas épocas donde por complicaciones del entorno socioeconómico o ciertos problemas coyunturales en la propia empresa (problemas con bajas de trabajadores, escasez de materias primas, etc.) que sea necesario realizar cierto aprovisionamiento para poder hacer frente a eventuales problemas. Entonces se empezará a producir o fabricar para que esté listo antes de la fecha prevista de salida.

Entonces aquí junto con el departamento encargado de preparar el embalaje para dar salida al material se debe ver cuando tiempo se necesita para preparar ciertos tipos de contenedores. Entonces se trataría de pulsar sobre esta acción, y que en la tabla se muestren todos los pedidos ya finalizados y entonces ver cuando está previsto que salga cada uno para así después poder ver que lleva cada uno de esos pedidos (la consulta de que lleva cada pedido se realizará en otras pestañas ya explicadas) y según lo que lleve se ve el tiempo que se necesita ver ya cuales son los pedidos a preparar. Es decir, simplemente sería establecer un orden de prioridades, para así ir preparando el envío, con el único fin de que todos los pedidos salgan en la fecha prevista, ni antes, ni después.



UNIVERSITAS Miguel Hernández

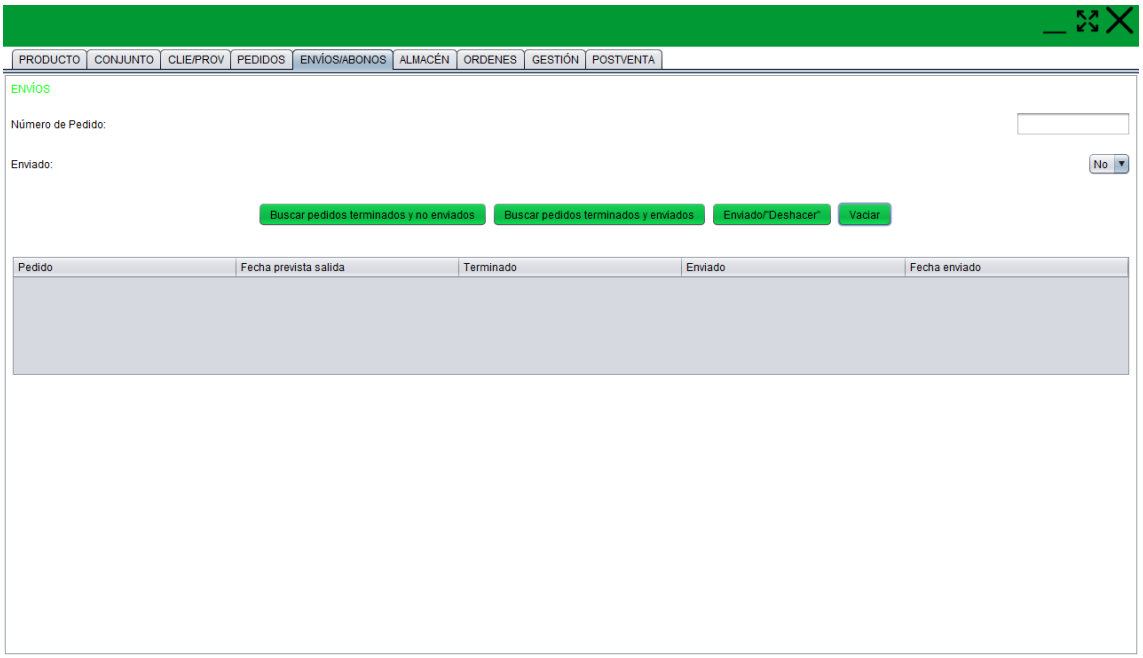


*Figura 29 – Apariencia de la pestaña.*

*FUENTE: Elaboración propia.*

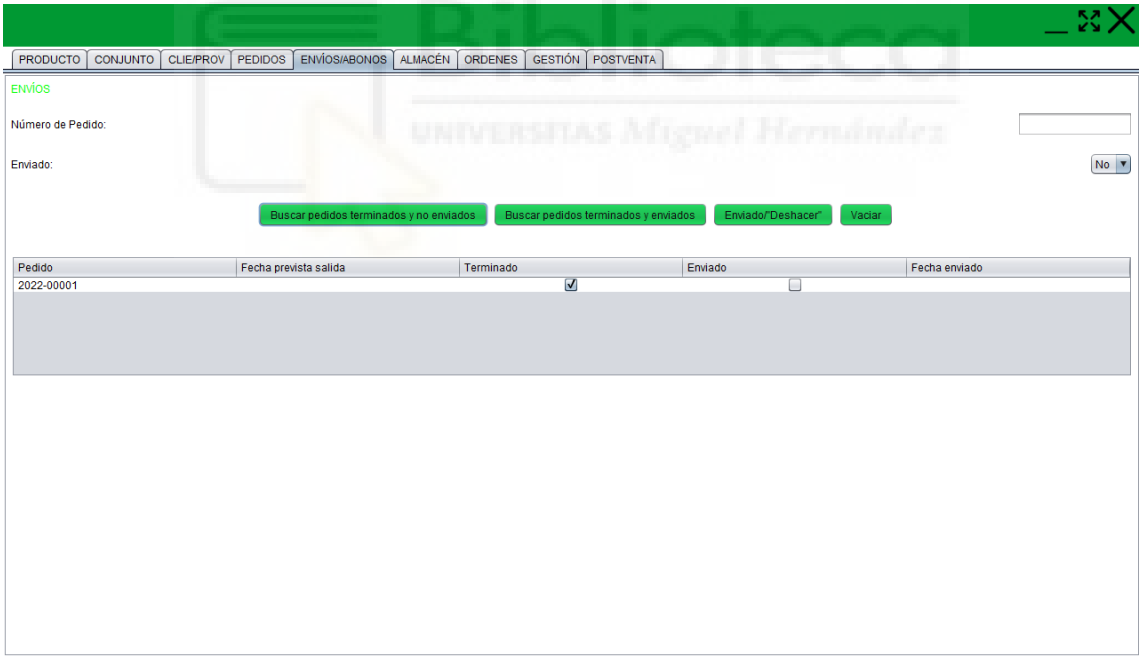


*Figura 30 – Cuando se pulsa la acción "BUSCAR PEDIDOS TERMINADOS Y NO ENVIADOS"*

*. FUENTE: Elaboración propia.*

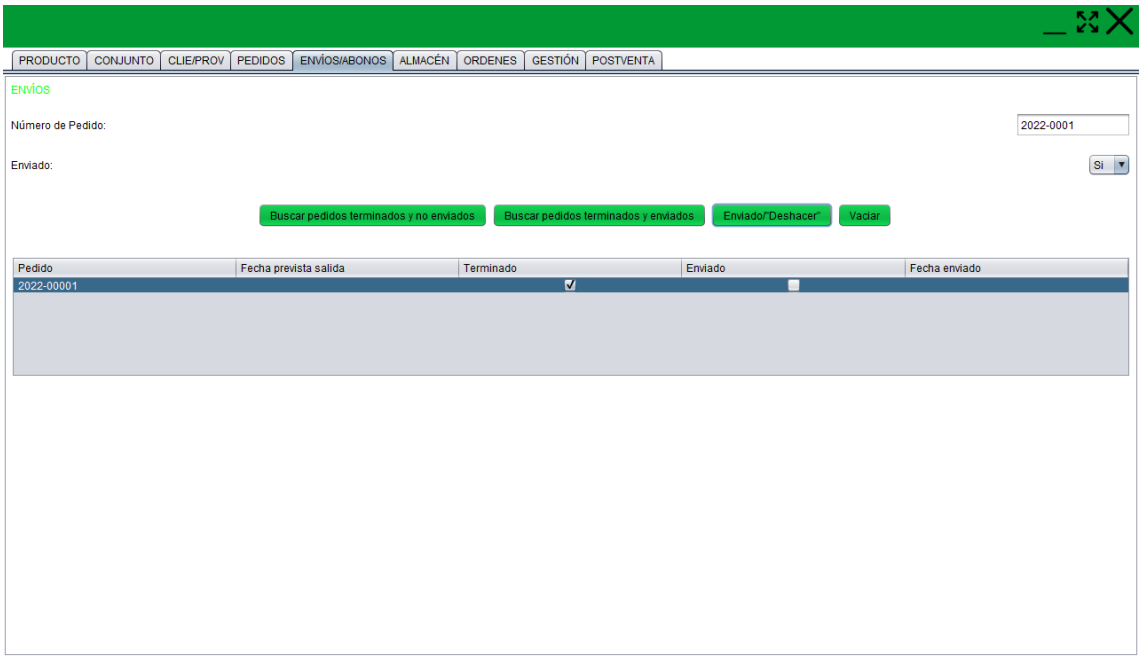


*Figura 31 – Pasos para declarar un pedido enviado.*

*FUENTE: Elaboración propia.*

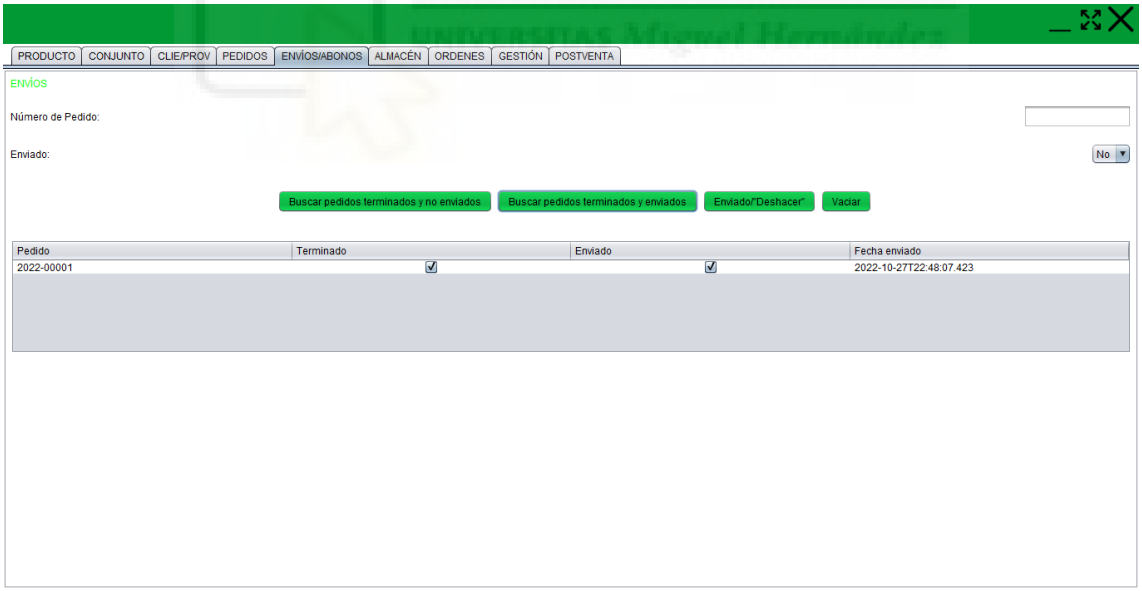


*Figura 32 – Resultado de haber enviado un pedido.*

*FUENTE: Elaboración propia.*

Como aquí se está teniendo en cuenta un campo de información como el de fecha de salida prevista, se añade la acción para poder cambiar este campo, ya que puede ocurrir que el cliente solicite a la empresa un retraso en el envío de la maquinaría, ya que por ejemplo sus instalaciones no están preparadas para recibirla, o todo lo contrario, sino que les corra prisa y que necesitan la máquina a la mayor brevedad posible.

La siguiente acción es, una vez ya se ha terminado de preparar un pedido y queda pendiente de que el transportista se pase a recogerlo de forma inmediata, es cuando, en el campo de texto que hay justo arriba, se escribe el número de ese pedido y se pulsa sobre la acción "ENVIADO/DESHACER", con esto, si en el desplegable aparece un SÍ (de que se ha enviado), se modificará ese campo booleano en la base de datos. Por supuesto, al hacer este cambio, ya para futuras búsquedas, ese pedido que se ha enviado dejará de aparecer cuando se pulse el botón/acción explicada anteriormente.

Otra opción que se da para esta acción, es poder deshacer que un pedido se haya enviado. Es por eso que está el desplegable en el que se puede elegir la opción NO (cuando antes para marcar como enviado se ha elegido SI). Para eso lo que hay que hacer es escribir el número del pedido, seleccionar NO y pulsar en el botón de esa acción.

Una diferencia que ocurre con otras pestañas, es que aquí, para editar alguna fila de una tabla, simplemente bastaba con pulsar sobre la fila a editar. En cambio, en este caso hay que escribir el número del pedido, ya que al pulsar es más fácil poder equivocarse, de la otra forma, al escribir el número del pedido, es más fácil poder darse cuenta de si se ha escrito correctamente o no.

En este apartado aparece una de las ventajas en las que solo se vacía la tabla cuando se pulsa sobre la acción "VACIAR", porque si se ha cometido algún error pulsando sobre alguna otra acción o editando algún campo, todavía se puede ver la fila con los datos antiguos y poder deshacer el error.

Ahora se llega al siguiente apartado de esta pestaña, en él se tratará el tema de los abonos. Esto es para cuando se ha realizado un pedido, y se ha visto que hay algún error en algunas de las referencias, o que simplemente el cliente se ha confundido pidiéndolo y quiere

devolverlo. Simplemente, lo que habría que hacer es devolverle el dinero de los productos que quiera devolver con sus respectivas cantidades.

Las acciones que primero aparecen en esta pestaña son:

- Crear.
- Buscar.
- Eliminar.
- Vaciar.

Aquí, para poder crear la devolución, los campos que hay que rellenar es el que número se le otorgará a la devolución, a continuación a que número de pedido va asociada y aunque ya aparezca en el pedido, también es interesante volver a indicar quien es el cliente al que se le hace la devolución. Por último aparece un campo que es el de nota, para poder indicar una frase corta del motivo de la devolución.

A continuación, para poder realizar la búsqueda, permite aplicar tres filtros de búsqueda, ya sea buscar por el número de devolución, por el número de pedido e incluso por cliente, según que información interese obtener.

Con la acción de "ELIMINAR", pulsando sobre esa devolución permite borrarla de la base de datos. Y por último, la acción que permite borrar todo el contenido que aparece en la tabla.

Como ocurre con las pestaña de pedidos, a parte de poder ver cuantas devoluciones hay, se puede ver el contenido de estas devoluciones. Por que aunque la devolución haga referencia a un pedido, no quiere decir que se vaya a devolver todo el pedido. Es decir, de 10 productos que puede tener un pedido, solo se quiere devolver la cantidad referida a 4 de ellos, por ejemplo.

Los campos que aparecen en la tabla que muestra los productos que se van a devolver en una devolución en concreto son:

- Devolución.
- Referencia.

- Cantidad.
- Precio x cantidad.

Las acciones que se podrían realizar para esta tabla sería la de añadirle unos productos a la devolución, con su respectiva cantidad. Según el precio unitario de ese producto se multiplicará por la cantidad para poder incluir el precio total a devolver. Lo siguiente que se podrá hacer es buscar según devolución, eliminar un producto de la devolución. Y finalmente generar una tabla en Excel del contenido de una devolución para así poder pasarlo a una plantilla y generar un pdf de la devolución.

## 6.6 ALMACÉN

Una vez ya están las órdenes de trabajo creadas, y es necesario preparar este material para sacarlo en el momento que toca. Para eso toca ver que órdenes están pendientes de sacar al almacén, y además de esto, debe de ir acompañado con la información de cuando está previsto de sacar al almacén, para así evitar sacar una orden antes que otra que tenga más prioridad.

Debe existir una tabla en la que aparezcan todos los productos que ocupen un lugar físico en el almacén. Ya que todo lo que se quede guardado debe quedar registrado, y como cada tipo de producto se debe guardar en cierta zona en concreta. Esto quiere decir que tenemos dos tablas que registran los productos, una que como ya se ha comentado se llama productos y la otra almacén, pero en una están todos los productos “activos” y en la otra, como ya se ha dicho, productos almacenados (ocupan espacio físico), entonces puede ocurrir, que por algún motivo, se haya borrado un producto de la tabla “almacén” pero no de la de “productos, pero en cambio, al revés no es posible, ya que no puede haber un producto que ocupe un espacio, cuando no se puede comercializar con ese mismo producto.

Entonces las acciones que se encuentran en esta primera parte de la pestaña son:

- Buscar órdenes.
- Buscar pendientes de sacar.



UNIVERSITAS Miguel Hernández

- “Sacado/Deshacer”.
- Vaciar.

Para la acción de “BUSCAR ÓRDENES” permite filtrar por el número de orden, pero la acción más importante en esta parte de la pestaña es la de “BUSCAR PENDIENTES DE SACAR”, aquí lo que se trata es que en la tabla se muestre todas las órdenes que han sido creadas, pero que no se han empezado a tratar todavía, y así, en la tabla, se puede ordenar en orden alfabético, parar que así muestre primero las ordenes que tienen que sacarse antes al taller. No confundir con filtrar con las ordenes que se tienen que mandar antes, porque quizás una orden lleva mucho tiempo de fabricación y eso por eso que se debe de empezar a fabricar con anterioridad.

Entonces, una vez ejecutada esta acción, que realiza una búsqueda en la base de datos y devuelve cada una de las filas que cumpla la condición comentada, se debe estudiar cual es la orden que se debe empezar a tramitar de forma inmediata. Una vez se conoce cual será, lo que se debe de hacer es pulsarlo esa fila de información en la tabla, y justo pulsar sobre la acción “SACADO/DESHACER”, en ese momento indicas que se va a sacar todo el material relacionado con esa orden.

Puede ocurrir, que por error se haya pulsado una orden que no es, entonces, se le ha dicho que se ha sacado del almacén al taller para que se prepare, cuando realmente, no es el caso. Se puede deshacer. Para eso, se debe primero buscar por número de orden (que es la primera acción explicada) pulsar sobre la orden en la que anteriormente el usuario se haya equivocado. Y a continuación en el desplegable con nombre de “sacar del almacén”, habría que indicar no, y justo después volver a pulsar sobre la acción “SACADO/DESHACER”, y lo que hace es que para la columna de sacado de la base de datos, le indica que NO se ha sacado (esta información se pasará un dato booleano).

Pasando a la segunda parte de esta pestaña del almacén, es la que permitirá poder decir que productos hay almacenados en este, que cantidad de estos, cuantos se reponen y cuantos se sacan. Para poder detallar que hay en este apartado, se tienen que comentar las acciones. Estas son:

- Crear Prod Alm.



