Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Desarrollo de una aplicación para la gestión de un almacén caótico

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Andreu Juan Luna

Tutor: Antonio Martí Campoy

2019 - 2020

Resumen

En el presente trabajo final de grado, se ha diseñado e implementado, una aplicación capaz de gestionar un almacén de productos no perecederos. La gestión operativa de un almacén contempla una serie de procesos de los cuales algunos son de carácter organizativo, y es sobre todo estos en los que se centra este trabajo. Los tres principales procesos son la entrada de mercancía, la gestión del stock actual en el almacén y la preparación de pedidos. El proyecto propone una mejora organizativa y operativa centrándose en estos procesos para ahorrar costes.

Como principal objetivo del proyecto, se busca optimizar el proceso de preparación de pedidos, es decir, la solución usada para preparar un pedido para cada caso dado debe ser la óptima o la mejor posible. Y para ello, en este trabajo se realiza un análisis de las variables que afectan a cada uno de los diferentes entornos dentro del almacén, se trabaja en un esquema de almacén desordenado o caótico y se aplica la algorítmica que proporciona el estudio del problema del vendedor ambulante. Cabe informar que la búsqueda, hoy, no resulta ser exitosa pero el porcentaje de optimalidad se acerca a la solución óptima para cada caso. Aunque el éxito no haya sido encontrado, el avance operativo dentro de la gestión del almacén es significativo y grandes empresas como Amazon aplican la ingeniería operativa en sus almacenes para aumentar sus beneficios reduciendo costes.

Para cumplir con dicho fin, se ha desarrollado una aplicación de escritorio que permite al usuario gestionar la entrada de mercancía, la gestión del stock actual y la gestión de la salida de mercancía del almacén.

Palabras clave: almacén caótico, operativa, vendedor ambulante, óptima, gestión, TSP.

Resum

En aquest treball final de grau, s'ha dissenyat i implementat, una aplicació capaç de gestionar un magatzem de productes no peribles. La gestió operativa d'un magatzem contempla una sèrie de processos dels quals alguns són de caràcter organitzatiu, i és sobretot amb aquests en els quals se centra aquest treball. Els tres principals processos són l'entrada de mercaderia, la gestió de l'estoc actual en magatzem i la preparació de comandes. El projecte proposa una millora organitzativa i operativa centrant-se en aquests processos per a estalviar costos.

Com a principal objectiu del projecte, es busca optimitzar el procés de preparació de comandes, és a dir, la solució usada per a preparar una comanda per a cada cas donat ha de ser l'òptima o la millor possible. I per a això, en aquest treball es realitza una anàlisi de les variables que afecten a cadascun dels diferents entorns dins del magatzem, es treballa en un esquema de magatzem desordenat o caòtic i s'aplica l'algorítmica que proporciona l'estudi del problema del venedor ambulant. Cal informar que la cerca, hui, no resulta ser exitosa però el percentatge d'optimalitat s'acosta a la solució òptima per a cada cas. Encara que l'èxit no haja sigut trobat, l'avanç operatiu dins de la gestió del magatzem és significatiu i grans empreses com *Amazon apliquen l'enginyeria operativa en els seus magatzems per a augmentar els seus beneficis reduint costos.

Per complir amb aquesta finalitat, s'ha desenvolupat una aplicació d'escriptori que permet a l'usuari gestionar l'entrada de mercaderia, la gestió de l'estoc actual i la gestió de l'eixida de mercaderia del magatzem.

Paraules clau: magatzem caòtic, operativa, venedor ambulant, òptima, gestió, TSP.

Abstract

This final degree project aims to design and install an application able to manage a warehouse of non-perishable products. The operational management of a warehouse includes several processes, and some of them are of an organisational nature, and this work will be mainly focused on these processes. The three main processes are the receipt of goods, the management of the current stock in the warehouse and the preparation of orders. The project proposes an organisational and operational improvement by focusing on these processes in order to save costs and, consequently, money.

The main objective of the project is to optimise the order preparation process, i.e. the solution used to prepare an order for each given situation must be the optimum or the best. And to do this, in this work an analysis is made on the variables that affect each of the different environments within the warehouse, a disordered or chaotic warehouse scheme is developed and the algorithmic that provides the study of the problem of the street vendor is applied. It is worth pointing out that nowadays the search does not turn out to be successful, but the percentage of optimality approaches the optimal solution for each situation. Although success has not been found, the operational progress within the warehouse management is significant and large companies such as Amazon apply the operational engineering in their warehouses to increase their profits by reducing costs.

To achieve this goal, a desktop application has been developed which allows the user to manage the entry of goods, the management of the current stock and the management of the exit of goods from the warehouse.

Keywords: chaotic warehouse, operational, travelling salesman, optimal, management, TSP.

Agradecimientos

Tras un intenso periodo dedicado al grado y acompañado de diversas experiencias laborales, presento mi trabajo combinando mucho de lo que he podido aprender.

Una vez llegado a este punto, siento una gran satisfacción por haber cumplido mis objetivos. Cuando era niño, la informática despertó en mí una curiosidad que más adelante se convertiría en una parte esencial de mi vida. Empecé rompiendo el ordenador de mi padre, entre otros aparatos, pero él siempre lo arreglaba y de alguna manera me estaba enseñando algo. Así que el primero de los agradecimientos se lo dirijo a él; un policía aficionado a la electrónica. No puedo olvidarme de mi madre y agradecerle todo el apoyo y esfuerzo brindado durante todos estos años. Gracias mamá por confiar siempre en mí y por enseñarme a trabajar duro. Os quiero mucho a los dos y espero que estéis orgullosos.

A mi tutor del trabajo, Antonio Martí Campoy, le agradeceré toda mi vida, el trato y la ayuda recibida durante el trascurso de mis estudios de grado. Para mí, es todo un referente como persona y profesor.

También, me gustaría agradecer a mis padrinos y a sus hijas todo el apoyo recibido durante, no solo mi etapa como estudiante, sino toda mi vida. A Inés Juan, Paco "*el cuadrao*", Agnes Rico y Eva Rico. Tendréis por siempre todo mi afecto.

A los padres de mi pareja, Manuel Martínez y María de los Ángeles Lledó, que me acogieron y apoyaron desde el primer momento que los conocí. Sois más que familia, os quiero.

Son muchas, las personas que me han apoyado durante el trascurso de mis estudios, pero la que más ha influido en mí, es mi pareja. En nueve años me ha cambiado la vida. Me ha enseñado mucho la forma de ver las cosas, de entenderlas y de afrontarlas. Este proyecto te lo dedico a ti, Mireia.

¡Muchas gracias a todos!

Tabla de contenidos

Tabla de ilustraciones

Índice de tablas

1. Introducción

En los últimos años, los sistemas de información han constituido uno de los principales ámbitos de estudio en el área de organización de empresas. La creciente globalización, el incremento de competencia en los mercados y la internacionalización de estas son factores que han hecho que las Tecnologías de la Información (TI) jueguen un papel muy importante dentro de las organizaciones.

Pero *¿por qué aplicar una transformación digital en el negocio*? El uso de la tecnología en el contexto de la organización siempre persigue un objetivo claro: aumentar el rendimiento general de la empresa mejorando siempre los procesos que influyen sobre la eficacia y que directamente reducen costes.

Las variables que afectan al organigrama empresarial son diversas en magnitud y cantidad. Es por eso por lo que, para este trabajo, solo se hará enfoque sobre el estudio de la organización de un almacén y las variables que lo rodean en los procesos de entrada y salida de mercancía. Además, este organigrama tendrá una serie de particularidades que serán argumentadas durante el transcurso del trabajo. La primera de ellas es que el tipo de gestión de ubicaciones es caótica o desordenada; según es realizada la recepción de mercancía en el almacén, ésta es asignada a un hueco aleatorio en función de los espacios que haya disponibles en ese preciso momento. Otra particularidad es que el ser humano no puede ser consciente, en cada momento, de las unidades que tiene almacenadas un almacén para cierto artículo, ya que las unidades de este se encuentran dispersas por varias ubicaciones, por lo que es necesario tener un sistema informático capaz de brindar una solución a este problema. Una particularidad más es que en el proceso de preparación de pedidos, la aplicación usará una librería llamada *"OR-Tools"* con un elemento generador de soluciones heurísticas llamado "*solver*" (herramienta que busca el valor óptimo a un determinado problema) que tratará de buscar la ruta óptima para la preparación de los pedidos. Como última particularidad, el almacén solo podrá operar con productos no perecederos (de larga durabilidad, el paso del tiempo no les afecta mucho) ya que la gestión del almacén permitirá que exista una rotación en la salida de los artículos, pero no se podrá discriminar en un mismo artículo si debe salir una unidad determinada u otra.

Actualmente, en el mercado, existen muchas empresas que hacen uso del concepto caótico en sus almacenes, aunque también lo implantan muchos programas de gestión empresarial, que no solo administran empresas de gran tamaño como Amazon, sino que este concepto cada vez se implanta más en empresas de tamaño pequeño y medio.

1.1 Motivación

En mi segundo año de grado empecé mi experiencia laboral en una empresa desarrollando módulos pertenecientes a un ERP (sistema de plantificación de recursos empresariales). Desde entonces he adquirido ciertos conocimientos en el ámbito de la gestión empresarial y de sus diferentes áreas. Además de la experiencia profesional, este sector ha despertado cierta inquietud en mí que ha hecho que estudie las problemáticas que tiene y como la informática y la ingeniería operativa pueden aplicarse para resolverlas.

Entonces el día 14 de noviembre de 2019, fui a una conferencia que impartió Rubén Ruiz García (Doctor en Ingeniería Informática y profesor de la *Universitat Politècnica de València*) en la que se habló de la investigación operativa. La conferencia tenía como título "El fin de los problemas imposibles". En dicha conferencia se introdujo el estudio de problemas que surgen en la empresa y como conseguir la solución óptima a dichos problemas. Como ejemplo, surgió el problema del vendedor ambulante o TSP (*Travelling Salesman Problem*) y las opciones computacionales que se disponen hoy en día para intentar resolverlo.

Tal fue la fascinación que provocó en mi la explicación del profesor que en ese momento pensé en trasladar esa problemática al problema de reducción de tiempo en la preparación de pedidos dentro de un almacén con productos no perecederos.

Como se ha comentado en el apartado de introducción, existen diferentes aplicaciones empresariales que han adaptado el concepto caótico en la operativa de almacén. Mi intención no es mejorar la funcionalidad que ofrecen cada una de ellas; es más un reto personal que me he propuesto.

1.2 Objetivos

Se pretende crear una aplicación que gestione la mercancía que es movida dentro de un almacén de productos no perecederos. Además, los tiempos medios que son precisados en la gestión de productos en la entrada y salida de mercancía deben ser más económicos que en un almacén con gestión de ubicaciones fijas (también nombrada durante el proyecto como gestión convencional, al ser la más utilizada por las empresas de comercialización de productos). La gestión sustentará tres procesos principales:

- Entrada de mercancía: la mercancía que entra en el almacén (acompañada de un albarán de compra) deberá ser gestionada y distribuida conforme al concepto caótico, resolviendo los problemas operativos de almacenado de la misma que se presentan en un almacén con gestión convencional.

- Gestión de mercancía: aunque la distribución de la mercancía entre las ubicaciones del almacén sea caótica, el sistema debe proporcionar al usuario, la información que el mismo requiera sobre el stock que se guarda en cada una de las ubicaciones. Además, se podrán realizar movimientos de mercancía entre ubicaciones y almacenes para asegurar un control del stock de la empresa.

- Salida de mercancía: la mercancía que sale del almacén deberá ser gestionada y preparada conforme al concepto caótico y utilizando las herramientas necesarias que obtengan la ruta mínima en el recorrido de preparación de pedidos de venta.

Para que el programa sea lo más accesible posible por personas sin conocimientos en la utilización del terminal de comandos del sistema operativo, se tiene también como objetivo, la adaptación del código a una interfaz gráfica con un menú de acceso a los diferentes puntos funcionales propuestos.

1.3 Impacto esperado

Como se ha comentado en el punto anterior, la reducción de los tiempos medios operativos en un almacén es uno de los objetivos a llevar a cabo, sin embargo, conlleva la realización de un estudio de las variables que lo involucran.

Con esto, se espera ofrecer una gestión de almacén diferente a la más utilizada convencionalmente (almacén organizado por ubicaciones fijas) y mucho más eficiente ya que:

- Se reduce el tiempo medio de preparación de pedidos.
- Se reduce la gestión operativa en la entrada de mercancía.
- Se reducen costes operativos al otorgar al aplicativo el poder de organización de artículos y no depender de un estudio previo de organización.

1.4 Metodología

La metodología que ha de seguir este trabajo es el modelo en cascada. Es un enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior. Al final de cada etapa, el modelo está diseñado para llevar a cabo una revisión final, que se encarga de determinar si el proyecto está listo para avanzar a la siguiente fase.

La metodología de desarrollo en cascada que se sigue en este proyecto es:

1. Análisis de requisitos.
2. Diseño de la base de datos.
3. Diseño de la aplicación de escritorio.
4. Implementación.
5. Pruebas.
6. Mantenimiento

Para alcanzar cada uno de los objetivos descritos y lograr el impacto esperado, se va a estructurar y nombrar los principales recursos utilizados:

- Paquete *OR-Tools* de *Google* : *software* de código abierto, rápido y portátil para la optimización combinatoria que está pensado para intentar abordar los problemas más difíciles del mundo en enrutamiento de vehículos, flujos, programación entera y lineal, y programación de restricciones. Sin embargo, la aplicación desarrollada en el trascurso del proyecto hará uso del "*solver*" que tiene la herramienta para aplicar el problema del vendedor ambulante o TSP (*Travelling Salesman Problem*) sobre un almacén dado.

- Instancia de base de datos Oracle sobre una máquina virtual de Windows 10 de 64 bits. Con esto, también se simulará la conexión de la aplicación con un servicio de base de datos externo.

- *PL/SQL Developer* como entorno de programación e integración de objetos de la base de datos como tablas, procedimientos, secuencias, disparadores, paquetes y funciones.

- *Visual Studio Enterprise* versión 2017 como entorno de desarrollo de la interfaz gráfica, de la comunicación con la base de datos y la integración operativa de los diferentes procedimientos que serán usados para la gestión de la mercancía junto con el "*solver*" de *Google*, utilizando el lenguaje C#.

- Documento de Microsoft Excel que contiene una plantilla de la matriz de distancias necesaria para resolver el problema de la ruta mínima.

1.5 Estructura de la memoria

Para abordar la memoria del proyecto, ésta se ha estructurado en los siguientes capítulos.

Capítulo 1. Introducción: introducción sobre el tema principal y aquellos aspectos que han motivado su elección.

Capítulo 2. Estado del arte: exposición que detalla el problema operativo que presenta un almacén de productos no perecederos, las diferentes opciones que se han estudiado y los motivos por los cuales se ha escogido la gestión caótica.

Capítulo 3. Análisis del problema: exposición que detallada las diferentes problemáticas que presentan los principales procesos operativos del almacén que gestiona la aplicación, con comparativas de diferentes tipos de gestiones y los motivos por los cuales se ha escogido el tipo de gestión caótica combinado con el estudio del problema del algoritmo TSP.

Capítulo 4. Análisis de requisitos: exposición de los diferentes requisitos que debe tener la aplicación en cada uno de los puntos operativos que en ella se gestionan.

Capítulo 5. Diseño: exposición detallada del modelo de diseño de *software* elegido y de la estructura y diseño de la base de datos relacional.

Capítulo 6. Implantación: contendrá los requisitos de infraestructura *software* y *hardware* mínimos para la implantación, la estructura de la base de datos que ha sido utilizada, además de los pasos seguidos para implantar la aplicación de escritorio en un equipo.

Capítulo 7 Pruebas: contiene toda la información obtenida en la realización de pruebas atacando los tres puntos operativos que ofrece la aplicación.

Capítulo 8. Conclusiones: contendrá las diferentes conclusiones que se han obtenido durante la realización del proyecto.

Capítulo 9. Futuras mejoras: capítulo que contendrá las diferentes mejoras que añadirían valor en la funcionalidad de la aplicación planteada y que se han ido planteando durante el desarrollo del proyecto.

Capítulo 10. Bibliografía.

1.6 Convenciones

Las palabras extranjeras aparecerán distinguidas con letra cursiva y las citas textuales, expresiones y las preguntas retóricas al lector irán remarcadas entre comillas.

2. Estado del arte

El concepto almacén puede sugerir diferentes cosas dependiendo de las experiencias profesionales que cada individuo haya vivido.

Etimológicamente, la palabra almacén proviene del término árabe "*almazán*" que hace referencia en su concepción clásica a un espacio físico destinado a albergar mercancías de cualquier especie. Actualmente, esta forma de entender el concepto almacén ha quedado obsoleta, ya que, el paso del tiempo, la influencia de la globalización y la tecnología han hecho que adquiera cierto grado de complejidad, realizando más actividades que la simple custodia de las mercancías y con capacidad de especialización.

Cada almacén cuenta con una operativa y unos requerimientos que dependen de múltiples factores, como puede ser la mercancía almacenada: no funciona de igual manera un almacén de alimentos perecederos que uno de alimentos no perecederos.

Dependiendo del sector al cual está destinado el sector de la empresa y del uso posterior que se va a hacer de la mercancía, se podrán clasificar los almacenes en:

- **Almacenes de materias primas:** son aquellos almacenes que albergan las materias y materiales necesarios para la producción o comercialización de algún producto específico. [1]

- **Almacenes de materiales intermedios:** estos almacenes albergan productos que se sitúan a mitad de la cadena de producción. [1]

- **Almacenes de producto terminado:** estos almacenes albergan productos terminados, totalmente operativos para ser suministrados al cliente. Lo interesante de esta clasificación es que el valor del producto almacenado es mayor que en los anteriores ya que el producto ya ha pasado a una fase final. [1]

Una empresa que se dedica a la fabricación y venta de productos terminados precisa tener una zona para almacenar la materia prima (preparada para empezar la producción), una zona para almacenar materia intermedia (solo necesaria para almacenar productos que se ensamblan para poder formar el producto acabado u otros productos intermedios necesarios para terminar el producto final ya que al tratar la materia prima no se puede conseguir dicho destino) y una zona para almacenar producto terminado.

Este proyecto se centrará únicamente en la clasificación de los almacenes de producto terminado, ya que se va a trabajar con una operativa de transacción de mercancía y no de transformación de esta, aunque podría adaptarse a empresas que trabajasen ofreciendo a otras empresas productos intermedios, ya que, para la empresa vendedora, el producto que ofrece está listo para la venta (no para el cliente final, pero si para un cliente intermedio no final que transformará el producto).

En los años recientes, el almacén ha dejado de ser un sencillo espacio en el que albergar los productos para convertirse en parte crucial del negocio.

Hoy en día, el marco emergente en los mercados demanda una reducción de tiempo de respuesta, un óptimo servicio de entrega y, todo ello, sin perder calidad en la gestión del almacén.

Por ello, el concepto de almacenaje ha ido evolucionando y en la actualidad engloba otras actividades, más allá del mero hecho de guardar y custodiar mercancía.

Dichas actividades se englobarán y ordenarán en un concepto, el cual será nombrado como, operativa de almacén, que incluye:

1. **Recepción de productos:** el personal adecuado comprobará y registrará la mercancía recibida, así como la calidad, la cantidad y otras características, de acuerdo con los requerimientos establecidos entre la empresa y el correspondiente proveedor.

2. **Almacenamiento:** proceso que consiste en almacenar la mercancía en accesos o ubicaciones. Dependiendo de la colocación de las unidades de carga existen varias estrategias:

    - **Gestión de ubicaciones fija o específica:** durante el proyecto se la refiere como gestión convencional, ya que es la más utilizada hasta el momento. Cada artículo es asignado a una posición o número de ubicación de antemano que, como ventaja, los operarios de almacén pueden memorizar en gran medida sin recurrir a recursos informáticos. Pero como desventaja, la capacidad efectiva se ve muy reducida proporcionalmente al aumento de la capacidad y número de ubicaciones del almacén. La evidencia es que en un almacén siempre existen artículos que tienen un número de transacciones y de almacenado más elevado que otros. Entonces existirán productos mucho más difíciles de ubicar que otros por la saturación de existencias en sus ubicaciones, en cambió otros productos menos frecuentados, logísticamente hablando, serán fáciles de ubicar ya que sus estancias estarán prácticamente vacías. Por lo que, cuanto mayor sea el almacén y más referencias existan, la capacidad efectiva del mismo se verá disminuida. Otra desventaja significativa es que el cambio de ubicaciones de artículos resulta un problema ya que el operario debe memorizar estos cambios en el momento para no disminuir la eficacia de preparación de pedidos.

    - **Almacén caótico, desordenado o aleatorio:** la mercancía se ubica en cualquier localidad o ubicación disponible, siguiendo previamente una lógica establecida y parametrizada en el "*SGA*" (sistema de gestión de almacenes). Una vez introducidos todos los datos de entrada de mercancía en el sistema, éste le indica al operario dónde ha de colocar cada una de las referencias y la cantidad de estas. Con esto se consigue acercar la capacidad efectiva del almacén a la capacidad física, pudiendo superar el 92% de ésta. *¿Esto que significa?* Al contrario que en la gestión anterior, la capacidad efectiva se aprovecha en gran medida ya que, las ubicaciones se rellenan aleatoriamente sin discriminar zonas o ubicaciones por tipo de artículo, u otro factor. La desventaja principal es que es difícil aplicar este tipo de gestión a los alimentos "no perecederos" ya que precisan una gestión de salida "FIFO" en función de la caducidad de estos y requeriría modificar el algoritmo que obtiene las ubicaciones de recogida de productos para que tuviera en cuenta la salida de estas referencias priorizando sus caducidades de forma descendente.

- **Gestión mixta:** combina el sistema fijo y el caótico, con la asignación de cada uno de ellos en función del tipo de producto. La ubicación fija es utilizada para productos de alto consumo, que se encuentran cerca de las zonas de "*picking*" o de los muelles, mientras que el caótico se deja para el resto de los productos y zonas de reserva.

