UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Implementación de procesos de negocio en motores de código abierto: un estudio de caso

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Iván Aguilar Cuba

Tutor: Juan Sánchez Díaz

Curso 2018 - 2019

Resumen

El constante crecimiento de las organizaciones y empresas requieren de una mejora continua en sus métodos de trabajo y en la gestión de sus procesos comerciales para crear dinámicas de trabajo eficientes y adecuadas a la situación de competitividad actual.

En este trabajo se propone el estudio de un caso en el que se trata la situación de una empresa textil cuyas metodologías de trabajo son muy rudimentarias, lo que está provocando que se queden atrás frente a sus competidores. El objetivo fundamental es modelar e implementar los procesos de negocio de la empresa. Para ello, se ha elaborado un estudio sobre la empresa, generando diferentes diagramas que muestren información objetiva sobre la misma, con el objetivo final de automatizar el mayor número de tareas. Se seleccionará un proceso típico de compra de productos vía Web y se mostrarán las ventajas de su automatización.

Palabras clave: métodos, dinámicas, diagramas, automatizar, tareas.

Resum

El constant creixement de les organitzacions i empreses que requereixen d'una millora continua en els seus mètodes de treball i en la gestió dels seus processos comercials per tal de crear dinàmiques de treball eficients i adequades a la situació de competitivitat actual.

A aquest treball es proposa l'estudi d'un cas al que es tracta la situació d'una empresa del sector tèxtil fictícia quines metodologies de treball son rudimentaries, el que està provocant que es queden enrere davant els seus competidors. L'objectiu fonamental és modelar i implementar els processos de negoci de la empresa. Per això, s'ha elaborat un estudi sobre la empresa, generant diferents diagrames que mostren informació objectiva de la mateixa, amb l'objectiu final d'automatitzar el major nombre de tasques que es realitzen. Es farà un procés típic de compra de productes via Web i es mostraran les ventatges de la automatització.

Paraules clau: mètodes, dinàmiques, diagrames, automatitzar, tasques.

Abstract

The constant growth of organizations and companies requires continuous improvement in their working methods and in the management of their business processes in order to create efficient work dynamics that are appropriate for the current competitive situation.

This paper proposes a case study that deals with the situation of a fictitious textile company whose work methodologies are very rudimentary, which is causing them to fall behind their competitors. The main objective is to model and implement the businesses process of this company. For this, a study on the company has been prepared, generating different diagrams that show objective information about it, with the final goal of automating the largest number of tasks that are performed. A typical process of web browsing buying will be done and the advantages of automation will be shown.

Keywords: methods, dynamics, diagrams, automating, tasks.

Índice de contenidos

Índice de figuras

1. Introducción

1.1. Justificación

Con el paso del tiempo, la modernización y el auge de las nuevas tecnologías digitales, la competitividad de mercado en cualquier sector es increíblemente alta y una empresa que resultaba desconocida, puede llegar a ser una gran potencia mundial en un corto espacio temporal. Para estar a la altura de seguir compitiendo en el mercado de una forma simple, es decir, recortando gastos, surgieron una serie de modelos de procesos con el fin de agilizar y optimizar los procesos de una empresa de cara a una venta o para adquirir beneficio.

Uno de estos modelos de procesos, convertido ya en estándar, es BPMN (Business Process Model and Notation, [2.1]), que, mediante su notación gráfica, permite crear modelos basados en flujos de trabajo. Esta notación es realmente interesante porque resulta fácilmente entendible por todos los usuarios del negocio y desde su nacimiento está experimentando un crecimiento de uso notable, formando parte del plan estratégico de gran cantidad de organizaciones. La figura 1.1 muestra un proceso genérico en el que podemos distinguir ítems fundamentales del estándar, así como las calles, los eventos o las actividades, por ejemplo.

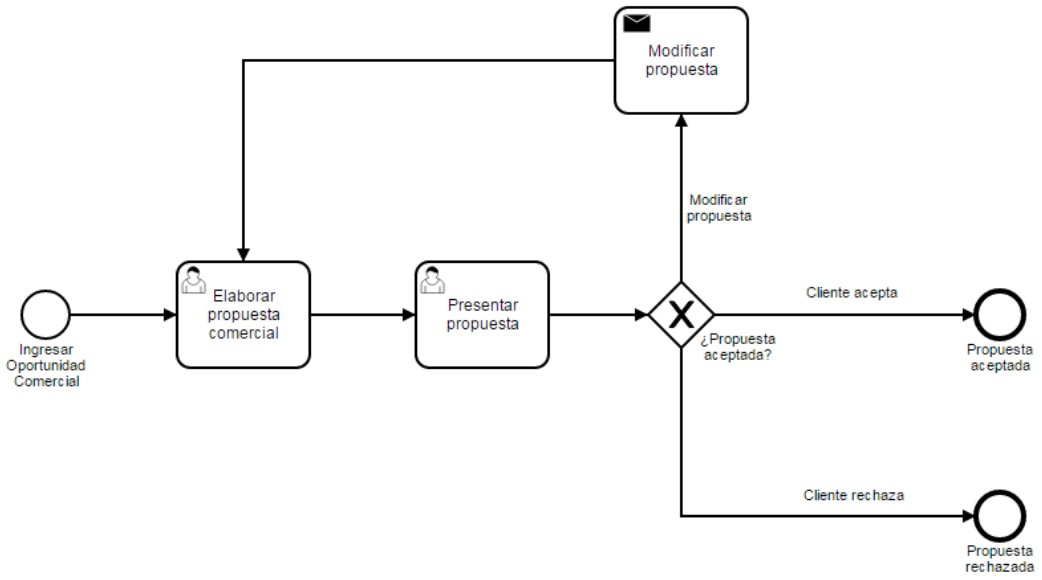


Figura 1.1. Ejemplo de modelo de proceso con elementos básicos.

Basándonos en el estándar BPMN proponemos realizar la mejora del proceso de compra vía web de una empresa dedicada a la fabricación y venta de productos textiles, fundamentalmente centrada en el mercado mayorista. Para poder realizar todos los modelos correspondientes, debemos primero conocer los fundamentos de la empresa, cómo se estructura y tener en cuenta lo que la misma empresa está buscando mediante el control de flujo de procesos que se propondrá. Una vez se establezca el modelo

correspondiente, se procederá a su implementación mediante el editor Camunda Modeler, una plataforma open-source que permite trabajar con flujos de trabajo basados en BPMN que se adapta a las necesidades del proyecto.

1.2. Objetivos

El principal objetivo de este Trabajo de Fin de Grado es implementar procesos de negocio usando motores de código abierto basándonos en un caso de venta online de una empresa textil. Se creará una interfaz de cliente que interaccione con una REST API del motor de procesos de negocio

Para ello, se tienen en cuenta los siguientes subobjetivos:

- Realizar un estudio del entorno de la empresa.
- Realizar un estudio de la estructura de la empresa.
- Realizar un modelo de procesos de la venta online del producto.
- Implementar el modelo usando Camunda.
- Implementar una interfaz personalizada para la web con ASP y C#.
- Utilizar la API REST de Camunda para realizar la comunicación cliente-servidor.
- Evaluar el funcionamiento del servicio de venta online una vez implementado.
- Extraer conclusiones en cuanto al uso de implementar automatización y optimización de procesos.

1.3. Estructura del Trabajo de Fin de Grado

La memoria del Trabajo de Fin de Grado está organizada en diez capítulos y tres anexos.

Capítulos

1. **Introducción:** Se expondrá brevemente la justificación del proyecto, los objetivos del mismo y su estructura.
2. **Planteamiento del problema**. Se explicará con detalle la situación actual de la empresa, factores externos e internos y por qué es necesario proponer una solución.
3. **Modelado de procesos del caso de estudio**. Se propondrá un modelado como solución al problema actual y se explicará con detalle cómo se ha llegado hasta el mismo.
4. **Arquitectura del sistema.** Se explicará la arquitectura utilizada en el proyecto y cómo se estructura en diferentes capas.