UNIVERSITAS Miguel Hernández

- Buscar por Referencia.
- Buscar por nombre.
- Introducir Producto(s).
- Retirar Producto(s).
- Eliminar del Almacén.
- Stock seguridad.
- Vaciar.

La primera acción que aparece es la de “CREAR PROD ALM”, aquí lo que se trata es que cuando se dé de alta un producto para la venta, se tiene que declarar en esta pestaña, esto es para poder darlo de entrada del almacén, para que se pueda sacar al taller y se pueda reponer. Para poder buscar que productos hay, para poder realizar alguna de las funciones que viene después es la de “BUSCAR POR REFERENCIA”. Al poder filtrar por referencia permite ver información como el stock que hay en el almacén y cual es el stock de seguridad. Lo mismo ocurre si se pulsa sobre la acción de “BUSCAR POR NOMBRE” con la diferencia que en este caso te permite filtrar por el nombre, escribiendo en el campo de nombre de producto.

Además de esta información comentada en el párrafo inmediatamente anterior, en la tabla también aparece la última vez que se sacó del almacén como mínimo un producto y cuando fue la última vez que en el almacén se guardó un producto en este. Esta información sirve para ver cuánto se utiliza un producto para así poder sacar conclusiones:

- Dependiendo la última vez que se sacara un producto del almacén, que se pueda ver si ese producto es muy usado o no. Porque viendo si se usa mucho, se puede ver si representa un conjunto como tal o si se usa como “uno” solo, para así poder ver si sería interesante dar de baja ese tipo de producto o al menos estudiar porque no se usa. Y para el caso contrario, exactamente lo mismo, porque siempre que se estudia fechas, siempre es reciente que se ha sacado ese producto. Porque eso quiere decir, que puede que se estropee mucho y haya que reemplazarlo a menudo, o que sea un kit de un robot que sea muy bien acogido por los clientes, y te interese potenciarlo.

- Dependiendo la última vez que entro ese material en el almacén. Esta información puede servir para ver si es necesario acudir muy pronto al proveedor para pedirle material, porque esto puede llevar a ver, que unido con información recopilada en otras pestañas, que se tenga que acudir de forma asidua al proveedor, por problemas de este en tiempos de envío, o escasez de confianza debido a fallos en el envío del material.

La acción que aparece a continuación es la "INTRODUCIR PRODUCTO(s)". Esta acción se usa en los casos en los que una vez se ha pedido un material al proveedor, este se recibe y se prepara para almacenarlo. Entonces justo antes en el momento de almacenarlo, se debe ver cuanto material se ha recibido y entonces se debe indicar la cantidad y además indicar la referencia del producto, y ya en ese momento se podría pulsar sobre esta acción, y lo que se hará es que en la cantidad que se va a guardar, se le sumará a la cantidad que aparece que queda en la base de datos, así en el momento que se pulse sobre buscar un producto, permita mostrar la cantidad actualizada. Y siempre que se pulse esta acción, guardará la fecha exacta del momento en el que se ha pulsado.

En cambio, para la acción de "SACAR PRODUCTO(s)" ocurre lo contrario. Este es el caso, en el que el taller de producción necesite recibir material para su transformación, o directamente o un producto ya terminado, pero que se almacenó para tener stock y se quiera enviar al cliente, porque se ha vendido a posteriori. Lo que se debe hacer es indicar la cantidad de producto que se saca, escribir la referencia de este y ya pulsar sobre esta acción. En ese instante lo que realizará es que, a la cantidad que aparezca que esté almacenada en ese instante en el almacén, le restará la cantidad introducida.

En esta acción ocurre lo mismo que en la anterior. Cada vez que se pulse este botón, guardará la fecha en el campo de fecha sacar producto.

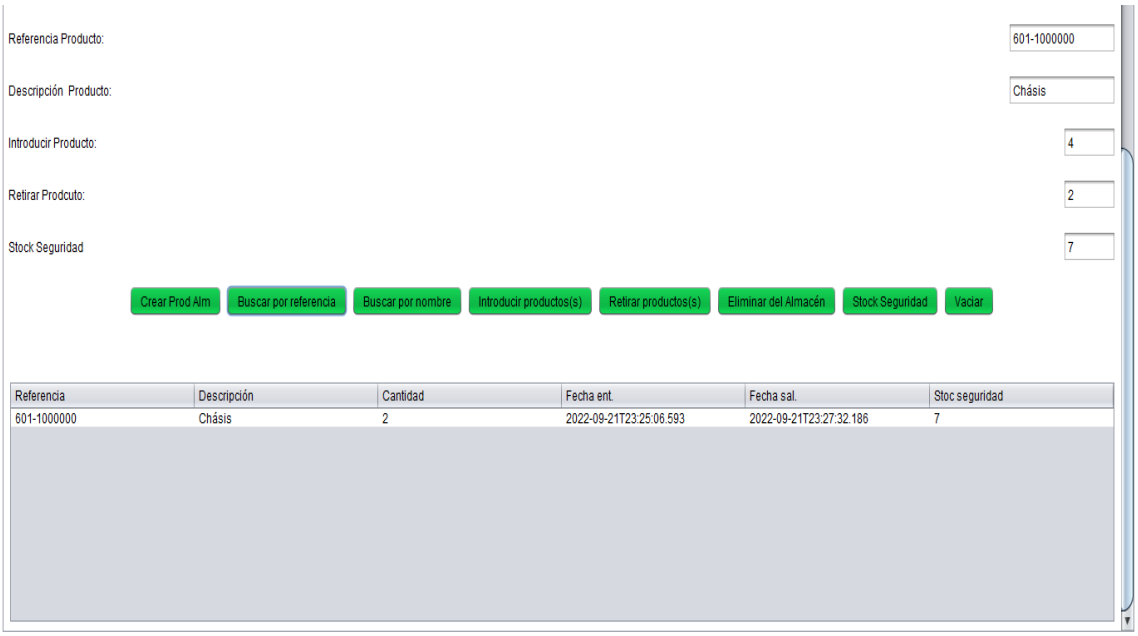


*Figura 33 – Información acerca de l*

*os movimientos de los productos en el almacén.*

*FUENTE: Elaboración propia.*

La siguiente acción que aparece es la de "ELIMINAR DEL ALMACÉN", esto es para cuando ya se decide borrar un producto de la tabla de productos, no tiene ningún sentido que siga apareciendo como que tiene un hueco en el almacén, por lo tanto, lo que se trata es de poder eliminar ese producto. Y para eso, se pulsa sobre el producto (que aparecerá en la tabla de debajo) que se desea eliminar y a continuación, pulsar el botón para esta acción.

A continuación, aparece una de las acciones más importantes que hay en esta pestaña que es la "STOCK SEGURIDAD". Como su propio nombre indica, este hace referencia al stock de seguridad que debe de haber de ese producto. Para facilitar su comprensión, el stock de seguridad se trata de la cantidad mínima de producto que siempre debe haber en un almacén para hacer frente a los imprevistos que puedan surgir (como una variación excesiva en la demanda, o problemas varios con el suministro del producto en cuestión o del proveedor).

Por lo tanto, lo que se trata es ir realizando búsquedas de todos los productos, para ver en qué punto la cantidad actual de producto en el almacén se está acercando a la cantidad de

stock de seguridad para poder realizarle pedido al proveedor para poder aumentar esta cantidad y así alejarse de la del stock de seguridad. También es muy importante estudiar los tiempos que se han comentado arriba, porque no se trata de siempre mantener unos niveles de almacenamiento 5 veces por encima del stock de seguridad, ya que si se trata de unos productos que no se usan casi nunca, y que encima el proveedor suministra de forma rápida, se está ocupando espacio y dinero de mantenimiento un material que no es necesario tener

## 6.7 ÓRDENES

En esta pestaña se trata de relacionar un pedido con una o varias órdenes de trabajo. Como ya se ha explicado anteriormente, un pedido puede llevar varías ordenes, porque una orden implica la realización de un trabajo o de un conjunto de trabajos que se encuentren relacionados entre sí. Por ejemplo, si se compran 2 conjuntos que se refieren a partes diferentes del robot/máquina, o incluso a máquinas diferentes, pues para ese pedido llevaría relacionado 2 órdenes de trabajo. Todo este trabajo debe ser gestionado por el departamento de producción, que tiene que ver que hace falta, como se va a hacer y cuando.

Entonces, la primera parte fundamental que debe aparecer en esta pestaña, ver que pedidos han entrado nuevos. Es decir, que el departamento de producción no haya empezado a gestionar. Entonces las acciones que aparecen en esta primera tabla son:

- Actualizar.
- Vaciar.
- Tramitar.

Cada vez que se pulsa “ACTUALIZAR” lo que ocurre es que en la tabla aparecen todos los pedidos que no se han empezado a gestionar, entonces para empezar a poder gestionarlo, habría que pulsar sobre ese pedido y después pulsar en tramitar. Pero antes de eso, habría que pasar al apartado inferior, que es para poder crear las órdenes.

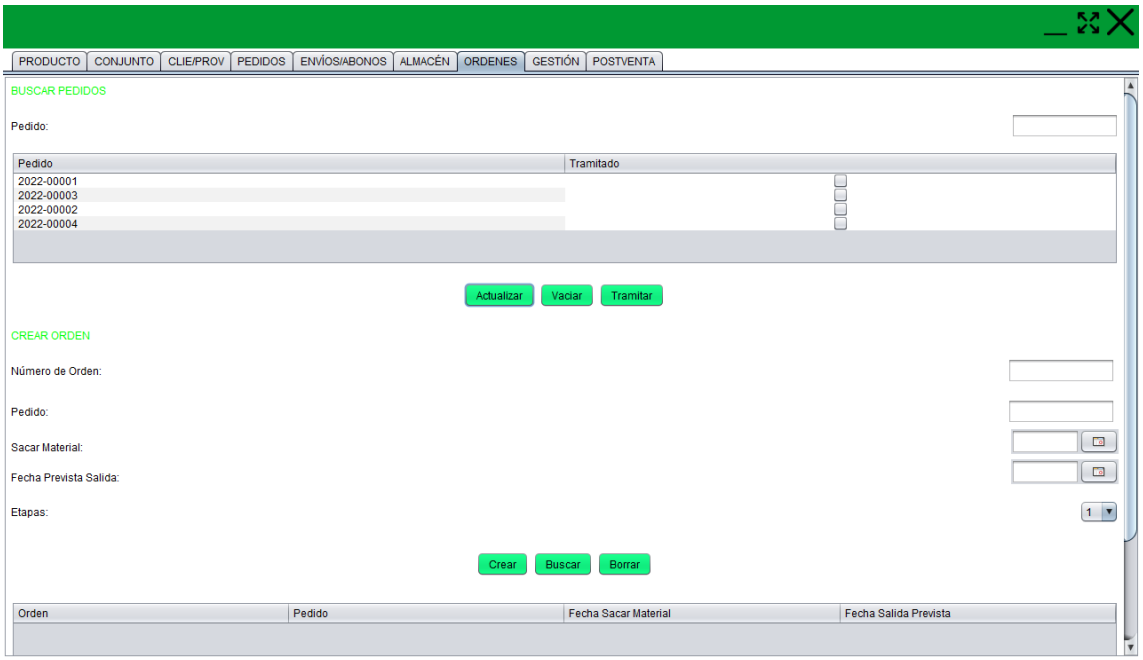


*Figura 34 – Aspecto de la pestaña órdenes.*

*FUENTE: Elaboración propia.*

Por lo tanto, aquí aparece la relación entre las dos partes. Arriba aparecen los pedidos que hay que tramitar, y debajo es la parte de donde se crean las órdenes. Entonces habrá que chequear el pedido, ver cuantas serían las mejores órdenes para completar el pedido. Entonces esas órdenes se crearían en la parte de debajo, se crearían tantas como hicieran falta. Una vez estas órdenes ya estén creadas, simplemente habría que indicar que el pedido ya se ha tramitado, para que así esa información le pueda aparecer al departamento del Almacén y así empezar a preparar el material cuando mejor toque según las fechas.

Las acciones que aparecen en la parte de debajo son:

- Crear.
- Buscar.
- Borrar.

Para poder crear la orden, se necesita añadir información tal como un número de identificación, indicar a que pedido irá relacionada. Además, habrá que indicar 2 fechas, una de ellas será la fecha en la que este material se deberá sacar al taller para montarlo en el tiempo suficiente para que sea terminarlo, embalarlo y enviarlo, sin que pase un tiempo almacenado en el almacén. La otra fecha que hay que indicar, es la fecha prevista de salida

para así que esa información pueda ir relacionada con la fecha anterior, para no perderla de vista. Y, por último, hay que indicar un valor que hace referencia a cuantas fases se compone el producto que se fa a montar en esa orden de fabricación. Aunque en esta pestaña pueda no entenderse el motivo de porque indicar que etapas lo forma.

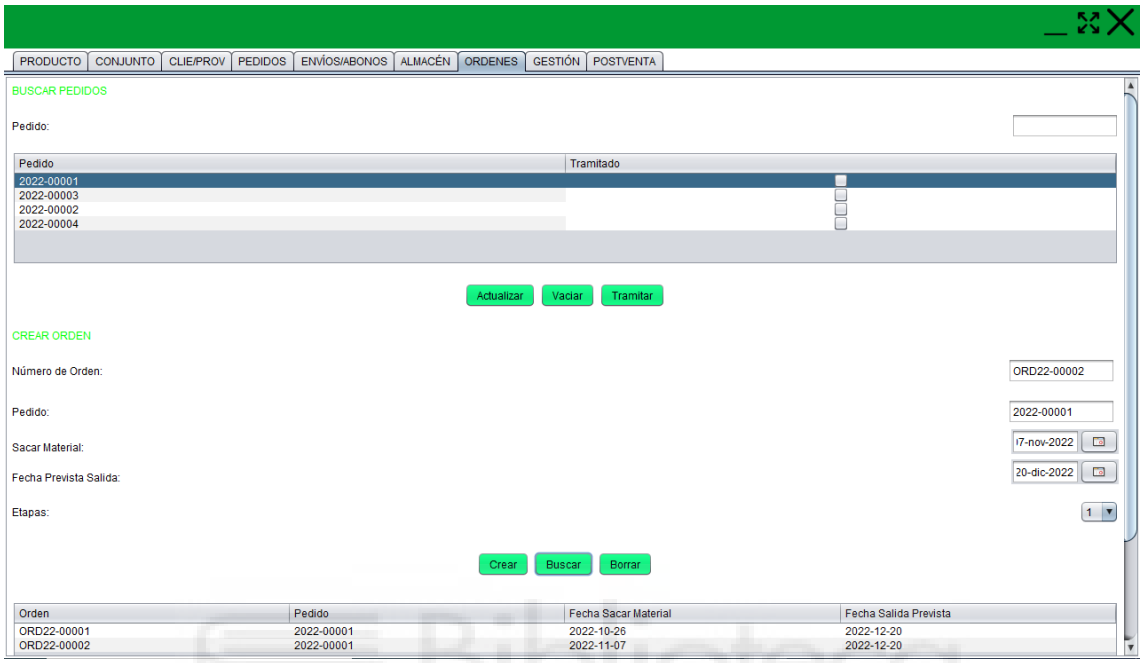


*Figura 35 – Momento antes de declarar un pedido "TRAMITADO".*

*FUENTE: Elaboración propia.*

La siguiente acción, la de "BUSCAR", simplemente permite filtrar por el número de orden, para que poder obtener la información básica a la hora de crear, editar o borrar una orden en caso de que sea necesario.

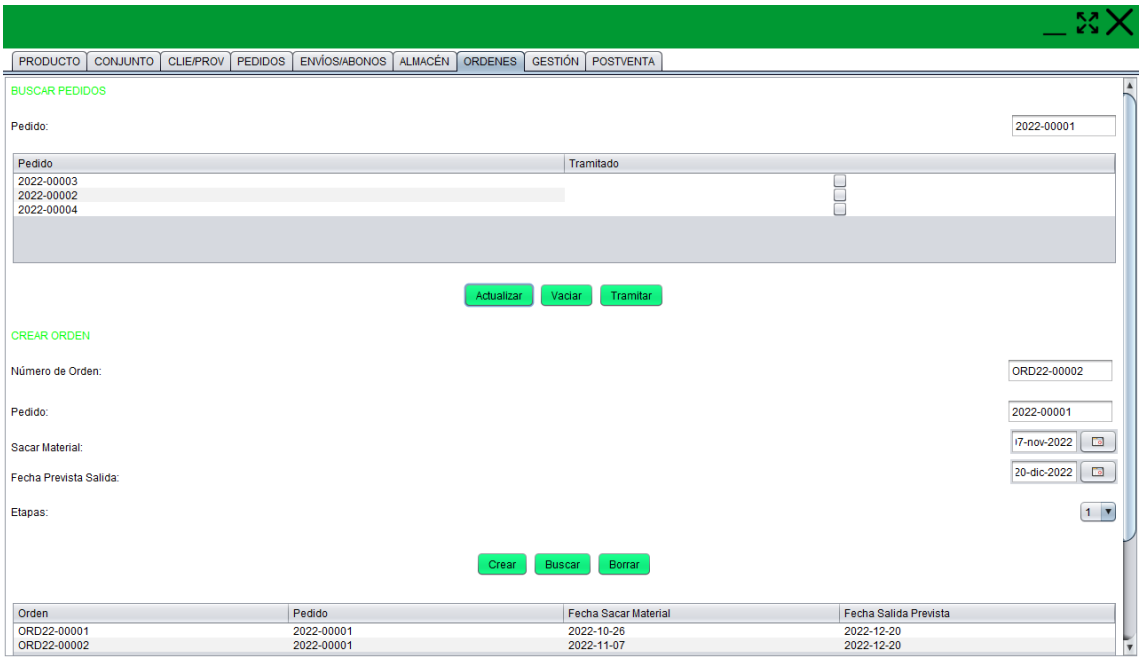


*Figura 36 – Los pedidos "TRAMITADOS" ya no aparecen.*

*FUENTE: Elaboración propia.*

También ocurre que un pedido no es urgente hasta que por algún motivo empieza a serlo, e incluso el caso contrario, en el que en algún momento el cliente pide que le mandes el producto más tarde lo que había pedido, porque quizás por algún motivo, no tenga las instalaciones preparadas para recibirlo. Es por esto que es necesario modificar las fechas que se marcaron en el momento de crear la orden de trabajo. Porque lo fundamental que se quiere conseguir, es reducir la cantidad de producto terminado que se encuentre en espera en el almacén.