3. **Conservación y mantenimiento:** la mercancía deberá ser tratada en consideración a las normas de seguridad, salud y otros requerimientos vigentes para que el estado de esta sea siempre el adecuado. Además, es necesaria la realización de revisiones periódicas del estado de capacidad y otros factores que afectan directa e indirectamente a la mercancía guardada en el almacén. Como ha podido verse en el punto dos, existe una gestión de almacén caótica, en la cual se complica la obtención de estudios y análisis de datos que referencian a productos guardados en el almacén cuando son guardados de forma aleatoria.

   Es necesario que, en este caso, se haga uso de un sistema informático que controle el estado de cada una de las ubicaciones y de la mercancía que guardan.

4. **Preparación de pedidos:** el operario obtiene un documento de albarán (referenciado a un pedido hecho al proveedor) en forma de listado que puede estar en formato digital o físico y que detalla los artículos y cantidades de cada uno de ellos necesarios para la realización del pedido. Este tipo de documento se estructura en dos partes principalmente:

   a) **Cabecera del pedido:** primordialmente contiene el número de documento, los datos fiscales de la entidad que vende y de la que va a comprar, aunque también aparecen datos propios del documento como la fecha, forma de pago a utilizar y la dirección de entrega.

   b) **Detalle del pedido:** conforma un listado de registros con las siguientes columnas: código de artículo y cantidad. Con estos datos el operario podría realizar el pedido siempre que el sistema de gestión del almacén sea de ubicaciones fijas y éste sepa donde están ubicadas cada una de las referencias citadas. En el caso de utilizar el sistema de gestión caótica, se deberán incluir las ubicaciones a visitar (una columna más) en cada una de las líneas del listado ya que el operario no tiene conocimiento de que referencias y cuanta cantidad de ellas existe en cada ubicación. Otros datos, como el importe de línea y precio por unidad también aparecen en el albarán, aunque estos no confieren ninguna función relacionada con la gestión del stock del almacén.

   Dentro de este punto, es necesario hacerse la siguiente pregunta: "*¿Cuál es la ruta óptima para la realización de un pedido concreto en un almacén dado?*". Si en una empresa de comercialización de productos se propone a distintos operarios que realicen un mismo pedido partiendo de la misma ubicación (ubicación inicial: zona de carga), es improbable que todos y cada uno de ellos ejecuten el mismo tiempo de preparación del pedido. En ello, aparecen dos factores:

   - **Factor humano:** cada operario trabaja y opera de una manera distinta a otro. Habrá operarios que de promedio trabajen más rápido y otros que lo hagan más lento e incluso, habrá uno de ellos que tenga el mejor promedio y otro que tenga el peor. Las capacidades, habilidades, destrezas y otras aptitudes y actitudes del día a día del operario influyen en el tiempo promedio de preparación de un pedido.

Realmente es muy difícil controlar estos factores, aunque existen estrategias empresariales que pueden favorecer y potenciar muchos de ellos. Por este motivo, la mejora de estos factores queda fuera del alcance de este proyecto.

- **Factor operativo:** las diferentes decisiones operativas que puede establecer (o le es establecido a) un operario afecta sustancialmente al tiempo promedio de preparación de un pedido.

Dentro de este punto, es necesario hablar del concepto de "*Principio de optimalidad de Bellman*". La definición que proporciona la "*Real Academia de Ingeniería*" es: "Principio aplicado en programación dinámica que consiste en que una secuencia óptima de decisiones que resuelve un problema debe cumplir la propiedad de que cualquier subsecuencia de decisiones, que tenga el mismo estado final, debe ser también óptima respecto al subproblema correspondiente." [2]

Adaptando esta definición a un ejemplo de preparación de un pedido dentro de un almacén se puede decir: dado un pedido, se precisa visitar un número *N* de ubicaciones de modo que, la secuencia de visita de dichas ubicaciones sea óptima. "*¿Qué significa realmente que la secuencia sea óptima?*" Quiere decir que, de todos los caminos posibles con el mismo estado final, el camino elegido debe ser el mejor (ruta más corta). Puede haber uno o más de uno, pero en tal caso el coste debe ser el mismo para todas y cada una de las soluciones óptimas.

Alcanzado este punto, es necesario realizar una aclaración. Cuando se habla de *óptimo* se dice que es el mejor. Es incorrecto usar la expresión "*ruta más óptima*" ya que una ruta es óptima cuando las rutas restantes no lo son, con lo que, consecuentemente en ningún caso podrá usarse la expresión "*las demás rutas son menos óptimas*", ya que no existen rutas más óptimas que otras. Solo hay una que es la mejor, aunque dependiendo del problema, podría darse el caso que más de una ruta resultara ser óptima. En tal caso, el coste de todas ellas seria el mismo.

Dado un grafo completo (tabla 1), construido a partir de una matriz de distancias (ilustración 1), donde los vértices representan ubicaciones y las aristas la distancia entre los dos vértices extremos, se pretende hallar un camino óptimo que parta de un vértice dado, recorra los demás y vuelva al vértice inicial. Se podría adaptar este ejemplo a un caso real si el vértice inicial fuera la ubicación perteneciente a la zona de carga, los demás vértices fueran ubicaciones por las cuales debe pasar el operario para realizar el pedido completo y por último el operario volvería a la ubicación inicial para depositar el pedido terminado en la zona de carga.

Tabla 1: Matriz de distancias de ejemplo

d(i,j)	1	2	3	4	5	6	7
1	0						
2	37	0					
3	39	40	0				
4	**37**	36	23	0			
5	28	**9**	**19**	23	0		
6	33	13	**17**	**5**	31	0	
7	**22**	**8**	40	14	31	23	0

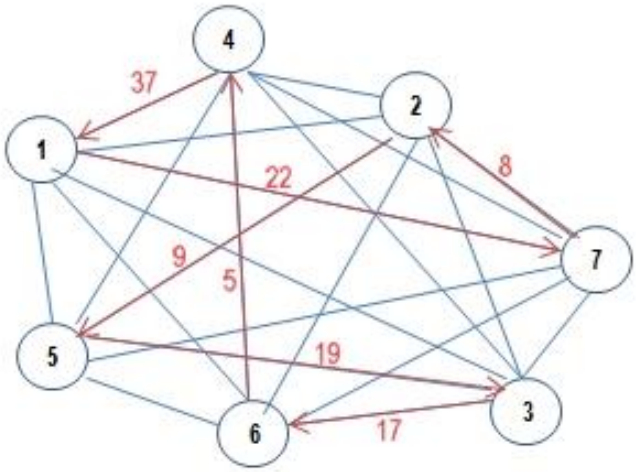


Ilustración 1: Grafo completo del ejemplo planteado en tabla 1

Este rompecabezas se asemeja al problema del vendedor ambulante o comúnmente llamado TSP de sus siglas en inglés "*Travelling Salesman Problem*" . Este modelo tiene como objetivo crucial encontrar un itinerario completo que una todos los nodos de una red, visitándolos tan solo una vez y volviendo al punto inicial, y asimismo minimice la distancia total de la ruta, o aplicando el tiempo, una reducción del tiempo total del recorrido. Este problema dispone de una variación considerable, y es que las distancias entre un nodo y otro puede que sean simétricas o no. Para no reducir la generalidad y aplicabilidad de este trabajo, se considerará que el coste de ir de *A* a *B* puede ser diferente que de ir de *B* a *A*.

El número de posibles rutas *R* en una red de nodos está determinado por la ecuación 1 donde *n* es el número de ubicaciones:

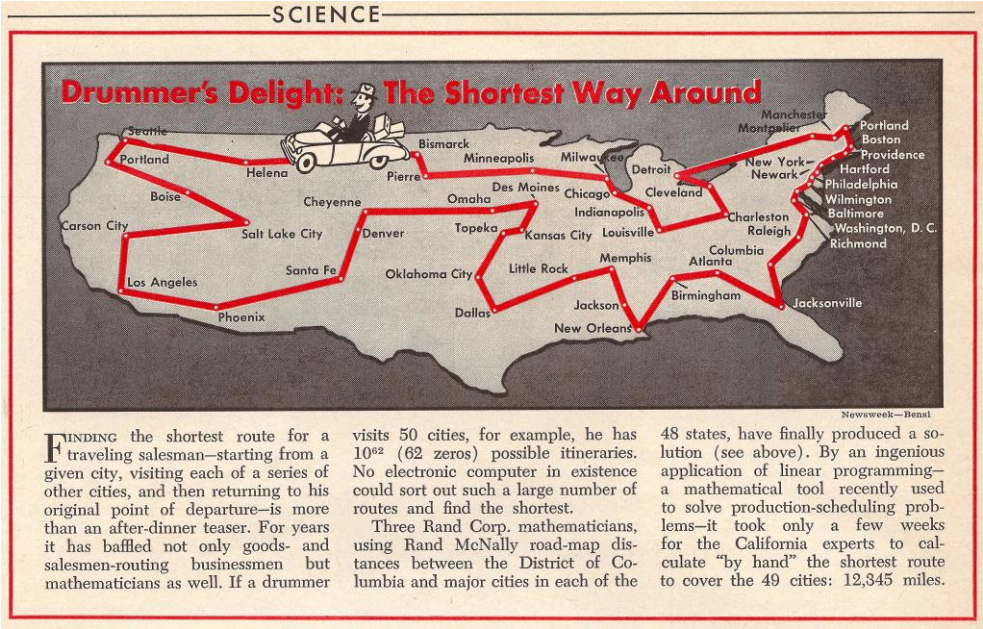
$$R = (n-1)! \quad (1)$$

En cambio, si la distancia entre nodos fuera simétrica, el número de posibilidades se vería reducido a la mitad. La ecuación 2 muestra el resultado en distancias simétricas:

$$R = \frac{(n-1)!}{2} \quad (2)$$

El ahorro de tiempo de procesamiento es significativo. Para poder trabajar con casos reales y aplicar el algoritmo a un estándar donde pueda haber muchos tipos de almacenes se deberá trabajar pensando en que la distancia entre los nodos puede no ser simétrica.

Las posibles soluciones al problema del vendedor ambulante han generado un interés a lo larga de los años, sobre todo desde que la revista Newsweek planteara el 26 de Julio de 1954 (ilustración 2), el problema del TSP sobre 50 ciudades estadounidenses.



SCIENCE

Drummer's Delight: The Shortest Way Around

Newsweek—Bensi

FINDING the shortest route for a traveling salesman—starting from a given city, visiting each of a series of other cities, and then returning to his original point of departure—is more than an after-dinner teaser. For years it has baffled not only goods- and salesmen-routing businessmen but mathematicians as well. If a drummer visits 50 cities, for example, he has 10^{62} (62 zeros) possible itineraries. No electronic computer in existence could sort out such a large number of routes and find the shortest.

Three Rand Corp. mathematicians, using Rand McNally road-map distances between the District of Columbia and major cities in each of the 48 states, have finally produced a solution (see above). By an ingenious application of linear programming—a mathematical tool recently used to solve production-scheduling problems—it took only a few weeks for the California experts to calculate "by hand" the shortest route to cover the 49 cities: 12,345 miles.

Ilustración 2: Newsweek, 26 Julio, 1954 [3]

Aplicando la ecuación 1, referenciada anteriormente (distancias no simétricas entre nodos) para obtener el número de rutas posibles, se obtiene la siguiente cifra:

$$R = (50 - 1)! = 6.0828186403427\ x\ 10^{62}\ caminos\ posibles$$

Este colosal número, según la investigación que hizo Rubén Ruiz (profesor y catedrático de Estadística e Investigación Operativa de la UPV) en la conferencia "El fin de los problemas imposibles", se asemeja a la suma de todos los átomos de más de 300 billones de tierras.

Si se dispusiera de un equipo computacional capaz de calcular 100 millones de soluciones por segundo, se tardaría 9,644246 x 10^{48} años.

Es posible afirmar que utilizando el método de la fuerza bruta, que consiste en realizar el cálculo de todos los posibles recorridos, hoy en día no existe ningún equipo computacional lo suficientemente potente para calcular todos los resultados en un espacio de tiempo aceptable y útil. Este método es extremadamente ineficiente e imposibilita prácticamente el cálculo del camino óptimo en problemas donde la cantidad de vértices es de gran tamaño. Pero, a lo largo de los años, se han desarrollado otras soluciones heurísticas que abaratan el cálculo de la solución subóptima. Estas se refieren al uso de algoritmos genéticos, pertenecientes a la familia de algoritmos evolutivos. Son una clase de metaheurísticas basadas en la teoría de la selección natural, la evolución y la genética. Sus métodos más utilizados son:

- **Método del vecino más cercano:** se basa en un algoritmo heurístico esbozado para generar una solución al problema del viajero ambulante. Este método no ratifica una solución óptima, sin embargo, es frecuente que aporte una buena solución y el tiempo de cálculo que exige es aceptable. Este método consiste en:

1. Se parte del nodo inicial y se escoge como nodo destino el siguiente que tenga la menor distancia hacia el nodo actualmente situado.

2. Una vez, en el segundo nodo, se escoge el siguiente nodo destino realizando la misma estrategia que en el punto anterior, pero sin tener en cuenta los nodos ya visitados.

3. Se repite el paso número 2 para los nodos restantes hasta llegar al último que debe tener, obligatoriamente como siguiente, el nodo inicial.

- **Método de ramificación y acotación (*Branch and bound*):** este método proporciona un conjunto de soluciones ramificadas que son dispuestas en forma de árbol, donde cada nodo representa un problema lineal. Los nodos hijo heredan el problema del padre, pero con una restricción más añadida que su correspondiente predecesor y el resultado de este movimiento es obligar a que una de las variables obtenidas en la resolución del nodo padres sea menor o igual que la parte entera de la solución óptima obtenida para dicha variable, consiguiendo de esta manera dos nuevos problemas objeto de estudio.

  Este método se basa en el concepto "*divide y vencerás*", donde se divide la región factible hasta que no queden más elementos objeto de estudio y se haya encontrado la solución óptima. A medida que avanza el cálculo de los nodos del árbol las cotas inferiores y superiores se irán actualizando y se habrá encontrado una solución óptima cuando estas dos cotas sean iguales. El siguiente ejemplo de TSP parte de la ubicación "0" y se pretende visitar las demás ubicaciones sin repetir ninguna y en el menor tiempo posible (la numeración de los vértices indica el coste de tiempo en minutos). La ilustración 3, muestra un árbol de búsqueda y la solución óptima con la secuencia de nodos 0-1-2-3-0.

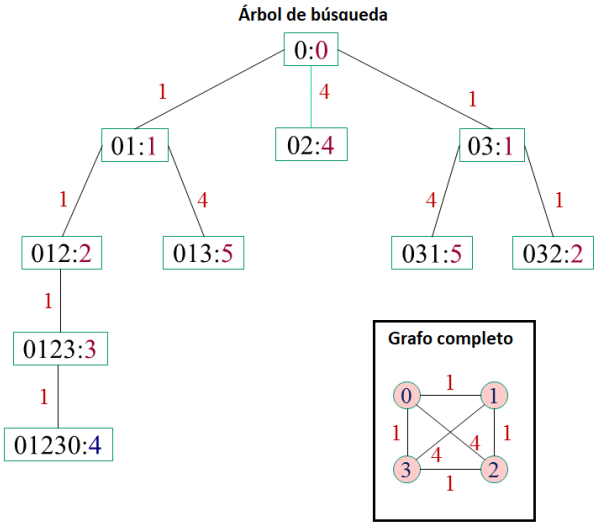


Ilustración 3: Grafo completo del ejemplo planteado con árbol de búsqueda desarrollado

La ramificación se ha realizado por la izquierda, pero lo que interesa de este árbol son las podas que se han realizado en las secuencias 0-1-3, 0-3-1 y 0-3-2 ya que las cotas inferiores de estos eran iguales o superiores a las actualizadas en ese momento y por eso no interesa continuar ramificando.

La solución de los nodos hijos empeoraría la solución ya obtenida y como ya no hay más nodos por los cuales ramificar, se ha obtenido la solución óptima. El gran inconveniente se presenta cuando la talla del problema es bastante extensa y la ramificación está muy cargada. Entonces, el tiempo exigido para el cálculo de la solución óptima es muy extenso, ya que no se dispone hoy en día de equipos lo suficientemente potentes para calcular la solución óptima de grandes problemas en un espacio aceptable de tiempo.

La eficacia y la eficiencia son dos variables fundamentales a tener en cuenta para la aplicación de un método heurístico:

- **Eficacia:** puede identificarse este término como la capacidad de alcanzar el objetivo. Si no se encontrara el camino más corto o camino óptimo se debe lograr encontrar uno que se acerque mucho al óptimo.

- **Eficiencia:** este término arroja una respuesta relacionada con los recursos utilizados, como el tiempo de cómputo utilizado para encontrar la solución.

5. **Transporte:** una vez que el pedido está preparado, se ensambla y se envía según características al destino indicado. Esta parte no se implementará en el proyecto ya que se entiende que pertenece al servicio de expediciones o de logística externa más que al servicio o departamento de almacenaje de la empresa.

Es necesario recalcar que todo tipo de empresa que precise atender un mercado de producción o consumo requiere de un servicio de almacenaje, el cual disponga de una estructura ya sea propia o a través de un tercero, y siempre haciendo énfasis en los primeros cinco puntos ya que son clave a la hora de gestionar la operativa del almacén.

Una vez, vistos los métodos más utilizados en la resolución del problema del viajero ambulante o sus derivados, hay que tener en cuenta que en el día a día de un almacén existen una gran cantidad de ubicaciones y que para el trazado óptimo de un subconjunto de ellas y operar con unos tiempos computacionales aceptables es necesario hacer uso de herramientas *software* heurísticas. Existen muchas herramientas en el mercado y algunas de ellas se basan en un concepto llamado "*Solver*".

- **Solver:** es una herramienta *software* matemática, conformada como un programa informático independiente o como una biblioteca o paquete *software*, que resuelve un problema matemático hallando un valor óptimo que puede ser un valor mínimo o un valor máximo dependiendo de la fórmula.

En este proyecto, se utilizará un "*solver*" que pueda adaptarse en forma de paquete a un proyecto de Visual Studio 2017 creado con el lenguaje C# y que utilice el método "*Branch and Bound*" ya que es rápido para encontrar la solución óptima en problemas de talla grande. Una de las grandes ventajas que ofrece la utilización de un "*solver*" es que ya está implementado y puede ser integrado en una aplicación multiplataforma mediante el uso de bibliotecas. Por esto, en este proyecto, se ha decidido utilizar una herramienta ya implementada.

El "*solver*" no garantiza que la solución obtenida sea la óptima cuando la talla de problema es extensa, pero éste hace uso de las cotas para poder aproximarse a la solución óptima.

En el planteamiento de un problema heurístico (como el cálculo del camino más corto), existe una característica que permite la obtención de cotas, mediante la técnica de relajación (que ignora algunas restricciones) y permite encontrar una solución bastante aproximada a la óptima en problemas de talla grande.

Además, el "*solver*" utiliza el algoritmo deseado para calcular los valores mínimos y máximos para una secuencia parcial fijada. Estos valores o cotas se obtienen evaluando el tiempo de procesamiento restante, a partir de un método basado en la determinación de las restricciones de cada nodo y éste en consecuencia actúa como cuello de botella. Cada secuencia parcial se asocia con cada nodo del árbol de exploración de tal forma que todos los nodos creados a partir de este coinciden con la secuencia parcial definida.

Existen multitud de herramientas *software*, pero para este proyecto se precisa de las que puedan ser adaptadas al entorno de programación C# y Visual Studio 2017:

- **LocalSolver:** herramienta implementada en lenguaje C++ que puede adaptarse mediante API (interfaces de programación de aplicaciones) e integrarse por completo con aplicaciones comerciales .NET. Esta herramienta es útil para resolver el problema de la mochila que consiste en rellenar la misma utilizando objetos con peso y valor específicos y siempre intentado maximizar el valor total sin exceder el peso máximo.

- **Google OR-Tools:** paquete *software* de código abierto orientado a la optimización de problemas de enrutamiento de gran dificultad. Ofrece un gran número de Solvers adaptados a la necesidad del problema.

- **Microsoft Solver Foundation:** herramienta .NET para modelado, optimización y simulación de problemas de optimización. La opción más eficiente es construir un modelo en Microsoft Excel y luego codificarlo en C#. El gran inconveniente es que se precisa de un *software* no gratuito de Microsoft.

Después de haber realizado distintas pruebas con las diferentes herramientas Solver indicadas anteriormente, la que más se adapta por facilidad de uso y eficiencia en problemas de talla grande es Google OR-Tools. Se ha podido apreciar la rapidez de esta herramienta al compararla con la llamada al proceso incrustado en Microsoft Excel y en problemas de talla grande Google OR-Tools tiene una respuesta mucho más rápida que Microsoft Excel.

3. Análisis del problema

La operativa que gestiona un almacén contempla muchas actividades y depende del sector y la logística empresarial, el tipo de producto e incluso el tipo de almacén.

El prototipo de almacén sobre el que se va a trabajar estará enfocado en el sector de la distribución de productos terminados donde no se incluyen alimentos perecederos ya que conlleva una gestión diferente a la que plantea el concepto caótico de almacén. La gestión operativa en este tipo de productos debe seguir un sistema FIFO (*First Int First Out*) en el tratamiento de mercancía. No es factible trabajar con este tipo de productos ya que, factores como la caducidad, afectarían al sistema de preparación de pedidos, por lo que la gestión caótica se vería severamente afectada al tener que dar prioridad de salida a determinados productos.

3.1 Estructura del almacén

El almacén estará ubicado en una zona estratégicamente preparada para la llegada de mercancía a través vehículos logísticos. Además, estará dotado de una serie de elementos característicos de un almacén convencional:

- **Estanterías:** estructuras (normalmente metálicas) independientes del edificio que son fijadas al suelo. Estas estarán compuestas por cinco localidades, divididas en tres alturas y cada una de estas divisiones representará una ubicación con el formato denominativo "*PxAyUz*". Dicha normativa, utilizada en la denominación de la ubicación, ha sido planteada con tal de poder identificar de una mejor manera cada una de las ubicaciones.

    - **Px:** donde "*P*" representa el pasillo donde reside la estantería y "*x*" el número del pasillo. Como ejemplo para ilustrar la estructura, se definirán, tres pasillos.