5. **Lenguajes y herramientas de desarrollo**. Se explicará con detalle qué herramientas y lenguajes de programación se han utilizado para resolver el problema.
6. **Camunda: configuración, despliegue de procesos y servicios web REST**. Se explicará en detalle cómo se ha configurado Camunda y cómo se despliegan los procesos, además de dar algún detalle sobre cómo funcionan los servicios web basados en la API REST.
7. **Implementación**. Se darán algunas pinceladas sobre cómo se ha creado el código cliente, las demás clases auxiliares y la base de datos y sobre su funcionamiento.
8. **Conclusiones**. Se extraerá una valoración del proyecto y unas conclusiones en torno a si merece la pena en términos de eficiencia para la empresa, así como posibles mejoras para el futuro.
9. **Relación con estudios cursados.** Se analizará la relación que ha tenido el proyecto con los estudios que se han realizado.
10. **Bibliografía**. Se citarán todas las referencias bibliográficas.

Anexos

1. **Listado de términos.** Lista con algunos términos que se usan en el documento y una pequeña aclaración de su significado.
2. **Manual de usuario e instalación**. Instrucciones para realizar la instalación del proyecto para el usuario.
3. **Modelo de procesos.** Se adjuntará la figura correspondiente al modelo de procesos de la empresa

2. Planteamiento del problema

2.1. Información general del caso y la empresa

Vaya Tela S.L. es una empresa mediana y familiar fundada en el año 1996 por Rodrigo Leal Ruiz y se dedica a la fabricación y venta al por mayor principalmente de productos textiles, incluyendo hilaturas, telas y ropa.

La empresa cuenta con una amplia cartera de clientes fijos, debido a los años en el negocio, pero estos no aportan los suficientes beneficios como para expandir el negocio. Es por esto que la junta directiva de Vaya Tela S.L. propuso realizar, por una parte, una mejora en cuanto a la publicidad de la empresa y una inversión para captar clientes, y por otra parte, realizar una revisión de los procesos de negocio junto con una implementación de una web de compra online, ya que hasta ahora los pedidos se hacían mediante llamada telefónica a nuestras oficinas.

2.2. Factores internos y externos de la empresa

Para entender algo más el funcionamiento y las causas de la situación actual de la empresa es necesario conocer tanto sus factores internos como externos. Los factores internos hacen referencia a aspectos como la gestión, el modelo de negocio, la cultura, etc, de la empresa. Por otro lado, los factores externos hacen referencia a aspectos que no confieren directamente a la empresa como impuestos, subvenciones, competidores, etc.

De esta forma, en cuanto a **factores internos** tenemos:

- **Propietarios**: grupo de la sociedad limitada. Todos pertenecen a la familia del fundador de la empresa, aunque cualquier empleado o interesado en la empresa puede ser socio.
- **Misión**: objetivo fundamental de la empresa. En este caso, la misión de Vaya Tela S.L. es llegar a ser un referente a nivel estatal en su sector mediante la utilización de los mejores materiales y teniendo muy en cuenta las opiniones de sus clientes.
- **Junta directiva**: órgano que toma las decisiones a medio y largo plazo de la empresa. Actualmente está formada por cuatro personas, encargadas de proponer decisiones de cuatro áreas diferentes de la empresa, que finalmente se votarán entre todos ellos.
- **Estructura organizacional**: como ya se ha comentado, la empresa que se estudia es una empresa mediana que, concretamente cuenta con 26 empleados, de los cuales 8 de ellos son miembros familiares del propietario, y además son socios de la empresa. La empresa cuenta con cuatro áreas fundamentales de funcionamiento: dirección, secretaría, taller y almacén. La dirección es el área encargada de las decisiones estratégicas de negocio. El área de secretaría es la encargada de recibir las llamadas entrantes o dudas referentes a pedidos o al

propio funcionamiento de la empresa, así como de llevar las cuentas de la empresa y cumplir los aspectos legales. El taller es el lugar donde se fabrican los productos que se venden a los clientes y, por último, el almacén es el área encargada de guardar, etiquetar y enviar los pedidos que llegan fabricados desde el taller. De los 26 empleados que tiene la empresa pertenecen: 4 a dirección, 4 a secretaría, 10 a taller y 8 a almacén. Esta división de trabajadores en las diferentes áreas de la empresa se debe a que se tienen que tramitar los pedidos de forma manual, así como realizar las cuentas y planificaciones. Una vez se modifique el sistema de recepción de pedidos, no será necesario tener tanto personal de secretaría, ya que dejará de ser una de las tareas que esta área realizará.

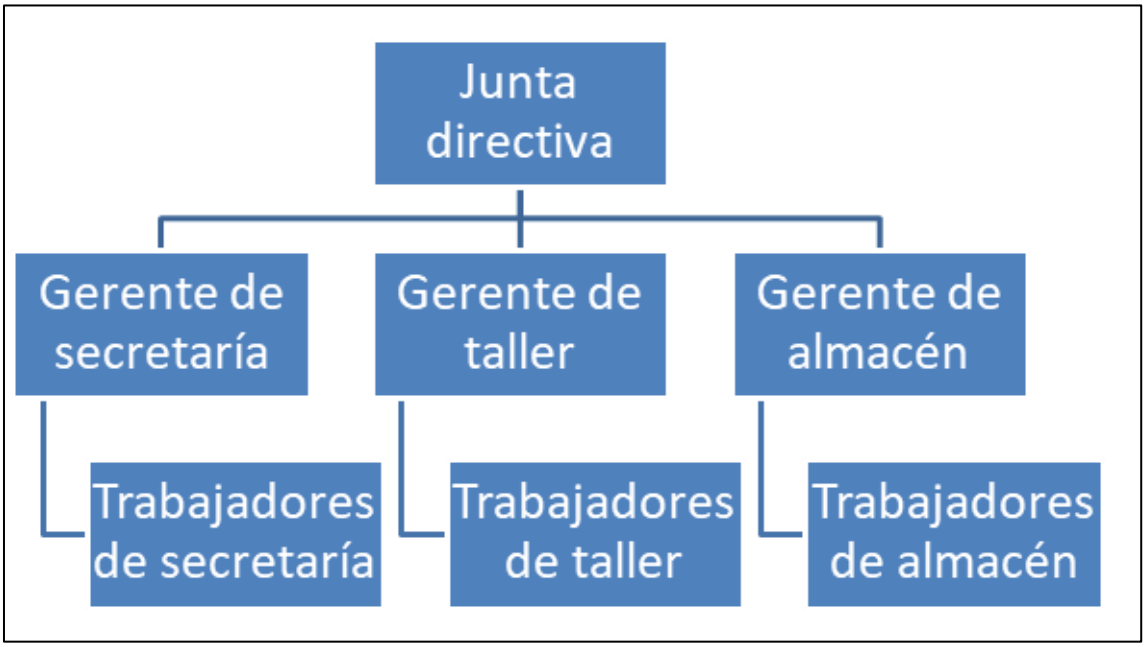


Figura 2.1. Estructura organizacional de Vaya Tela S.L.

- **Comunicación**: entre las diferentes áreas de la empresa. Se realiza de forma jerárquica, es decir, de arriba hacia abajo, o viceversa, en la estructura organizacional. Hasta la fecha se está realizando de forma informal y verbal.
- **Cultura**: infundida en la organización. En este caso es una cultura basada en la familia, ya que se trata de una empresa familiar, como ya se ha comentado.
- **Aprendizaje**: una vez llega un nuevo trabajador o si se relocaliza a un trabajador actual. Se basa en que alguien con más experiencia en ese ámbito que trabaja actualmente en la empresa le instruya.
- **Capital**: del que dispone la empresa. Básicamente la empresa cuenta con las instalaciones y la maquinaria del taller.

En contrapartida tenemos los **factores externos**, divididos en el entorno específico y el entorno general. El entorno específico "*comprende aquellos elementos externos a la empresa que están relacionados estrechamente con ella y, por tanto, tienen una influencia muy directa*", mientras que el entorno general engloba *"factores no tan*

directamente vinculados a la empresa pero que en muchas ocasiones tienen una influencia decisiva".

Comenzando con el **entorno específico**:

- **Proveedores**: a los que la empresa encarga sus *"inputs"*. En este caso, los proveedores son fundamentalmente de materias primas para el taller como lino, lana, etc. Aunque también hay otros apartados que necesitan proveedor como los productos de limpieza, por ejemplo.
- **Clientes**: a los que la empresa vende sus productos. Básicamente se trata de otras grandes superficies de venta al por menor que tienen un stock de cada producto que van reponiendo, aunque también se puede dar el caso de ventas a diseñadores, constructores, etc.
- **Competidores**: con los que la empresa tiene que compartir mercado. Son las grandes empresas manufactureras del sector textil nacional.