## 6.8 GESTIÓN

La siguiente pestaña se trata de la de gestión. Aunque la palabra gestión pueda tener un significado muy amplio, ya que se puede gestionar cualquier departamento o ámbito de la empresa. Pero en este caso en concreto se refiere a controlar el departamento de producción. Concretamente a controlar el tiempo que se dedica a cada orden, para así poder ver si se está produciendo cerca del tiempo óptimo, o por si algún caso hay algún desvío, ya sea, tanto por arriba como por debajo.

El primer apartado de esta pestaña está titulado trabajo diario, en el que simplemente se trata de que cada trabajador, en el momento que llega a su puesto de trabajo tiene que indicar con que orden va estar trabajando y en el momento que se vaya, tiene que indicar que lo va a dejar.

Y no solo esto, también tiene que indicar los cambios de órdenes que hay durante su jornada laboral. Es decir, si el trabajador va a estar todo el día con la misma orden de trabajo lo que tiene que hacer es indicarse con su ID personal e indicar con que orden va a empezar a trabajar ese día, y por último, para irse le indicará una hora final de trabajo diario de esa orden. En cambio, si en un día va a estar con dos o más órdenes, la forma de indicarlo es, decir con que orden empieza, y si saliera algún tema urgente (o simplemente esa orden de montaje ya ha concluido por el producto ya está listo) lo que deberá hacer es darle una hora de fin a la orden y a continuación volver a indicar una hora de inicio para la nueva orden que vaya a iniciar a continuación.

Entonces, las acciones que acompañan a la tabla de este apartado son:

- Buscar por trabajador.
- Buscar por orden.
- Empezar trabajo.
- Cerrar trabajo.
- Vaciar (misma función en todas las tablas).

Para la primera acción, la de BUSCAR POR TRABAJADOR, lo que te permite es buscar en la base de datos según la ID del trabajador, así se podrá ver un histórico de ese trabajador a lo largo del tiempo, para así poder ver su evolución diaria viendo cuanto tiempo a dedicado a cada orden ese trabajador. Para eso hay que introducir la ID del trabajador en el primer campo de texto, y al pulsar sobre esta acción se realizará la búsqueda a la base de datos.

La segunda acción es la de BUSCAR POR ORDEN. Con esta acción lo que se consigue es ver que trabajadores han estado trabajando con esa orden y durante que tiempo, ya que dependiendo del trabajo que se tenga que realizar en esa orden o la urgencia en finalizar esos trabajos, pues se tenga que dedicar más personal o no. La forma de realizar la

búsqueda es como todas, en este caso introducir el número de orden en el segundo campo de texto que aparece en este apartado y a continuación pulsar sobre esta acción.

Para la acción que aparece ahora, la de EMPEZAR TRABAJO, lo que habría que hacer es indicar que trabajador es el que va a empezar el trabajo, y en qué orden (todo esto se indicará en los dos campos de texto que aparecen inmediatamente arriba, que ya se han mencionado en los dos párrafos anteriores). Una vez escritos estos datos se pulsará sobre la acción. En ese mismo momento lo que se estará haciendo sobre la base de datos es registrar el día y el momento exacto en el que se ha pulsado sobre la acción. Así se guardará que trabajador ha cogido X orden y el momento exacto que la ha empezado.

Para la siguiente acción ocurre lo mismo, la de CERRAR TRABAJO. Lo que hay que indicar es que trabajador va a cerrar ese trabajo que lleva empezado y en qué orden. Y cada vez que se pulse sobre la acción, aparte de que se guarde los dos campos arriba mencionados se guarda automáticamente la hora a la que se ha pulsado sobre la acción.

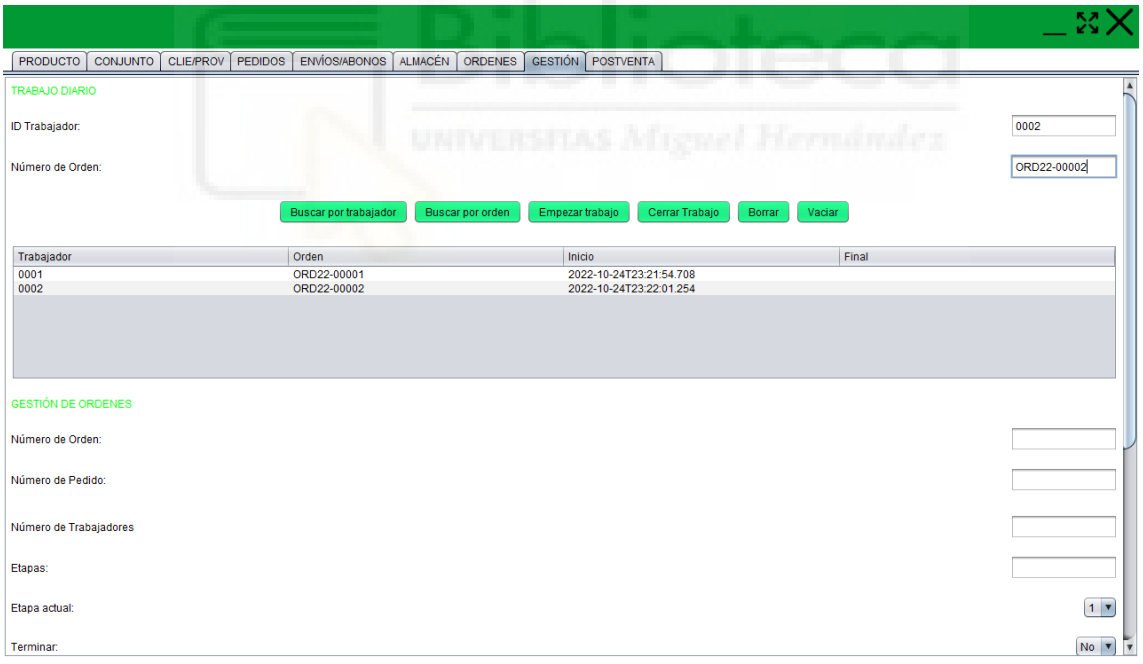


*Figura 37 – Registrar el inicio del trabajo diario de una orden por parte de un trabajador.*

*FUENTE: Elaboración propia*

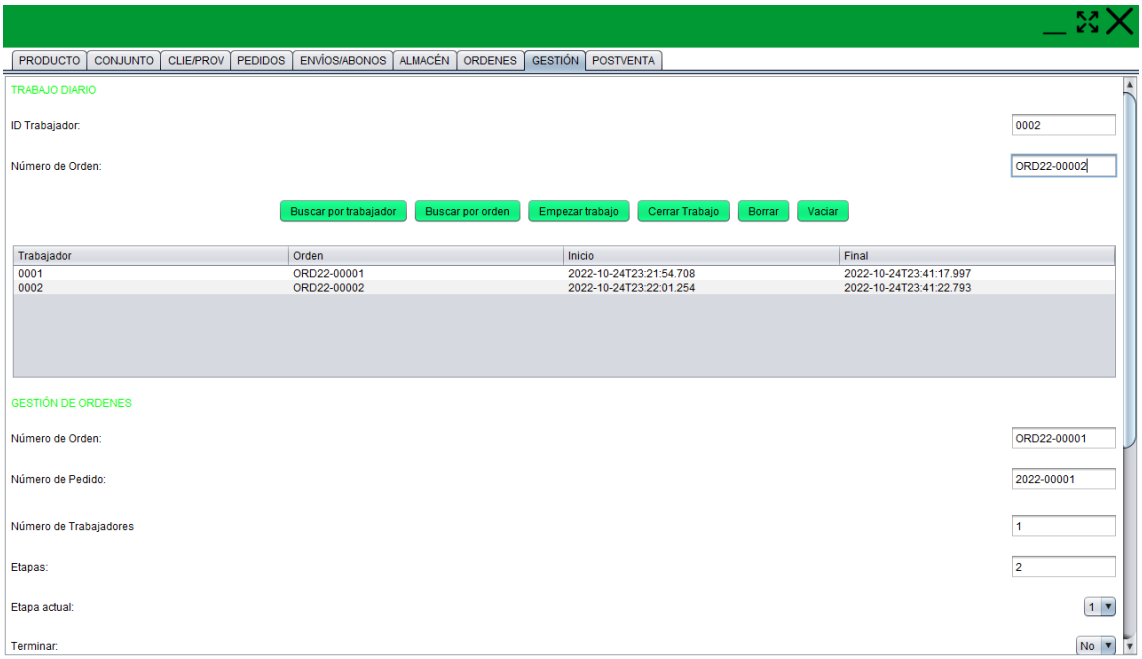


*Figura 38 – Registrar el fin del trabajo diario con esa orden.*

*FUENTE: Elaboración propia.*

Con estas dos acciones, y la tabla que te permite visualizar toda la información, se puede realizar un control de la producción diaria, ya que se ve en que trabajos ha estado cada trabajador durante cada día.

Aunque no se tenga acceso general a las aplicaciones del software ERP, para un correcto uso de este apartado se deben disponer de dispositivos para que los propios empleados del taller, para que puedan registrar la información de la forma más rápida posible.

Como esta pestaña se compone de dos apartados, una vez se ha comentado el de “trabajo diario” se llega al apartado de “gestión de órdenes”. Aquí es donde se crearán las órdenes y se les añadirá la información de cada orden a lo largo del tiempo. Aquí ya no importa tanto quien realice cada orden, sino que lo que importa es la evolución de la orden a lo largo del tiempo hasta que llega a su fin.

Las acciones para este apartado son:

- Crear.
- Empezar etapa.
- Cerrar etapa.

- Buscar por orden.
- Buscar por etapa.
- Añadir nota.
- “Terminado/Deshacer”.
- Borrar.
- Vaciar. (Sirve para vaciar el contenido de cualquier tabla)

Para la acción de CREAR, lo que se trata es de añadir una línea de datos a la tabla en la que se le añadirá información tal como la evolución de esa orden en el tiempo. Por lo tanto, los datos a añadir en los campos de texto que aparecen justo arriba de las tablas son la del número de esa orden, el número del pedido al que va relacionado esa orden (toda esta información ya fue usada en la pestaña de órdenes, en el apartado de crear órdenes). Además de los dos campos mencionados, también hay que añadir información sobre cuantos trabajadores se van a destinar a la realización de esta orden (ya que según los tipos de trabajos que se tengan que realizar y lo urgente que sea esta orden o no, habrá que añadir más o menos trabajadores.

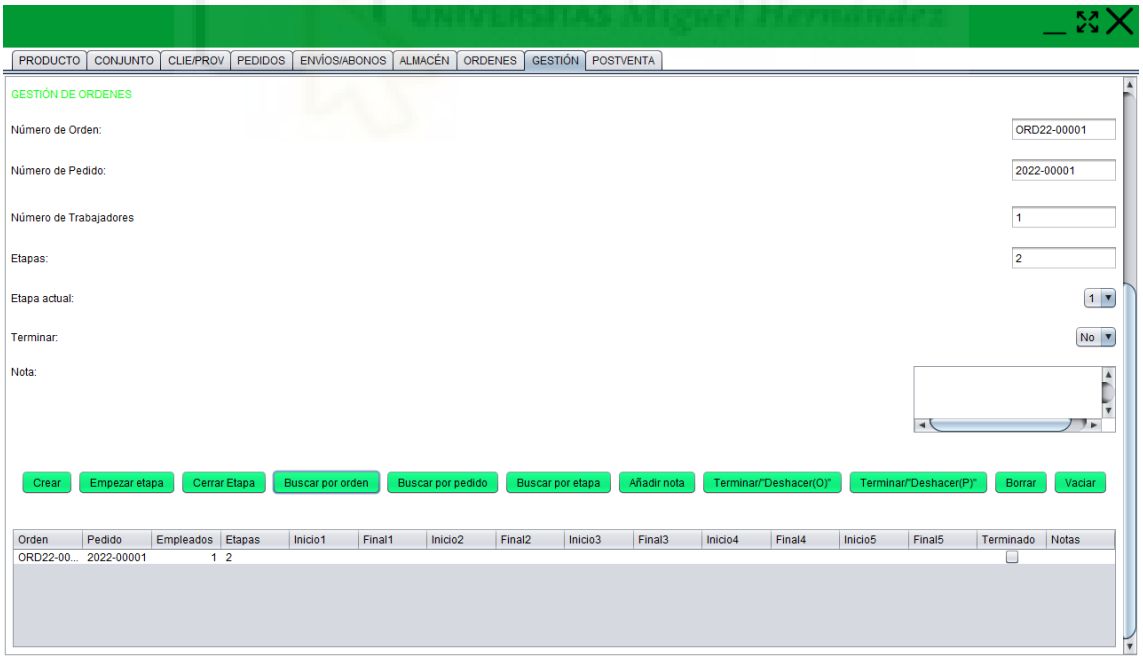


*Figura 39 – Inicio de registro general de una orden.*

*FUENTE: Elaboración propia.*

La última información que hay que añadir para poder gestionar una orden que se acaba de crear es el número de etapas de la que está compuesta. Para eso, en el departamento de producción se habrá realizado un trabajo exhaustivo en el que se vea para la realización de cada producto de la empresa, en cuantas etapas se divide la producción de ese producto y que se realiza en cada una de ellas. Para así después poder controlar cada una de esas etapas.

La siguiente acción es la de EMPEZAR ETAPA. Cada vez que se empieza una etapa de las que se compone la orden (como se ha mencionado en el párrafo anterior, crea en el momento que se añade la línea de la orden) hay que pulsar sobre este botón para así dejar reflejado en que día y en qué hora se está empezando esta etapa. Como no, en el campo desplegable que hay arriba llamado etapas, se le debe indicar que etapa es la que se está empezando, ya que, si no se modifica, por defecto siempre se estará añadiendo información en la etapa 1.

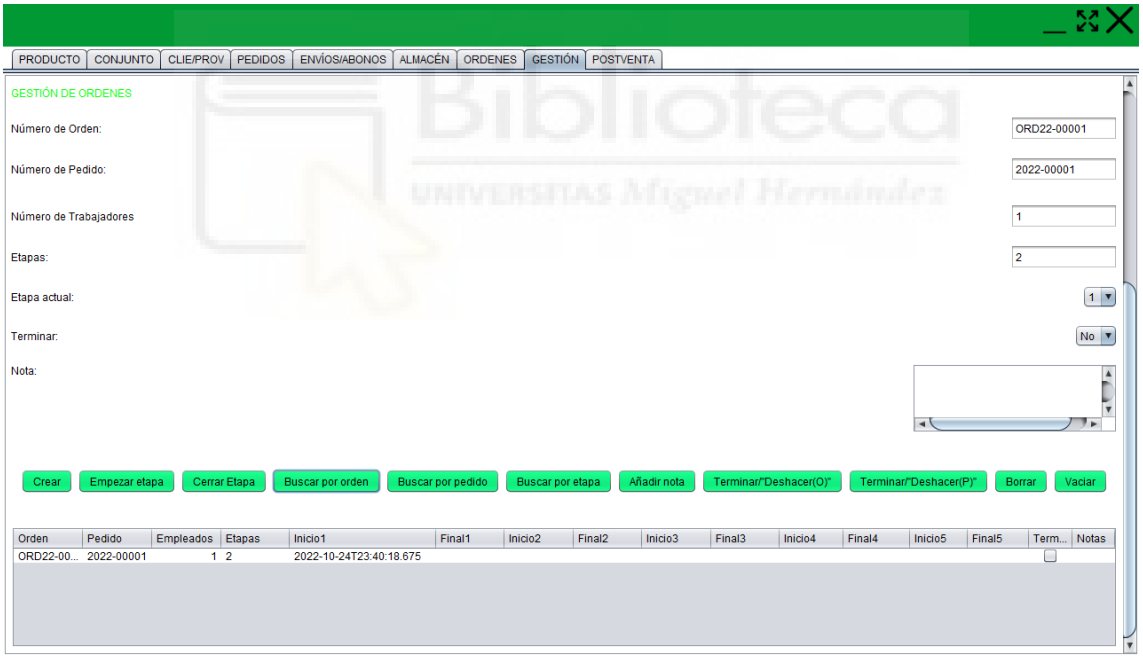


*Figura 40 – Muestra los datos guardados del inicio de una etapa de la orden.*

*FUENTE: Elaboración propia.*

Lo mismo ocurre para la siguiente acción, que es la de CERRAR ETAPA, que, a la mayor celeridad posible, cada vez que se finalice una etapa se debe pulsar sobre esta acción para así dejar constancia de la fecha y de la hora a la que se ha terminado la etapa. Ocurre lo

mismo que con la acción anterior, hay que indicar en el desplegable de etapas, que etapa es en la que se está poniendo fin a esa etapa.

Gracias a la información recopilada en estas últimas dos acciones, la persona encargada de la gestión del departamento de producción puede estudiar el tiempo que ha llevado cada etapa (ya que se dispone de la fecha de inicio y de fin). Así se puede ver si está dentro de los estándares marcados por la empresa.