    - **Ay:** donde "*A*" representa la altura en la cual está situada la ubicación y "*y*" el nivel de la altura. Como ejemplo para ilustrar la estructura, se definirán tres alturas por pasillo.

    - **Uz:** donde "*U*" representa una localidad y "z" el número que identifica a dicha. Como ejemplo para ilustrar la estructura, se definirán cinco localidades por pasillo.

    Con lo que, si hay 3 pasillos, 3 alturas por pasillo y 5 localidades por pasillo, en total, habrá 45 ubicaciones organizadas en estanterías. Pero, además, también habrá 2 ubicaciones adicionales de carácter especial:

    - **Zona de carga:** esta ubicación tiene un fin estratégico. Para resolver el problema del camino óptimo (TSP), el repartidor debe salir de una ubicación, recorrer, las ubicaciones requeridas para la preparación del pedido y volver a la ubicación inicial. Por eso, se establecen como ubicación inicial, la zona de carga, para que el repartidor salga de esta y termine en la misma donde depositará el pedido una vez ya terminado. Esta ubicación no almacenará ningún producto, ya que solo está destinada a almacenar pedidos preparados y pendientes de servir.

- **Picking:** los diferentes productos pueden diferir en tamaños, y las ubicaciones no tienen una capacidad ilimitada, por lo que cada una de las ubicaciones nombradas anteriormente (sin tener en cuenta la ubicación de zona de carga) tienen una capacidad limitada y los artículos estarán registrados en el sistema con un número de capacidad por unidad. Por lo que, es necesario tener una ubicación con una capacidad ilimitada (relativamente, ya que solo será así a efectos del sistema.

  En realidad, toda ubicación, por muy grande que sea, siempre tendrá una capacidad limitada, pero es responsabilidad del personal administrativo, tener en cuenta que dicha ubicación no llegue a llenarse nunca). Existen varias razones por las cuales utilizar esta ubicación:

  - Cuando llega una mercancía que no cabe en ninguna de las ubicaciones por que supera la capacidad restante de cada una de ellas.

  - Cuando llega una mercancía de gran tamaño que no cabe en ninguna ubicación, aunque estas estén vacías.

- **Zonas de paso:** también llamadas comúnmente "*pasillos*", son zonas de movilidad y operatividad de los operarios, además pueden contar con el posible movimiento de vehículos. Es necesario reservar un espacio en el almacén para permitir el paso de los operarios y vehículos necesarios para la realización de pedidos.

- **Zona de descarga o entrada:** zona destina a descargar y cargar los vehículos logísticos. Tendrá el espacio suficiente para albergar de forma separada e independiente estas dos funciones ya que puede darse el caso (y es habitual) que mientras se está cargando un vehículo con pedidos preparados, al mismo tiempo llegue un vehículo con mercancía a descargar.

Así que, con lo dispuesto en este apartado, puede ser planteado un esquema visual del almacén (ilustración 4) con 47 ubicaciones (de las cuales una es la zona de carga y otra la zona de *picking*) donde pueden apreciarse:

- Zona de carga junto a la entrada del almacén.

- Tres pasillos con las siguientes características:

  - Tres alturas.

  - Cinco localidades.

- Zona de *picking*.

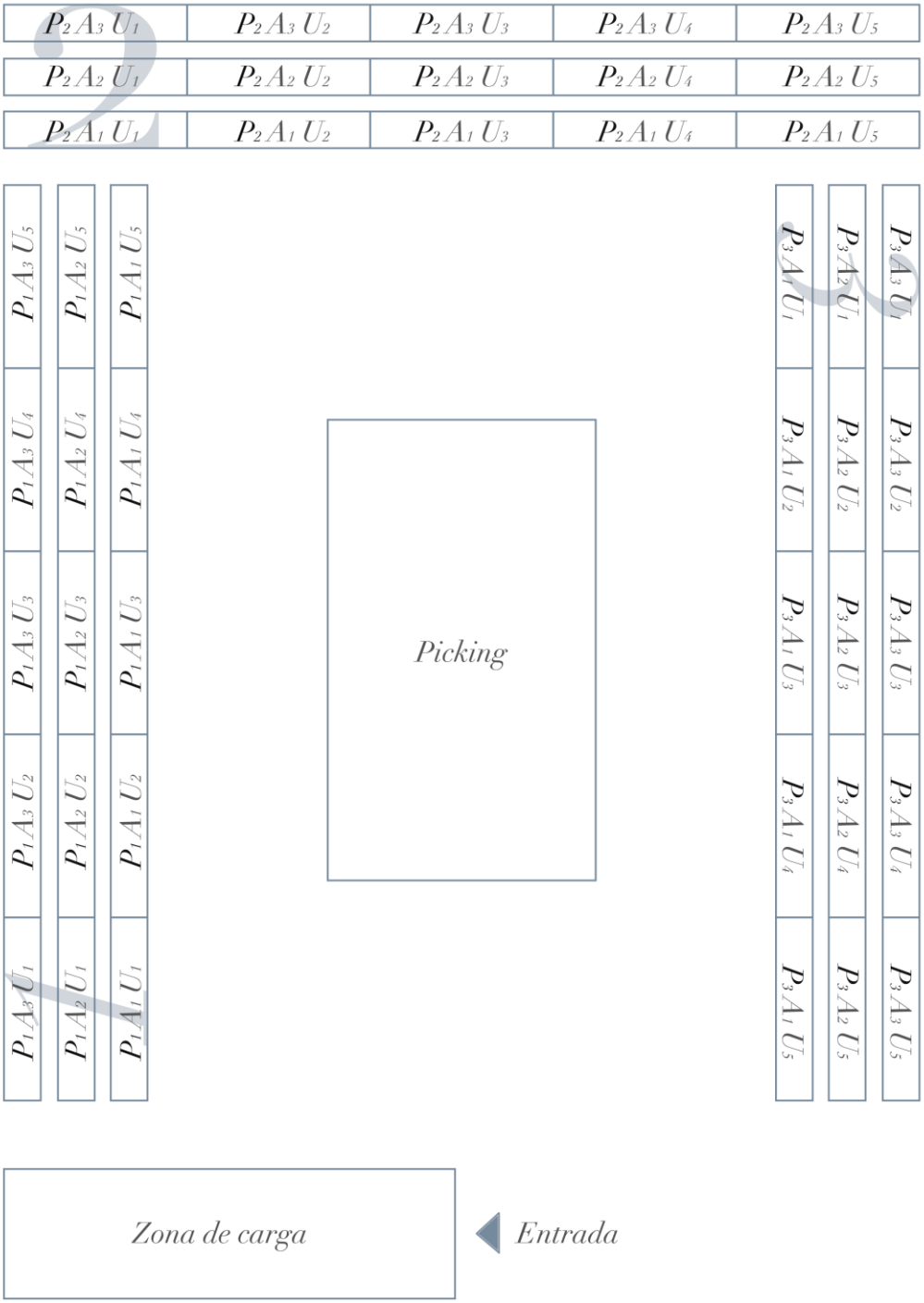


Ilustración 4: Esquema gráfico que representa la estructura y organización del almacén planteado

3.2 Principales procesos del almacén

Los procesos que son realizados en un almacén diariamente son diversos, y existen distintos procesos que a su vez se dividen en varios subprocesos.

Ha de tenerse en cuenta que, muchos de los procesos y subprocesos que a continuación se verán no pueden ser descritos en su totalidad, ya que se ven relacionados con diversos departamentos del entorno empresarial y para entender mejor el funcionamiento del almacén solo se deben hacer referencia a los procesos que conciernen a la operativa del almacén.

Son tres los procesos que se consideran principales y de los cuales se hará estudio de los subprocesos y tareas que los componen. Para ello se hará uso de los diagramas BPMN (*Business Process Model And Notation*):

- **Proceso de entrada de mercancía:** el proceso se inicia cuando un camión o vehículo logístico llega a las inmediaciones del almacén y el conductor entrega al responsable de almacén el Albarán de Compra correspondiente a la mercancía que transporta. A continuación, se cotejan las líneas del albarán recibido con el pedido correspondiente que hizo el departamento de compra en su momento. Una vez realizada la comprobación, un operario cualificado descarga el camión mientras otro comprueba el estado del palet que se está descargando y lo coteja con la línea que le corresponde del albarán recibido. Si existiera cualquier problema, se informaría al departamento de compras, que realizaría las pertinentes gestiones acorde a la incidencia. Una vez descargados todos los palets, se registra el albarán en el sistema para que la mercancía se registre como una entrada de stock en la aplicación y seguidamente se obtiene la lista de ubicaciones donde el sistema, virtualmente, ha alojado dicha mercancía.

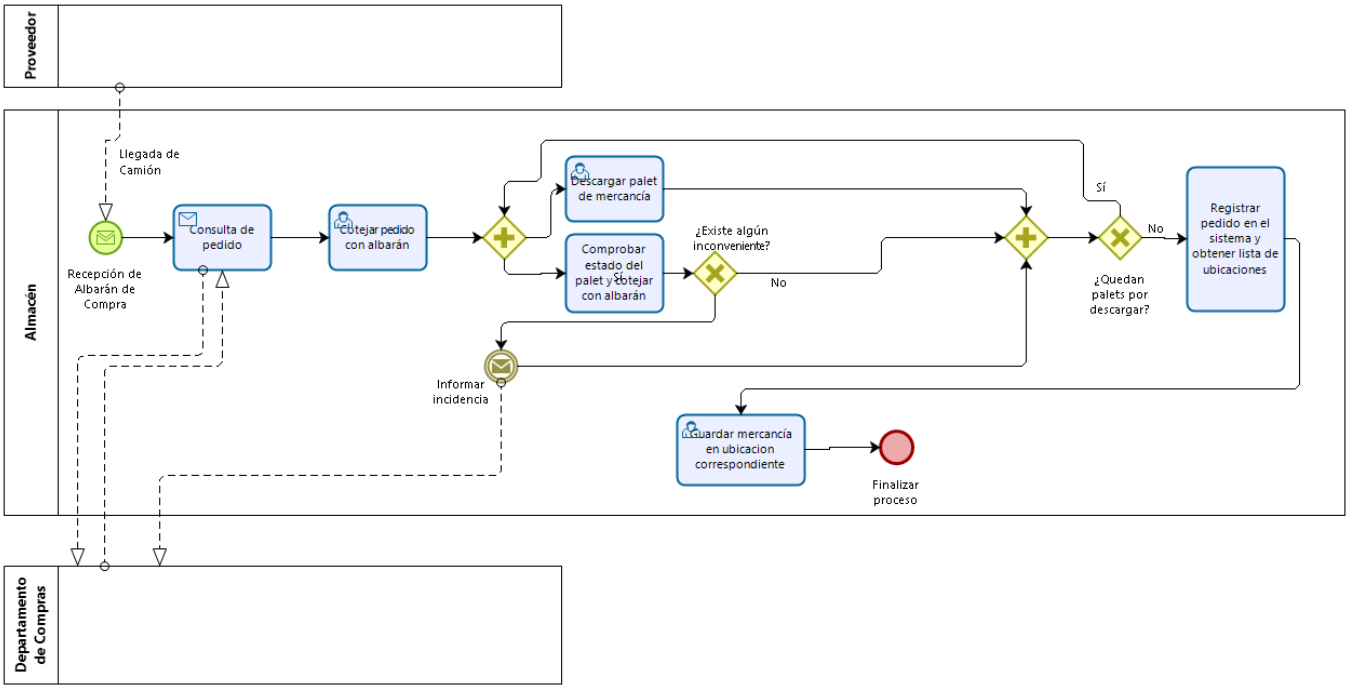


Ilustración 5: BPMN de entrada de mercancía

- **Estudio de situaciones:** en este punto se analizarán las diferentes situaciones que pueden surgir en este proceso. La responsabilidad de algunas de ellas puede recaer, en parte, sobre el proveedor de la mercancía, pero, es importante recalcar, que la empresa compradora es también responsable de la supervisión y almacenaje de la mercancía.

  - La mercancía puede llegar a la zona de descarga en mal estado. En este caso el operario encargado de la tarea debe informar al departamento de compras y este hará las operaciones pertinentes.

  - La mercancía puede dañarse mientras es descargada.

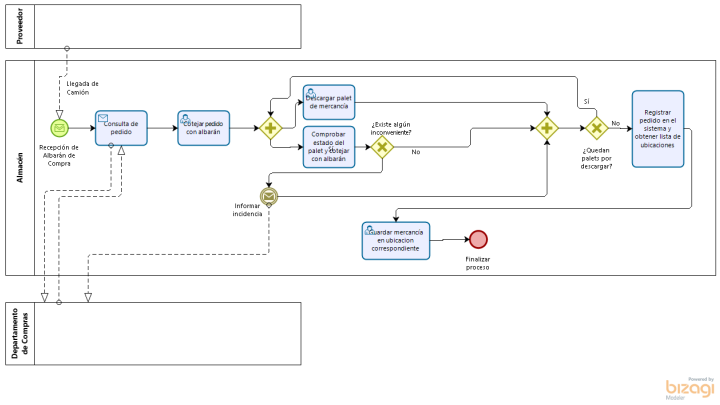
  - El operario que almacena una referencia puede equivocarse al guardarla y almacenarla en otra ubicación.

  - El operario que coteja la mercancía puede equivocarse e introducir en el sistema una referencia o cantidad incorrecta.

  Los errores humanos son difíciles de suprimir por completo, pero con una buena organización y buenas prácticas de funcionamiento de procesos se pueden ver disminuidos significativamente.

Tabla 2: proceso 1

Nombre del proceso:	Entrada de mercancía
Objetivo:	Se registra el albarán de compra y con ello se realizan los registros de entrada de mercancía pertinentes. La mercancía queda guardada en las ubicaciones que indica el sistema al hacer la entrada.
Departamento / Persona Responsable:	Encargado de almacén
Departamento/ Personas Implicadas:	Almacén / Proveedor / Departamento de Compras
Subprocesos: Lista de subprocesos	- Consulta de pedido - Cotejar pedido con albarán - Descargar palet de mercancía - Comprobar estado del palet y cotejar con el albarán - Registrar pedido en el sistema y obtener lista de ubicaciones

	- Guardar mercancía en ubicación correspondiente
Diagrama:	

- **Proceso de gestión de mercancía:** este proceso se inicia cuando un operario desea realizar una acción sobre el stock que se encuentra registrado en el sistema. Este puede hacer un ajuste de stock, donde puede mover la mercancía de una ubicación a otra o ajustar el stock de una ubicación, ya que en ocasiones surgen discrepancias entre lo que se tiene registrado en el sistema y lo que realmente se encuentra en la ubicación.

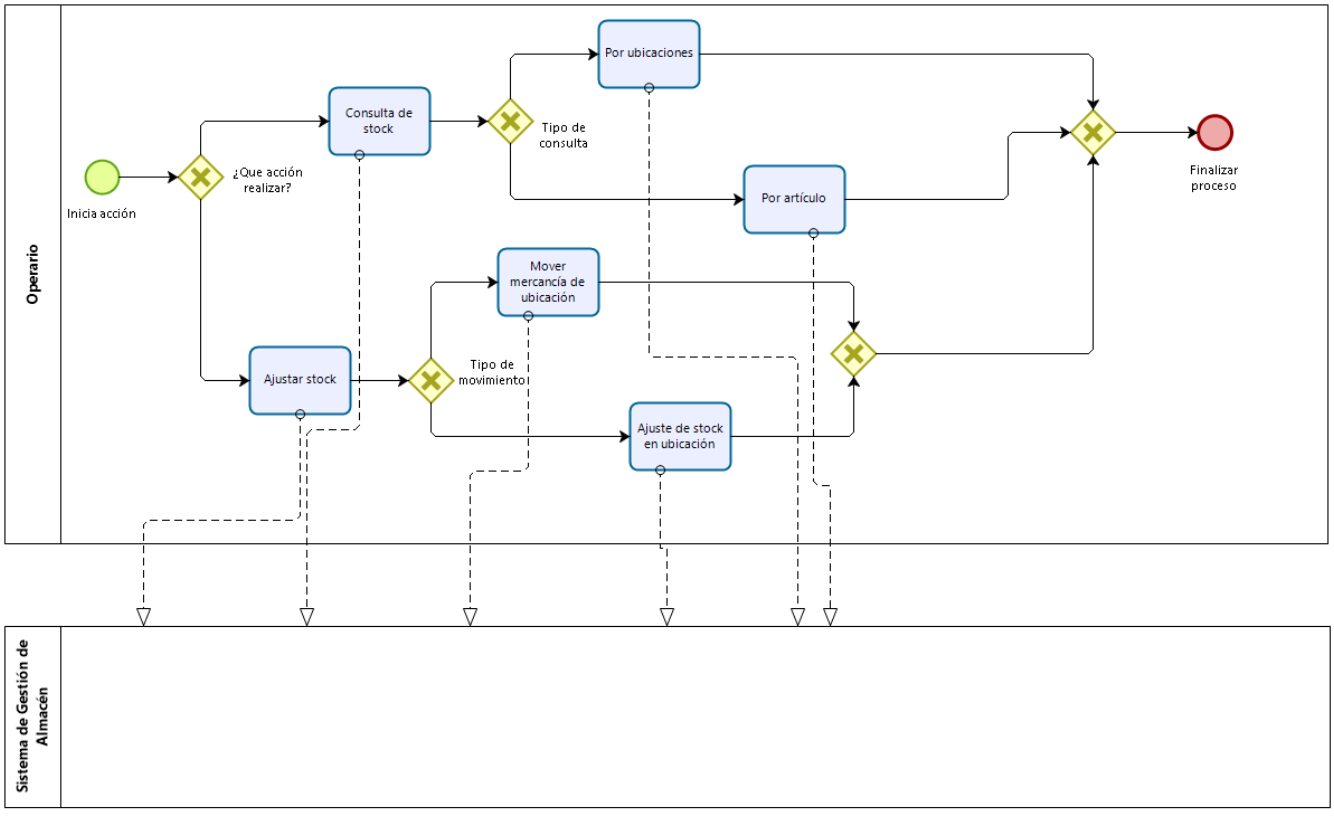
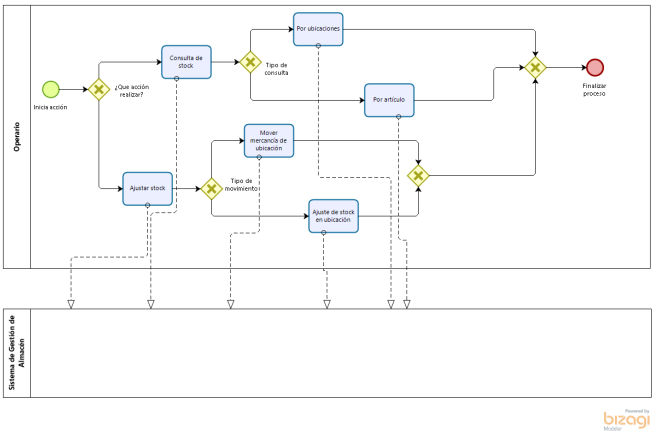


Ilustración 6: BPMN de gestión de mercancía

Además de un ajuste de stock, el operario puede consultar el mismo; y lo puede hacer por ubicaciones, donde aparece toda la mercancía que se encuentra en las mismas o por artículo, donde aparece la cantidad total que existe. en el almacén y en que ubicaciones se encuentra.

- **Estudio de situaciones:** este es un proceso dedicado a realizar un seguimiento general de la actividad del almacén y a solucionar los problemas que generan las diversas situaciones que se producen en el almacén. No presenta situaciones en sí, sino que ofrece una variedad de herramientas que permiten llevar un control de la mercancía que se aloja en el almacén.

Tabla 3: proceso 2

Nombre del proceso:	Gestión de mercancía
Objetivo:	Permite realizar un seguimiento de los artículos albergados en el almacén. Además, también permite realizar un traspaso de mercancía entre ubicaciones o almacenes, en caso de requerir una regularización del stock.
Departamento / Persona Responsable:	Encargado de almacén
Departamento/ Personas Implicadas:	Operario / Sistema de gestión de almacén
Subprocesos: Lista de subprocesos	- Consulta de stock por ubicación - Consulta de stock por artículo - Mover mercancía de ubicación - Ajuste de stock en ubicación
Diagrama:	

- **Proceso de salida de mercancía:** este es un proceso que se lanza cuando se ve iniciada la jornada de trabajo del almacén. Siempre habrá un número de operarios destinados a realizar exclusivamente este proceso ya que es aquel que genera directamente los beneficios en la empresa.

  Siempre que existan pedidos a realizar, los operarios deben tomar cada uno de ellos, siguiendo un orden preestablecido y atacando cada una de las líneas de forma ordenada tal y como aparece en el listado que proporciona el sistema. Una vez el operario hace enfoque en un pedido va línea por línea recorriendo las ubicaciones indicadas y recogiendo la mercancía que corresponde a cada una de las líneas, aunque puede darse el caso que, llegados a una ubicación, la mercancía no esté disponible, aunque en el sistema indique que sí. Eso quiere decir que, existe un desajuste en el stock, por lo cual se debe informar al departamento de ventas para que realice una modificación en el pedido y reajuste el stock de la referencia indicada. Una vez finalizada la recogida de productos, el operario debe depositar los mismos en la zona de carga para que los operarios del departamento de logística la carguen en el vehículo correspondiente. Cuando se hayan terminan de realizar todos los pedidos de la jornada, y ya no existan más a realizar, el proceso se dará por terminado.