Y finalizando con el **entorno general**:

- **Componentes económicos**: como la salida de la reciente crisis económica, las tasas o impuestos que hay que pagar al estado, etc. En España las tasas que tiene que pagar la empresa son mucho más altas que si operara en un país Oriental, por ejemplo.
- **Componentes tecnológicos**: referentes a maquinaria e intrusión de nuevas tecnologías. En cuanto a maquinaria la empresa está bien situada, pero no hace un uso completo de las nuevas tecnologías por lo que se están implementando mejoras.
- **Componentes político-legales**: como las normas de contratación de empleados, las especificaciones que tiene que cumplir la empresa, etc. La empresa sigue firmemente las especificaciones regulatorias del marco europeo y español y el área de secretaría está al tanto para realizar las modificaciones oportunas en algún aspecto de este ámbito si fuese necesario.
- **Componentes demográficos**: como la edad media de la población, el nivel de ingresos medio, etc. No es un aspecto fácilmente perceptible y sería necesario realizar estudios para comprobar la influencia de este componente.
- **Componentes socioculturales**: como el estilo de vida, los hábitos, etc. España es un país multicultural y se tiene en cuenta a la hora de crear un catálogo de productos.
- **Componentes medioambientales**: como el cambio climático o el daño causado al medio ambiente. La empresa trata de ajustarse a los marcos establecidos por la Unión Europea y España, pero sin poner mucho esfuerzo adicional ya que supondría un sobrecoste muy alto.

2.3. Funciones internas en la empresa

Una vez conocidos los componentes internos y externos que influyen en mayor o menor medida en la empresa, es necesario poner cierto énfasis en conocer las tareas que se realizan dentro de la misma. Con este fin se exponen a continuación los casos de uso referentes a cada tipo de empleado para ampliar la información mencionada anteriormente:

- **Empleado de dirección**

El empleado de dirección o directivo se encarga de realizar el planteamiento estratégico de la empresa, tanto a largo como a medio plazo, y de la toma de decisiones en cuestiones imprevistas y comunicarlas a los demás empleados debidamente.

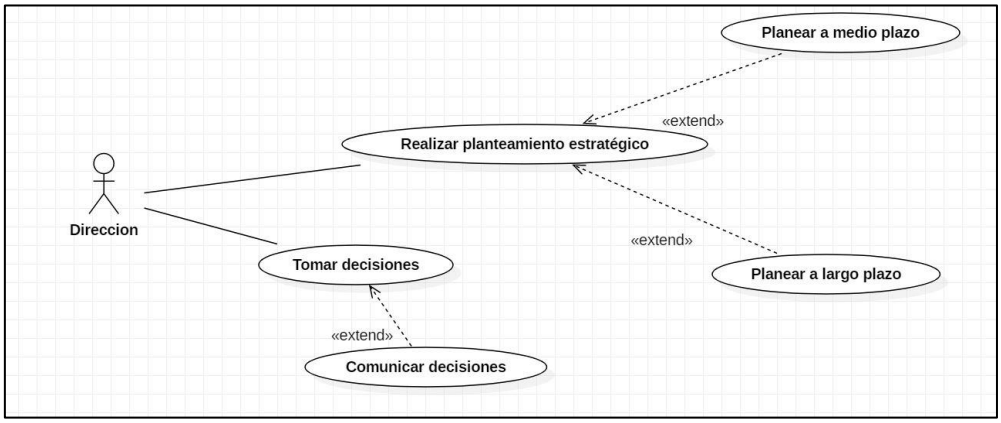


Figura 2.2. Funciones de un empleado de dirección.

- **Empleado de secretaría**

El empleado de secretaría tiene muchas funciones. Recepcionan los pedidos y se los envían al taller, además de atender las quejas de los clientes o de proporcionarles información si se la requieren. Además, controlan los gastos, facturaciones y presupuestos de la empresa.

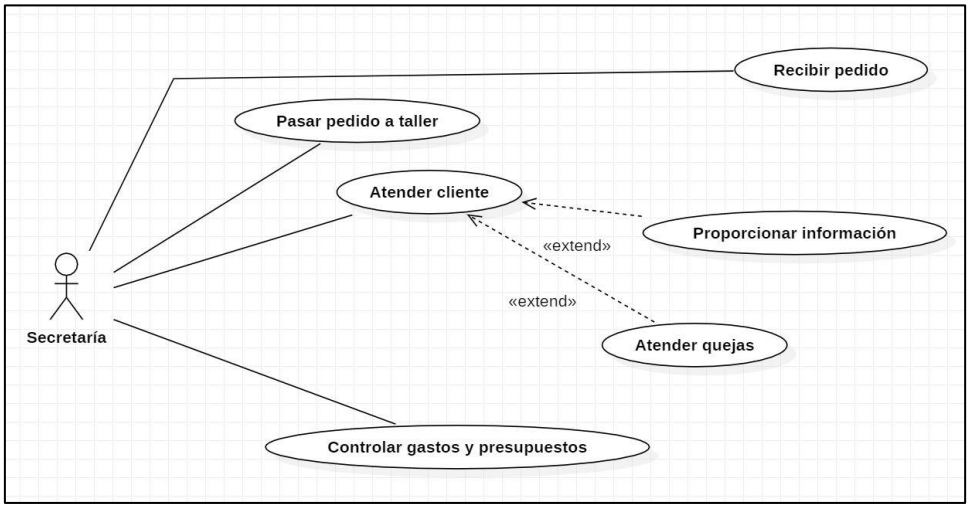


Figura 2.3. Funciones de un empleado de secretaría.

- **Empleado de taller**

El empleado de taller se encarga del mantenimiento de la maquinaria del taller, de fabricar los pedidos que se van realizando con la maquinaria y materiales de los que dispone. Si necesita algo extra, puede realizar pedido de materiales al proveedor.

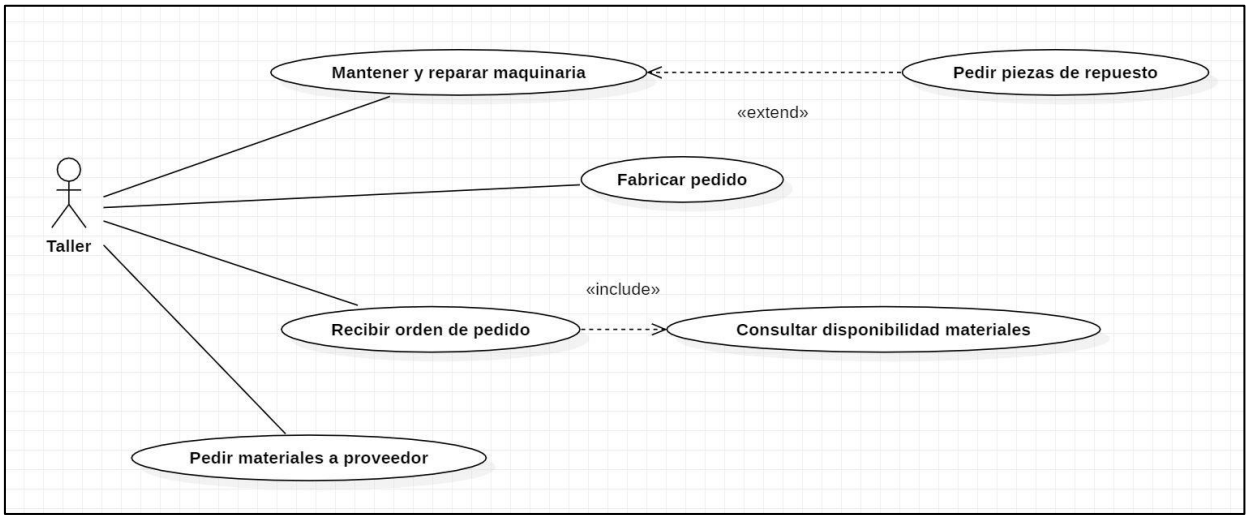


Figura 2.4. Funciones de un empleado de taller.

- **Empleado de almacén**

El empleado de almacén se encarga de recoger los pedidos fabricados para posteriormente prepararlos para su envío. Además, puede comprobar el stock actual de materiales y los pedidos que todavía no han sido enviados.