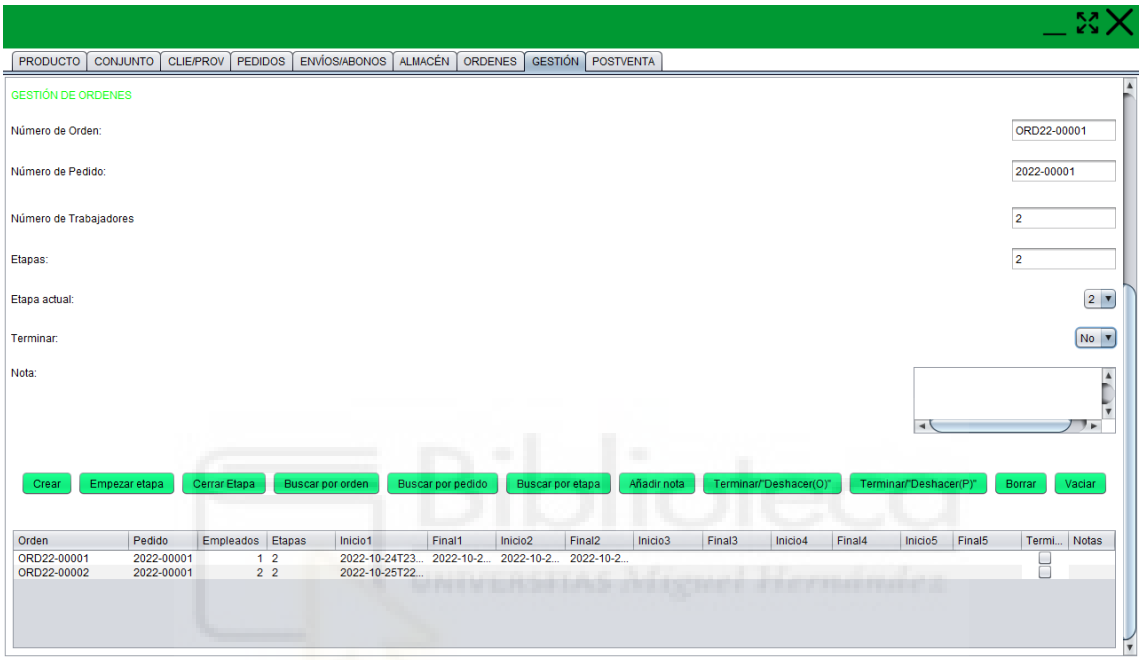


*Figura 41 – Muestra la información de trabajo de cada una de las etapas de una orden.*

*FUENTE: Elaboración propia.*

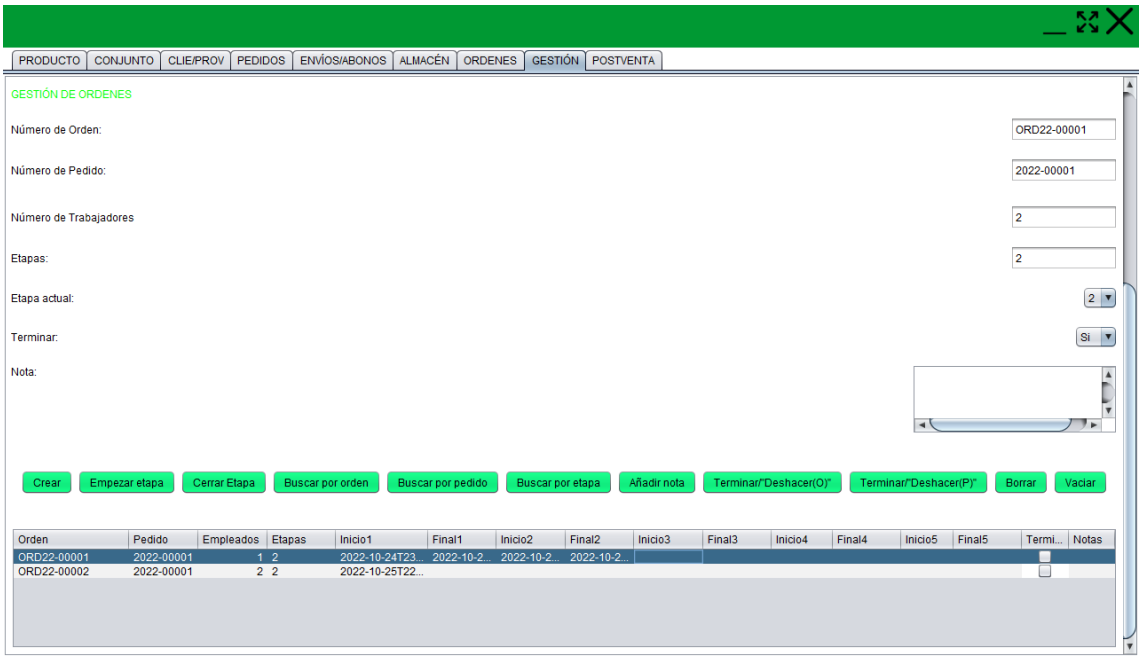


*Figura 42 – Una vez terminada la orden, muestra el procedimiento para declararla terminada.*

*FUENTE: Elaboración propia.*

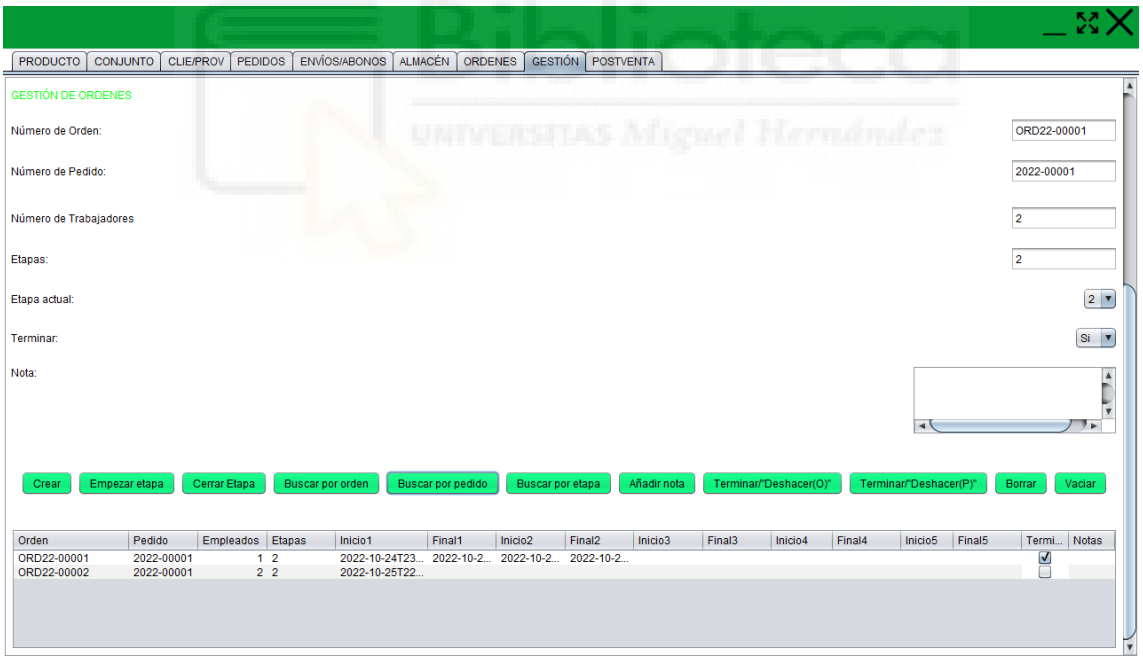


*Figura 43 – Muestra esa orden, con la información de que se encuentra "TERMINADA".*

*FUENTE: Elaboración propia.*

Como cada vez que se añade información a la tabla, no se va mostrando, hay que añadir ciertas acciones para que puedan mostrar el contenido de la tabla de cierta forma para que esta información sea lo más fácil posible de entender de un solo vistazo. Entonces

encontramos dos acciones similares que son la de BUSCAR POR ORDEN y la de BUSCAR POR ETAPA. Con la primera permite filtrar por el número de orden para así poder ver sus evoluciones en el tiempo. En cambio, en la otra acción busca todas las líneas de ordenes que coincidan con el número de etapas marcadas en el desplegable. Así de un simple vistazo se puede ver la duración de las etapas de órdenes de trabajo similares. Con esto te permite encontrar irregularidades de una forma más sencilla.

La acción que aparece a continuación es de las más importantes ya que te permite contextualizar la información que aparece en cada línea de la orden. Y esta no es otra que la de AÑADIR NOTA.

En el campo de texto que aparece justo de arriba llamado nota te permite añadir diferentes frases que son necesarias para explicar quizás posibles retrasos o adelantos en la ejecución de una orden. Porque es posible que en el alguna etapa se produzcan retrasos considerables provocados por falta de material debido a una crisis en el suministro o a una huelga de transporte, entonces esto se debe indicar en el apartado de notas.

Pero al contrario lo mismo, ya que, si se han indicado los retrasos, también puede pasar lo mismo, pero con adelantos. Ya que es posible que si hay algún pedido que sea muy urgente, desde producción se haya decidido reforzar con personal extra para acelerar su producción, pues quizás a partir de la segunda etapa se ha reforzado el personal. Pues en este caso sería conveniente añadir esta nota, para poder explicar ese cambio extraño en la producción, para no concluir que se ha ido rápido de más y ya de por si indica un problema.

Con la información de esta tabla, si se combina con la información aportada por la tabla comentada en el apartado anterior, te permite ver en concreto que trabajador ha estado en cada orden y como ha ido la ejecución de esa orden. Así si hay algún problema con la ejecución de las órdenes, ver si hay alguna relación con el personal que las realiza.

Hasta ahora, se ha ido añadiendo la fecha y la hora de cuando se empieza una etapa y cuando se cierra. Pero además es conveniente de indicar cuando se ha terminado esa orden, y para eso se utiliza la acción de TERMINADO/DESHACER. Así cuando se ha terminado, en el desplegable cuyo título es el de terminado, se indica “si” y se pulsa sobre



UNIVERSITAS Miguel Hernández

esta acción, así ya permite al departamento de producción darla por cerrada y terminada. También existe la posibilidad, de que se haya indicado que se ha terminado una orden por error. Entonces lo que se podría hacer es indicar en el desplegable con el título de terminado y se selecciona que "no", en ese momento, en la tabla, se pulsa sobre la orden que se quiera indicar que no se ha terminado, y se pulsa sobre la acción de TERMINADO/DESHACER, y listo.

La siguiente acción que aparece es la de BORRAR, esto quiere decir, que cuando haya un error creando una línea de orden para su gestión en la producción, para poder borrarla, lo que hay que hacer es pulsar sobre esa misma línea, y después pulsar sobre borrar.

## 6.9 POSTVENTA

En esta pestaña, es la dedicada al departamento de POSTVENTA, para poder registrar todas las comunicaciones/interacciones entre los clientes y la empresa, una vez ya se les ha vendido la máquina/producto y este ya lo tienen ellos. La información recogida por este departamento es muy importante ya que es la que servirá de retroalimentación. Es decir, todos los demás departamentos se deben nutrir de la información que sea capaz de recopilar este departamento para así poder ver que fallos cometen cada uno. Porque toda organización debe estar enfocada en el cliente y su satisfacción.

En este primer apartado que aparece en esta pestaña es el de las INTERACCIONES. Aquí se tratará de registrar cada una de las interacciones que se le realicen con los clientes, independientemente de la vía por la que se hagan (ya sea tanto por e-mail, por llamada telefónica e incluso por mensaje al móvil). La clave de ese apartado, aparte de registrar la interacción es poder registrar el motivo de que el cliente haya contactado a la empresa, para así ser capaz de poder ir filtrando.

Las acciones que aparecen en la tabla de este apartado para poder ir guardando y mostrando los datos son:

- Crear.
- Buscar por cliente.
- Buscar por motivo.



UNIVERSITAS Miguel Hernández

- Buscar por cliente y motivo.
- Borrar.
- Vaciar. (sirve para vaciar el contenido de la tabla)

Para la primera acción, la de CREAR una nueva interacción una vez se ha mantenido una comunicación con un cliente habría que indicar en el primer campo de texto, el número/id de esa interacción, a continuación, indicar que cliente es el que ha realizado la interacción. Lo siguiente que habría que indicar, es en un desplegable, cual es el motivo por el que el cliente ha querido ponerse en contacto con nosotros.

Los diferentes motivos están encapsulados dentro de un desplegable, y se debe seleccionar uno de ellos. Esto es así ya que, al ser una elección cerrada, posteriormente te permitirá filtrar por cada uno de los motivos. Estos motivos son: “garantía”, “repuestos”, “solicitar información”, “reportar fallo”, “solicitar documentación” y por último “quejas”.

Aunque algunos de los motivos parezcan similares y/o sinónimos, no lo son, ya que la confusión puede estar entre los términos de “solicitar información” y “solicitar documentación”. Que, aunque son similares, y es verdad que se podrían considerar como que hacen referencia a lo mismo. Pero en este caso en concreto se requiere diferenciarlos, ya que el término de SOLICTAR DOCUMENTACIÓN está más pensado en que el cliente solicite algún documento de la máquina tales como esquema eléctrico, neumático (si fuera el caso), manual de despiece o manual de usuario. En cambio, el otro término hace referencia a por ejemplo si se solicita algún kit/ampliación de la máquina o producto, solicita de que se le envíe información de como instalarlo y hacerlo funcionar.

Los demás motivos aparecen claros, ya que el de “garantía” es que haya fallado una pieza que se encuentre dentro del periodo de garantía y se deba reponer. “Repuestos” es para cuando el cliente quiere solicitar algún repuesto para la máquina/robot que tiene allí y el de “reportar fallo” es cuando no se puede hacer un uso normal del producto y se comunica a la empresa fabricante que no funciona y se solicita ayuda para solucionarlo. Por último, las “quejas” simplemente es para reportar problemas en el funcionamiento de la empresa, tales como que no se entrega la información suficiente para poder saber cómo funciona, o plazos de entrega muy elevados, temas de precios, etc.

El siguiente campo por rellenar con información cuando se crea la interacción es la de indicar el tipo de fallo que es la interacción. Por lo tanto, al explicar esto, no quiere decir que sea siempre un campo de texto que sea obligatorio rellenar. Solo hay que rellenar este campo, cuando en el desplegable anterior se haya indicado que la interacción con el cliente se haya debido a un fallo habrá que indicar en este campo que tipo de fallo ha sido. Por ejemplo, fallo por desgaste en una pieza mecánica, problema en algún sensor, mal montaje del producto antes de enviarlo, problema en el envío, etc.

Y, por último, aparece el campo de notas, y es por si se quisiera explicar algo con más detalle, o quizás dejar constancia de algo para poder recordarlo en el futuro, este sería ese campo.

La siguiente acción por comentar es la de BUSCAR POR CLIENTE. Pulsando sobre esta acción lo que te permite es poder ver todas las interacciones que ha hecho un cliente. Así se puede ver que tipos de interacciones suele reportar. Porque como cada cliente es de una forma se puede llegar a buscar un patrón de interacciones por zonas y además permite conocer a cada cliente para que después el departamento comercial en un futuro pueda enfocar mejor su estrategia con cada uno de los clientes. Entonces para que haga este tipo de búsqueda, en el campo de texto del cliente hay que escribir su nombre, para así después pulsar sobre esta acción. Al final lo que se trata de conseguir es de poder recabar cuantas más información posible, para así ir aplicando las técnicas de LEAN MANUFACTURING mencionadas en capítulos anteriores para así poder optimizar todos los recursos de la empresa.

También existe la posibilidad de poder BUSCAR POR MOTIVO. Para poder realizar esto, en el desplegable, hay que indicar el tipo de motivo por el que buscar, y justo después pulsar sobre esta acción. Así lo que mostrará será todas las interacciones que coincidan con el motivo que se quiere buscar.

Otro tipo de búsqueda es el de la siguiente acción, que permite buscar tanto por cliente como por motivo (la acción es la de BUSCAR POR CLIENTE Y MOTIVO), así filtra mucho más la información. Esta sería como combinar los dos tipos de búsquedas anteriores, pero en uno solo. Para eso, en los campos de arriba, habría que indicar el



UNIVERSITAS Miguel Hernández

cliente y el motivo para que así cuando se pulse sobre la acción pueda filtrar, ya que, si no se indica nada, no habrá filtros y mostrará toda la información

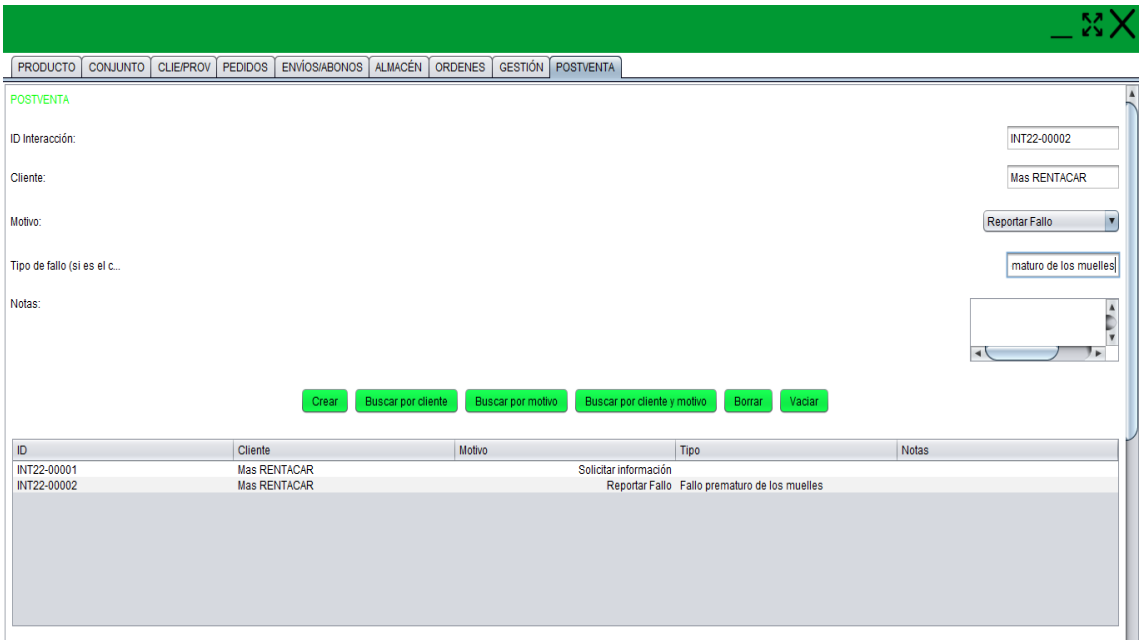


*Figura 44 – Se trata de mostrar por cliente, justo una vez se ha creado la segunda interacción.*

*FUENTE: Elaboración propia.*

La última acción a comentar en este apartado es la "BORRAR", en la que simplemente pulsando sobre una interacción en la tabla, (que es la que se quiere borrar) y después pulsando sobre la acción, ya quedará la interacción borrada definitivamente de la tabla de la base de datos.

A continuación, dentro del mismo apartado aparece otra tabla debajo con diferentes acciones. Esta parte de este apartado de interacciones es controlada por el personal del departamento de postventa (igual que la parte anterior), ya que inmediatamente después de cada interacción, lo que se debe de realizar es añadir una nota numérica (entre 0 y 10 por ejemplo), para así poder valorar el estado de satisfacción del cliente una vez se ha resuelto el problema (o mejor dicho se ha terminado la interacción con él). Para así poder realizar posteriores estudios con ellas y ver si hay que cambiar ciertos enfoques y poder centrarse en personalizar las estrategias de venta por clientes, que es lo que persigue con este software de gestión.