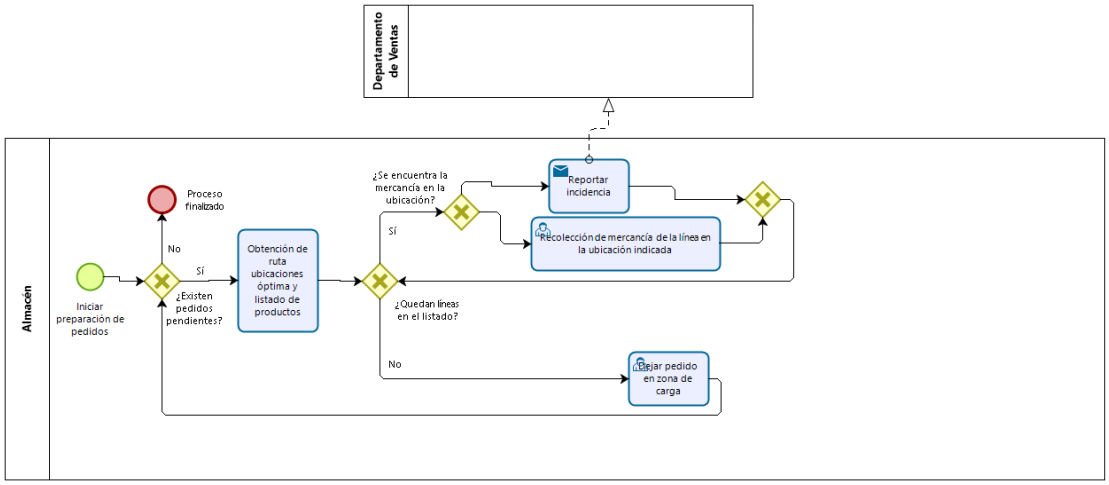


Ilustración 7: BPMN de salida de mercancía

  - **Estudio de situaciones:** en este punto se analizan los principales problemas que pueden surgir desde que el operario empieza a preparar un pedido hasta que éste lo deposita en la zona de carga.

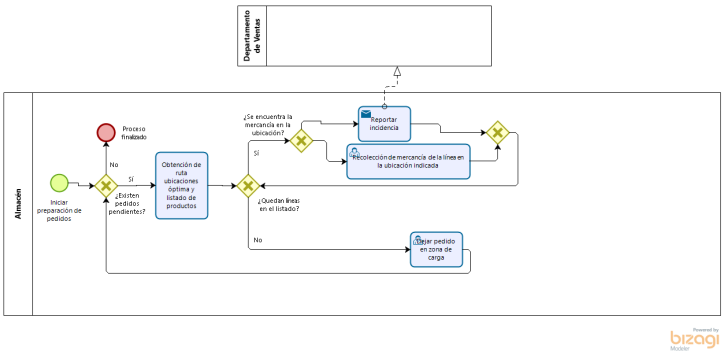
    - El operario acude a la ubicación dada a recoger una referencia y se equivoca con la cantidad o con la referencia.

    - El operario acude a la ubicación dada a recoger una referencia y la misma no se encuentra. Hay un error en el stock.

    - El operario que se encarga de cargar la mercancía al camión (departamento de logística) se equivoca al llevarse un pedido que no es de su ruta de reparto.

Como se ha podido analizar antes, los errores humanos son difíciles de controlar, pero algunos de los ahora comentados son fáciles de detectar cuando se aumentan los controles de stock en el proceso de consulta de mercancía.

Tabla 4: proceso 3

Nombre del proceso:	Salida de mercancía
Objetivo:	Se da salida a la mercancía en forma de pedido destinado a ser servido al cliente. Los operarios destinados a ello realizan la preparación de pedidos visitando las ubicaciones en el orden que indica el sistema mediante un listado estratégicamente ordenado para abordar la organización caótica del almacén. Una vez realizados los pedidos, estos se van depositando uno a uno en la zona de carga.
Departamento / Persona Responsable:	Encargado de almacén
Departamento/ Personas Implicadas:	Almacén / Departamento de Ventas
Subprocesos: Lista de subprocesos	- Obtención de ruta ubicaciones óptima y listado de productos - Reportar incidencia - Recolección de mercancía de la línea en la ubicación indicada - Dejar pedido en zona de carga
Diagrama:	

Hasta el momento, se ha podido analizar los tres principales procesos que se gestionan diariamente en el almacén modelo planteado para el proyecto. Este análisis permite ver los puntos operativos que pueden ser mejorados, pero, como se ha visto, existen algunos que dependen mucho del rigor y esmero de los operarios.

Los procedimientos (o procesos) de organización del almacén son un elemento clave a tener en cuenta en la gestión del mismo ya que permiten elevar la eficacia operativa del día a día. El concepto caótico (o desordenado) que se ha planteado permite economizar estos procesos de los siguientes modos:

- No se precisa ubicar la mercancía de forma ordenada cuando esta entra en el almacén.

- No es necesario organizar el almacén por secciones o grupos de artículos, por lo cual tampoco es necesario que el encargado de la tarea organice y controle la mercancía que se halla en el mismo.

- La ruta de obtención de artículos de un pedido es la óptima ya que se obtiene mediante el cálculo del camino mínimo. Este punto es el más importante de todos ya que la economización de este proceso aporta:

  - Disminución del proceso de preparación de pedidos, con lo que se aprovecha mucho mejor el tiempo preparando más pedidos en el mismo plazo de tiempo (comparando con el método convencional, referido al método de organización de almacén más utilizado hasta el momento, organizado los artículos en ubicaciones preestablecidas).

La organización caótica presenta como ventaja operativa, el gran ahorro en la organización del almacén y como ventaja productiva, la conversión de un tiempo mal aprovechado para preparar pedidos en tiempo eficaz respecto a la organización convencional.

4. Análisis de requisitos

En este apartado serán definidas las distintas funcionalidades necesarias para el cumplimiento del proyecto. Para poder abordar la implementación de la aplicación de una forma estructurada es necesaria la recopilación de dichos requisitos.

La definición de requisitos se estructurará en cuatro puntos. En primer lugar, se definirán los requisitos previos, después los de inicio de sesión, pantalla de menú, entrada de mercancía, gestión de mercancía, y salida de mercancía.

- **Requisitos previos:**

Tabla 5: requisito 1

Número	1
Requisito	La aplicación será multiempresa
Prioridad	Alta
Descripción	La aplicación permitirá la gestión de varias empresa o multiempresa. Para ello todas las pantallas de la aplicación deberán tener instanciada una variable con el código de empresa. Todas las transacciones que realiza la aplicación con la base de datos precisarán del código de empresa, ya que con, esto cada transacción opera en una gestión de empresa diferente.

Tabla 6: requisito 2

Número	2
Requisito	La aplicación permitirá la gestión de varios almacenes
Prioridad	Alta
Descripción	La aplicación permitirá la gestión de varios almacenes por empresa. Para ello todas las pantallas de la aplicación que gestionen una operación que referencia un artículo deberán tener instanciada una variable con el código de almacén.

- **Inicio de sesión:**

Tabla 7: requisito 3

Número	3
Requisito	Tomar usuario y contraseña en pantalla de inicio
Prioridad	Alta
Descripción	Será de carácter obligatorio estar identificado en el sistema. La pantalla inicial deberá estar compuesta de dos campos de texto necesarios para que el usuario pueda identificarse, además también habrá dos botones; el primero permitirá validar los datos y entrar en el sistema, y el segundo cancelará la acción y cerrará la aplicación. El usuario y contraseña estarán previamente definidos en la base de datos. El operario deberá introducir sus credenciales para poder acceder a la pantalla de menú de la aplicación y hacer las gestiones que allí se ofrecen. Además, será necesario introducir el código de empresa mediante un elemento de entrada de texto.

- **Menú principal**

Tabla 8: requisito 4

Número	4
Requisito	Botones de menú para acceder a las pantallas principales
Prioridad	Media
Descripción	La pantalla de menú estará dotada con tres botones que darán acceso a las pantallas referentes a los tres principales procesos. Un primer botón para permitir la entrada de mercancía en el sistema en forma de albarán de proveedor. Un segundo botón que permitirá gestionar el stock existente en el almacén. Y un tercer botón, que permita la salida de mercancía mediante un albarán de venta.

- **Entrada de mercancía:**

Tabla 9: requisito 5

Número	5
Requisito	Introducir albarán de compra
Prioridad	Media
Descripción	La mercancía que llega al almacén, lo hace a través de un vehículo logístico y es entregada a su destinatario junto con un albarán de compra, así que, una pantalla será necesaria para albergar los elementos que harán posible dicha entrada. También serán necesarios una serie de elementos de entrada de texto que realizarán una función de filtrado para poder instanciar un único albarán. Además, será preciso introducir un elemento cuadrícula para mostrar las líneas de este.

Tabla 10: requisito 6

Número	6
Requisito	Introducir datos del albarán en el sistema
Prioridad	Media
Descripción	Para poder introducir los datos de las líneas de detalle del albarán, será necesario introducir un botón en la pantalla de entrada de mercancía que inicie las gestiones de entrada oportunas. Además, dichas gestiones realizarán los movimientos de mercancía oportunos.

Tabla 11: requisito 7

Número	7
Requisito	Consultar el stock de una ubicación
Prioridad	Baja
Descripción	En la pantalla de gestión de mercancía se podrá consultar el stock actual de una ubicación y almacén dados. Para ello, será necesaria la introducción dos elementos de entrada de texto para el almacén y la ubicación, un tercer elemento de cuadrícula para mostrar los datos resultantes y un botón para mostrar dichos datos.

- **Gestión de mercancía:**

Tabla 12: requisito 8

Número	8
Requisito	Consultar el stock de un artículo
Prioridad	Baja
Descripción	En la pantalla de gestión de mercancía se podrá consultar el stock actual de un artículo en un almacén dado. Para ello, será necesaria la introducción dos elementos de entrada de texto para el almacén y el artículo, un tercer elemento de cuadrícula para mostrar los datos resultantes y un botón para mostrar dichos datos.

Tabla 13: requisito 9

Número	9
Requisito	Mover mercancía de ubicación
Prioridad	Media
Descripción	Ubicar en la pantalla de gestión de mercancía una zona en la cual poder registrar un traspaso de la cantidad de un artículo de una ubicación a otra. Para ello, serán necesarios cinco elementos de entrada de texto que harán referencia al almacén, las ubicaciones origen y destino, la cantidad y el artículo respectivamente. Además, un botón realizará el inicio de la transacción.

- **Salida de mercancía:**

Tabla 14: requisito 10

Número	10
Requisito	Obtener los datos de un albarán de venta y la lista de ruta
Prioridad	Alta
Descripción	Será necesaria una pantalla para mostrar el detalle de líneas de un albarán de venta dado. Para ello, como elementos de entrada, serán necesarios aquellos campos que puedan instanciar un albarán. Los resultados serán mostrados en un elemento cuadrícula y un botón será necesario para refrescarlos. La ruta de ubicaciones y cantidades será obtenida mediante un botón que permita iniciar los trámites de preparación del pedido. Dichos trámites también realizarán los movimientos de almacén oportunos e iniciarán el algoritmo de búsqueda de ruta mínima.

5. Diseño

En este apartado se describirán tanto la arquitectura escogida para el diseño de la aplicación como la base de datos relacional que lo soporta.

5.1 Arquitectura cliente-servidor

En esta aplicación, se ha optado por seguir una arquitectura cliente-servidor.

"La arquitectura cliente-servidor es un modelo de diseño de *software* en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da la respuesta." [4]

La separación lógica entre el cliente y el servidor genera un balanceo sobre la carga computacional de la aplicación. Esta arquitectura ha cambiado el almacenamiento y la gestión de datos del escritorio a una aplicación de red, ya que permite centralizar la gestión de datos y pasar muchas de las aplicaciones *CRUD* (procesamiento de búsquedas) del equipo cliente al servidor con las bases de datos. Con esto, la comunicación entre un equipo proveedor y otro cliente es mucho

Antiguamente, cada cliente mantenía una conexión exclusiva con el equipo servidor de la base de datos, lo cual limitaba la escalabilidad de la aplicación La lógica de negocio alojada en el cliente y el código utilizado para acceder a la base de datos no podía ser compartido con otras

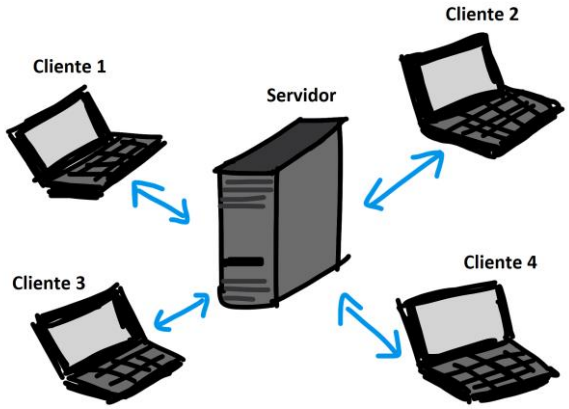


Ilustración 8: diagrama de red local (Licencia: dominio público)

aplicaciones.

El tipo de arquitectura cliente-servidor que será usada para crear la aplicación será la arquitectura de tres capas:

1. Capa de presentación: se encarga de la representación de la información. Los equipos cliente que solicitan recursos, estarán equipados con una interfaz de usuario para la representación de estos.

2. Capa de negocio: se encarga de la logística de la información Es donde se dice que se hace con los datos. La aplicación diseñada en este proyecto tendrá un repositorio con librerías (que actuará como servidor intermedio) capaces de realizar la conexión con la base de datos del servidor y tratar ficheros externos.

3. Capa de datos (o persistencia): se encarga de guardar los datos y gestionar todo lo relativo a la base de datos y a la creación, edición y borrado de esta. En la aplicación habrá un equipo servidor, que proporcione y gestione los datos de la aplicación.

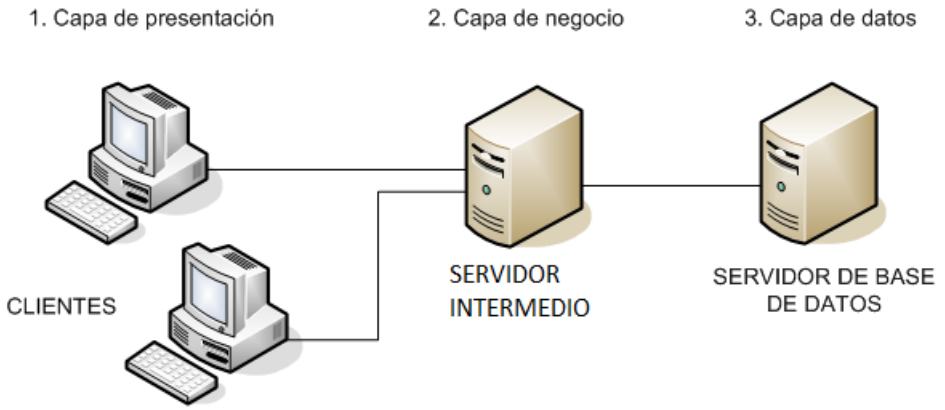


Ilustración 9: esquema arquitectura de tres capas (Licencia: dominio público)

El diagrama de red de la aplicación a desarrollar en este proyecto podría plantearse como el de la siguiente imagen (ilustración 10). Este es un ejemplo en el que tres clientes distribuidos geográficamente en tres puntos distintos del planeta y con conexión a Internet, podrán conectarse al servidor y utilizar la aplicación y los datos sin problema. La conexión de los equipos cliente y servidor a la red de Internet mediante la utilización de servicios y la aplicación de protocolos hace posible el funcionamiento de la aplicación sin necesidad de depender de una red de área local.

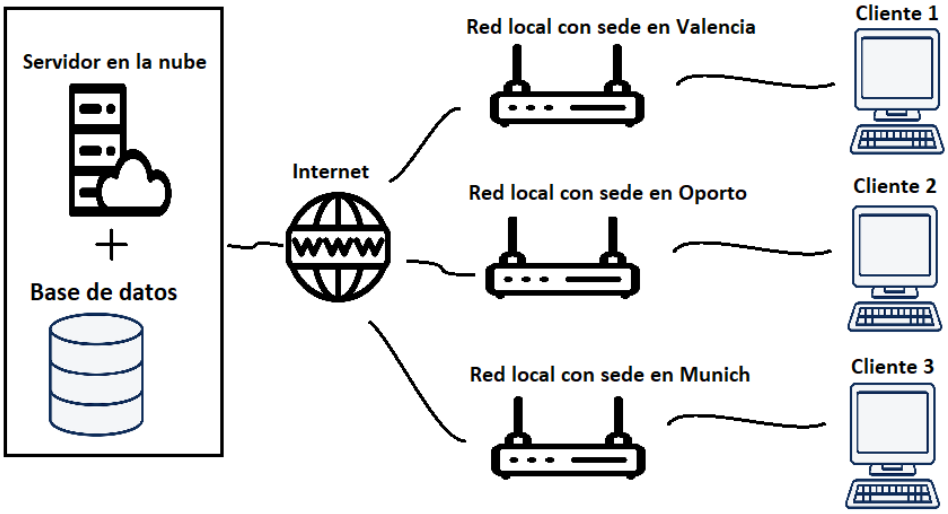


Ilustración 10: diagrama de red planteado

5.2 Base de datos relacional

La aplicación requerirá de un gestor de bases de datos relacional que almacene la información necesaria para que la gestión de los datos sea lo más eficiente.

En la actualidad, existen diferentes gestores de bases de datos en el mercado. Elegir uno u otro dependerá de cuál de ellos se adapta mejor a la idea funcional de la aplicación, dependiendo también de la arquitectura cliente-servidor. Aun realizando este filtrado, el abanico es bastante extenso. El gestor escogido debe estar expandido entre muchas compañías ya que esto permitirá la fácil adaptación de la aplicación a otros aplicativos de gestión empresarial externos, y así, junto a otros aplicativos destinados a suplir otras áreas empresariales, pueden formar un sistema de gestión empresarial completo.

Los dos gestores más populares son: *Microsoft Sql Server* y *Oracle Data Base* . Las diferencias más significativas entre ambos son:

Tabla 15:diferencias entre Oracle Data Base y Microsoft Sql Server

	Gestores de bases de datos	
Propiedad	**Oracle Data Base**	**Microsoft Sql Server**
Programación	PL/SQL	T-SQL
Sintaxis	Simple	Compleja pero más eficiente
Soporte Sistema Operativo	Multiplataforma	Solo disponible en Windows
Optimización	No dispone	Usa el método de optimización de consulta en estrella
Interfaz de usuario	Simple	Compleja
Transacciones	Las transacciones no se confirman hasta que es ejecutado el comando *COMMIT*.	Si no son ejecutados los comandos apropiados, las transacciones se ejecutan y son confirmadas individualmente

Además, *Oracle Data Base* está orientado a trabajar con grandes bases de datos y proporciona unas estrategias de recuperación de la base de datos que pueden ser automatizadas. Dichas estrategias permiten devolver la base de datos a un estado consistente.

Las inconsistencias aparecen cuando una transacción no ha sido procesada correctamente.

Procesar correctamente una transacción significa:

a) todas las operaciones de la transacción se ejecutan con éxito y sus cambios (actualizaciones) quedan grabados permanentemente en la base de datos,

o bien

b) la transacción no tiene ningún efecto en la base de datos.

Al menos debe cumplirse uno de estos dos objetivos para garantizar el correcto procesamiento.

Los elementos utilizados en el procesamiento de transacciones y la estrategia de recuperación en el sistema gestor de base de datos de Oracle son: buffer de datos, buffer de diario, fichero de diario, espacio deshacer.

Los bloques del espacio deshacer guardan un histórico de movimientos de las últimas actualizaciones. Al realizar una operación de recuperación en la base de datos, este fichero será un elemento clave en la estrategia de recuperación escogida y dependiendo de la operación podrá rehacer la operación, deshacerla o no realizar ninguna acción sobre ella.

Oracle utiliza una organización particular para gestionar el espacio disponible en disco. El espacio que está destinado a datos dentro del disco se organiza en espacios de distinto tamaño llamados "*tablespace*".

Es necesario que, antes de crear la base de datos y sus elementos, se realice un modelo relacional de las tablas que estarán involucradas en la misma. Este modelo es una herramienta usada en modelado y gestión de bases de datos y está basada en la lógica de predicados y la teoría de conjuntos.

Una de las ventajas obtenidas al realizar el modelo de datos es que favorece la normalización por ser comprensible y aplicable. La normalización de una base de datos es un proceso que consiste en designar y aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelado entidad-relación al modelo relacional.

La ilustración 11 muestra el modelo de datos creado con la herramienta "*Sql Developer Data Modeler*". Este modelo servirá como referencia para realizar la implementación tanto de la base de datos y sus elementos como de los demás componentes de la aplicación.

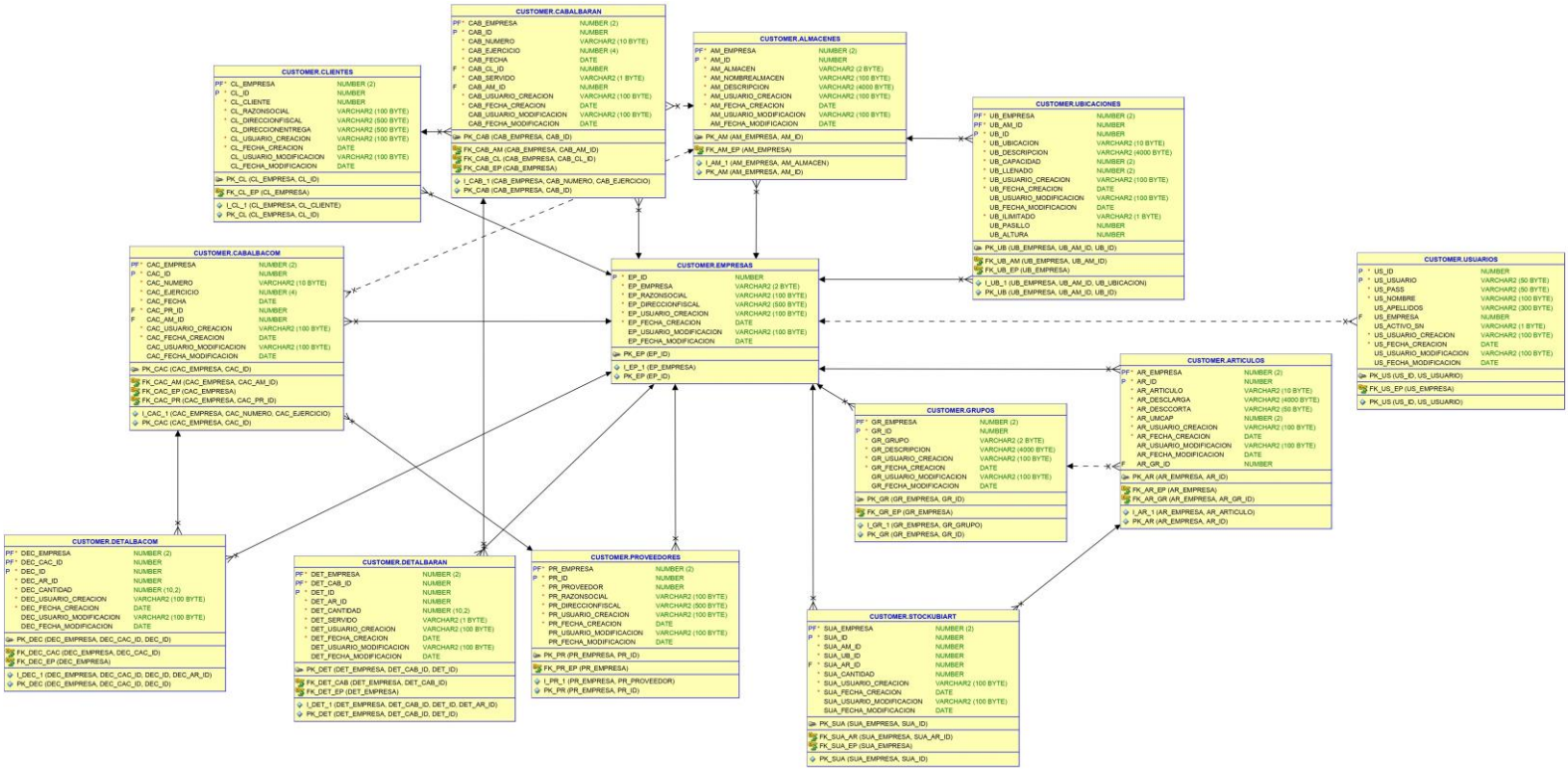


Ilustración 11: modelo de datos realacional

Los tipos utilizados y referenciados en los campos utilizados en el modelo son:

- *Number*: almacena números fijos y en punto flotante. Admiten hasta 38 dígitos y opcionalmente se le puede indicar la precisión (número total de dígitos) y la escala (número de dígitos a la derecha de la coma, decimales, los cogerá de la precisión indicada).