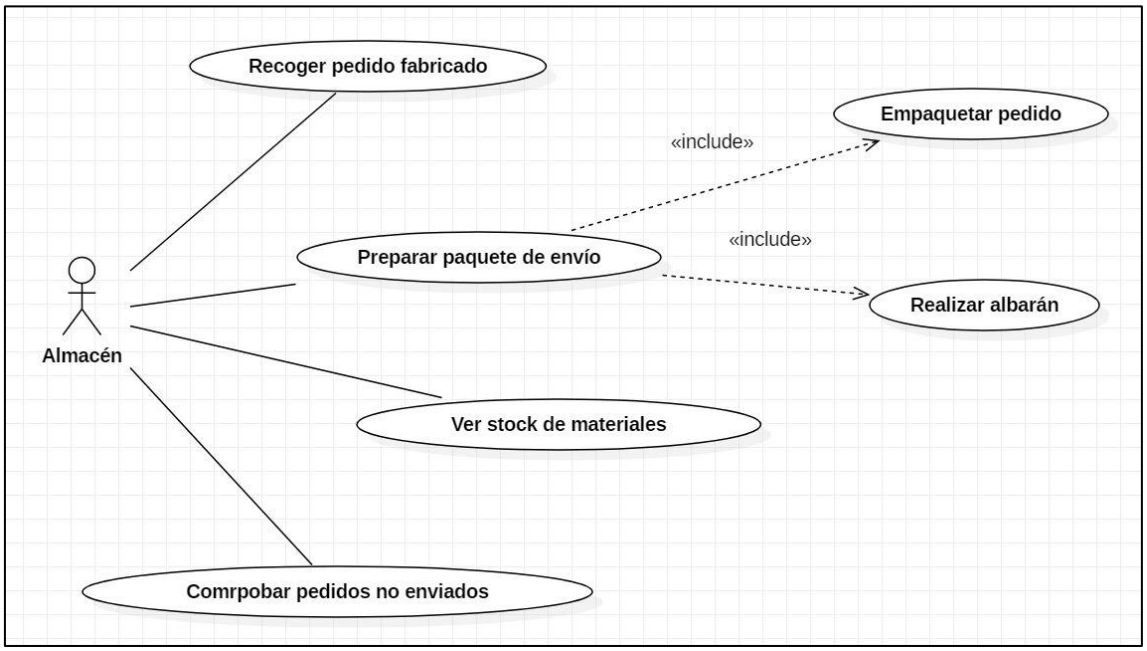


Figura 2.5. Funciones de un empleado de almacén.

2.4. Modelado inicial de la interfaz

Con el objetivo de crear una aplicación web que represente y englobe todas las ideas vistas anteriormente tal que sea un reflejo de la propia organización, y además sirva de forma eficiente para lograr el fin marcado, que no es otro que el de conseguir automatizar algunos procesos de la empresa, se han creado una serie de modelos iniciales mediante MockFlow, que servirán de base para el proyecto de la aplicación web.

En la siguiente imagen podemos ver lo que sería la página inicial, desde la cual se pueden añadir líneas de pedido, que son tuplas de artículo y cantidad, acceder a ver la cesta, finalizar pedido e iniciar sesión. Además, se podrá acceder al menú de navegación desde todas las páginas.

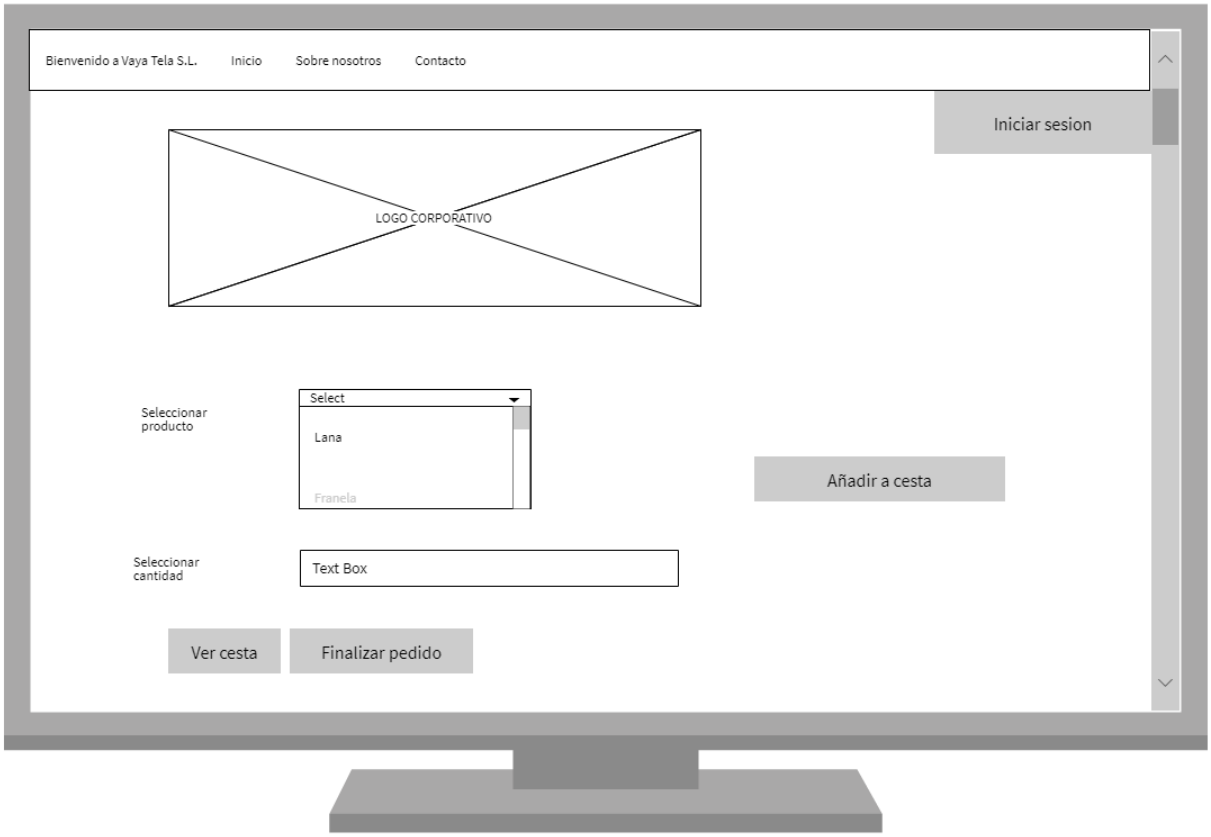


Figura 2.6. Primer modelo de página inicial.

En la siguiente imagen se pueden observar los elementos que podría tener la página correspondiente a ver la cesta del pedido. Parece bastante simple, pero puede resultar de utilidad ya que debería permitir borrar líneas de pedido previas.

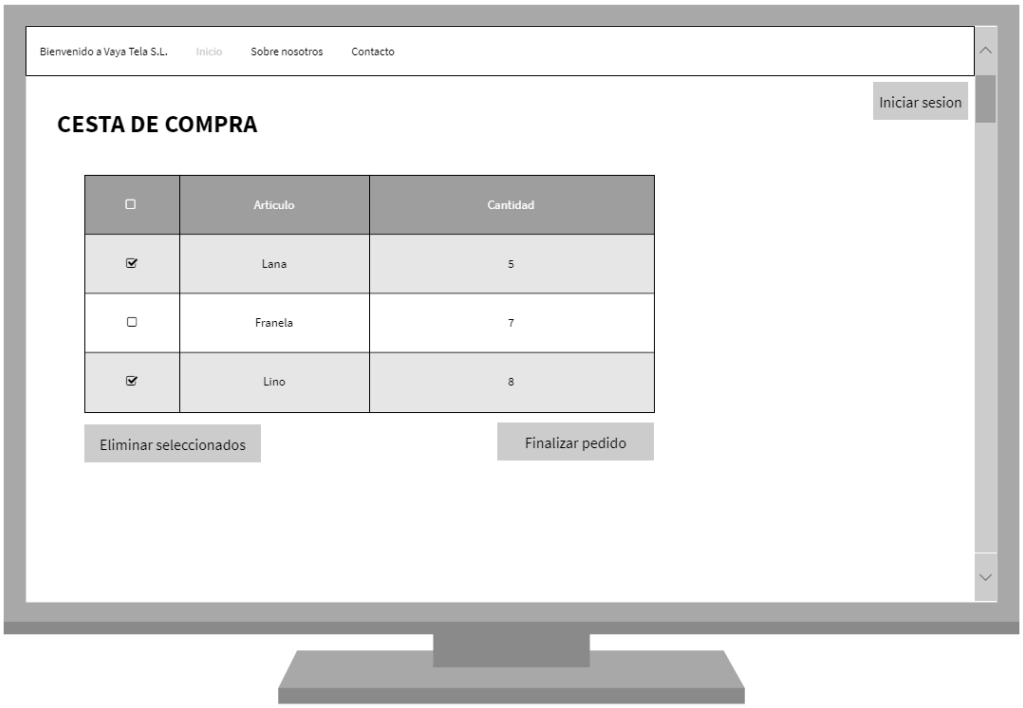


Figura 2.7. Primer modelo de página de cesta.