Por lo tanto, las acciones que aparecen para esta tabla son las siguientes:

- Crear.
- Buscar por cliente.
- Calcular media.
- Borrar.
- Vaciar (vaciar el contenido de la tabla que hay debajo).

La acción de “CREAR” es para cuando una vez se ha creado la línea de interacción en la tabla de arriba, y se haya terminado de hablar con el cliente, lo que habría que hacer es en los dos campos inmediatamente superiores a esta tabla, añadir el nombre del cliente con el que se ha tenido la interacción y añadir una nota, que venga de la valoración de cuál ha sido la satisfacción del cliente una vez ha hablado con nosotros.

A continuación, lo que aparece es la acción de “BUSCAR POR CLIENTE”. Así una vez se introduce el nombre del cliente en el campo de arriba se pulsa sobre la acción y te permite mostrar todas las notas de ese cliente después de haber realizado una interacción, y se puede conocer la evolución de su estado de ánimo. También se añade la opción de, además de poder ver la nota de todas y cada una de las interacciones que ha hecho un cliente en concreto, está la opción de poder calcular la nota media del conjunto de interacciones que ha hecho ese cliente. Esto se consigue, escribiendo el nombre del cliente en el campo de texto que hay arriba y pulsando sore la acción de “CALCULAR MEDIA”.

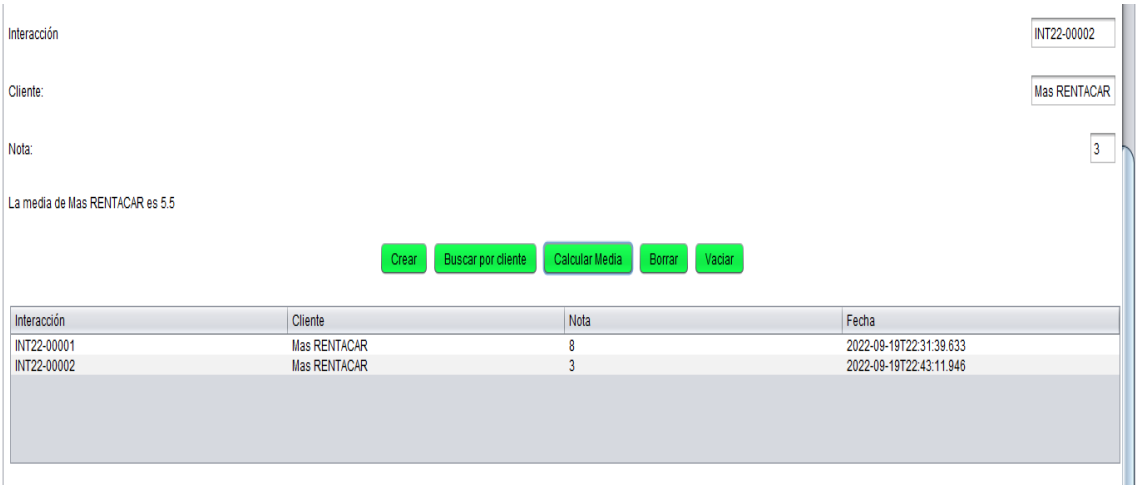


*Figura 45 – Calificar una interacción.*

*FUENTE: Elaboración propia.*

La siguiente acción es la de “BORRAR”, en la que simplemente, cuando se quiera borrar la nota de una interacción, lo que habrá que realizar es pulsar sobre esta nota de interacción en la tabla, y después pulsar sobre el botón de la acción.

Una vez comentados estos dos apartados, se llega ya al último. El último apartado es el que sería el más visual de todos, ya que permite crear gráficas. Estas gráficas serían del tipo gráficas de barras en las que se encargaría de realizar la suma de cada uno de los diferentes motivos de interacciones que han sido especificados en el primer apartado de esta pestaña (garantía, repuestos, solicitar información, reportar fallo, solicitar documentación y quejas). Pero puede mostrar 2 tipos de información. Una es un gráfico en el que muestra el conjunto de cada una de las interacciones que ha recibido la empresa, esto se conseguiría pulsando sobre la acción “CREAR GRÁFICA EN GENERAL”, y en cambio, hay otra acción que permite mostrar en una gráfica la suma de cada uno de los motivos de las interacciones realizadas por el cliente. Para esto se consigue escribiendo el nombre del cliente en el campo de texto de arriba, y a continuación pulsar sobre la acción “CREAR GRÁFICA POR CLIENTE”.

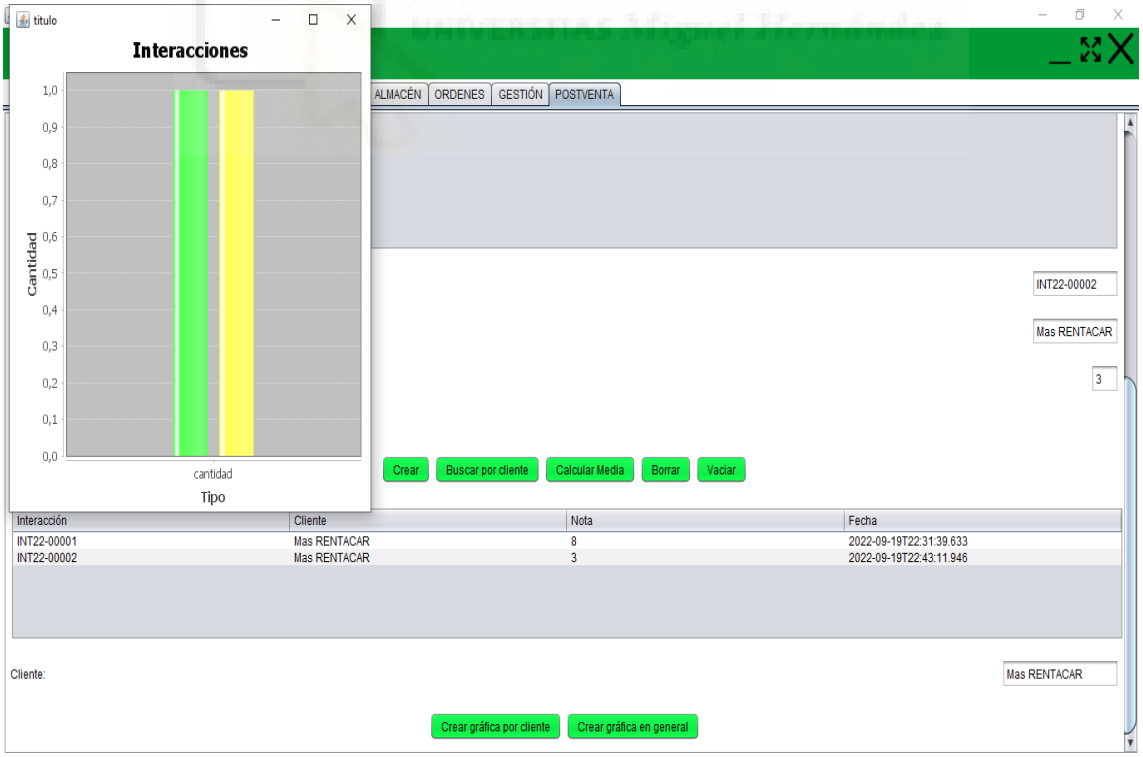


*Figura 46 – Gráfica que muestra todas las interacciones de un solo cliente.*

*FUENTE: Elaboración propia.*

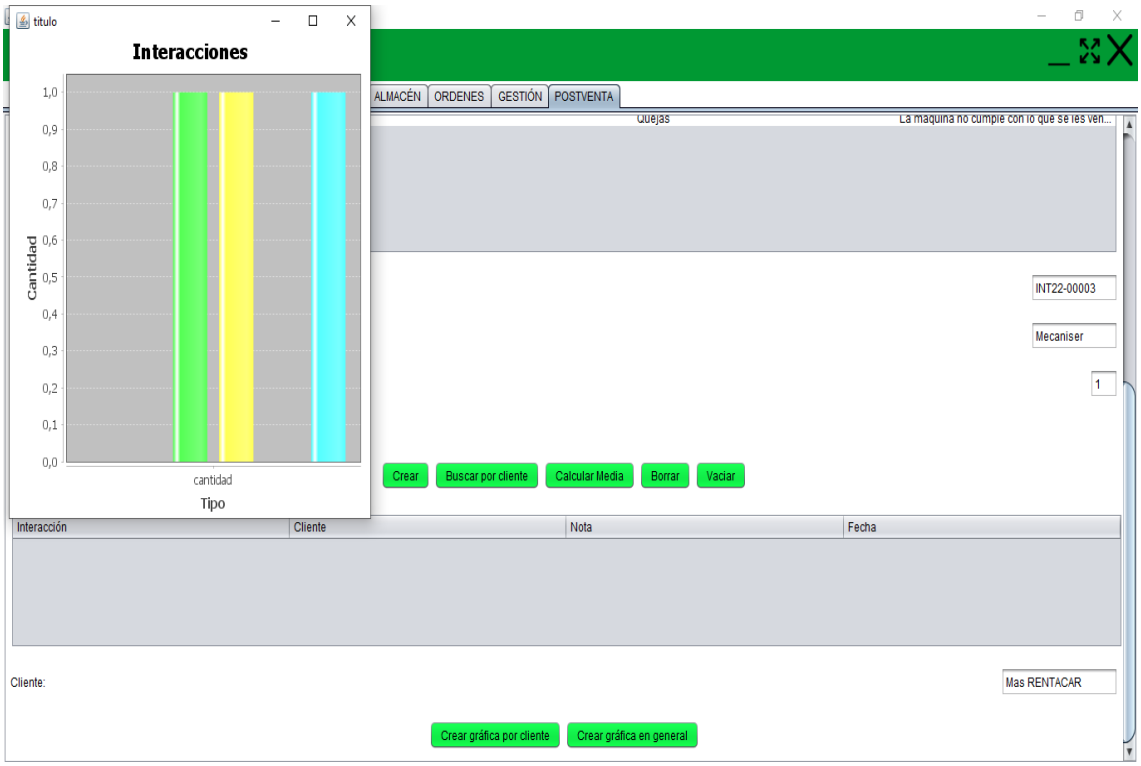


*Figura 47 – Muestra todas las interacciones de todos los clientes.*

*FUENTE: Elaboración propia.*

Biblioteca
UNIVERSITAS Miguel Hernández

# 7. CONCLUSIÓN Y TRABAJOS FUTUROS

Una vez entendido como funciona una empresa de carácter industrial, a que retos y actividades se enfrenta cada día, había que encontrar un método o una forma de trabajo en poder reducir problemas que puedan aparecer (y por lo tanto reducir costes).

La filosofía de este software se basa en dos pilares. El pilar fundamental se trataría el de recabar la mayor información posible. Y conviene matizar, la mayor información "necesaria" posible. Porque puede haber muchos datos recabados que sean innecesarios, es decir, que no aporten información. Por eso en este trabajo se ha querido dar mucho énfasis en el integrar el desarrollo y diseño informático involucrando la gestión industrial. Ya que por muchas herramientas de programación informática que haya, se necesita involucrar perfiles de gestión industrial para así poder orientar donde se encuentran los datos importantes y por aconsejar como mostrarlos.

Porque una vez comentado lo del párrafo anterior nos lleva al otro pilar. Ese pilar sería el de poder mejorar la comunicación entre los diferentes departamentos, porque la forma en la que se muestre ayudará mejor a su comprensión y a la integración de los departamentos. Ya que muchas veces la comunicación no es la mejor en las empresas. Pues con este software se forzaba a que cada departamento recibiera la información necesaria en el momento que otro departamento realizaba sus actividades.

Para que pueda existir una buena comunicación es necesario que la información sea fácilmente accesible. Por lo tanto, las tablas de las bases de datos contienen más información que las tablas que aparecen en el interfaz. Ya que lo que se buscaba es que en cada pestaña apareciera la información justa y necesaria. Aquí aparece una de las posibles mejoras del software, que sería la de crear diferentes niveles de acceso. Esto quiere decir que las personas de un departamento solo puedan acceder a la información que les compete.



UNIVERSITAS Miguel Hernández

Aunque no solo distinguir por departamentos, sino por cargo. Es decir, el responsable del departamento de logística que pueda acceder también a la información del estado del almacén. O también, que el encargado del departamento de producción pueda acceder a la información del departamento de postventa, para que así pueda conocer qué clase de reportes se están recibiendo por parte de los clientes, y cuáles pueden ser las notas de satisfacción de los clientes.

Otra de las mejoras que se podría implementar es la de una versión del software para smartphone. Esta versión más que introducción de datos, que pueda ser la de consulta, ya que así permitiría de un vistazo rápido poder repasar stocks, o poder revisar los últimos pedidos que se nos hayan realizado.

Por último, para terminar de completar el software, sería la de poder integrar un módulo de contabilidad financiera y fiscal. Para que así puedan utilizar toda la información de los pedidos para poder realizar estas actividades. Y que toda actividad realizada por la empresa se quede guardad en la misma base de datos.

Biblioteca UNIVERSITAS Miguel Hernández

# 8. ANEXO – CÓDIGO FUNCIONES BASE DE DATOS

```
package javafxapplication4;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.time.LocalDateTime;
import javax.swing.table.DefaultTableModel;
//import sun.tools.jar.resources.jar;

/**
 *
 * @author USUARIO
 */
public class basedatos {
   Connection cnx = null;

   public void conectar() throws SQLException, ClassNotFoundException{
      cnx = DriverManager.getConnection("jdbc:mysql://localhost:3306/tfm?useLegacyDatetimeCode=false&serverTimezone=Europe/Madrid",
"root", "skynetJm");
    }

   public void insertarproducto(String referencia, String descripcion, int precio, boolean propio, boolean conjunto, String nombre_proveedor, String
pais, int coste ) throws SQLException, ClassNotFoundException {
   conectar();
   try (PreparedStatement ps = cnx.prepareStatement("INSERT INTO productos VALUES (NULL, ?, ?, ?,?,?,?,?,?)")) {
         ps.setString(1, referencia);
         ps.setString(2, descripcion);
         ps.setInt(3, precio);
         ps.setBoolean(4, propio);
         ps.setBoolean(5, conjunto);
         ps.setString(6, nombre_proveedor);
         ps.setString(7, pais);
         ps.setInt(8, coste);
         ps.executeUpdate();
         ps.close();
      }

   cnx.close();
   }

   public DefaultTableModel buscarproducto(DefaultTableModel mod, String parte_busqueda, int busqueda) throws SQLException,
ClassNotFoundException{
      conectar();
      PreparedStatement ps1,ps2;
      ps1= cnx.prepareStatement("select * from productos where referencia LIKE ?");
      ps2= cnx.prepareStatement("select * from productos where descripción LIKE ?");
      if (busqueda==0){
         ps1.setString(1, "%"+parte_busqueda+"%");
         ResultSet rs0 = ps1.executeQuery();
         Object[] row;
         while (rs0.next()){
            row = new Object[8];
            row[0]=  rs0.getString(2);
            row[1]= rs0.getString(3);
            row[2]= rs0.getInt(4);
            row[3]= rs0.getBoolean(5);
            row[4]= rs0.getBoolean(6);
            row[5]=rs0.getString(7);
            row[6]= rs0.getString(8);
            row[7]= rs0.getString(9);
            mod.addRow(row);
         }
      }
      else {
         ps2.setString(1, "%"+parte_busqueda+"%");
         ResultSet rs1 = ps2.executeQuery();
         Object[] row1;
         while (rs1.next()){
```

```
            row1 = new Object[8];
            row1[0]=  rs1.getString(2);
            row1[1]= rs1.getString(3);
            row1[2]= rs1.getInt(4);
            row1[3]= rs1.getBoolean(5);
            row1[4]= rs1.getBoolean(6);
            row1[5]=rs1.getString(7);
            row1[6]= rs1.getString(8);
            row1[7]= rs1.getString(9);
            mod.addRow(row1);
        }
    }
    ps1.close();
    ps2.close();
    cnx.close();
    return mod;
}

public void borrarclienteseleccionado(String referencia) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps = cnx.prepareStatement("DELETE FROM productos WHERE referencia=?");
    ps.setString(1, referencia);
    ps.executeUpdate();
    ps.close();
    cnx.close();
}

public void editarproducto (String referencia, int esp, String texto) throws SQLException, ClassNotFoundException{
    conectar();
    switch (esp){
        case 0:
            System.out.println("dentro del case 0");
            System.out.println(texto);
            PreparedStatement ps0 = cnx.prepareStatement("UPDATE productos SET descripción=? WHERE referencia=?");
            ps0.setString(1, texto);
            ps0.setString(2, referencia);
            ps0.executeUpdate();
            ps0.close();
            break;
        case 1:
            System.out.println("dentro del case 1");
            int precio = Integer.parseInt(texto);
            System.out.println(precio);
            PreparedStatement ps1 = cnx.prepareStatement("UPDATE productos SET precio=? WHERE referencia=?");
            ps1.setInt(1, precio);
            ps1.setString(2, referencia);
            ps1.executeUpdate();
            ps1.close();
            break;
        case 2:
            System.out.println("dentro del case 2");
            PreparedStatement ps2 = cnx.prepareStatement("UPDATE productos SET nombre_proveedor=? WHERE referencia=?");

            ps2.setString(1, texto);
            ps2.setString(2, referencia);
            ps2.executeUpdate();
            ps2.close();
            break;
        case 3:
            System.out.println("dentro del case 3");
            PreparedStatement ps4 = cnx.prepareStatement("UPDATE productos SET pais=? WHERE referencia=?");
            ps4.setString(1, texto);
            ps4.setString(2, referencia);
            ps4.executeUpdate();
            ps4.close();
            break;
        case 4:
            System.out.println("dentro del case 4");
            int precio2 = Integer.valueOf(texto);
            PreparedStatement ps3 = cnx.prepareStatement("UPDATE productos SET coste=? WHERE referencia=?");
            ps3.setInt(1, precio2);
            ps3.setString(2, referencia);
            ps3.executeUpdate();
            ps3.close();
            break;
```