- *Varchar2*: cadena de caracteres alfanuméricos de longitud variable. Entre paréntesis se especifica un límite de número de *bytes*. El tamaño del campo dependerá del valor que contenga.

- *Date*: almacena un punto en el tiempo (fecha y hora). Los datos se almacenan en campos de longitud fija de siete octetos cada uno, correspondiendo al siglo, año, mes, día, hora, minuto, y al segundo.

6. Implementación

En este apartado se describirán los elementos necesarios para poder implantar la aplicación en un sistema *cliente-servidor*.

6.1 Base de datos

Para la elaboración de este proyecto se usará la siguiente versión del gestor de bases de datos *Oracle Database*: "*Oracle Database 18c Express Edition*". Además, solo se precisará de un "*tablespace*" que será nombrado como "*CUSTOMER*".

Para que la base de datos adquiera una estructura funcional, necesitará componerse de una serie de elementos lógicos y únicos, donde cada uno de ellos realizará una función concreta:

- Tablas: se refieren al tipo de modelado de datos donde son guardados los datos recogido por el programa. Se componen de dos estructuras, campos y registros. El campo se refiere al nombre de la columna, debe tener un tipo de dato asociado y además ha de ser único en la tabla. El registro se refiere a cada una de las filas que componen la tabla donde se integran los datos y registros que eventualmente pueden ser nulos si funcionalmente se requiriera. Las tablas pueden ser persistentes en el tiempo o temporales, y esto depende de la finalidad funcional que tienen los datos que las albergan. Según el modelo de datos construido, las tablas necesarias son:

    - **USUARIOS:** alberga los datos de registro de los usuarios. Estos son necesarios para validar al usuario que intenta iniciar sesión en el programa.

    - **EMPRESAS:** alberga los datos que identifican cada una de las empresas registradas en la empresa.

    - **ALMACENES:** alberga los datos que identifican cada uno de los almacenes que hay en la empresa.

    - **UBICACIONES:** alberga los datos que identifican cada una de las ubicaciones registradas en un almacén y empresa dados. Las ubicaciones tendrán un límite de capacidad identificada mediante un campo. Además, otros datos como la altura y el pasillo serán necesarios para poder identificar físicamente la ubicación.

    - **ARTICULOS:** alberga los datos que identifican cada una de las diferentes referencias que tiene registradas una empresa. Cada artículo tendrá identificado mediante un campo la capacidad que ocupa por unidad.

    - **GRUPOS:** alberga los datos que identifican cada uno de los grupos de artículos que puede haber en una empresa. El concepto de grupo o familia es utilizado para para poder agrupar la información de artículos por grupos.

    - **STOCKUBIART:** alberga los datos que identifican cada uno de los registros de entrada de mercancía que se encuentran disponibles en el almacén. Cuando un artículo es

sustraído de una ubicación dada, la entrada de mercancía que le referenciaba es eliminada de esta tabla. El agrupamiento de los diferentes registros y la operativa adecuada permiten controlar el stock actual del almacén.

- **PROVEEDORES:** alberga los datos que identifican cada uno de los proveedores de productos registrados en la empresa.

- **CABALBACOM:** alberga los datos que identifican cada una de las cabeceras que se referencian con albaranes de compra realizados a proveedores. Cada cabecera es referenciada por un albarán y contiene los datos identificativos del documento, los datos fiscales del comprador y los del vendedor.

- **DETALBACOM:** alberga los datos que identifican cada una de las líneas de detalle de albarán pertenecientes a una cabecera referenciada a un albarán de compra dado.

- **CLIENTES:** alberga los datos que identifican cada uno de los clientes registrados en la empresa.

- **CABALBARAN:** alberga los datos que identifican cada una de las cabeceras que se referencian con albaranes de venta destinados a clientes. Cada cabecera es referenciada por un albarán y contiene los datos identificativos del documento, los datos fiscales del comprador y los del vendedor.

- **DETALBARAN:** alberga los datos que identifican cada una de las líneas de detalle de albarán pertenecientes a una cabecera referenciada a un albarán de venta dado.

Todas las tablas disponen de un campo "*ID*" (numérico) utilizado para identificar e indexar un registro de estas, pero solo la tabla empresa dispone de una unicidad con este único campo. Las demás tablas deben conseguir la unicidad combinando el campo "*ID*" con otros por cuestiones estructurales. Este campo se usará como herramienta interna para indexar los registros ya que redunda en ventajas de rendimiento y será totalmente transparente al usuario. Pero, para que el usuario pueda identificar los registros, habrá en cada tabla otro campo (de tipo alfanumérico, no tiene por qué ser numérico. De esta manera el cliente identificará mejor los registros agregando una codificación más amigable a estos) con la misma combinatoria de unicidad que el campo "*ID*" y será elegido por el usuario que realiza la entrada del registro en el sistema. Esta separación hace que el sistema trabaje con índices numéricos y la indexación es mucho más rápida y eficiente.

Otra peculiaridad de las tablas es que cada una de ellas dispondrá de cuatro registros de auditoria: usuario de creación, fecha de creación, usuario de modificación y fecha de creación. Con esto, se puede identificar al usuario que ha creado el registro y al último que lo ha modificado, además de sus respectivos valores de tiempo.

Y como última particularidad, todas las tablas, a excepción de las tablas *EMPRESA* y *USUARIO*, tendrán como primer índice de la tabla el *ID* de empresa. Al ser una aplicación multiempresa, se debe separar el conjunto de registros por empresa.

La tabla de usuarios no se rige por esta última premisa ya que cada usuario puede escoger la empresa con la que quiere operar al iniciar la aplicación, es decir, los usuarios no pertenecen a una sola empresa, sino a todas las que se gestionan en la aplicación (aunque normalmente se gestione una única empresa, esto serviría para gestionar un grupo de empresas).

- Secuencias: son bloques escritos en código PL/SQL usados para generar números enteros. Son muy útiles en la generación automática de valores de clave primaria ya que pueden autoincrementarse, independientemente de que la transacción se confirme o retroceda. Cada una de los "*triggers*" que han sido creados para la aplicación tienen una secuencia para incrementar el campo "*ID*" de la tabla a la cual se asocian.

- Disparadores o *triggers*: son bloques escritos en código PL/SQL que se asocian a una tabla y se ejecutan automáticamente cuando se producen ciertos eventos asociados a la tabla. Son muy útiles en la prevención de transacciones erróneas e implementación de restricciones de integridad o seguridad.

  Cada una de las tablas de la base de datos de la aplicación tienen un "*trigger*" asociado y será ejecutado (o disparado) justo antes de insertar los datos. Cuando se usa la cláusula "*INSERT*", el campo "*ID*" se informar en la misma, pero con el uso de un "*trigger*" esto puede evitarse, ya que, a través del uso de la programación, se rellenará ese campo con el uso de un elemento secuencia.

- Paquetes: son bloques escritos en código PL/SQL que sirven para agrupar y organizar funcionalidades en la base de datos. Estos se componen principalmente de un elemento especificación (o cabecera) y un elemento cuerpo. El primero establece la interfaz de la funciones y procedimientos que se van a implementar en el cuerpo, y el segundo no es más que la implementación de lo especificado en el elemento anterior. La base de datos de la aplicación está compuesta por diferentes paquetes agrupados por funcionalidades:

  - **FUNC_ALMACENES:** contiene los bloques de código que afectan directamente a los registros de la tabla de mantenimiento de almacenes. Dependiendo de la operación escogida, la información puede ser consultada, modificada o eliminada.

  - **FUNC_ARTICULOS:** contiene los procedimientos y funciones asociados al tratamiento de los registros de la tabla de mantenimiento de artículos. El acceso a esta tabla solo permite la consulta, no puede modificarse, ya que los datos denominativos de los artículos solo sirven de referencia en la operativa del almacén. Otro departamento de la empresa será el encargado de realizar cambios sobre estos registros. Además, el usuario no tiene competencia administrativa para modificarlas.

  - **FUNC_CLIENTES:** contiene los procedimientos y funciones asociados a la consulta de registros de la tabla clientes. Los registros de dicha tabla no pueden modificarse, ya que los datos de clientes se utilizan para referenciar los datos de cliente de un albarán de venta y no están fuertemente relacionados con el stock del almacén. Además, el usuario no tiene competencia administrativa para modificar dichos registros.

- **FUNC_VENTAS:** contiene los procedimientos y funciones asociados al proceso de expedición de mercancía mediante un albarán de venta. Dicho proceso afecta a las tablas mantenimiento de stock y albaranes de venta.

- **FUNC_EMPRESAS:** contiene los procedimientos y funciones que referencian directamente los registros de la tabla de mantenimiento de empresas. Esta tabla tampoco puede ser modificada, ya que el usuario de la aplicación no tendrá competencia administrativa para ello, solo para consultarla.

- **FUNC_UBICACIONES:** contiene los procedimientos y funciones que afectan directamente a los registros de mantenimiento de la tabla de ubicaciones. Dependiendo de la operación puede insertarse una nueva ubicación, modificar una ya existente o eliminar una ubicación existente.

- **FUNC_PROVEEDORES:** contiene los procedimientos y funciones asociados a la consulta de registros de la tabla proveedores. Los registros de dicha tabla no pueden modificarse, ya que los datos de proveedores se utilizan para referenciar los datos de proveedor de un albarán de compra y no están fuertemente relacionados con el stock del almacén. Además, el usuario no tiene competencia administrativa para modificar dichos registros.

- **FUNC_COMPRAS:** contiene los procedimientos y funciones asociados al proceso de incorporación de mercancía mediante un albarán de compra. Dicho proceso afecta a las tablas mantenimiento de stock y albaranes de compra.

- **FUNC_UTIL:** contiene los procedimientos y funciones que no se asocian a ninguno de los apartados anteriores, sino que más bien son de ámbito general o de apoyo a otras funcionalidades.

6.2 Requisitos mínimos de implantación

La programación implantada en los paquetes de la base de datos ha sido planifica con el fin de disminuir al máximo la carga de cómputo del cliente y maximizar la del servidor. Por lo tanto, los requisitos mínimos del equipo servidor (donde está alojada la base de datos) son más exigentes que en el equipo cliente.

En las siguientes tablas se describen los requisitos mínimos recomendados de *hardware* que la aplicación demanda para que el rendimiento de la misma sea eficiente.

Tabla 16: requisitos de hardware mínimos recomendados

Requisitos *Hardware*		
Requisito	**Cliente**	**Servidor**
Arquitectura del procesador	32 bits	64 bits
Número de hilos del procesador	5	12
Número de núcleos del procesador	5	6
Frecuencia del procesador	2 gigahercios	4 gigahercios
Velocidad tarjeta de red	100 gigabits por segundo	1000 gigabits por segundo

Tabla 17: requisitos de software mínimos recomendados

Requisitos S*oftware*		
Requisito	**Cliente**	**Servidor**
Sistema Operativo	Windows 10 Professional Edition	Windows 10 Profesional Edition
.NET Framework	Versión 4.7.2	Versión 4.7.2
Microsoft Excel	Versión 2019 o 365	No es necesario.
ODBC	Instant Client ODBC 19c	Instant Client ODBC 19c
Oracle Database	No es necesario.	Versión 12c

Estos valores no son exactos y están pensados para una empresa que disponga de tres almacenes, cincuenta ubicaciones y cincuenta usuarios (clientes) conectados a la vez en una jornada diaria de trabajo (8 horas), ya que dependiendo del volumen de estos habrá requisitos que puedan prescindir de otros requerimientos más elevados por parte del equipo servidor.

Las versiones de las aplicaciones *software* de terceros que se han elegido son aquellas que disponen de un servicio de mantenimiento por parte de los propietarios de estas.

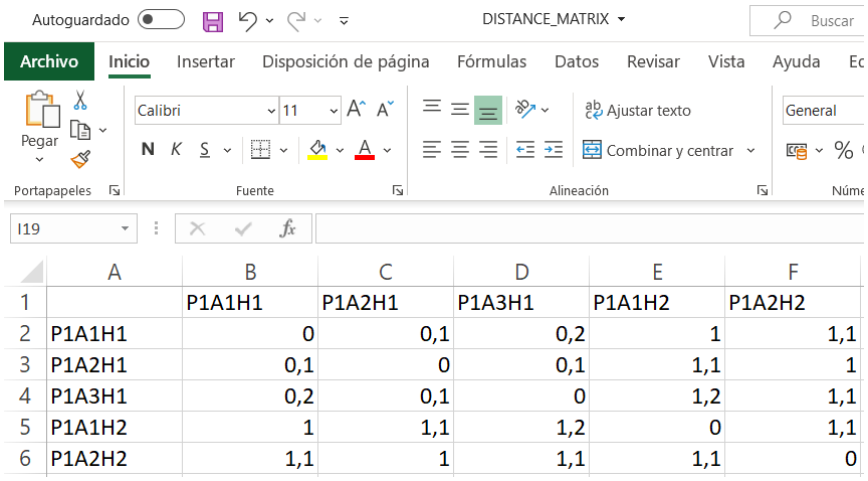
6.3 Matriz de distancias

La aplicación requiere de un fichero de Microsoft Office Excel que contiene la matriz de distancias entre ubicaciones. Esta es una matriz cuyos elementos representan las distancias entre dos ubicaciones. Además, es una matriz simétrica de tamaño NxN (dado un conjunto de N puntos) conteniendo números reales no negativos como elementos.

El fichero Excel contiene como elementos cada una de las ubicaciones de que dispone un almacén, así que, es necesario tener una matriz de distancias por cada almacén de la empresa que se gestione de manera caótica.

El "*solver*" de Google funciona con número enteros, por lo que si las distancias entre las ubicaciones no son números enteros, se deberá redondear para que lo sean. Para evitar la pérdida de precisión en los resultados provocada por el redondeo, las distancias se almacenen en centímetros.

La ilustración 12 muestra una submatriz extraída de la matriz utilizada para la realización de pruebas.



	A	B	C	D	E	F
1		P1A1H1	P1A2H1	P1A3H1	P1A1H2	P1A2H2
2	P1A1H1	0	0,1	0,2	1	1,1
3	P1A2H1	0,1	0	0,1	1,1	1
4	P1A3H1	0,2	0,1	0	1,2	1,1
5	P1A1H2	1	1,1	1,2	0	1,1
6	P1A2H2	1,1	1	1,1	1,1	0

Ilustración 12: submatriz de distancias planteada como ejemplo

6.4 Aplicación en cliente

La interfaz de cliente utilizada es "*Windows Forms*", desarrollada con el entorno de desarrollo integrado para Windows "*Microsoft Visual Studio*" en su versión "*Enterprise 2017*".

La ilustración 13 muestra el explorador de soluciones del entorno con cada una de las implementaciones que se han precisado para el funcionamiento de la aplicación.

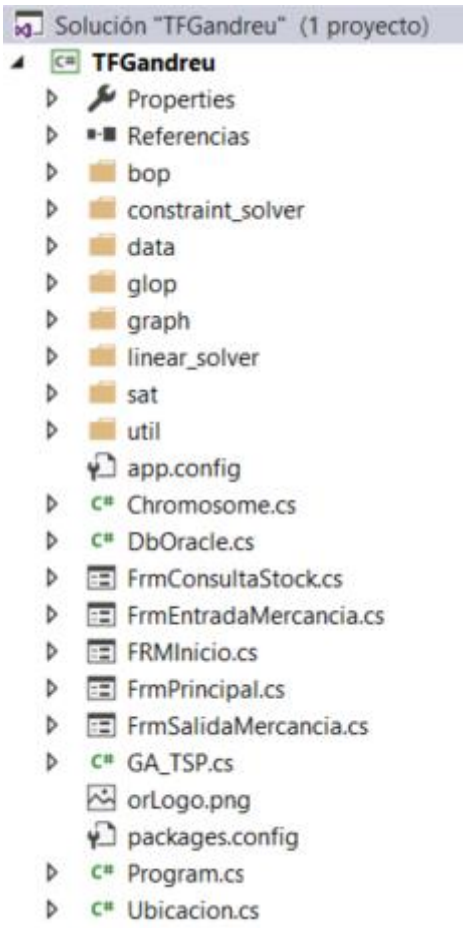


Ilustración 13: explorador de soluciones de Microsoft Visual Studio 2017

Dejando a un lado las carpetas (que contienen utilidades de bibliotecas y de la solución), esta estructura se divide en tres partes:

- **Propiedades de la solución:** permiten realizar cambios generales en la aplicación como la plata forma, el nombre de ensamblado, el framework y otras características.

- **Inicialización del programa:** representa el fichero de inicialización de la aplicación. Este fichero suele tener el nombre estandarizado "*Program*" y dispone de un método de entrada ejecutable llamado "*EjecutarSolver()*". En su interior se realizará una llamada al formulario de inicio de sesión.

- **Interfaces de usuario de Windows:** también llamados formularios, son elementos dotados de una interfaz gráfica y componentes funcionales, que, junto con la programación en lenguaje "*C#*", dotarán de una o varia funcionalidades en función de los objetivos marcados para ello. Los formularios implementados en este proyecto son los siguientes:

1. Pantalla de inicio de sesión: contiene dos elementos de entrada de texto en los cuales el usuario debe introducir sus datos de registro y un tercer campo que carga las descripciones de las empresas registradas en la base de datos, y que el usuarió debe elegir. Este último campo es guardado en una variable local para cada formulario que tramite una comunicación con la base de datos ya que la variable empresa se pasar como índice en cada trámite o comunicación con la base de datos. Los otros dos campos son botones que realizan sus correspondientes acciones; el primero inicia el trámite que, mediante una consulta a la base de datos, comprueba la validez de los datos introducidos y seguidamente accede a la pantalla de menú principal, y el segundo sale de la pantalla.

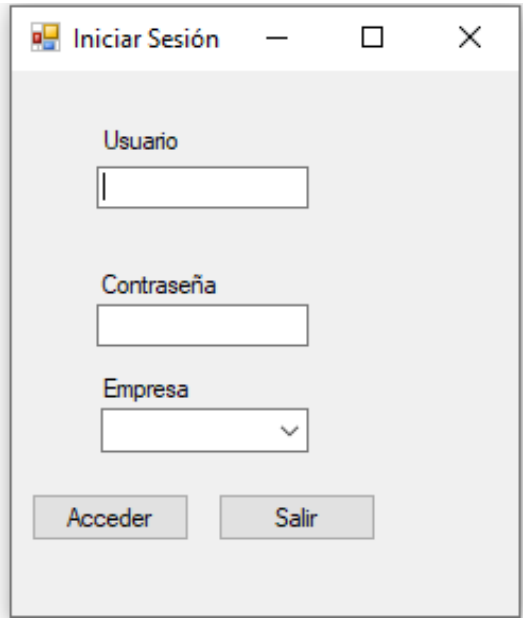


Ilustración 14: pantalla de inicio de sesión

2. Pantalla de menú: contiene tres botones que dan acceso a las principales funcionalidades de la aplicación. El primer botón se refiere a la entrada de mercancía mediante la tramitación de entrada a través de un albarán de compra preinsertado en la base de datos. El segundo botón se refiere a la gestión de stock y a las opciones que en esta se incluyen. Y el tercer botón se refiere a la salida de mercancía mediante la tramitación de salida a través de un albarán de venta preinsertado en la base de datos. Los aplicativos que son gestionados en la aplicación conciernen solo a la gestión de almacén, por lo que la introducción de documentos de venta o compra deben estar ya preinsertados al pertenecer esa gestión a otro departamento de la empresa (normalmente concierne al departamento comercial).