Por último, se muestra también el primer modelo de página de inicio de sesión. Resulta interesante recalcar la posibilidad de incluir un botón para regresar a la página anterior, además del botón de inicio. Desde esta página de inicio de sesión, se podrá acceder mediante las credenciales de cada usuario, además de poder acceder a la página de registro, si el usuario es nuevo.

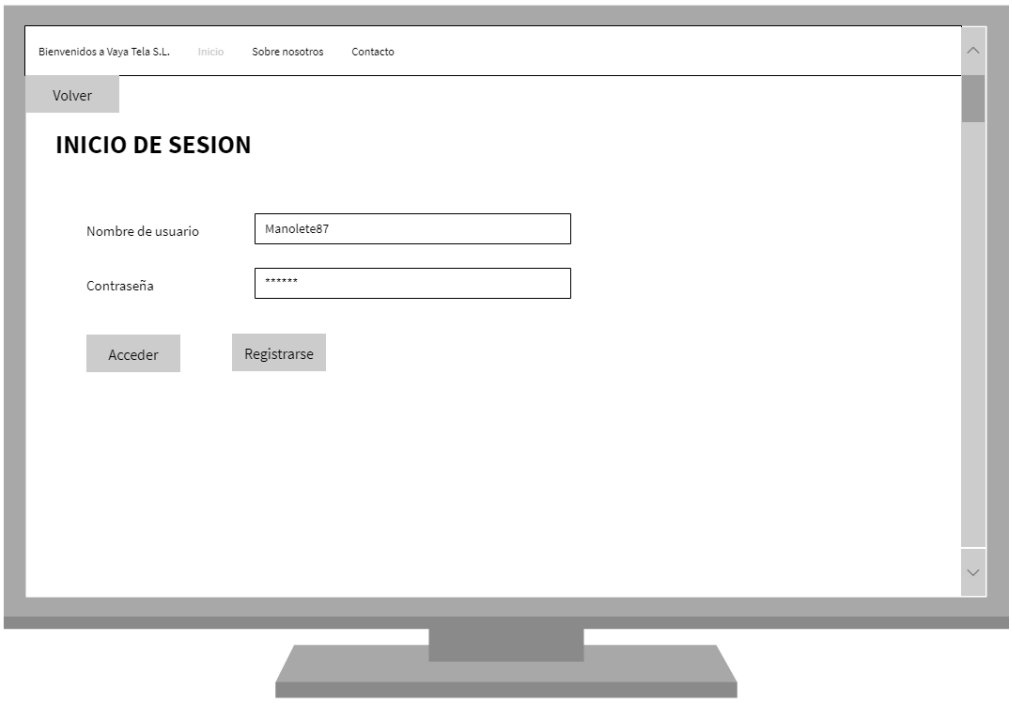


Figura 2.8. Primer modelo de página de inicio de sesión.

3. Modelado de procesos

3.1. Origen e historia del modelado BPMN

La notación BPMN (Business Process Model and Notation) es un nuevo estándar de modelado de procesos de negocio. La notación está basada en los *flowchart* y es lo suficientemente expresiva para que pueda ser entendida tanto en el ámbito del negocio como en el ámbito del desarrollo de software, es decir por analistas de negocio y por analistas informáticos.

Los modelos de BPMN pueden utilizarse para documentar los procesos de una organización o los procesos que se ejecutan en varias organizaciones. Pueden también ser empleados como punto de entrada para un procedimiento de automatización de procesos de negocio.

Además, la notación gráfica facilita el entendimiento del rendimiento de colaboraciones y transacciones de negocio entre organizaciones. Esto asegura que los negocios se entenderán a sí mismos y a los participantes en sus procesos y permite a los negocios ajustarse a cambios en la organización, así como a eventos B2B (Business 2 Business) rápidamente.

El origen de BPMN se remonta a inicios de siglo. El BPMI (Business Process Management Initiative, [2.4]) se fundó en agosto del 2000 que, sin ánimo de lucro pretendía promover la estandarización de procesos de negocio para la mejora empresarial en ámbitos de *e-commerce* (comercio electrónico) y B2B. Esta estandarización que promovía BPMI tenía la misión de "promover y desarrollar el uso de Procesos de Control de Negocio a través del establecimiento de estándares para diseño, ejecución, mantenimiento y optimización de estándares de proceso".

En el año 2004 el BPMI presentó su primera iniciativa de BPMN, lo que provocó que en 2005 el OMG (Object Management Group, [2.2]) se interesara en ella, para que finalmente, ambas organizaciones trabajasen en conjunto, se combinasen, según explica OMG. Por tanto, tras esta primera versión presentada en mayo de 2004 por el BPMI, OMG se hace cargo de BPMN, que seguiría el esquema propuesto de BPD (Business Process Diagram, [2.6]), en la que, por medio de elementos gráficos y usando la técnica de diagrama de flujo se provee toda la información necesaria para el análisis.

En mayo de 2006 OMG lanza su primera versión de BPMN, la versión 1.0, que ya incluía todo un conjunto de elementos gráficos, así como una semántica informal y las primeras herramientas compatibles.

En su versión 1.1, lanzada en enero de 2008, aparecen muchas más herramientas para dar soporte a la notación, y algún cambio menor, mientras que, en su posterior versión, la 1.2, lanzada en enero de 2009, se corrigen algunos errores de la versión anterior.

Finalmente, en el año 2011, es lanzada oficialmente la versión 2.0, que ya incluye el formato de intercambio y describe semiformalmente la semántica.

BPMN 2.0 es la penúltima versión de BPMN. Existe una versión posterior, la versión 2.0.2, lanzada en enero de 2013, pero cuyos cambios apenas son destacables, ya que se centran sobre los formatos de intercambio de archivos.

3.2. Necesidad histórica

Históricamente ha existido el problema de la comunicación entre diferentes órganos de una empresa, así como entre diferentes empresas. En la gestión de procesos no ha sido diferente. Muchas veces ocurría el fenómeno del *misunderstanding* o malentendido debido a falta de especificación en las comunicaciones que se debían emplear, así como en los canales que se debían realizar o el idioma que se debía utilizar.

BPMN proporciona una notación gráfica estándar, fácil de entender y de leer para todos los interesados en los procesos del negocio. Estos interesados cuentan con analistas de negocio (aquellos que detectan y definen o mejoran los procesos), los desarrolladores (aquellos que implementan los procesos), gerentes, jefes, directores o administradores (aquellos que monitorizan y controlan los procesos) y demás empleados (aquellos que ejecutan los procesos).

3.3. Estructura BPMN

BPMN se puede estructurar siguiendo diferentes acuerdos, aunque lo más común es organizarla según los elementos que la componen. Estos elementos son los siguientes y siguen esta jerarquía:

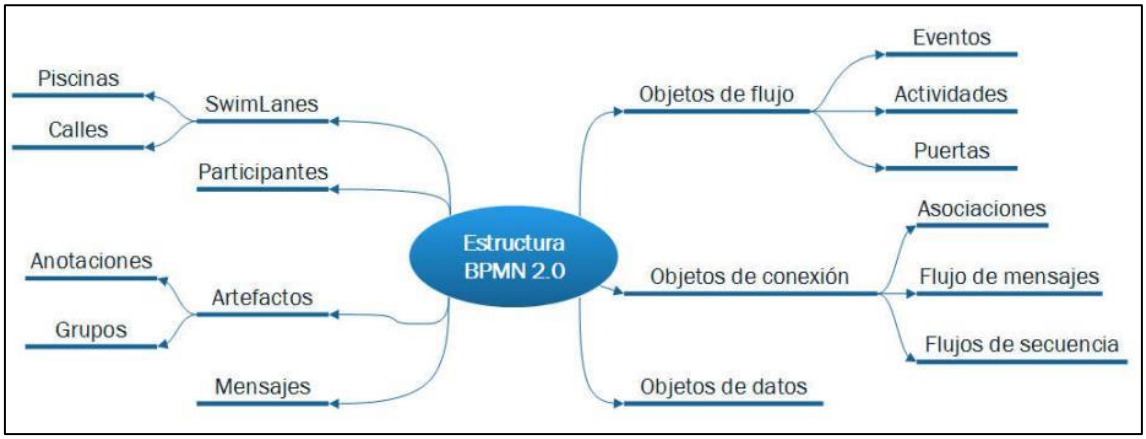


Figura 3.1. Estructura de elementos BPMN 2.0.