Biblioteca UNIVERSITAS Miguel Hernández

```
        }
      cnx.close();
   }

   public void insertarconjunto (String referencia, String descripcion, int precio) throws SQLException, ClassNotFoundException{
      conectar();
   try (PreparedStatement ps = cnx.prepareStatement("INSERT INTO productos VALUES (NULL, ?, ?, ?,?,?,null,null,null)")) {
         ps.setString(1, referencia);
         ps.setString(2, descripcion);
         ps.setInt(3, precio);
         ps.setBoolean(4, Boolean.TRUE);
         ps.setBoolean(5, Boolean.TRUE);
         ps.executeUpdate();
         ps.close();
      }

   cnx.close();

   }

   public void insertarproductoconjunto (String referenciaprod, String referenciaconj, int cantidad) throws SQLException, ClassNotFoundException{
      conectar();
      PreparedStatement ps = cnx.prepareStatement("INSERT INTO productos_conjuntos VALUES (NULL, ?,?,?)");
         ps.setString(1, referenciaprod);
         ps.setInt(2, cantidad);
         ps.setString(3, referenciaconj);
         ps.executeUpdate();
         ps.close();

         cnx.close();

   }

   public void borrarproductoconjunto (String referenciaprod, String referenciaconj) throws SQLException, ClassNotFoundException{
      conectar();
      PreparedStatement ps = cnx.prepareStatement("DELETE FROM productos WHERE referencia_producto=? AND referencia_conjunt=?");
      ps.setString(1, referenciaprod);
      ps.setString(2, referenciaconj);
      ps.executeUpdate();
      ps.close();
      cnx.close();

   }

   public DefaultTableModel buscarproductosconjuntos (DefaultTableModel mod, String  refodesc) throws SQLException, ClassNotFoundException{
   conectar();
   PreparedStatement ps1;
   System.out.println(" en la base de datos");
      ps1= cnx.prepareStatement("select * from productos_conjuntos where referencia_conjuto LIKE ?");
         //ps1.setString(1, "%"+refodesc+"%");
         ps1.setString(1, refodesc);
         ResultSet rs0 = ps1.executeQuery();
         Object[] row;
         while (rs0.next()){
            row = new Object[3];
            row[0]=  rs0.getString(2);
            row[1]= rs0.getInt(3);
            row[2]= rs0.getString(4);
            mod.addRow(row);
         }
      ps1.close();
      cnx.close();
   return mod;
   }

   public void borrarproductoconjuntoseleccionado (String referencia, int cantidad, String referencia_conjunto) throws SQLException,
ClassNotFoundException{
   conectar();
   PreparedStatement ps = cnx.prepareStatement("DELETE FROM productos_conjuntos WHERE referencia_producto=? AND cantidad_producto=?
AND referencia_conjuto=?");
   ps.setString(1, referencia);
   ps.setInt(2, cantidad);
   ps.setString(3, referencia_conjunto);
   ps.executeUpdate();
   ps.close();
```

```
    cnx.close();
    }


    public void insertarclienteproveedor (String nombre, int cif, String direccion, boolean proveedor, String email, int telefono) throws SQLException,
ClassNotFoundException{
    conectar();
    PreparedStatement ps = cnx.prepareStatement("INSERT INTO cliente_proveedores VALUES (NULL, ?,?,?,?,?,?)");
    ps.setString(1, nombre);
    ps.setInt(2, cif);
    ps.setString(3, direccion);
    ps.setBoolean(4, proveedor);
    ps.setString(5, email);
    ps.setInt(6, telefono);
    ps.executeUpdate();
    ps.close();

    cnx.close();

    }

    public DefaultTableModel buscarclientesproveedor (DefaultTableModel mod, String parte_busqueda, int busqueda) throws SQLException,
ClassNotFoundException{
    conectar();
    PreparedStatement ps1,ps2,ps3;
       ps1= cnx.prepareStatement("select * from cliente_proveedores where nombre LIKE ?");
       ps2= cnx.prepareStatement("select * from cliente_proveedores where cif LIKE ?");
       ps3= cnx.prepareStatement("select * from cliente_proveedores where nombre LIKE ? AND proveedor LIKE true");
       if (busqueda==0){
          ps1.setString(1, "%"+parte_busqueda+"%");
          ResultSet rs0 = ps1.executeQuery();
          Object[] row;
          while (rs0.next()){
             row = new Object[6];
             row[0]=  rs0.getString(2);
             row[1]= rs0.getInt(3);
             row[2]= rs0.getString(4);
             row[3]= rs0.getBoolean(5);
             row[4]= rs0.getString(6);
             row[5]=rs0.getInt(7);
             mod.addRow(row);
          }
       }
       else if (busqueda==1){
       ps2.setString(1, "%"+parte_busqueda+"%");
       ResultSet rs1 = ps2.executeQuery();
       Object[] row1;
       while (rs1.next()){
          row1 = new Object[6];
             row1[0]=  rs1.getString(2);
             row1[1]= rs1.getInt(3);
             row1[2]= rs1.getString(4);
             row1[3]= rs1.getBoolean(5);
             row1[4]= rs1.getString(6);
             row1[5]=rs1.getInt(7);
             mod.addRow(row1);
          }
       }

       else if (busqueda==2){
          System.out.println("estoy en el cif");
       ps3.setString(1, "%"+parte_busqueda+"%");
       ResultSet rs1 = ps3.executeQuery();
       Object[] row1;
       while (rs1.next()){
          row1 = new Object[6];
             row1[0]=  rs1.getString(2);
             row1[1]= rs1.getInt(3);
             row1[2]= rs1.getString(4);
             row1[3]= rs1.getBoolean(5);
             row1[4]= rs1.getString(6);
             row1[5]=rs1.getInt(7);
             mod.addRow(row1);
          }
       }
```

```
        ps1.close();
        ps2.close();
        ps3.close();
        cnx.close();
    return mod;
    }

    public void borrarclienteproveedor(int cif) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps = cnx.prepareStatement("DELETE FROM cliente_proveedores WHERE cif=?");
        ps.setInt(1, cif);
        ps.executeUpdate();
        ps.close();
        cnx.close();
    }

    public void editarnombrecliente (int cif, String nuevonombre) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps = cnx.prepareStatement("UPDATE cliente_proveedores SET nombre=? WHERE cif=?");
        ps.setString(1, nuevonombre);
        ps.setInt(2, cif);
        ps.executeUpdate();
        ps.close();
        cnx.close();
    }

    public void editarcifcliente (int nuevocif, String nombre) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps = cnx.prepareStatement("UPDATE cliente_proveedores SET cif=? WHERE nombre=?");
        ps.setInt(1, nuevocif);
        ps.setString(2, nombre);
        ps.executeUpdate();
        ps.close();
        cnx.close();
    }

    public void crearpedido (String numero, String cliente, String direccion, Boolean garantía, boolean pagado) throws SQLException,
ClassNotFoundException{
        conectar();
        PreparedStatement ps = cnx.prepareStatement("INSERT INTO pedido1 VALUES (NULL,?,?,NULL,?,?,NULL,?,NULL,
NULL,NULL,NULL,NULL,null)");
        ps.setString(1, numero);
        ps.setString(2, cliente);
        ps.setString(3, direccion);
        ps.setBoolean(4, garantía);
        ps.setBoolean(5, pagado);
        ps.executeUpdate();
        ps.close();
        cnx.close();
    }

    public DefaultTableModel buscarpedido (String textobuscar, int busqueda, DefaultTableModel mod) throws SQLException,
ClassNotFoundException{
        conectar();
        PreparedStatement ps1,ps2;
            ps1= cnx.prepareStatement("select * from pedido1 where numero LIKE ?");
            ps2= cnx.prepareStatement("select * from pedido1 where cliente LIKE ?");
            if (busqueda==0){
            ps1.setString(1, "%"+textobuscar+"%");
            ResultSet rs0 = ps1.executeQuery();
            Object[] row;
                while (rs0.next()){
                row = new Object[11];
                row[0]=  rs0.getString(2);
                row[1]= rs0.getString(3);
                row[2]= rs0.getDouble(4);
                row[3]= rs0.getString(5);
                row[4]= rs0.getBoolean(6);
                row[5]=rs0.getBoolean(7);
                row[6]=rs0.getBoolean(8);
                row[7]=rs0.getBoolean(9);
                row[8]=  rs0.getString(10);
                row[9]=rs0.getBoolean(11);
                row[10]=  rs0.getString(12);
                mod.addRow(row);
```

```
            }
          }
          else{
          ps2.setString(1, "%"+textobuscar+"%");
          ResultSet rs1 = ps2.executeQuery();
          Object[] row1;
          while (rs1.next()){
             row1 = new Object[11];
             row1[0]=  rs1.getString(2);
             row1[1]= rs1.getString(3);
             row1[2]= rs1.getDouble(4);
             row1[3]= rs1.getString(5);
             row1[4]= rs1.getBoolean(6);
             row1[5]=rs1.getBoolean(7);
             row1[6]=rs1.getBoolean(8);
             row1[7]=rs1.getBoolean(9);
             row1[8]=  rs1.getString(10);
             row1[9]=rs1.getBoolean(11);
             row1[10]=  rs1.getString(12);
             mod.addRow(row1);
             }
          }
       return mod;
       }

    public void editarpedido (int combo, boolean marcado, String numero) throws SQLException, ClassNotFoundException{
       conectar();
       if (combo==0){
          PreparedStatement ps = cnx.prepareStatement("UPDATE pedido1 SET garantía=? WHERE numero=?");
          ps.setBoolean(1, marcado);
          ps.setString(2, numero);
          ps.executeUpdate();
          ps.close();

       }
       else if (combo==1){
          PreparedStatement ps = cnx.prepareStatement("UPDATE pedido1 SET pagado=? WHERE numero=?");
          ps.setBoolean(1, marcado);
          ps.setString(2, numero);
          ps.executeUpdate();
          ps.close();
       }

       else {
          PreparedStatement ps = cnx.prepareStatement("UPDATE pedido1 SET tramitado=false WHERE numero=?");
          ps.setString(1, numero);
          ps.executeUpdate();
          ps.close();

       }
       cnx.close();
    }

    public void borrarpedido (String numero) throws SQLException, ClassNotFoundException{
       conectar();
       PreparedStatement ps = cnx.prepareStatement("DELETE FROM pedido1 WHERE numero=?");
       ps.setString(1, numero);
       ps.executeUpdate();
       ps.close();
       cnx.close();
    }

    public void instarproductospedido (String numeroped, String referencia, int cantidad, int descuento) throws SQLException,
ClassNotFoundException{
       conectar();


    }

    public DefaultTableModel calcularprecio (String referencia, int cantidad, int descuento, String pedido, DefaultTableModel mod, Boolean garantía)
throws SQLException, ClassNotFoundException{
       conectar();
       PreparedStatement ps,ps3;
        ps= cnx.prepareStatement("SELECT precio FROM productos WHERE referencia LIKE ?");
        ps.setString(1, referencia);
```

```
        ResultSet rs1 = ps.executeQuery();

        while (rs1.next()){
            double precio = rs1.getInt(1);
            System.out.println(precio);
            double precioxcantidadsd= precio*cantidad;
            System.out.println(precioxcantidadsd);
            double descuento1 = (descuento)*0.01;
            System.out.println(descuento1);
            double precioxcantidad = precioxcantidadsd - (descuento1*precioxcantidadsd);
            System.out.println(precioxcantidad);

            ps3= cnx.prepareStatement("INSERT INTO pedido1_producto VALUES (NULL, ?, ?, ?, ?, ?, ?,?)");
            ps3.setString(1, pedido);
            ps3.setString(2,referencia);
            ps3.setInt(3,cantidad);
            ps3.setDouble(4, precioxcantidadsd);
            ps3.setDouble(5, descuento);
            ps3.setDouble(6, precioxcantidad);
            ps3.setBoolean(7, garantía);
            ps3.executeUpdate();

            ps3.close();
            Object row[];
            row = new Object[7];
            row[0]= pedido;
            row[1]= referencia;
            row[2]=cantidad;
            row[3]=precioxcantidadsd;
            row[4]=descuento;
            row[5]=precioxcantidad;
            row[6]=garantía;
            mod.addRow(row);
            insertarpreciopedido(pedido);
            System.out.println(pedido);
        }
        ps.close();
        return mod;
    }

    public DefaultTableModel buscarprodpedido (String pedido, DefaultTableModel mod) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps1;
        ps1= cnx.prepareStatement("select * from pedido1_producto where pedido LIKE ?");
            ps1.setString(1, "%"+pedido+"%");
            ResultSet rs0 = ps1.executeQuery();
            Object[] row;
            while (rs0.next()){
                row = new Object[7];
                row[0]=  rs0.getString(2);
                row[1]= rs0.getString(3);
                row[2]= rs0.getInt(4);
                row[3]=rs0.getDouble(5);
                row[4]= rs0.getDouble(6);
                row[5]= rs0.getDouble(7);
                row[6]=rs0.getBoolean(8);
                mod.addRow(row);

            }
        return mod;
    }

    public void insertarpreciopedido (String referenciaped) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps2;
        ps2= cnx.prepareStatement("UPDATE pedido1 SET precio=(SELECT SUM(preciofinal) FROM tfm.pedido1_producto WHERE pedido=?)
WHERE numero=?");
        ps2.setString(1, referenciaped);
        ps2.setString(2, referenciaped);
        ps2.executeUpdate();
        ps2.close();
    }

    public void borrarproductodelpedido (String referenciaped, int numero) throws SQLException, ClassNotFoundException{
        conectar();
```

```
    PreparedStatement ps;
    ps = cnx.prepareStatement("DELETE FROM pedido1_producto WHERE referencia=? AND cantidad=? LIMIT 1");
    ps.setString(1, referenciaped);
    ps.setInt(2, numero);
    ps.executeUpdate();
    ps.close();
    cnx.close();
}

public void insertarfecha (String lct) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps = cnx.prepareStatement("INSERT INTO fechas VALUES (NULL,?)");
    ps.setString(1, lct);
    ps.executeUpdate();
    ps.close();
    cnx.close();
}

public String buscarfecha () throws SQLException, ClassNotFoundException{
String cadenalct = null;
conectar();
PreparedStatement ps1;
ps1= cnx.prepareStatement("select * from tfm.fechas");
ResultSet rs0 = ps1.executeQuery();
 while (rs0.next()){
cadenalct = rs0.getString(2);
 }
return cadenalct;
}

public int buscargarantias (String referencia) throws SQLException, ClassNotFoundException{
int numero=0;
conectar();
PreparedStatement ps1;
//ps1= cnx.prepareStatement("select conut(*) from tfm.pedido1_producto WHERE referencia = ? AND garantia=TRUE");
ps1= cnx.prepareStatement("select sum(cantidad) from tfm.pedido1_producto WHERE referencia = ? AND garantia=TRUE");
ps1.setString(1, referencia);
ResultSet rs0 = ps1.executeQuery();

 while (rs0.next()){
 numero = rs0.getInt(1);
 }

return numero;
}

public int buscargarantiast() throws SQLException, ClassNotFoundException{
int total=0;
conectar();
PreparedStatement ps1;
ps1= cnx.prepareStatement("select sum(cantidad) from tfm.pedido1_producto WHERE garantia=TRUE");
ResultSet rs0 = ps1.executeQuery();
 while (rs0.next()){
 total = rs0.getInt(1);
 }
return total;
}

public void introducirprodalm (String ref, String descr) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps = cnx.prepareStatement("INSERT INTO productos_almacen VALUES (NULL,?,?,0,0,0, 0)");
    ps.setString(1, ref);
    ps.setString(2, descr);
    ps.executeUpdate();
    ps.close();
    cnx.close();

}


public DefaultTableModel buscarprodalm (String ref, DefaultTableModel mod) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps1;
    ps1= cnx.prepareStatement("select * from productos_almacen where referencia LIKE ?");
       ps1.setString(1, "%"+ref+"%");
```

```
        ResultSet rs0 = ps1.executeQuery();
        Object[] row;
        while (rs0.next()){
           row = new Object[6];
           row[0]=  rs0.getString(2);
           row[1]= rs0.getString(3);
           row[2]= rs0.getInt(4);
           row[3]=rs0.getString(5);
           row[4]= rs0.getString(6);
           row[5]= rs0.getInt(7);
           mod.addRow(row);
        }
    return mod;
    }

public DefaultTableModel buscarproddescralm (String descr, DefaultTableModel mod) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps1;
    ps1= cnx.prepareStatement("select * from productos_almacen where descripcion LIKE ?");
        ps1.setString(1, "%"+descr+"%");
        ResultSet rs0 = ps1.executeQuery();
        Object[] row;
        while (rs0.next()){
           row = new Object[6];
           row[0]=  rs0.getString(2);
           row[1]= rs0.getString(3);
           row[2]= rs0.getInt(4);
           row[3]=rs0.getString(5);
           row[4]= rs0.getString(6);
           row[5]= rs0.getInt(7);
           mod.addRow(row);
        }
    return mod;
    }

public void buscarcantidadproducto (String referencia, int cantidad, int intext, String fecha) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps1;
    ps1= cnx.prepareStatement("select cantidad_almacen from productos_almacen where referencia LIKE ?");
        ps1.setString(1, "%"+referencia+"%");
        ResultSet rs0 = ps1.executeQuery();
        Object[] row;
        while (rs0.next()){
           int a =0;
           a=  rs0.getInt(1);
           System.out.println(a);
           System.out.println(intext);
           if (intext ==0){
              introduciralmacen(referencia, cantidad, fecha, a);
           }
           else if (intext ==1){
              sacaralmacen(referencia, cantidad, fecha, a);
           }
        }
    }

public void introduciralmacen (String referencia, int cantidad, String fecha_entrada, int sumar) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps2;
    System.out.println(cantidad);
    System.out.println(sumar);
    int total = cantidad + sumar;
    ps2= cnx.prepareStatement("UPDATE productos_almacen SET cantidad_almacen=?, fecha_entrada=? WHERE referencia=?");
    ps2.setInt(1, total);
    ps2.setString(2, fecha_entrada);
    ps2.setString(3, referencia);
    ps2.executeUpdate();
    ps2.close();
}

public void sacaralmacen (String referencia, int restar, String fecha_salida, int cantidad) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps2;
    System.out.println(cantidad);
    System.out.println(restar);
```