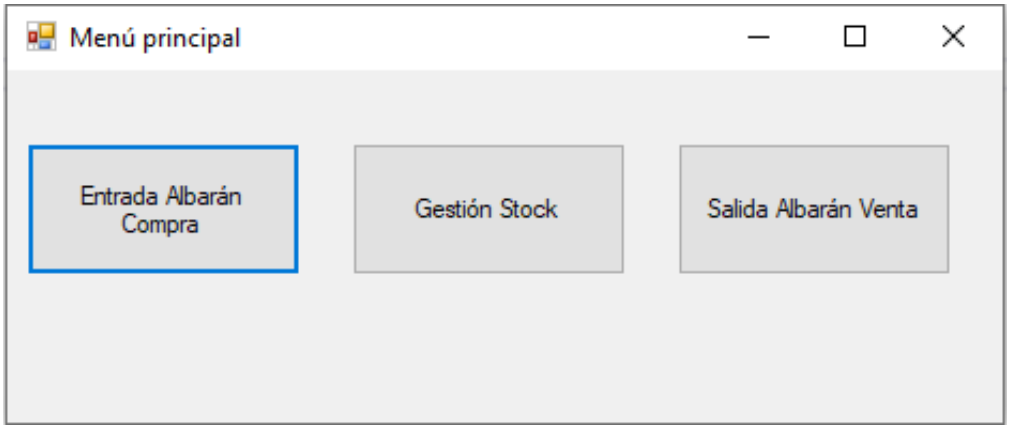


Ilustración 15: pantalla de menú principal

3. Pantalla Entrada Albarán Compra: está compuesta principalmente por dos bloques funcionales. El primero se refiere a la consulta del documento (albarán de compra). Antes de introducir la mercancía en el almacén se deben cotejar las líneas del documento referenciado y en caso de discrepancia, la cuadrícula central (utilizada para mostrar el detalle del documento) permite cambiar y corregir los datos antes de la introducción de mercanía. Para ello, se necesitan unos elementos de texto de entrada que realicen la función de filtrado (campos: ejercicio y número de albarán) y un botón que realice los trámites de consulta y mostrado del documento. El segundo bloque funcional realiza la entrada de mercancía en la base de datos y actualiza el albarán con los cambios que se realicen. Además, en la parte inferior de la pantalla aparece un campo informativo que muestra la sumarización de la cantidad de artículos del documento.

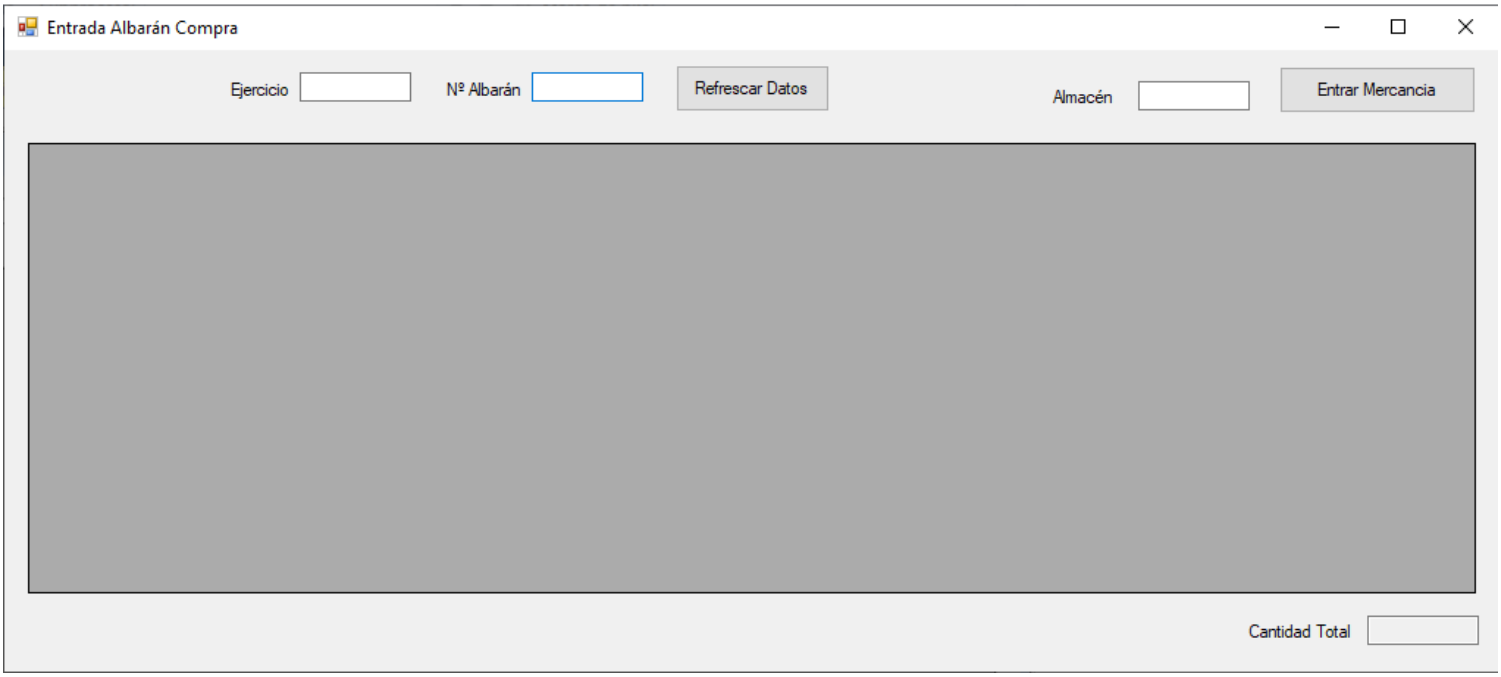


Ilustración 16: pantalla de entrada de albaranes de compra

4. Pantalla de gestión de stock: está compuesta principalmente por dos bloques funcionales. El primero se refiere a la consulta de stock de un artículo. Dicha consulta puede detallarse en dos niveles; a nivel de artículo donde se muestra la cantidad total del artículo en el almacén indicado, y un segundo nivel por ubicación, donde se muestran cada una de las ubicaciones que tienen, en el momento de realizar la consulta, almenos una referencia de dicho artículo almacenada y la cantidad de la misma. Los elementos necesarios para este primer bloque son: dos elementos de entrada de texto para el almacén y el artículo a consultar, dos botones de tipo radio para escoger el nivel de consulta, y un botón que tramite la consulta con la base de datos y refresque la cuadrícula central con la información recuperada. El segundo bloque (ubicado en la parte inferior de la pantalla) se refiere al movimiento de mercancía (o stock) de la cantidad de un artículo almacenada en una ubicación dada a otra del mismo almacén o de otro distinto. Para ello, se introducen los datos de filtrado necesarios para indexar un artículo de una ubicación origen. Y como datos necesarios para realizar el movimiento se tiene la cantidad a transpasar y la ubicación destino. Y por último, para realizar dicho traspaso, es necesario que un botón accione los accesos pertinentes a la base de datos.

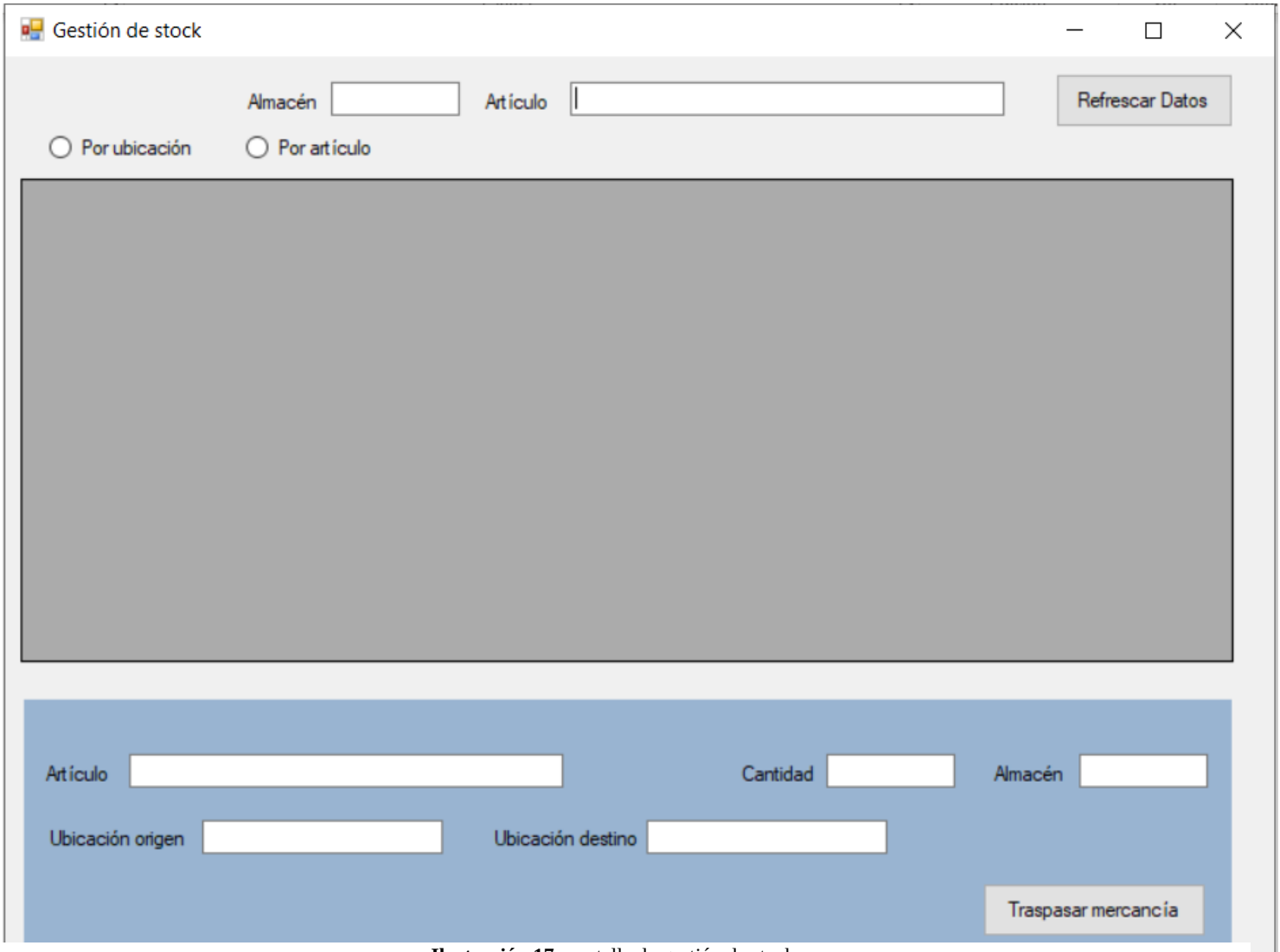


Ilustración 17: pantalla de gestión de stock

5. Pantalla Salida Albarán Venta: está compuesta principalmente por dos bloques funcionales. El primero se refiere a la consulta del documento (albarán de venta) ya que antes de recoger la mercancía de las ubicaciones correspondientes, se puede modificar en la cuadrícula (utilizada para mostrar el detalle del documento) algun dato. Para ello, se necesitan unos elementos de texto de entrada que realicen la función de filtrado (campos: ejercicio y número de albarán) y un botón que realice los trámites de consulta y mostrado del documento. El segundo bloque funcional realiza la salida de mercancía en la base de datos y actualiza el albarán con los cambios que se realicen. Además, el usuario tiene que introducir la ruta donde se encuentre el fichero con la matriz de distancias. Para que el proceso de cálculo obtenenga la submatriz con las ubicaciones obtenidas por el procedimiento almacenado en la base de datos, que obtiene la lista de ubicaciones a visitar y seguidamente la pone a disposición del proceso TSP para que calcule la ruta mínima. También se precisa de un elemento botón que inicie los trámites que generan la salida de mercancía. Además, en la parte inferior de la pantalla aparece un campo informativo que muestra la sumarización de la cantidad de artículos del documento.

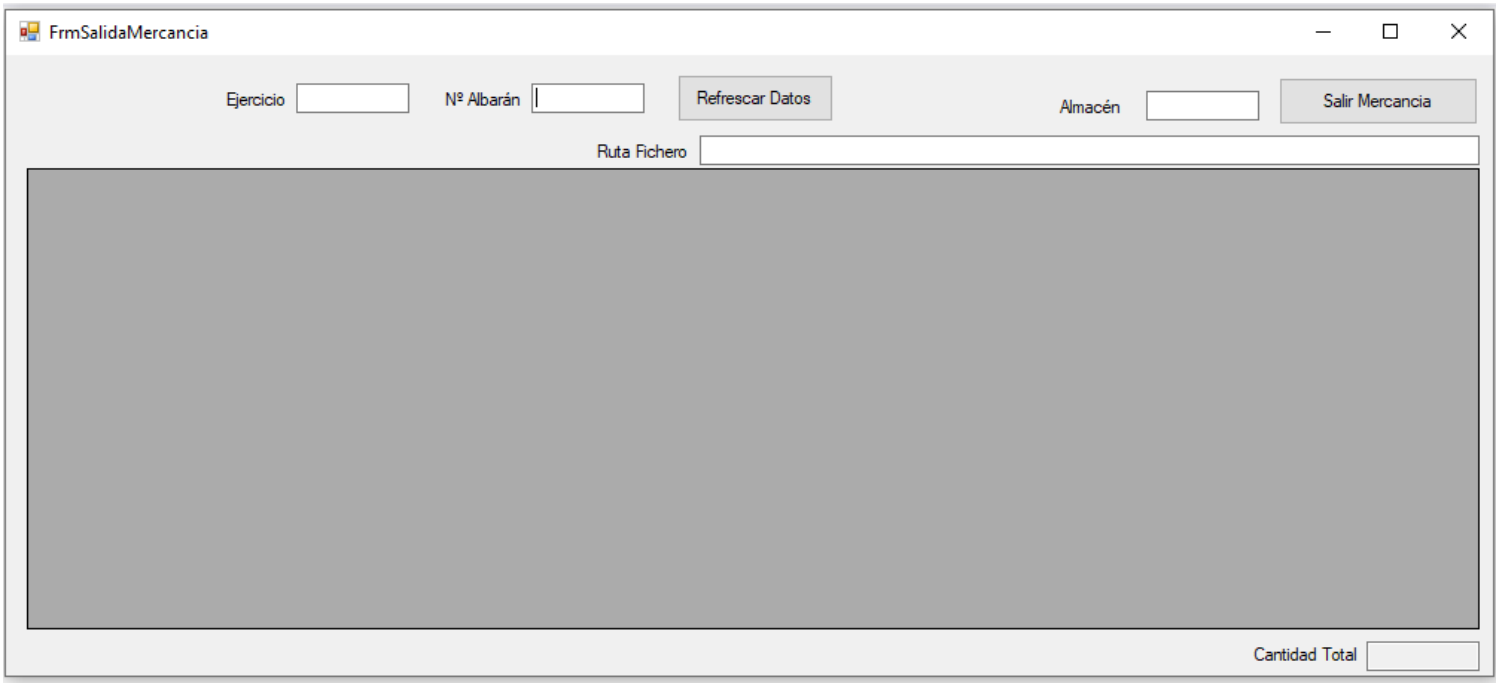


Ilustración 18: pantalla de salida de albarán de venta

- **Conexión con la base de datos:** la aplicación usa la biblioteca " *Oracle Managed Database*" para que las operaciones de consulta, modificación e inserción que se precisen puedan llevarse a trámite. Dicha biblioteca está integrada en el proyecto de Visual Studio. Es necesario inicializar una conexión cada vez que se intente acceder a la base de datos y cerrarla al finalizar el trámite. Como las conexiones son bastante habituales, dentro de una aplicación que trabaja con datos y éstos, en ocasiones, estan alojados en un equipo externo al del equipo cliente, la redundancia de código puede verse incrementada y además pueden surgir errores si, por ejemplo, una conexión no se cierra. La solución es crear un fichero genérico de conexión a la base de datos que tenga diferentes métodos dependiendo de si la operación es de consulta, modificación o inserción.

En la aplicación se usa el fichero "*DbOracle*" como fichero genérico de connexión con la base de datos, instanciado por los métodos que tramitan una operación con esta.

- **Elementos TSP:** una vez se obtiene la lista de ubicaciones a recorrer, se necesita saber cual es la combinación de todas ellas que tiene el recorrido más corto. Para ello, el programa inicializa el método "*Main*" del fichero "*GA_TSP*", pasándole como parámetros la lista de ubicaciones, incluyendo siempre la ubicación "*Zona de Carga*" ya que es el punto de partida del repartidor y al mismo tiempo el último punto del recorrido (donde se depositan los pedidos completos, listos para ser cargados). Este es el único proceso de toda la aplicación que tiene una carga computacional considerable para el cliente (siempre dependiendo del número de ubicaciones), por lo que se deben seguir unos pasos estructurados para evitar que la máquina del cliente pueda entrar en un estado de saturación computacional (para ello, se han definido unos requisitos mínimos recomendados en el equipo cliente). Los pasos que realiza la aplicación para resolver el problema del camino mínimo se estructuran en el fichero "*GA_TSP*":

  1. Se instancia el modelo de datos.

```
public void EjecutarSolver()
{
    // Instancia del problema.
    DataModel data = new DataModel();
```

Ilustración 19: código referente al método que instancia el problema.

  2. Se obtiene la submatriz de distancías a partir de la lista de ubicaciones a visita que incluye la zona de carga como primera y última ubicación. La ilustración 19 muestra la clase "*DataModel*" que inicializa la matriz de distancias correspondiente al campo "DistanceMatriz" y le asigna los datos de la submatriz ya calculada. El campo "*Depot*" se refiere al número de ubicación de inicio (o ubicación de partida) que estáticamente es siempre zero. Y por último, el campo "*VehicleNumber*" hace referencia al número de operarios que realizan la ruta (siempre será 1, ya que si fuera superior se estaría planteando el problema de enrutamiento de vehículos y este es otro problema distino).

```
class DataModel
{
    public long[,] DistanceMatrix = submatrizXLSX ;
    public int VehicleNumber = 1;
    public int Depot = 0;
};
```

Ilustración 20: código referente a la creación de la instancia del solver

  3. Se crea el modelo de enrutamiento mediante el administrador de índices ("*manager*"). El método "*manager._IndexToNode*" convierte los índices internos del "*solver*" en números para ubicaciones. Dichos números corresponden a los índices de la matriz de distancias. Los parámetros de entrada del método "*RoutingIndexManager*" son:

- El número de filas de la matriz de distancia, que es el número de ubicaciones (incluida la ubicación inicial, zona de carga).
- El número de vehículos (operarios) del problema.
- El nodo correspondiente a la ubicación inicial.

```
// Creación del índice del modelo de enrutamiento
RoutingIndexManager manager = new RoutingIndexManager(
    data.DistanceMatrix.GetLength(0),
    data.VehicleNumber,
    data.Depot);
```

Ilustración 21: código referente a la creación del índice del modelo de enrutamiento

4. Para usar el "*solver*" de enrutamiento se debe crear un "*callback*" de distancia con una función que tome cualquier par de ubicaciones y devuelva la distancia entre ellas. Para ellos, se obtendrán los datos de referencia de la matriz de distancias. La ilustración 22 muestra la función que crea el "*callbakc*" y lo registra con el "*solver*" como "*transit_callback_index*". Además esta función tomará dos índices (con referencia a ubicaciones) como parámetros de entrada y devuelve la correspondiente entrada de la matriz de distancias.

5. El "*solver*" necesita saber como calcular el coste del viaje entre dos ubicaciones cualesquiera (o dicho de otra manera, el coste de la línea que las une en la gráfica del problema). La ilustración 22 establece el evaluador de costes entre ubicaciones.

```
// Definir el coste de cada arco
routing.SetArcCostEvaluatorOfAllVehicles(transitCallbackIndex);

// Creación del modelo de enrutamiento
RoutingModel routing = new RoutingModel(manager);

int transitCallbackIndex = routing.RegisterTransitCallback(
  (long fromIndex, long toIndex) => {
      // Conversión de la variable de enrutamiento Index a la matriz de distancia NodeIndex.
      var fromNode = manager.IndexToNode(fromIndex);
      var toNode = manager.IndexToNode(toIndex);
      return data.DistanceMatrix[fromNode, toNode];
  }
);
```

Ilustración 22: código referente a la definición del coste y modelo de enrutamiento

6. Antes de ejecutar el "*solver*" de la herramienta "*OR-Tools*" se deben establecer los parámetros de búsqueda predeterminados y un método heurístico para encontrar la primera solución. La siguiente ilustración ("*nombre*") muestra el código utilizado en el cual se establece la primera estrategia de solución en la variable "*PatCheapestArc*", que crea una ruta inicial para el "*solver*" agregando repetidamente nodos con menor peso que no conducen a un nodo visitado anteriormente (que no sea el nodo inicial).

```
// Obtener una primera solución heurística
RoutingSearchParameters searchParameters =
  operations_research_constraint_solver.DefaultRoutingSearchParameters();
searchParameters.FirstSolutionStrategy =
```

Ilustración 23: código referente a la primera solución heurística

7. Una vez se han establecido las opciones de los puntos anteriores, el "solver" de "*OR-Tools*" puede ejecutarse y seguidamente obtener los resultados de la operación. La siguiente ilustración 24 muestra la llamada al proceso y al método que obtiene los resultados. El primer valor que devuelve es la distancia total recorrida por el operario, el segundo valor devuelve la ubicación de partida (o la inicial), y el tercer valor devuelve a modo de cadena la lista de ubicaciones separadas por comas en el orden que deben ser visitadas. Una vez se obtienen estos resultados, la aplicación gestiona la llamada a un proceso de la base de datos que genera una lista con el detalle de los artículos y las cantidades que deberán ser recogidas en cada ubicación de la lista. La ilustración 24 muestra el código asociado a lo explicado en este apartado. Además se eliminan las entradas correspondientes en la tabla "*STOCKUBIART*" para dejar constancia de la salida de mercancía que se ha efectuado.

```
static void ObtenerResultados(
    in RoutingModel routing,
    in RoutingIndexManager manager,
    in Assignment solution)
{
    long totalMetros = solution.ObjectiveValue();

    long routeDistance = 0;
    var index = routing.Start(0);
    string resulRuta = "";
    while (routing.IsEnd(index) == false)
    {

        resulRuta += manager.IndexToNode((int)index)); ;
        var previousIndex = index;
        index = solution.Value(routing.NextVar(index));
        routeDistance += routing.GetArcCostForVehicle(previousIndex, index, 0);
    }

}
```

Ilustración 24: código del método que obtiene el resultado del algoritmo

7. Pruebas

Las pruebas que se han realizado en este proyecto parten de una estructura organizada en un entorno cliente-servidor. El entorno gráfico de la aplicación se encuentra instalado en un equipo portátil con Windows 10, una arquitectura de procesador de 64 bits. La base de datos de Oracle se alojará en una máquina virtual de Windows 10 ubicada en un equipo distinto al del cliente (pero en la misma red local).