3.4. Modelo del caso de estudio

Este modelo se basa en el proceso de compra de cualquier cliente registrado vía web. Para ello, el cliente puede añadir y modificar líneas de pedido, y en cualquier momento iniciar sesión o registrarse. Una vez se obtiene un inicio de sesión válido, así como una cesta válida, se procede a finalizar la compra y a realizar el pago. Esto provoca que se genere un pedido en la base de datos con unas líneas de pedido determinadas que será perteneciente a ese cliente en concreto que será la información con la que tratará el equipo del taller. La imagen referente al modelo completo se encuentra en el anexo.

Explicación del modelo **por fases**:

1. **Fase de acceso y selección de pedido**
    - El cliente accede a la página web. Esto se recoge como una petición de acceso a la web que el servidor maneja y finalmente permite acceder al cliente. Posteriormente introduce los productos y cantidades que desea comprar e inicia sesión (o no) antes de darle a finalizar pedido. Puede también quitar líneas de pedido desde la vista de la cesta o registrarse si no tiene una cuenta creada.

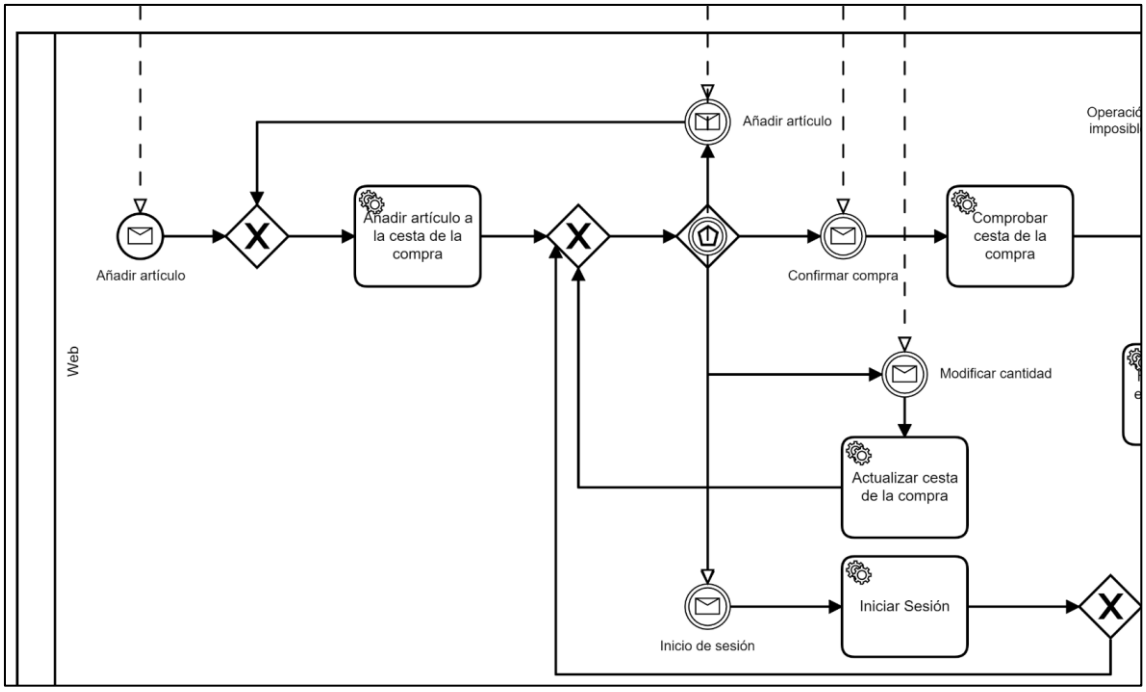


Figura 3.2. Fase de acceso y selección de pedido.

2. Fase de confirmación de compra

- Una vez el cliente ha finalizado su pedido y este es válido, ha de realizar el pago mediante su tarjeta de crédito e indicar la dirección de entrega del pedido. Además, si no tiene su cuenta iniciada, se obligará a iniciar una cuenta, y si no lo hace, se le redirige a la página principal.

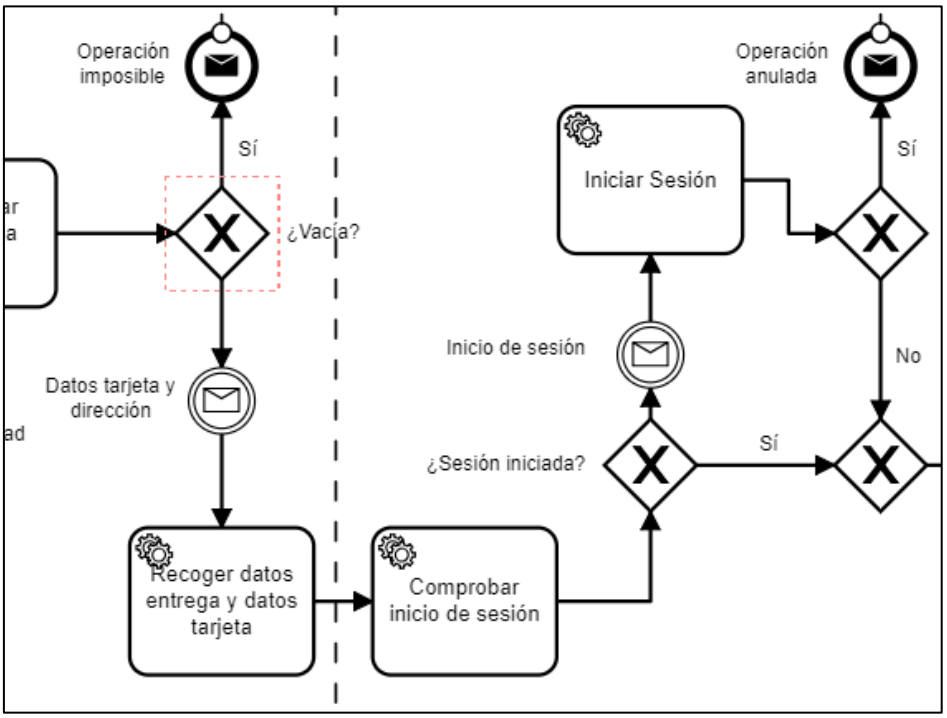


Figura 3.3. Fase de confirmación, parte 1.

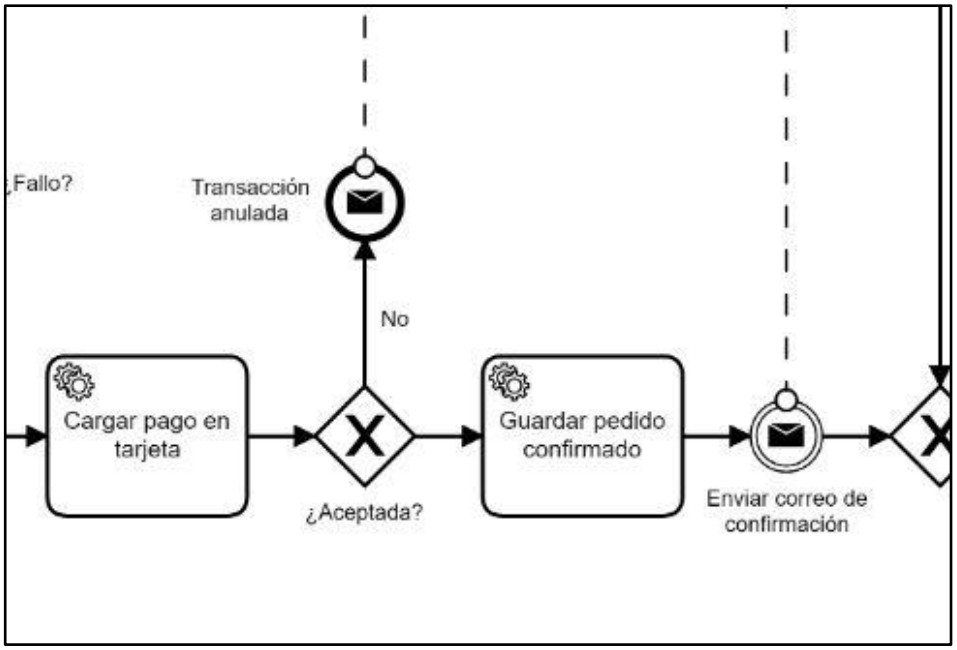


Figura 3.4. Fase de confirmación, parte 2.