UNIVERSITAS Miguel Hernández

```
        int total = cantidad-restar;
        ps2= cnx.prepareStatement("UPDATE productos_almacen SET cantidad_almacen=?, fecha_salidad=? WHERE referencia=?");
        ps2.setInt(1, total);
        ps2.setString(2, fecha_salida);
        ps2.setString(3, referencia);
        ps2.executeUpdate();
        ps2.close();
        cnx.close();
    }

    public void borrarproductoalmacen (String referencia, int numero) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps;
        System.out.println(referencia);
        System.out.println(numero);
        ps = cnx.prepareStatement("DELETE FROM productos_almacen WHERE referencia=? AND cantidad_almacen=? LIMIT 1");
        ps.setString(1, referencia);
        ps.setInt(2, numero);
        ps.executeUpdate();
        ps.close();
        cnx.close();
    }

    public void stockseguridad (String referencia, int stock) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps2;
        ps2= cnx.prepareStatement("UPDATE productos_almacen SET stock_seguridad=? A WHERE referencia=?");
        ps2.setInt(1, stock);
        ps2.setString(2, referencia);
        ps2.executeUpdate();
        ps2.close();
    }

    public DefaultTableModel buscarpedidononotratado (DefaultTableModel mod) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps1;
        ps1= cnx.prepareStatement("select numero, tramitado from pedido1  where tramitado is null or tramitado is false");
            ResultSet rs0 = ps1.executeQuery();
            Object[] row;
            while (rs0.next()){
                row = new Object[2];
                row[0]=  rs0.getString(1);
                row[1]= rs0.getBoolean(2);
                mod.addRow(row);
            }
            ps1.close();
            cnx.close();
        return mod;
    }

    public void tramitado (String referencia) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps1;
        ps1= cnx.prepareStatement("UPDATE pedido1 SET tramitado=TRUE WHERE numero=?");
        ps1.setString(1, referencia);
        ps1.executeUpdate();
        ps1.close();
        cnx.close();
    }

    public void crearorden (String orden, String pedido, String fechaalmacen, String fechasal, int etapas) throws SQLException,
ClassNotFoundException{
        conectar();
        PreparedStatement ps = cnx.prepareStatement("INSERT INTO orden VALUES (NULL,?,?,?,?,?, NULL, NULL)");
        ps.setString(1, orden);
        ps.setString(2, pedido);
        ps.setString(3, fechaalmacen);
        ps.setString(4, fechasal);
        ps.setInt(5, etapas);
        ps.executeUpdate();
        ps.close();
        cnx.close();
    }

    public DefaultTableModel buscarorden (DefaultTableModel mod, String orden) throws SQLException, ClassNotFoundException{
```

```
        conectar();
        PreparedStatement ps1;
        ps1= cnx.prepareStatement("select numero_orden, pedido, fecha_sacar_almacen, fecha_prevista_salida from orden where numero_orden LIKE
?");
            ps1.setString(1, "%"+orden+"%");
         ResultSet rs0 = ps1.executeQuery();
            Object[] row;
            while (rs0.next()){
               row = new Object[4];
               row[0]=  rs0.getString(1);
               row[1]=  rs0.getString(2);
               row[2]= rs0.getString(3);
               row[3]= rs0.getString(4);
               mod.addRow(row);
            }
            ps1.close();
            cnx.close();
        return mod;
    }

    public void borrarorden (String orden) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps;
        ps = cnx.prepareStatement("DELETE FROM orden WHERE numero_orden=? LIMIT 1");
        ps.setString(1, orden);
        ps.executeUpdate();
        ps.close();
        cnx.close();

    }

    public DefaultTableModel buscarportrabajador (DefaultTableModel mod, String id_trabajador) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps;
        ps = cnx.prepareStatement("SELECT * FROM gestion_trabajadores WHERE id_trabajador LIKE ?");
        ps.setString(1, "%"+id_trabajador+"%");
        ResultSet rs0 = ps.executeQuery();
            Object[] row;
            while (rs0.next()){
               row = new Object[4];
               row[0]=  rs0.getString(2);
               row[1]= rs0.getString(3);
               row[2]= rs0.getString(4);
               row[3]= rs0.getString(5);
               mod.addRow(row);
            }
        ps.close();
        cnx.close();
        return mod;
    }

    public DefaultTableModel buscarpororden gestion(DefaultTableModel mod, String id_orden) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps;
        ps = cnx.prepareStatement("SELECT * FROM gestion_trabajadores WHERE id_orden LIKE ?");
        ps.setString(1, "%"+id_orden+"%");
        ResultSet rs0 = ps.executeQuery();
        Object[] row;
            while (rs0.next()){
               row = new Object[4];
               row[0]=  rs0.getString(2);
               row[1]= rs0.getString(3);
               row[2]= rs0.getString(4);
               row[3]= rs0.getString(5);
               mod.addRow(row);
            }
        ps.close();
        cnx.close();
        return mod;
    }

    public void insertarlineatrabajotrab (String id_trabajador, String id_orden, String fecha_inicio) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps;
        ps= cnx.prepareStatement("INSERT INTO gestion_trabajadores VALUES (NULL,?,?,?, NULL)");
```

```
        ps.setString(1, id_trabajador);
        ps.setString(2, id_orden);
        ps.setString(3, fecha_inicio);
        ps.executeUpdate();
        ps.close();
        cnx.close();
    }

    public void editargestiontrabajadororden (String id_trabajador, String id_orden, String fecha_final) throws SQLException,
ClassNotFoundException{
        conectar();
        PreparedStatement ps1;
        ps1= cnx.prepareStatement("UPDATE gestion_trabajadores SET fecha_final=? WHERE id_trabajador=? and id_orden=?");
        ps1.setString(1, fecha_final);
        ps1.setString(2, id_trabajador);
        ps1.setString(3, id_orden);
        ps1.executeUpdate();
        ps1.close();
        cnx.close();

    }

    public void crearordenproduccion (String id_orden,String pedido, int num_empleados, int etapas) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps;
        ps= cnx.prepareStatement("INSERT INTO gestion_produccion VALUES (NULL,?,?,?,?,
NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL)");
        ps.setString(1, id_orden);
        ps.setString(2, pedido);
        ps.setInt(3,num_empleados);
        ps.setInt(4, etapas);
        ps.executeUpdate();
        ps.close();
        cnx.close();
    }

    public void empezarordenproduccion (int numero_etapa, String inicio, String id_orden) throws SQLException, ClassNotFoundException{
        conectar();
        PreparedStatement ps;
        if (numero_etapa ==0){
            ps= cnx.prepareStatement("UPDATE gestion_produccion SET inicio1=? WHERE id_orden=?");
            ps.setString(1, inicio);
            ps.setString(2,id_orden);
            ps.executeUpdate();
            ps.close();
        }
        else if (numero_etapa ==1){
            ps= cnx.prepareStatement("UPDATE gestion_produccion SET inicio2=? WHERE id_orden=?");
            ps.setString(1, inicio);
            ps.setString(2,id_orden);
            ps.executeUpdate();
            ps.close();
        }
        else if (numero_etapa ==2){
            ps= cnx.prepareStatement("UPDATE gestion_produccion SET inicio3=? WHERE id_orden=?");
            ps.setString(1, inicio);
            ps.setString(2,id_orden);
            ps.executeUpdate();
            ps.close();
        }
        else if (numero_etapa ==3){
            ps= cnx.prepareStatement("UPDATE gestion_produccion SET inicio4=? WHERE id_orden=?");
            ps.setString(1, inicio);
            ps.setString(2,id_orden);
            ps.executeUpdate();
            ps.close();
        }
        else if (numero_etapa ==4){
            ps= cnx.prepareStatement("UPDATE gestion_produccion SET inicio5=? WHERE id_orden=?");
            ps.setString(1, inicio);
            ps.setString(2,id_orden);
            ps.executeUpdate();
            ps.close();
        }
        cnx.close();
```

```
}

public void cerrarodenproduccion(int numero_etapa, String etapafinal, String id_orden) throws SQLException, ClassNotFoundException{
   conectar();
   PreparedStatement ps;
   if (numero_etapa ==0){
      ps= cnx.prepareStatement("UPDATE gestion_produccion SET final1=? WHERE id_orden=?");
      ps.setString(1, etapafinal);
      ps.setString(2,id_orden);
      ps.executeUpdate();
      ps.close();
   }
   if (numero_etapa ==1){
      ps= cnx.prepareStatement("UPDATE gestion_produccion SET final2=? WHERE id_orden=?");
      ps.setString(1, etapafinal);
      ps.setString(2,id_orden);
      ps.executeUpdate();
      ps.close();
   }
   if (numero_etapa ==2){
      ps= cnx.prepareStatement("UPDATE gestion_produccion SET final3=? WHERE id_orden=?");
      ps.setString(1, etapafinal);
      ps.setString(2,id_orden);
      ps.executeUpdate();
      ps.close();
   }
   if (numero_etapa ==3){
      ps= cnx.prepareStatement("UPDATE gestion_produccion SET final4=? WHERE id_orden=?");
      ps.setString(1, etapafinal);
      ps.setString(2,id_orden);
      ps.executeUpdate();
      ps.close();
   }
   if (numero_etapa ==4){
      ps= cnx.prepareStatement("UPDATE gestion_produccion SET final5=? WHERE id_orden=?");
      ps.setString(1, etapafinal);
      ps.setString(2,id_orden);
      ps.executeUpdate();
      ps.close();
   }
   cnx.close();
}

public DefaultTableModel buscarordenesgestion (DefaultTableModel mod, String id_orden) throws SQLException, ClassNotFoundException{
   conectar();
   PreparedStatement ps;
   ps = cnx.prepareStatement("SELECT * FROM gestion_produccion WHERE id_orden LIKE ?");
   ps.setString(1, "%"+id_orden+"%");
   ResultSet rs0 = ps.executeQuery();
   Object[] row;
      while (rs0.next()){
         row = new Object[16];
         row[0]=  rs0.getString(2);
         row[1] = rs0.getString(3);
         row[2] = rs0.getInt(4);
         row[3] = rs0.getInt(5);
         row[4]=  rs0.getString(6);
         row[5]=  rs0.getString(7);
         row[6]=  rs0.getString(8);
         row[7]=  rs0.getString(9);
         row[8]=  rs0.getString(10);
         row[9]=  rs0.getString(11);
         row[10]=  rs0.getString(12);
         row[11]=  rs0.getString(13);
         row[12]=  rs0.getString(14);
         row[13]=  rs0.getString(15);
         row[14]=  rs0.getBoolean(16);
         row[15]= rs0.getString(17);
         mod.addRow(row);
      }
   ps.close();
   cnx.close();
return mod;
}
```

```
public DefaultTableModel buscarpedidosgestion (DefaultTableModel mod, String pedido) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("SELECT * FROM gestion_produccion WHERE pedido LIKE ?");
    ps.setString(1, "%"+pedido+"%");
    ResultSet rs0 = ps.executeQuery();
    Object[] row;
        while (rs0.next()){
            row = new Object[16];
            row[0]=  rs0.getString(2);
            row[1] = rs0.getString(3);
            row[2] = rs0.getInt(4);
            row[3] = rs0.getInt(5);
            row[4]=  rs0.getString(6);
            row[5]=  rs0.getString(7);
            row[6]=  rs0.getString(8);
            row[7]=  rs0.getString(9);
            row[8]=  rs0.getString(10);
            row[9]=  rs0.getString(11);
            row[10]=  rs0.getString(12);
            row[11]=  rs0.getString(13);
            row[12]=  rs0.getString(14);
            row[13]=  rs0.getString(15);
            row[14]=  rs0.getBoolean(16);
            row[15]= rs0.getString(17);
            mod.addRow(row);
        }
    ps.close();
    cnx.close();
 return mod;
 }

public DefaultTableModel buscaretapasgestion (DefaultTableModel mod, int etapas) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("SELECT * FROM gestion_produccion WHERE etapas LIKE ?");
    ps.setString(1, "%"+etapas+"%");
    ResultSet rs0 = ps.executeQuery();
    Object[] row;
        while (rs0.next()){
            row = new Object[15];
            row[0]=  rs0.getString(2);
            row[1] = rs0.getString(3);
            row[2] = rs0.getInt(4);
            row[3] = rs0.getInt(5);
            row[4]=  rs0.getString(6);
            row[5]=  rs0.getString(7);
            row[6]=  rs0.getString(8);
            row[7]=  rs0.getString(9);
            row[8]=  rs0.getString(10);
            row[9]=  rs0.getString(11);
            row[10]=  rs0.getString(12);
            row[11]=  rs0.getString(13);
            row[12]=  rs0.getString(14);
            row[13]=  rs0.getString(15);
            row[14]=  rs0.getString(16);
            mod.addRow(row);
        }
    ps.close();
    cnx.close();
 return mod;
 }

public void notagestion (String nota, String id_orden) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("UPDATE gestion_produccion SET notas=? WHERE id_orden=?");
    ps.setString(1, nota);
    ps.setString(2, id_orden);
    ps.executeUpdate();
    ps.close();
    cnx.close();
}

public void borrargestion (String id_orden) throws SQLException, ClassNotFoundException{
```

```
    conectar();
    PreparedStatement ps;
    System.out.println(id_orden);
    ps = cnx.prepareStatement("DELETE FROM gestion_produccion WHERE id_orden=? LIMIT 1");
    ps.setString(1, id_orden);
    ps.executeUpdate();
    ps.close();
    cnx.close();
  }

  public void borrargestiontrab (String id_trabajador, String fecha_inicio) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    System.out.println(id_trabajador);
    ps = cnx.prepareStatement("DELETE FROM gestion_trabajadores WHERE id_trabajador=? and fecha_inicio=?");
    ps.setString(1, id_trabajador);
    ps.setString(2, fecha_inicio);
    ps.executeUpdate();
    ps.close();
    cnx.close();
  }


  public void crearinteraccionsinfallo (String id_interaccion, String cliente, String smotivo, String notas) throws SQLException,
ClassNotFoundException{
      conectar();
      PreparedStatement ps;
      ps= cnx.prepareStatement("INSERT INTO interaccion VALUES (NULL,?,?,?,NULL,?)");
      ps.setString(1, id_interaccion);
      ps.setString(2,cliente);
      ps.setString(3, smotivo);
      ps.setString(4, notas);
      ps.executeUpdate();
      ps.close();
      cnx.close();
   }

  public void crearinteraccionconfallo (String id_interaccion, String cliente, String smotivo, String tipo, String notas) throws SQLException,
ClassNotFoundException{
      conectar();
      PreparedStatement ps;
      System.out.print("dentro de la base de datos");
      ps= cnx.prepareStatement("INSERT INTO interaccion VALUES (NULL,?,?,?,?,?)");
      ps.setString(1, id_interaccion);
      ps.setString(2,cliente);
      ps.setString(3, smotivo);
      ps.setString(4, tipo);
      ps.setString(5, notas);
      ps.executeUpdate();
      ps.close();
      cnx.close();
   }

  public DefaultTableModel buscarporclienteinteraccion(DefaultTableModel mod, String cliente) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
     ps = cnx.prepareStatement("SELECT * FROM interaccion WHERE cliente LIKE ?");
     ps.setString(1, "%"+cliente+"%");
     ResultSet rs0 = ps.executeQuery();
     Object[] row;
        while (rs0.next()){
           row = new Object[5];
           row[0]=  rs0.getString(2);
           row[1]= rs0.getString(3);
           row[2]= rs0.getString(4);
           row[3]= rs0.getString(5);
           row[4]=rs0.getString(6);
           mod.addRow(row);
        }
     ps.close();
     cnx.close();
     return mod;

  }
```

```
public DefaultTableModel buscarmotivointeraccion (DefaultTableModel mod, int motivo) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
     ps = cnx.prepareStatement("SELECT * FROM interaccion WHERE motivo LIKE ?");
     ps.setInt(1, motivo);
     ResultSet rs0 = ps.executeQuery();
     Object[] row;
        while (rs0.next()){
           row = new Object[5];
           row[0]=  rs0.getString(2);
           row[1]= rs0.getString(3);
           row[2]= rs0.getInt(4);
           row[3]= rs0.getString(5);
           row[4]=rs0.getString(6);
           mod.addRow(row);
        }
     ps.close();
     cnx.close();
     return mod;

}

public DefaultTableModel buscarporclientemotivointeraccion(DefaultTableModel mod, String cliente, int motivo) throws SQLException,
ClassNotFoundException{
    conectar();
    PreparedStatement ps;
     ps = cnx.prepareStatement("SELECT * FROM interaccion WHERE cliente LIKE ? AND motivo LIKE ?");
     ps.setString(1, "%"+cliente+"%");
     ps.setInt(2, motivo);
     ResultSet rs0 = ps.executeQuery();
     Object[] row;
        while (rs0.next()){
           row = new Object[5];
           row[0]=  rs0.getString(2);
           row[1]= rs0.getString(3);
           row[2]= rs0.getInt(4);
           row[3]= rs0.getString(5);
           row[4]=rs0.getString(6);
           mod.addRow(row);
        }
     ps.close();
     cnx.close();
     return mod;

}

public void borrarinteraccion (String id) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("DELETE FROM interaccion WHERE id_interaccion=? LIMIT 1");
     ps.setString(1, id);
     ps.executeUpdate();
     ps.close();
     cnx.close();
}

public void introducirnota (String interaccion, int nota, String cliente, String fecha) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps= cnx.prepareStatement("INSERT INTO notas VALUES (NULL,?,?,?,?)");
    ps.setString(1, interaccion);
    ps.setString(2,cliente);
    ps.setInt(3,nota);
    ps.setString(4,fecha);
    ps.executeUpdate();
    ps.close();
     cnx.close();

}

public int total (String cliente) throws SQLException, ClassNotFoundException{
    conectar();
    int numtotal=0;
    PreparedStatement ps;
    ps= cnx.prepareStatement("SELECT COUNT(*)FROM notas WHERE cliente=?");
```

```
    ps.setString(1,cliente );
    ResultSet rs0 = ps.executeQuery();
    while (rs0.next()){
     numtotal = rs0.getInt(1);
    }
    return numtotal;
}

public int suma (String cliente) throws SQLException, ClassNotFoundException{
    conectar();
    int numero=0;
    PreparedStatement ps;
     ps= cnx.prepareStatement("select sum(nota) from notas WHERE cliente =?");
     ps.setString(1, cliente);
     ResultSet rs0 = ps.executeQuery();

     while (rs0.next()){
        numero = rs0.getInt(1);
     }
     return numero;
}

public DefaultTableModel buscarclientenota (DefaultTableModel mod, String cliente) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("SELECT * FROM notas WHERE cliente LIKE ?");
    ps.setString(1, "%"+cliente+"%");

    ResultSet rs0 = ps.executeQuery();
    Object[] row;
       while (rs0.next()){
          row = new Object[4];
          row[0]=  rs0.getString(2);
          row[1]=rs0.getString(3);
          row[2]= rs0.getInt(4);
          row[3]= rs0.getString(5);
          mod.addRow(row);
       }
    ps.close();
    cnx.close();
    return mod;
}

public void borrarnota (String fecha) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
   //System.out.println(id);
    System.out.println(fecha);
    ps = cnx.prepareStatement("DELETE FROM notas WHERE  fecha=?");
     //ps.setString(1, id);
     ps.setString(1, fecha);
     ps.executeUpdate();
     ps.close();
     cnx.close();
}

public int[] contarmotivocliente (String cliente) throws SQLException, ClassNotFoundException{
    conectar();
    int[] resultados = {0,0,0,0,0,0};
    PreparedStatement ps0,ps1,ps2,ps3,ps4,ps5;
    ps0= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Garantía' and cliente=?");
    ps0.setString(1,cliente );
    ResultSet rs0 = ps0.executeQuery();
    while (rs0.next()){
     resultados[0] = rs0.getInt(1);
    }
    ps1= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Repuestos' and cliente=?");
    ps1.setString(1,cliente );
    ResultSet rs1 = ps1.executeQuery();
    while (rs1.next()){
     resultados[1] = rs1.getInt(1);
    }
```