Esta estructura se ha establecido con tal de simular un entorno de empresa real, donde existen diversos equipos clientes con la aplicación de escritorio instalada y un único servidor de datos alojado de forma externa.

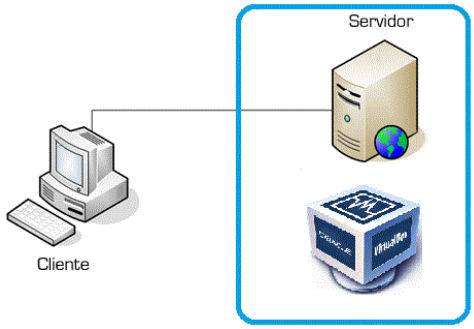


Ilustración 25: diagrama de red local utilizado en el proyecto (Licencia: dominio público)

Para poder realizar pruebas en la base de datos, esta debe estar dotada de información en sus tablas, ya que, como se ha mencionado anteriormente, existen tablas (albaranes de ventas y albaranes de compras) que deben ser mantenidas por otros departamentos de la empresa. Para ello, se ha desarrollado un script en lenguaje PL/SQL que rellena las tablas con datos de prueba que permiten realizar un trazado de los diferentes puntos operativos que tiene la aplicación.

Se han realizado dos pruebas para comprobar el correcto funcionamiento de los procesos operativos que ofrece la aplicación:

- Entrada de mercancía: para comprobar que este proceso funciona correctamente, se ha referenciado un albarán, y se ha comprobado que la entrada de los artículos que lo componen

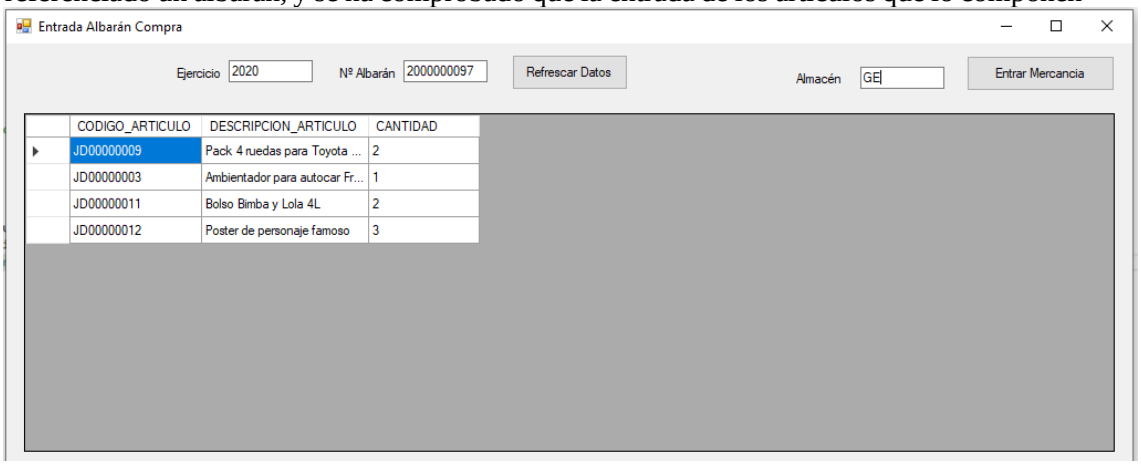


Ilustración 26: instantánea de pantalla de entrada de pedidos de compra

se ha hecho correctamente. La ilustración 26 muestra la pantalla de entrada de mercancía con el detalle del albarán de compra "*2000000097*".

Antes de realizar la entrada de mercancía, se consulta la cantidad de stock actual de cada uno de los artículos que componen el detalle del documento, pero para escenificar el proceso de una manera más clara y menos redundante, la comprobaciones que se muestran en las ilustraciones 27 y 28, referencian a un solo artículo del documento. Como ejemplo, se ha escogido el artículo con código "*JD00000009*". Se ha comprobado el stock actual que tiene el artículo en el almacén general (código de almacén "*GE*").

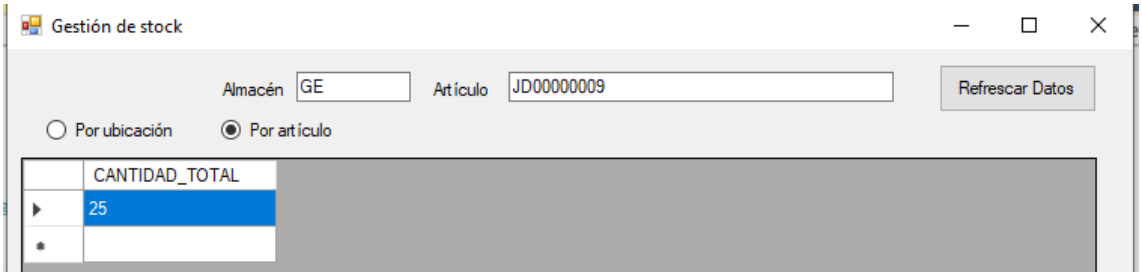


Ilustración 27: instantánea de pantalla de gestión de stock por artículo

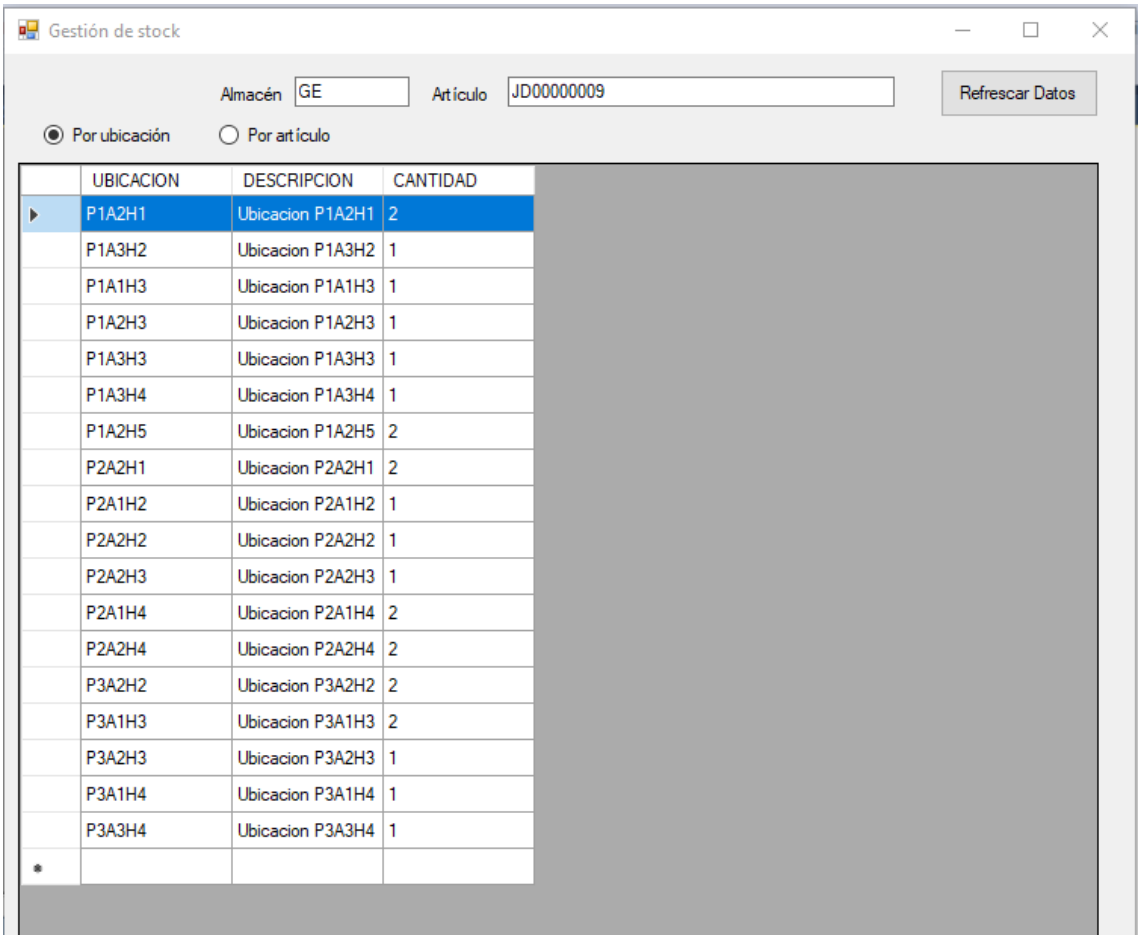


Ilustración 28: instantánea de pantalla de gestión de stock por ubicación

En el momento de realizar la consulta, este artículo tenia registradas en el sistema 25 unidades repartidas en distintas ubicaciones. Al accionar el botón que genera la entrada de mercancía (botón en ilustración 26), la información de este artículo deberia actualizarse aumentando la cantidad en dos unidades en el almacén general. La ilustraciones 29 y 30 muestran el resultado obtenido al realizar dicha actualización.

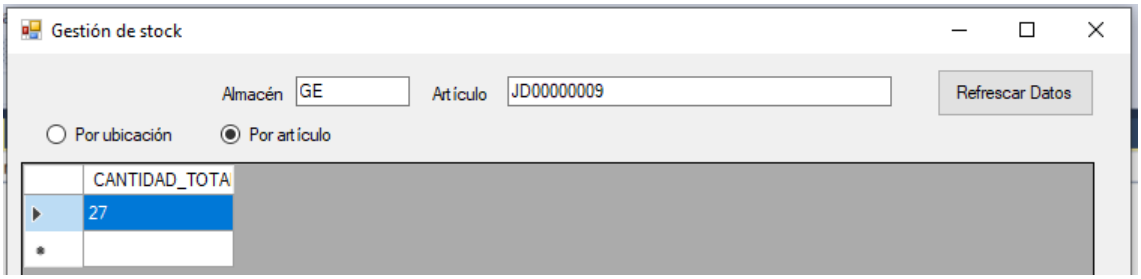


Ilustración 29: instantánea de pantalla de gestión de stock por artículo

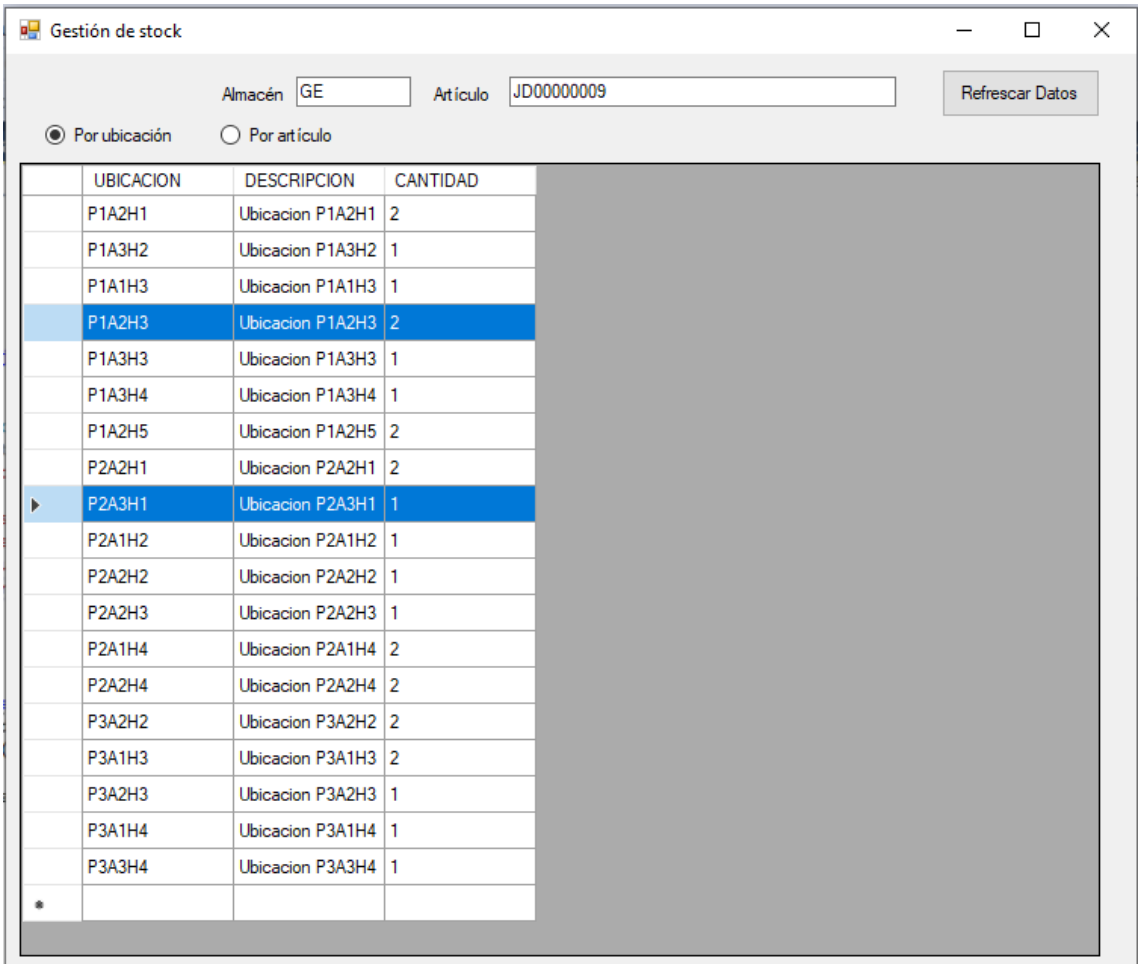


Ilustración 30: instantánea de pantalla de gestión de stock por ubicación

Como puede observarse en la ilustración 29, la cantidad ha aumentado en dos unidades, pero si se obtiene un listado de stock de este mismo artículo detallado por ubicación, puede observarse como, de las dos unidades insertadas, una de ellas se ha insertado en la ubicación "*P1A2H3*" (sumarizando la cantidad ya existente de este mismo artículo en dicha ubicación) y la otra en la ubicación "*P2A3H1*".

- Salida de mercancía: para escenificar este ejemplo, se ha seguido la misma estructura que la usada en el proceso anterior, pero referenciando un albarán de venta. La ilustración 31 muestra la pantalla de entrada de mercancía con el detalle del albarán de venta "*2000000084*".

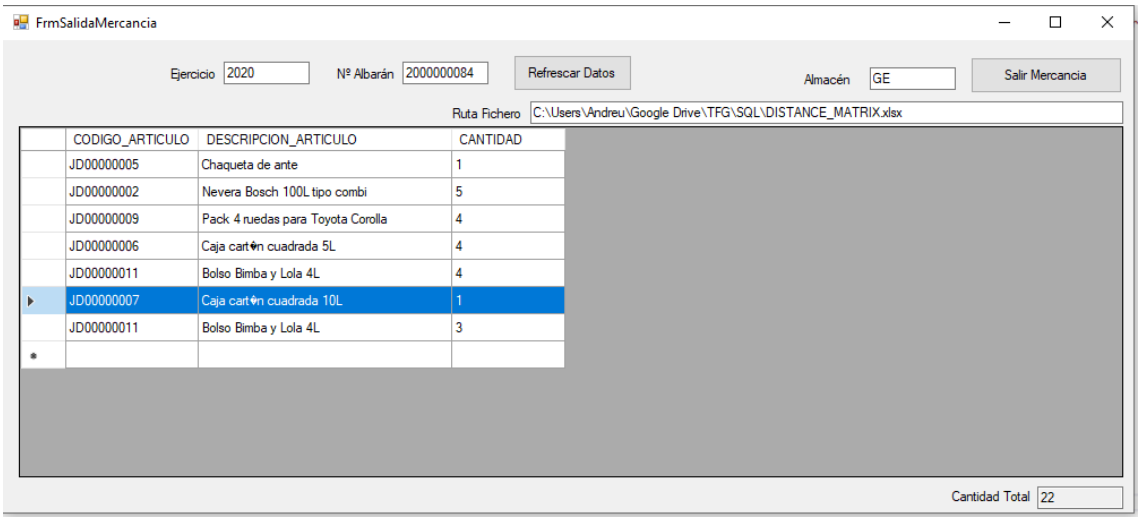


Ilustración 31: instantánea de pantalla de salida de pedidos de venta

Como en el caso anterior, antes de realizar la salida de mercancía, se sonsulta la cantidad de stock actual de cada uno de los artículos que componen el detalle del documento, pero para escenificar el proceso de una manera más clara y menos redundante, la comprobación se ha hecho de solo un artículo del documento. Como ejemplo, se ha escogido el artículo con código "*JD00000007*". Se ha comprobado el stock actual que tenia el artículo en el almacén general (código de almacén "*GE*"). La ilustraciones 32 y 33 muestran la consulta de este artículo por artículo y por ubicación respectivamente.

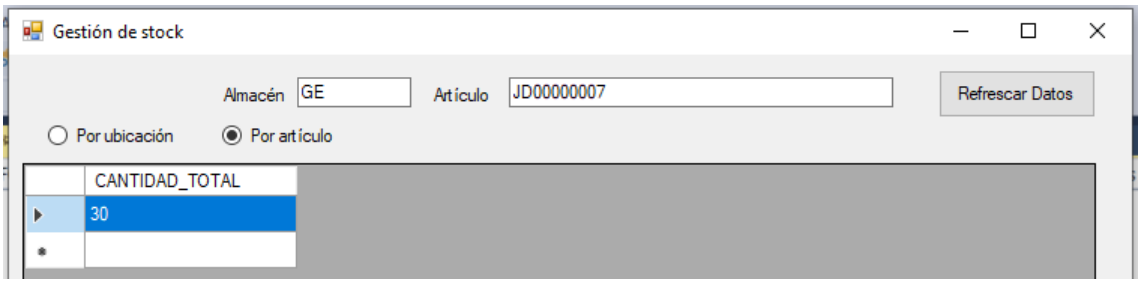
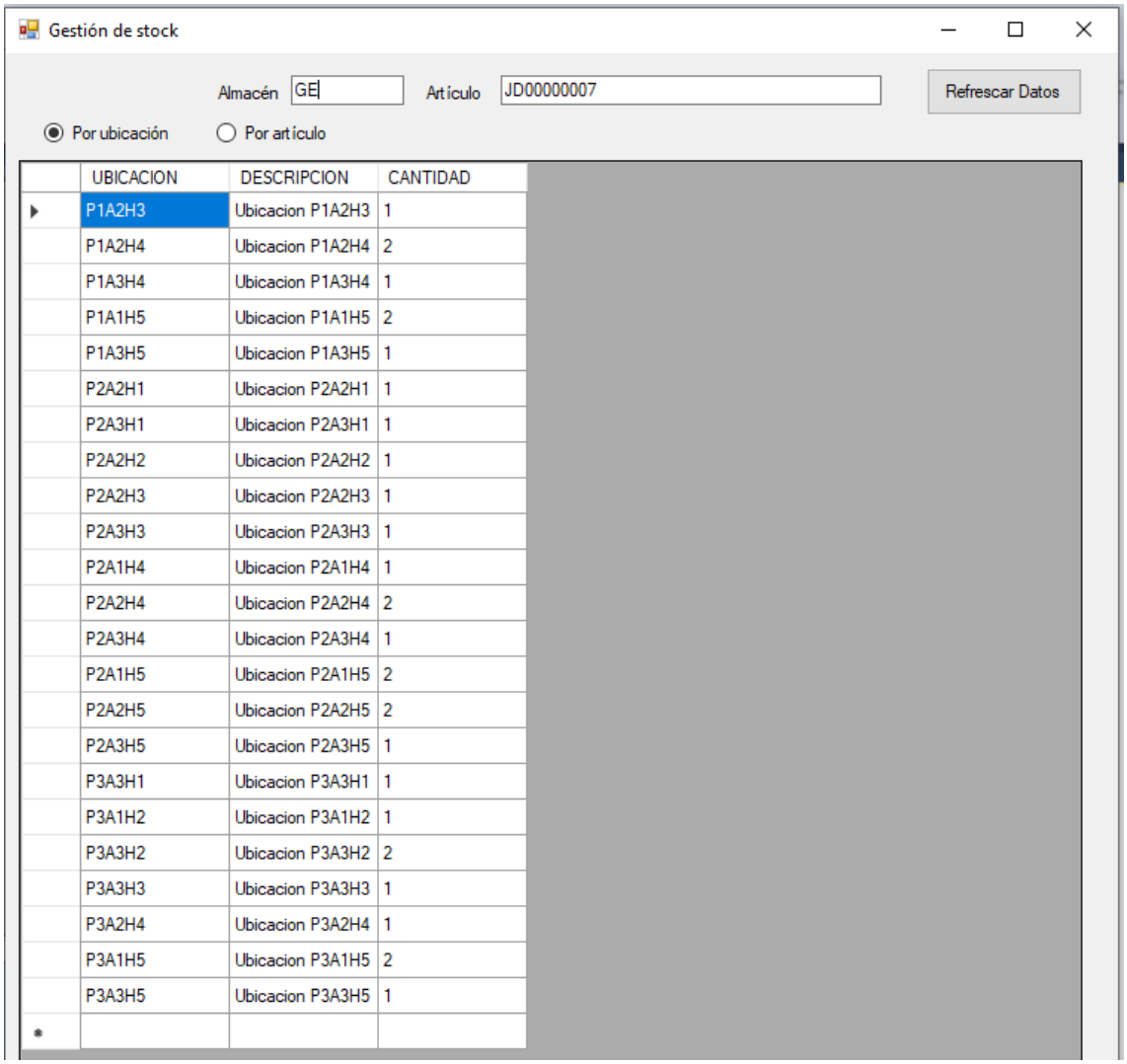


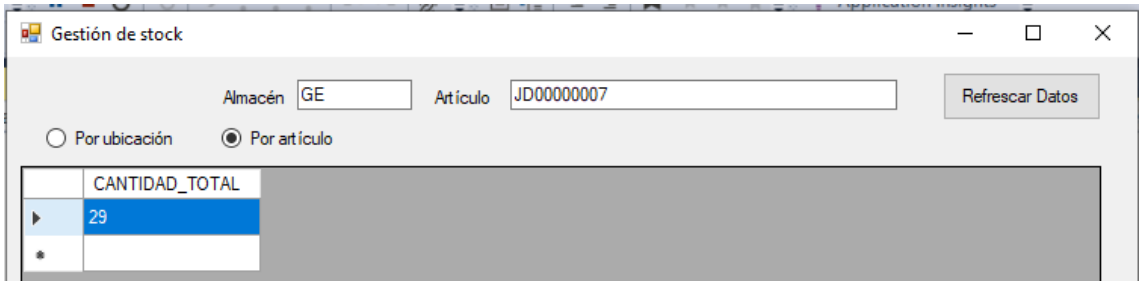
Ilustración 32: instantánea de pantalla de gestión de stock por artículo



UBICACION	DESCRIPCION	CANTIDAD
P1A2H3	Ubicacion P1A2H3	1
P1A2H4	Ubicacion P1A2H4	2
P1A3H4	Ubicacion P1A3H4	1
P1A1H5	Ubicacion P1A1H5	2
P1A3H5	Ubicacion P1A3H5	1
P2A2H1	Ubicacion P2A2H1	1
P2A3H1	Ubicacion P2A3H1	1
P2A2H2	Ubicacion P2A2H2	1
P2A2H3	Ubicacion P2A2H3	1
P2A3H3	Ubicacion P2A3H3	1
P2A1H4	Ubicacion P2A1H4	1
P2A2H4	Ubicacion P2A2H4	2
P2A3H4	Ubicacion P2A3H4	1
P2A1H5	Ubicacion P2A1H5	2
P2A2H5	Ubicacion P2A2H5	2
P2A3H5	Ubicacion P2A3H5	1
P3A3H1	Ubicacion P3A3H1	1
P3A1H2	Ubicacion P3A1H2	1
P3A3H2	Ubicacion P3A3H2	2
P3A3H3	Ubicacion P3A3H3	1
P3A2H4	Ubicacion P3A2H4	1
P3A1H5	Ubicacion P3A1H5	2
P3A3H5	Ubicacion P3A3H5	1

Ilustración 33: instantánea de pantalla de gestión de stock por ubicación

En el momento de realizar la consulta, este artículo tiene registradas en el sistema 30 unidades repartidas en distintas ubicaciones. Si se acciona el botón que genera la salida de mercancía (botón en ilustración 31), la información de este artículo deberia actualizarse disminuyendo la cantidad en una unidad en el almacén general. La siguientes ilustraciones 34 y 35 muestran el resultado obtenido al realizar dicha actualización.