3. **Fase de anulación**
   - Esta es una fase opcional en la que el cliente dispone de un espacio de tiempo en el que puede cancelar el pedido que ha realizado.

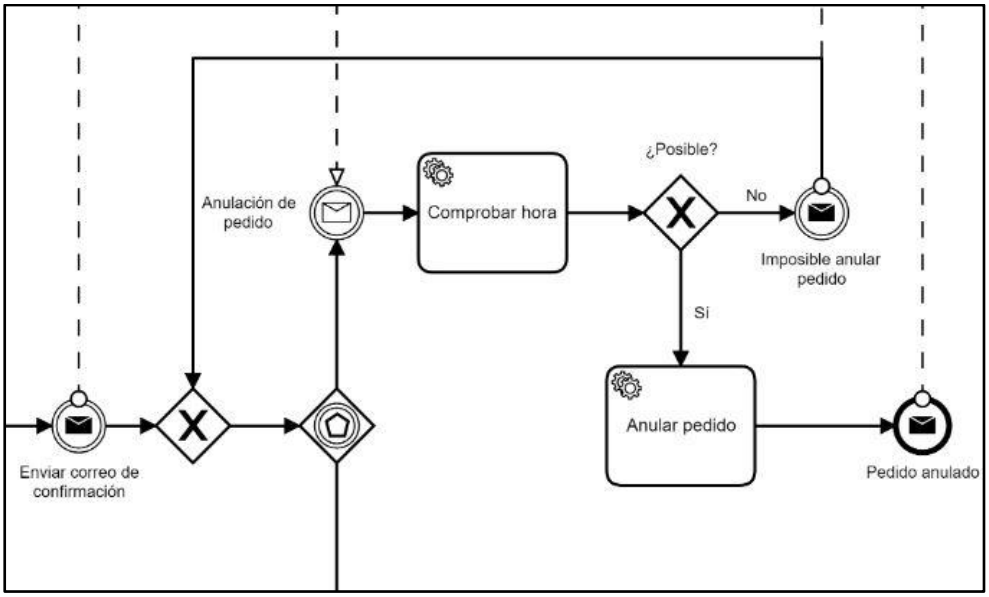


Figura 3.5. Fase de anulación.

4. **Fases de taller y almacén**
   - Son fases internas de la empresa en las que el cliente no interacciona más que para recibir el pedido de parte del transportista.

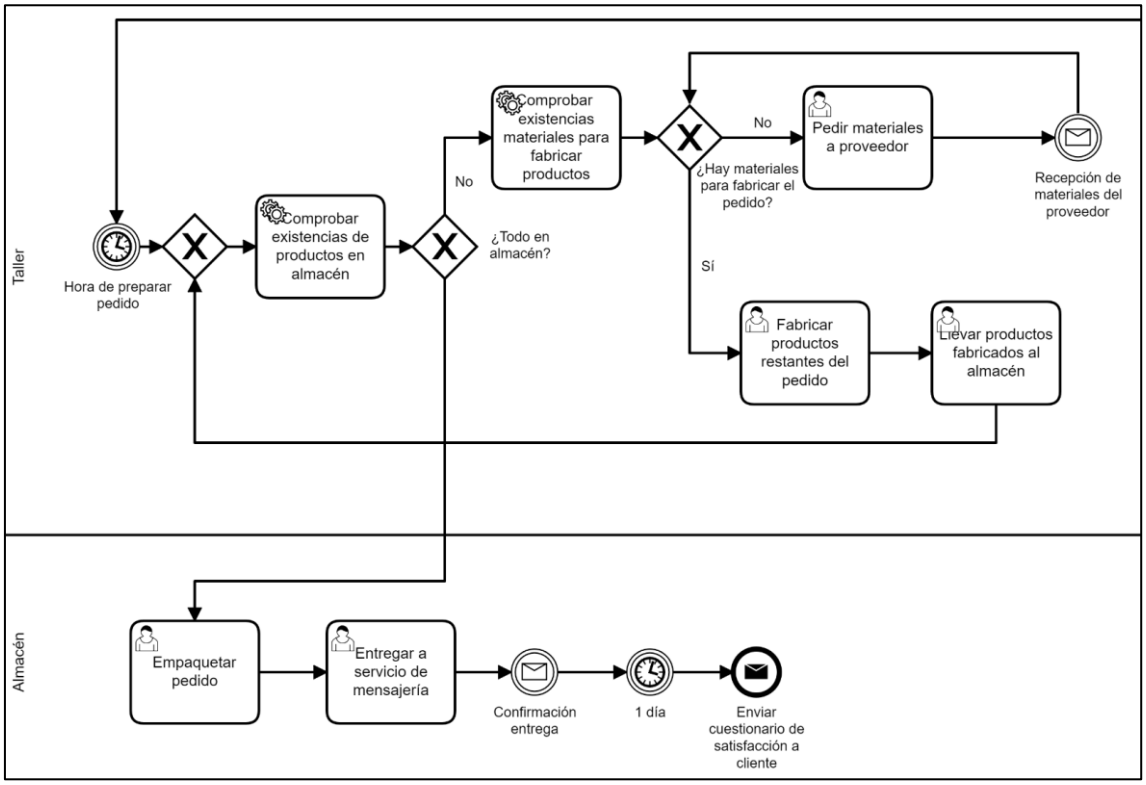


Figura 3.6. Fase de taller y almacén.

4. Arquitectura del sistema

4.1. Introducción a la arquitectura por capas

Cuando se realizan proyectos de programación de cierta magnitud, poder agrupar todo el código que se va escribiendo en diferentes grupos o capas facilita enormemente el trabajo continuado en el proyecto y es más fácil localizar cualquier fracción del código rápidamente.

Además, esta forma de trabajo añade robustez al proyecto, ya que si existe algún fallo inesperado cuando se ejecuta la aplicación, este fallo sólo afectará a su capa, siendo además fácilmente localizable ya que se reduce el *acoplamiento informático* (forma y nivel de dependencia entre módulos software). Todo ello se une mediante una API que intercambia información entre las capas.

Usualmente se utiliza el modelo de tres niveles o capas, ya que es simple diferenciarlas y aporta mucha solidez en comparación a no diferenciarlas, aunque también existen las arquitecturas de dos y cuatro capas.

4.2. Arquitectura del cliente ASP de Camunda

En nuestro caso, se ha optado por la arquitectura basada en tres niveles. Esto significa que el proyecto cuenta con tres capas claramente diferenciadas en las que interactúan tal como se puede observar en la imagen:

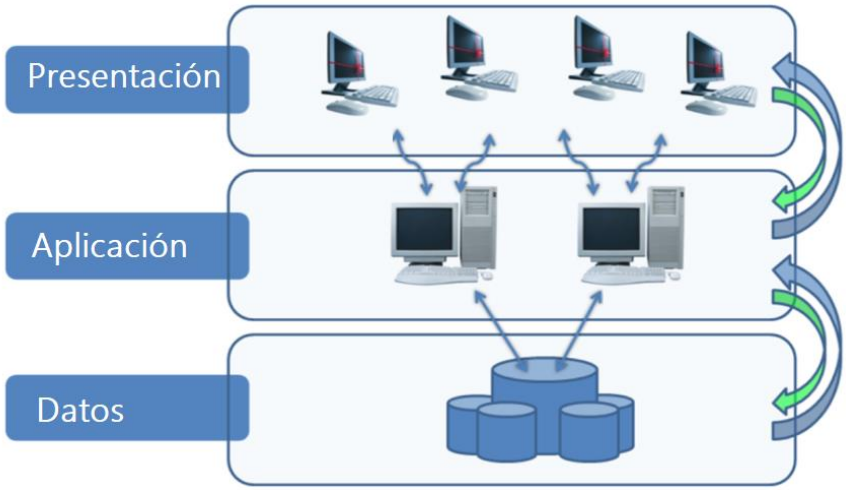


Figura 4.1. Arquitectura de tres capas.