Biblioteca UNIVERSITAS Miguel Hernández

```
    ps2= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Solicitar información' and cliente=?");
    ps2.setString(1,cliente );
    ResultSet rs2 = ps2.executeQuery();
    while (rs2.next()){
     resultados[2] = rs2.getInt(1);
    }
    ps3= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Reportar Fallo' and cliente=?");
    ps3.setString(1,cliente);
    ResultSet rs3 = ps3.executeQuery();
    while (rs3.next()){
     resultados[3] = rs3.getInt(1);
     System.out.println(resultados[3]);
    }


    ps4= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Solicitar Documentación' and cliente=?");
    ps4.setString(1,cliente );
    ResultSet rs4 = ps4.executeQuery();
    while (rs4.next()){
     resultados[4] = rs4.getInt(1);
    }
    ps5= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Quejas' and cliente=?");
    ps5.setString(1,cliente );
    ResultSet rs5 = ps5.executeQuery();
    while (rs5.next()){
     resultados[5] = rs5.getInt(1);
    }

    return resultados;
}

public int totalcliente (String cliente) throws SQLException, ClassNotFoundException{
    conectar();
    int total=0;
    PreparedStatement ps0;
    ps0= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE cliente=?");
    ps0.setString(1,cliente);
    ResultSet rs0 = ps0.executeQuery();
    while (rs0.next()){
     total = rs0.getInt(1);
    }
    return total;
}

public int[] contarmotivo () throws SQLException, ClassNotFoundException{
    conectar();
     conectar();
    int[] resultados = {0,0,0,0,0,0};
    PreparedStatement ps0,ps1,ps2,ps3,ps4,ps5;
    ps0= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Garantía'");

    ResultSet rs0 = ps0.executeQuery();
    while (rs0.next()){
     resultados[0] = rs0.getInt(1);
    }
    ps1= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Repuestos'");

    ResultSet rs1 = ps1.executeQuery();
    while (rs1.next()){
     resultados[1] = rs1.getInt(1);
    }



    ps2= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Solicitar información'");
    ResultSet rs2 = ps2.executeQuery();
    while (rs2.next()){
     resultados[2] = rs2.getInt(1);
    }
    ps3= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Reportar Fallo'");

    ResultSet rs3 = ps3.executeQuery();
    while (rs3.next()){
     resultados[3] = rs3.getInt(1);
     System.out.println(resultados[3]);
```

```
    }


    ps4= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Solicitar Documentación'");

    ResultSet rs4 = ps4.executeQuery();
    while (rs4.next()){
     resultados[4] = rs4.getInt(1);
    }
    ps5= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion WHERE motivo='Quejas'");

    ResultSet rs5 = ps5.executeQuery();
    while (rs5.next()){
     resultados[5] = rs5.getInt(1);
    }

    return resultados;
  }

  public int totalsincliente () throws SQLException, ClassNotFoundException{
    conectar();
    int total=0;
    PreparedStatement ps0;
    ps0= cnx.prepareStatement("SELECT COUNT(*)FROM interaccion");
    ResultSet rs0 = ps0.executeQuery();
    while (rs0.next()){
     total = rs0.getInt(1);
    }
    return total;
  }

  public DefaultTableModel buscarordenalmacen (DefaultTableModel mod, String orden) throws SQLException, ClassNotFoundException{
      conectar();
      PreparedStatement ps1;
      ps1= cnx.prepareStatement("select numero_orden, pedido, fecha_sacar_almacen, fecha_prevista_salida, sacado, terminado from orden where
numero_orden LIKE ?");
        ps1.setString(1, "%"+orden+"%");
       ResultSet rs0 = ps1.executeQuery();
        Object[] row;
        while (rs0.next()){
          row = new Object[6];
          row[0]=  rs0.getString(1);
          row[1]= rs0.getString(2);
          row[2]= rs0.getString(3);
          row[3]= rs0.getString(4);
          row[4]= rs0.getBoolean(5);
          row[5]= rs0.getBoolean(6);
          mod.addRow(row);
        }
        ps1.close();
        cnx.close();
      return mod;
  }

   public DefaultTableModel buscarordensacar(DefaultTableModel mod) throws SQLException, ClassNotFoundException{
      conectar();
      PreparedStatement ps1;
      ps1= cnx.prepareStatement("select numero_orden, pedido, fecha_sacar_almacen, fecha_prevista_salida, sacado, terminado from orden where
sacado is null or sacado is false");
       ResultSet rs0 = ps1.executeQuery();
        Object[] row;
        while (rs0.next()){
          row = new Object[6];
          row[0]=  rs0.getString(1);
          row[1]= rs0.getString(2);
          row[2]= rs0.getString(3);
          row[3]= rs0.getString(4);
          row[4]= rs0.getBoolean(5);
          row[5]= rs0.getBoolean(6);
          mod.addRow(row);
        }
        ps1.close();
        cnx.close();
      return mod;
  }
```

```
  public void ordensacaralm (String id_orden) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("UPDATE orden SET sacado=TRUE WHERE numero_orden=?");
    ps.setString(1, id_orden);
    ps.executeUpdate();
    ps.close();
    cnx.close();
  }

  public void ordensacardeshacer (String id_orden) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("UPDATE orden SET sacado=FALSE WHERE numero_orden=?");
    ps.setString(1, id_orden);
    ps.executeUpdate();
    ps.close();
    cnx.close();
  }

  public void terminarorden (String id_orden) throws SQLException, ClassNotFoundException{
     conectar();
    PreparedStatement ps, ps1;
    ps = cnx.prepareStatement("UPDATE orden SET terminado=TRUE WHERE numero_orden=?");
    ps.setString(1, id_orden);
    ps.executeUpdate();
    ps.close();
    ps1 = cnx.prepareStatement("UPDATE gestion_produccion SET oterminada=TRUE WHERE id_orden=?");
    ps1.setString(1, id_orden);
    ps1.executeUpdate();
    ps1.close();
    cnx.close();
  }

  public void terminarpedido (String pedido) throws SQLException, ClassNotFoundException{
     conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("UPDATE pedido1 SET terminado=TRUE WHERE numero=?");
    ps.setString(1, pedido);
    ps.executeUpdate();
    ps.close();
    cnx.close();
  }


  public void terminarordendesh (String id_orden) throws SQLException, ClassNotFoundException{
     conectar();
    PreparedStatement ps,ps1;
    ps = cnx.prepareStatement("UPDATE orden SET terminado=FALSE WHERE numero_orden=?");
    ps.setString(1, id_orden);
    ps.executeUpdate();
    ps.close();
    ps1 = cnx.prepareStatement("UPDATE gestion_produccion SET oterminada=FALSE WHERE id_orden=?");
    ps1.setString(1, id_orden);
    ps1.executeUpdate();
    ps1.close();
    cnx.close();
  }

  public void terminarpedidodesh (String pedido) throws SQLException, ClassNotFoundException{
     conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("UPDATE pedido1 SET terminado=FFALSE WHERE numero=?");
    ps.setString(1, pedido);
    ps.executeUpdate();
    ps.close();
    cnx.close();
  }

  public DefaultTableModel buscarterminado (DefaultTableModel mod) throws SQLException, ClassNotFoundException{
    conectar();
    PreparedStatement ps;
    ps = cnx.prepareStatement("select numero, terminado, enviado, fechaenviado from pedido1 where terminado is true and enviado is false\n" +
"union\n" +
```

```
"select numero, terminado, enviado, fechaenviado from pedido1 where terminado is true and enviado is null");
      ResultSet rs0 = ps.executeQuery();
          Object[] row;
          while (rs0.next()){
             row = new Object[4];
             row[0]=  rs0.getString(1);
             row[1]= rs0.getBoolean(2);
             row[2]= rs0.getBoolean(3);
             row[3]= rs0.getString(4);
             mod.addRow(row);
          }
       ps.close();
       cnx.close();
       return mod;
   }

   public DefaultTableModel buscarterminadoyenviado (DefaultTableModel mod) throws SQLException, ClassNotFoundException{
     conectar();
     PreparedStatement ps;
     ps = cnx.prepareStatement("select numero, terminado, enviado, fechaenviado from pedido1 where terminado is true and enviado is true");
     ResultSet rs0 = ps.executeQuery();
          Object[] row;
          while (rs0.next()){
             row = new Object[4];
             row[0]=  rs0.getString(1);
             row[1]= rs0.getBoolean(2);
             row[2]= rs0.getBoolean(3);
             row[3]= rs0.getString(4);
             mod.addRow(row);
          }
       ps.close();
       cnx.close();
       return mod;
   }

   public void enviarpedido (String pedido) throws SQLException, ClassNotFoundException{
       conectar();
     PreparedStatement ps;
     ps = cnx.prepareStatement("UPDATE pedido1 SET enviado=TRUE WHERE numero=?");
     ps.setString(1, pedido);
     ps.executeUpdate();
     ps.close();
     cnx.close();
   }

   public void enviarpedidodesh (String pedido) throws SQLException, ClassNotFoundException{
       conectar();
     PreparedStatement ps;
     ps = cnx.prepareStatement("UPDATE pedido1 SET enviado=FALSE WHERE numero=?");
     ps.setString(1, pedido);
     ps.executeUpdate();
     ps.close();
     cnx.close();
   }

   public void fechaenviado (String fecha, String pedido) throws SQLException, ClassNotFoundException{
     conectar();
     PreparedStatement ps;
     ps = cnx.prepareStatement("UPDATE pedido1 SET fechaenviado=? WHERE numero=?");
     ps.setString(1, fecha);
     ps.setString(2, pedido);
     ps.executeUpdate();
     ps.close();
     System.out.println(fecha);
     cnx.close();

   }

   public void stock (String ref, int stock) throws SQLException, ClassNotFoundException{
     conectar();
     PreparedStatement ps;
     ps = cnx.prepareStatement("UPDATE productos_almacen SET stoc_seguridad=? WHERE referencia=?");
     ps.setInt(1, stock);
     ps.setString(2, ref);
     ps.executeUpdate();
```

```
        ps.close();
        cnx.close();

    }
}
```

Biblioteca
UNIVERSITAS Miguel Hernández

# 9. ANEXO 2 – CÓDIGO BASE DE DATOS

```
CREATE TABLE clientes(
id_cliente INT AUTO_INCREMENT,
nombre_cliente VARCHAR(33),
dir_clente VARCHAR(55),
pais_cliente VARCHAR(11),
tel_cliente INT,
 PRIMARY KEY (id_cliente)
);

CREATE TABLE pedido(
id_pedido INT AUTO_INCREMENT,
codemp_pedido VARCHAR(16) DEFAULT NULL,
id_cliente INT NOT NULL,
precio_pedido DECIMAL (8,2) DEFAULT NULL,
acept_pedido CHAR(1) DEFAULT NULL,
preparado_pedido CHAR(1) DEFAULT NULL,
enviado_pedido CHAR(1) DEFAULT NULL,

PRIMARY KEY (id_pedido),
FOREIGN KEY (id_cliente) REFERENCES clientes (id_cliente)
);

CREATE TABLE pedido_productos(
id_pedido_productos INT AUTO_INCREMENT,
codemp_pedido VARCHAR(16) DEFAULT NULL,
cantidad INT DEFAULT NULL,
referencia VARCHAR(16) DEFAULT NULL,
precio_nproductos DECIMAL (8,2) DEFAULT NULL,
descuento INT DEFAULT NULL,
precio_descuento DECIMAL (8,2) DEFAULT NULL,

PRIMARY KEY (id_pedido_productos)
);

CREATE TABLE desglose_conjuntos(
id_desglose_conjuntos INT AUTO_INCREMENT,
id_conjunto INT DEFAULT NULL,
descripción_conjunto VARCHAR(33) DEFAULT NULL,
referencia_producto VARCHAR(10) DEFAULT NULL,
cantidad INT DEFAULT NULL,

PRIMARY KEY (id_desglose_conjuntos)
);

CREATE TABLE productos_conjuntos(
id_productos_conjuntos INT AUTO_INCREMENT,
conjunto BOOLEAN,
referencia_producto_conjunto VARCHAR(12) DEFAULT NULL,
descripción VARCHAR(33) DEFAULT NULL,
precio DECIMAL (7,2) DEFAULT NULL,
proveedor VARCHAR(33) DEFAULT NULL,

PRIMARY KEY (id_productos_conjuntos)
);

CREATE TABLE tiempos_pedidos(
id_tiempos_pedidos INT AUTO_INCREMENT,
id_pedido INT,
fecha_oferta DATE DEFAULT NULL,
fecha_aceptar_pedido DATE DEFAULT NULL,
fecha_aceptar_produccion DATE DEFAULT NULL,
fecha_terminar_produccion DATE DEFAULT NULL,
fecha_preparado_salir DATE DEFAULT NULL,
fecha_salir DATE DEFAULT NULL,

PRIMARY KEY (id_tiempos_pedidos),
FOREIGN KEY (id_pedido) REFERENCES pedido (id_pedido)
);
```

```
CREATE TABLE productos_conjuntos(
id_conjunto INT AUTO_INCREMENT,
referencia_producto char(11),
cantidad_producto INT,
referencia_conjuto CHAR(11),


PRIMARY KEY (id_conjunto)
);


CREATE TABLE pedido1_producto(
id_pedido1_producto INT AUTO_INCREMENT,
pedido CHAR(20),
referencia CHAR(20),
cantidad INT,
precioxcantidad double,
descuento double,
preciofinal double,
garantia boolean,


PRIMARY KEY (id_pedido1_producto)
-- FOREIGN KEY (id_cliente) REFERENCES clientes (id_cliente)
);

CREATE TABLE fechas(
 id int,
 fecha char(30),

PRIMARY KEY (fecha)

);

CREATE TABLE productos_almacen(
 id int AUTO_INCREMENT,
 referencia char(10),
 descripcion char(30),
 cantidad_almacen int,
 fecha_entrada char(30),
 fecha_salidad char(30),
 stoc_seguridad int,

PRIMARY KEY (id)
);


CREATE TABLE orden(
 id int AUTO_INCREMENT,
 numero_orden char(10),
 pedido char(30),
 fecha_sacar_almacen char(30),
 fecha_prevista_salida char(30),
 etapas int default null,
 etapa1 char(30) default null,
 etapa2 char(30) default null,
 etapa3 char(30) default null,
 etapa4 char(30) default null,
 etapa5 char(30) default null,

PRIMARY KEY (id)
);



CREATE TABLE gestion_trabajadores(
 id_gestion_trabajadores int AUTO_INCREMENT,
 id_trabajador char(10),
 id_orden char(30),
 fecha_inicio char(30),
 fecha_final char(30) default null,

PRIMARY KEY (id_gestion_trabajadores)
);
```

```
CREATE TABLE gestion_produccion(
identificacion int AUTO_INCREMENT,
id_orden char(30),
num_empleados int,
etapas int,
inicio1 char(30),
final1 char(30),
inicio2 char(30),
final2 char(30),
inicio3 char(30),
final3 char(30),
inicio4 char(30),
final4 char(30),
inicio5 char(30),
final5 char(30),
notas char(200),
PRIMARY KEY (identificacion)
);


CREATE TABLE interaccion(
id int AUTO_INCREMENT,
id_interaccion char(30),
cliente char(20),
motivo int,
tipo_fallo char(30),
notas char(200),
PRIMARY KEY (id)
);


CREATE TABLE pedido1(
id_pedido int AUTO_INCREMENT,
numero char(10),
cliente char(10),
precio double,
direccion char(20),
garantia boolean default false,
pedido_aceptado boolean default false,
pagado boolean default false,
tramitado boolean default false,
fechapresitasal char(20) default null,
terminado boolean default false,
fechasalida char(20) default null,
PRIMARY KEY (id_pedido)
);

CREATE TABLE orden(
id int AUTO_INCREMENT,
numero_orden char(30),
pedido char(20),
fecha_sacar_almacen char(50),
fecha_prevista_salida char(50),
etapas int,
etapa1 char(50),
etapa2 char(50),
etapa3 char(50),
etapa4 char(50),
etapa5 char(50),
PRIMARY KEY (id)
);


CREATE TABLE notas(
id int AUTO_INCREMENT,
cliente char(30),
nota int,
fecha char(30),
PRIMARY KEY (id)
);
```

Biblioteca UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

# 10. BIBLIOGRAFIA

APD. (2019). *ASOCIACIÓN PARA EL PROGRESO DE LA DIRECCIÓN*. Obtenido de https://www.apd.es/ventajas-y-desventajas-sistema-erp/

Fantino, J. (2021). Obtenido de https://www.crehana.com/blog/desarrollo-web/que-es-netbeans/

Guevara, J. M. (s.f.). Fundamentos de programación en Java. EME.

Gustavo. (2022). Obtenido de https://www.hostinger.es/tutoriales/que-es-mysql

Outvio, D. B.-W. (2020). Obtenido de https://outvio.com/es/blog/mejores-erp/

RAE. (s.f.). *RAE*. Obtenido de https://dle.rae.es/empresa

VELNEO, F. F. (2020). Obtenido de https://www.velneo.com/blog/historia-de-erp-pasado-presente-y-futuro

Vizán, J. C. (2013). *LEAN MANUFACTURING - Conceptos, técnicas e implantación.* EOI.

Walton, A. (2018). Obtenido de https://javadesdecero.es/fundamentos/breve-historia-caracteristicas-y-aplicaciones/