CANTIDAD_TOTAL
29

Ilustración 34: instantánea de pantalla de gestión de stock por artículo

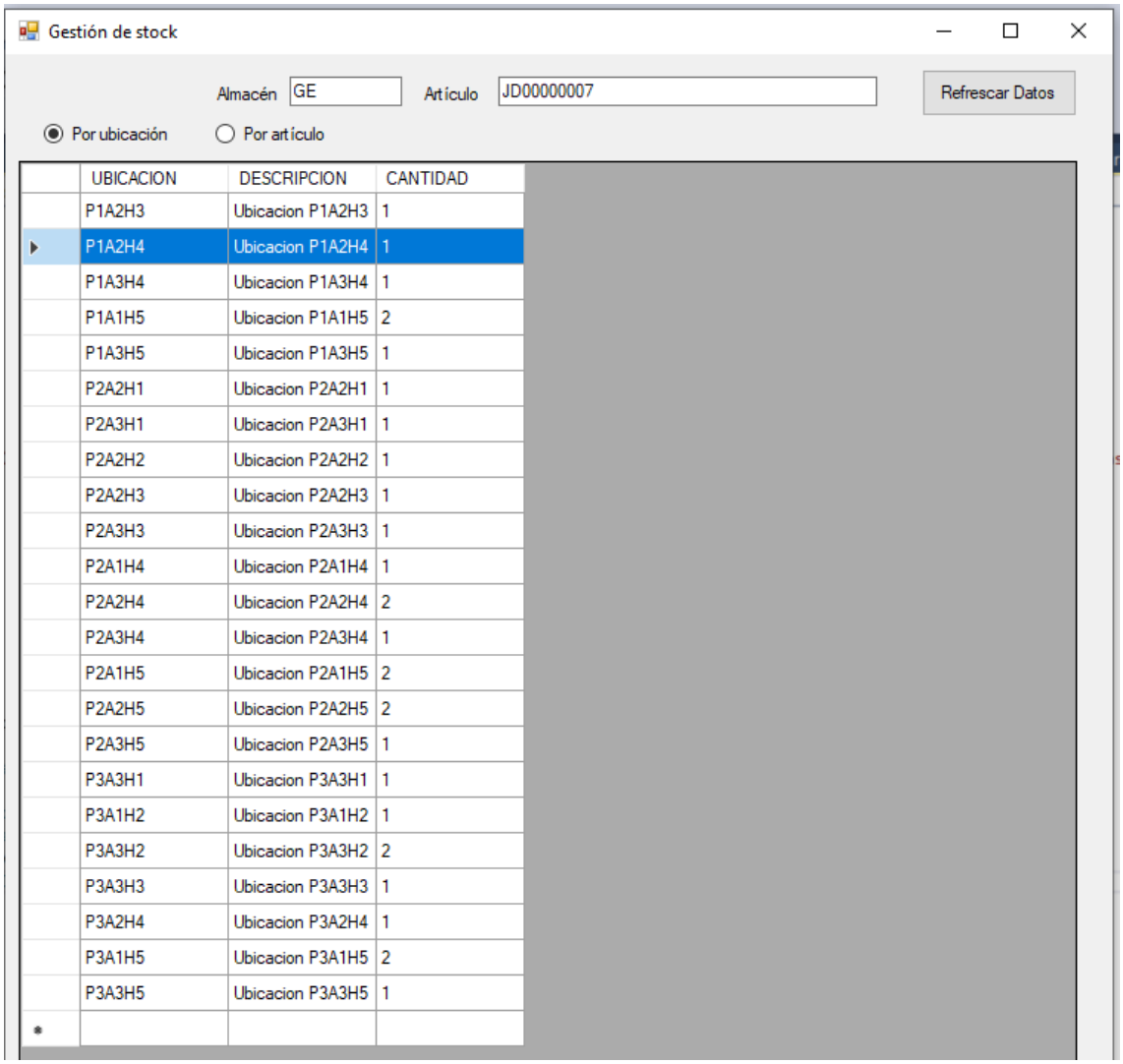


Ilustración 35: instantánea de pantalla de gestión de stock por ubicación

Como puede observarse en la primera ilustración, la cantidad ha disminuido en una unidad, pero, alobtener un listado de stock de este mismo artículo detallado por ubicación, puede observarse como la cantidad de la ubicación "*P1A2H3*" ha disminuido (restando la cantidad ya existente de este mismo artículo en dicha ubicación).

- Traspaso de mercancía: sobre los datos de la ilustración anterior ("*P1A2H3*"), se ha realizado un movimiento de mercancía de una ubicación a otra.

  Como ejemplo se ha tomado una unidad del artículo "*JD00000007*" en la ubicación "*P3A3H5*" y se ha movido a la ubicación "*P3A1H5*". La ilustración 36 muestra los datos de filtrado que se han necesitado para realizar dicho movimiento.

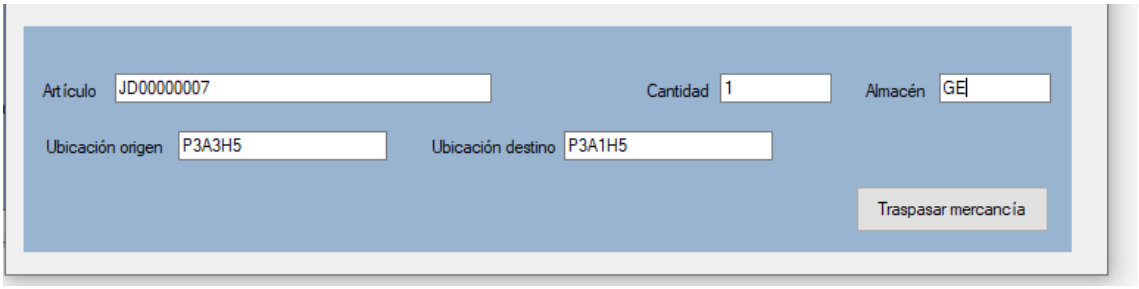


Ilustración 36: instantánea de pantalla de gestión de artículos. Traspaso de artículos

La aplicación comprueba si en la ubicación destino caben la unidades que van a ser traspasadas, además también verifica que las unidades introducidas son correctas. Si el movimiento no fuera posible, se muestra un mensaje de error al usuario. La ilustración 37 escenifica una prueba en la que se introdujo un cantidad errónea.

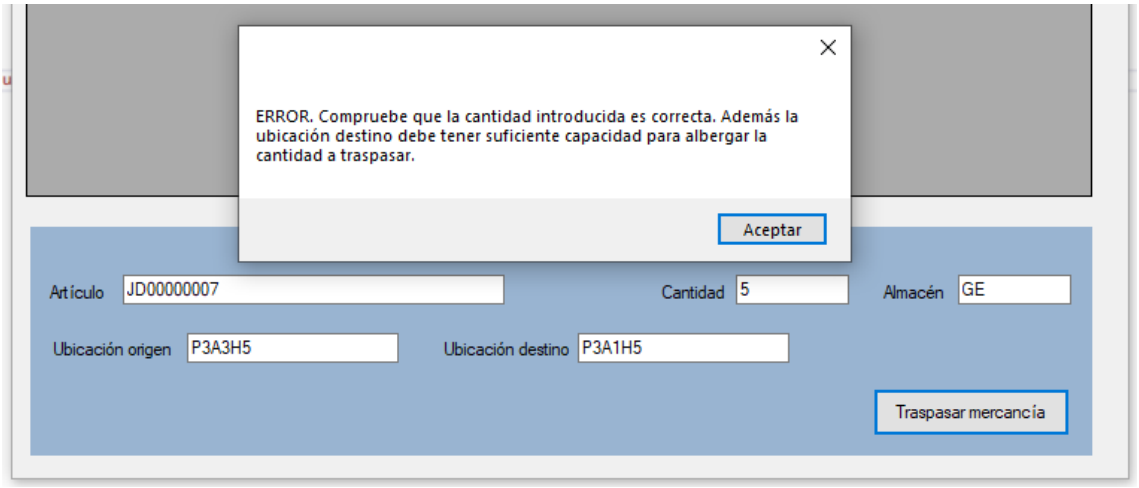
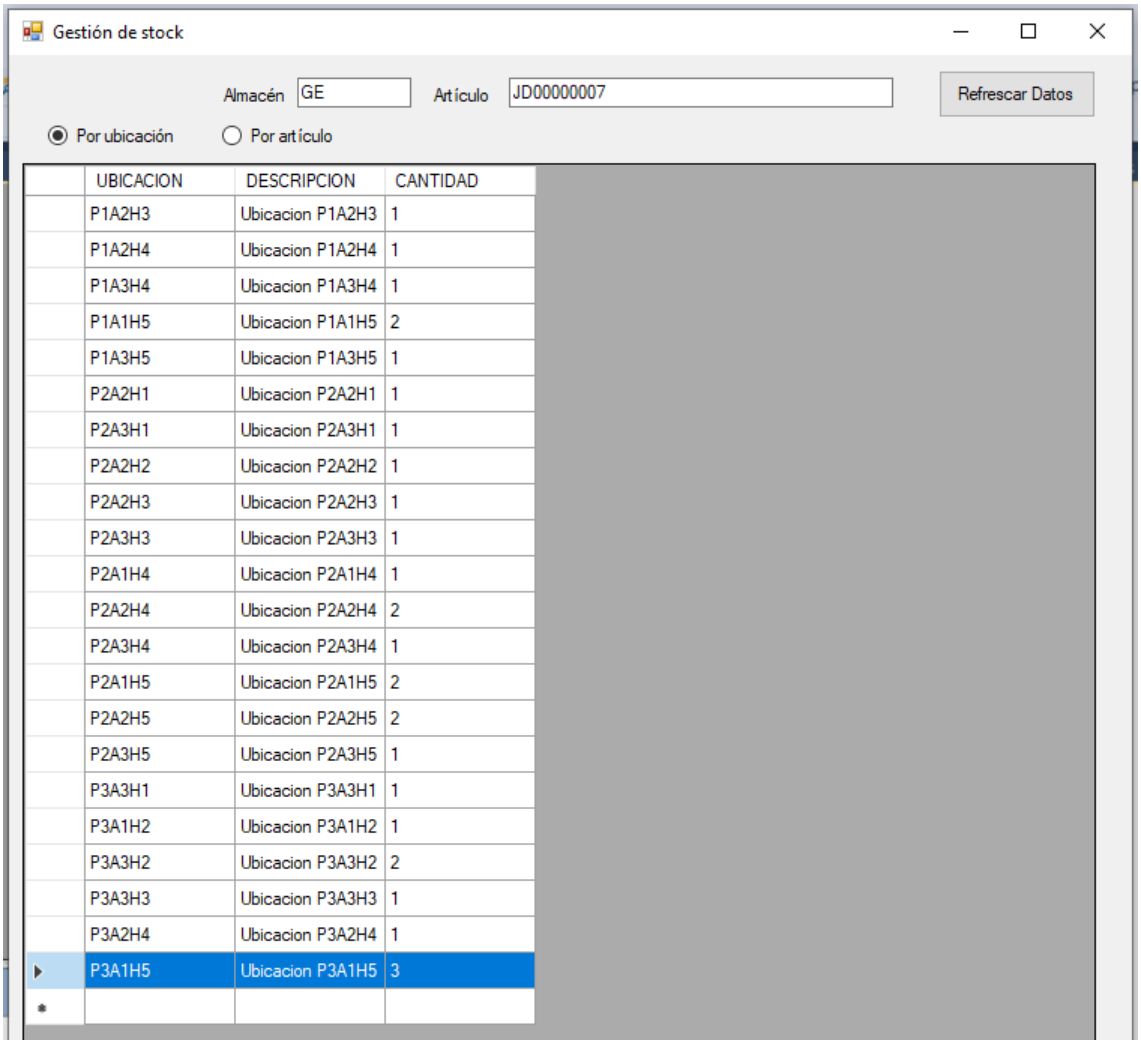


Ilustración 37: instantánea de pantalla de gestión de artículos. Traspaso de artículos con mensaje de error

Si la operación de traspaso ha sido existosa, el resultado de la consulta del stock por artículo detallado por ubicaciones del artículo "*JD00000007*" habrán cambiado.

La siguiente ilustración 38 muestra los resultados de la consulta con la operación de traspaso realizada correctamente.



UBICACION	DESCRIPCION	CANTIDAD
P1A2H3	Ubicacion P1A2H3	1
P1A2H4	Ubicacion P1A2H4	1
P1A3H4	Ubicacion P1A3H4	1
P1A1H5	Ubicacion P1A1H5	2
P1A3H5	Ubicacion P1A3H5	1
P2A2H1	Ubicacion P2A2H1	1
P2A3H1	Ubicacion P2A3H1	1
P2A2H2	Ubicacion P2A2H2	1
P2A2H3	Ubicacion P2A2H3	1
P2A3H3	Ubicacion P2A3H3	1
P2A1H4	Ubicacion P2A1H4	1
P2A2H4	Ubicacion P2A2H4	2
P2A3H4	Ubicacion P2A3H4	1
P2A1H5	Ubicacion P2A1H5	2
P2A2H5	Ubicacion P2A2H5	2
P2A3H5	Ubicacion P2A3H5	1
P3A3H1	Ubicacion P3A3H1	1
P3A1H2	Ubicacion P3A1H2	1
P3A3H2	Ubicacion P3A3H2	2
P3A3H3	Ubicacion P3A3H3	1
P3A2H4	Ubicacion P3A2H4	1
P3A1H5	Ubicacion P3A1H5	3

Ilustración 38: instantánea de pantalla de gestión de stock por ubicación

Tras el movimiento realizado, la ubicación "*P3A3H5*" ha desaparecido del listado ya que no contien ningun cantidad del artículo referenciado, pero la ubicación "*P3A1H5*" ha aumentado la cantidad en una unidad.

8. Conclusiones

La finalidad del proyecto consistía en diseñar una aplicación capaz de gestionar la mercancía que alberga un almacén caótico, contemplando la entrada y la salida de esta.

Tal como se observa en el capítulo de pruebas, la aplicación cumple con la especificación de requisitos expuesta en el apartado correspondiente de este documento.

En términos generales, se han cumplido los objetivos estimados al inicio del proyecto a excepción de uno. El cálculo de la ruta mínima en el proceso de preparación de pedidos no asegura que la ruta obtenida sea siempre la óptima, sobre todo cuando la talla del problema (número de ubicaciones) es elevado. Además, dicho proceso ha exigido mucho más tiempo de lo esperado. Esto se debe a que el proceso que obtiene la lista de ubicaciones a recorrer presentaba unas carencias en la eficiencia, ya que, al obtener dicha lista, el número de ubicaciones era distinto. No obstante, se modificó y repitió el proceso hasta que el resultado fuera el esperado.

En cuanto al apartado técnico, debo decir que la implementación de la base de datos y la conexión del entorno de desarrollo con esta presentó muchos problemas. Han surgido muchos inconvenientes relacionados con protocolos de red y autentificación de la base de datos. La búsqueda de información para resolver todas estas incidencias, han hecho que el tiempo invertido en ello sea elevado. En cuanto a la utilización del entorno de Visual Studio 2017, el lenguaje C# y la base de datos de Oracle, debo decir que es la primera vez que implementaba un proyecto utilizando la combinación de estas herramientas. Y nunca había realizado un desarrollo tan grande de *software* de esta envergadura. Por ese motivo necesité de un periodo de adaptación para poder realizar dicha combinación.

La experiencia en el ámbito profesional desarrollando herramientas para programas ERP me ha aportado muchos conocimientos prácticos que he podido destacar en este proyecto. Sin embargo, los conocimientos y la metodología de los procesos y lenguajes obtenidos en diversas asignaturas del Grado han sido un elemento clave en el desarrollo de este. Los contenidos de la asignatura de Calidad y Optimización fueron los que despertaron en mi la curiosidad de mejorar los procesos operativos de la empresa. Lo aprendido en la asignatura de Sistemas Integrados de Información en las Organizaciones me aportó una visión general de los procesos que en la empresa se llevan a cabo y además me enseñó la gran utilidad que tiene modelar un proceso (un aspecto necesario para realizar este tipo de proyectos). La elección del gestor de bases de datos fue a raíz de haber cursado la asignatura de Diseño y Gestión de Bases de Datos, con la que pude descubrir un gestor que aportaba muchas herramientas de tratamiento de datos. El lenguaje C# lo aprendí en la asignatura de Ingeniería del *Software*, además de las metodologías de programación y de las buenas prácticas de esta.

Este proyecto, junto con lo aprendido durante mis estudios de grado y la experiencia laboral, me ha servido para ver que la informática es un concepto que deber ser tratado con cautela, En pocos años, la informática nos ha cambiado la vida. Los avances en el sector evolucionan muy rápido, y en ocasiones no podemos asimilarlos correctamente. Este concepto, forma parte del día a día de gran parte de las personas que viven en este planeta, ya que estamos siempre rodeados de aparatos tecnológicos. Debemos saber utilizarlos y entender bien cuál es su fin.

9. Futuras mejoras

Una vez analizados los objetivos de desarrollo planteados al principio, durante el desarrollo y programación de la aplicación se plantearon nuevos objetivos que aportarían más funcionalidad a la misma. Estos se han planteado como futuras mejoras ya que, por diversas causas tales como la falta de tiempo en la realización, no se han podido realizar. Entre los mismos se pueden citar:

1. Generar un procedimiento que avise al usuario responsable del almacén cuando el stock de un artículo sea inferior a una cantidad dada para que así el operario pueda pedir más cantidad al proveedor.

2. Ofrecer al usuario pantallas para poder administrar y mantener los almacenes y ubicaciones.

3. Ofrecer la funcionalidad de preparación de pedidos en un aplicativo móvil para "*IOS*" y "*Android*", donde cada usuario obtenga la lista de ubicaciones a recorrer en el dispositivo y descuente la mercancía leída mediante lectura por código de barras.

4. Obtener la funcionalidad del problema de enrutamiento de vehículos (siglas en inglés "*VRP*") en la aplicación, para dividir un pedido grande entre varios operarios.

5. Interfaz de usuario más atractiva, introduciendo elementos más elegantes e intuitivos.

6. Construir de forma dinámica la matriz de distancias, almacenando los datos en una tabla temporal de la base de datos para evitar el acceso a un fichero externo.

7. Traspasar la funcionalidad de cálculo de la ruta más corta al equipo servidor para quitar esa carga computacional al cliente. Esto conllevaría un nuevo estudio de los requerimientos mínimos de *hardware* del sistema.

8. Clasificar los usuarios por roles y así añadir un nivel más de seguridad en la aplicación, clasificando las funciones que puede realizar cada rol dentro de la misma.

9. Cambiar la gestión de caótica a mixta. Este sistema aceptaría una combinación de productos perecederos dentro del almacén poniendo estos con el sistema fijo y el resto de los productos con sistema caótico.

10. Ajustar el stock que se encuentra en una ubicación dada. Permitir, desde el sistema, regenerar el stock de una ubicación introduciendo valores nuevos de situación.

10. Referencias

[1] Centro Europeo de Postgrado. Clasificación de los almacenes.

https://www.ceupe.com/blog/clasificacion-de-los-almacenes.html#:~:text=Clasificaci%C3%B3n%20de%20los%20almacenes%20en%20funci%C3%B3n%20de%20la%20naturaleza%20de,y%20almacenes%20de%20producto%20terminado.

[2] Real Academia de Ingeniería. Definición.

http://diccionario.raing.es/es/lema/principio-de-optimalidad-de-bellman#:~:text=Definici%C3%B3n%3A,%C3%B3ptima%20respecto%20al%20subproblema%20correspondiente

[3] Tour of 50 USA Landmarks, William Cook

http://www.math.uwaterloo.ca/tsp/usa50/index.html

[4] Wikipedia. Definición cliente-servidor.

https://es.wikipedia.org/wiki/Cliente-servidor

[5] Pixabay.com. Imágenes de dominio público

https://pixabay.com/es/vectors/equipo-de-datos-base-de-datos-1294359/