Contenido y funcionalidad de cada capa:

1. **Capa de presentación** (o de usuario)

La capa de usuario engloba todo aquello que puede observar el usuario. Por ello, debe ser amigable con el cliente para que este sepa o intuya cómo realizar cada acción de forma sencilla y sin generar demasiadas complicaciones.

En esta capa es donde se recoge la información, en nuestro caso de pedidos, que es introducida por el usuario.

2. **Capa de aplicación** (o de negocio)

La capa de negocio agrupa todo aquello que interacciona, o bien con la capa de presentación, o bien con la capa de datos. Fundamentalmente se trata del código escrito de la aplicación en el que se establecen las reglas de introducción de datos por parte del cliente, para posteriormente tratar estos datos y enviarlos, modificarlos o borrarlos desde la capa de datos.

En nuestro caso, esta capa está compuesta por el código adherido a cada vista de la página, además de una clase de *querys* que genera oraciones para enviarlas a la clase de "control" que finalmente interactúa con la base de datos, y de la interacción existente entre el código escrito y el motor de procesos de Camunda.

La arquitectura utilizada, en resumen, contendría los siguientes elementos:

- Microsoft Visual Studio, el IDE que se ha usado para desarrollar el proyecto y engloba:
  - Formularios HTML, con la codificación de la web.
  - Diagrama BPMN, creado mediante Camunda Modeler.
  - Otro código .NET, como el de las clases intermedias que se explicarán más adelante.
  - Trabajadores externos, que realizan funciones auxiliares.
  - Enlace con la base de datos de la organización.
- Todo ello unido mediante un sistema de ficheros.

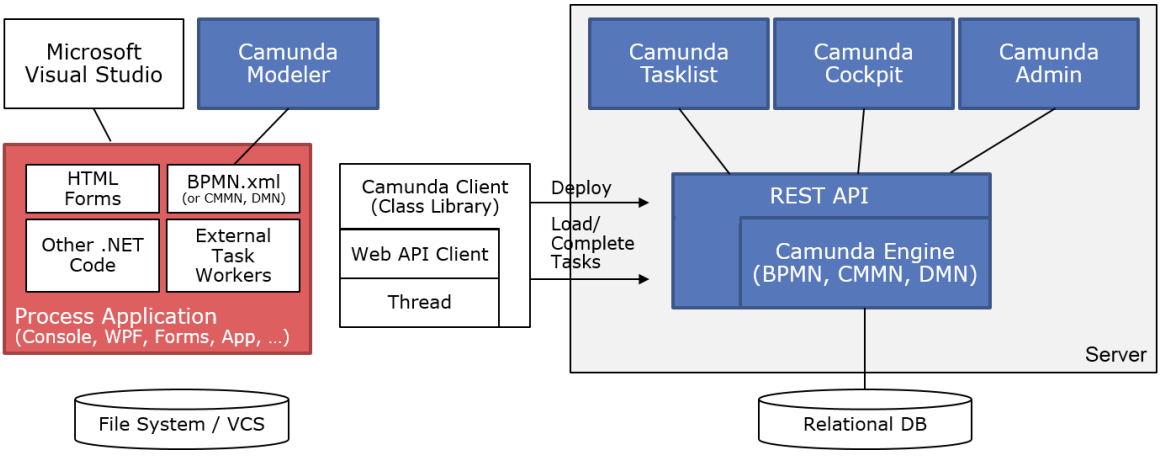


Figura 4.2. Esquema de arquitectura de la aplicación

3. Capa de datos

La capa de datos contiene las estructuras de los datos y los propios datos de la organización. Como se ha comentado, se accede a ellos desde la clase de control de la capa de aplicación vía consultas MySQL.

Se trata de una base de datos relativamente sencilla, estructurada de la siguiente forma:

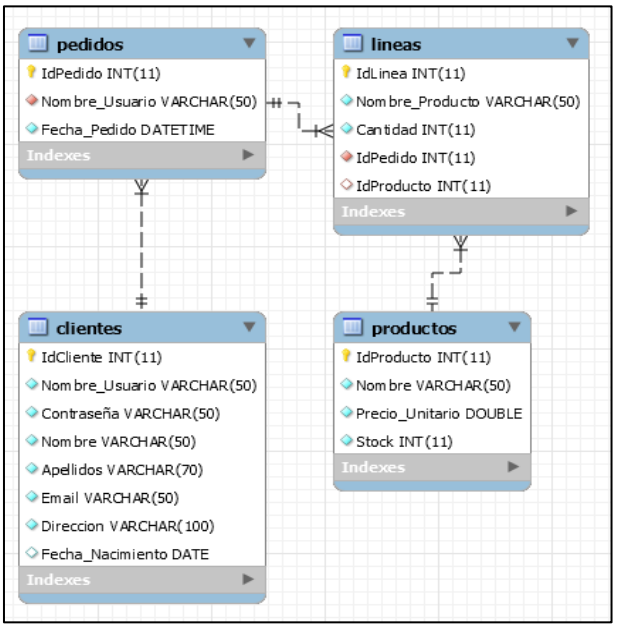


Figura 4.3. Esquema de la base de datos.

4.3. Arquitectura de Camunda

En cuanto a la arquitectura del motor de procesos de Camunda, pese a que esta fue originalmente diseñada para ser usada junto a Java, se ha conseguido poder acoplarla a su uso en C#.

La arquitectura del motor de procesos es la siguiente:

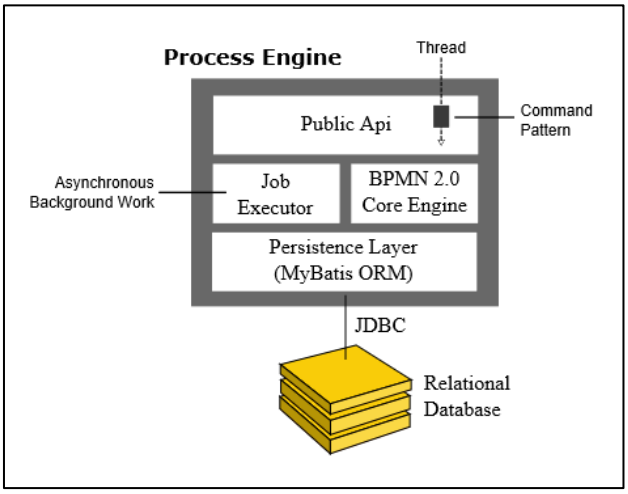


Figura 4.4. Esquema de arquitectura del motor de procesos.

Se encuentran los siguientes elementos:

- **API pública del motor de proceso**

Se trata de una API orientada al servicio que permite a aplicaciones Java (o compatibles), interaccionar con el motor de procesos. Las diferentes responsabilidades del motor de procesos se separan en servicios individualizados.

- **BPMN 2.0 Motor Núcleo**

Engloba un motor ligero de ejecución de estructuras de grafos, un analizador que transforma ficheros BPMN 2.0 XML en objetos Java y un set de implementaciones de comportamiento BPMN.

- **Ejecutor de trabajos**

Es la parte responsable de procesar el trabajo de fondo asíncrono como contadores de tiempo o continuaciones asíncronas de un proceso.

- **Capa de persistencia**

La capa de persistencia del motor, que se comunica directamente con la base de datos relacional de Camunda, se utiliza para salvar los procesos existentes.

5. Lenguajes y herramientas

5.1. Herramientas de modelado

- **Camunda Modeler**

Camunda Modeler es una aplicación de escritorio que permite crear modelos de flujo BPMN, así como modelos de decisión DMN. Permite a múltiples desarrolladores trabajar en conjunto en los mismos diagramas.

En concreto se ha trabajado con su versión 3.1.0, lanzada el 26 de abril de 2019, que soporta BPMN 2.0, CMMN 1.1 y DMN 1.1, aunque nos centraremos en el uso enfocado a BPMN, ya que permite crear modelos ejecutables, que es en lo que se basa fundamentalmente el proyecto.

Estos modelos ejecutables son ficheros con extensión XML que pueden ser leídos, manejados y ejecutados por diferentes entornos de desarrollo como NetBeans, Eclipse o Visual Studio. En este caso, se utilizará Visual Studio debido a que resulta compatible con este tipo de ficheros, además de ser el IDE (Integrated Development Environment) por excelencia en cuanto a programación en C# se refiere.

Se puede acceder a esta herramienta de forma gratuita desde la página web oficial de Camunda.

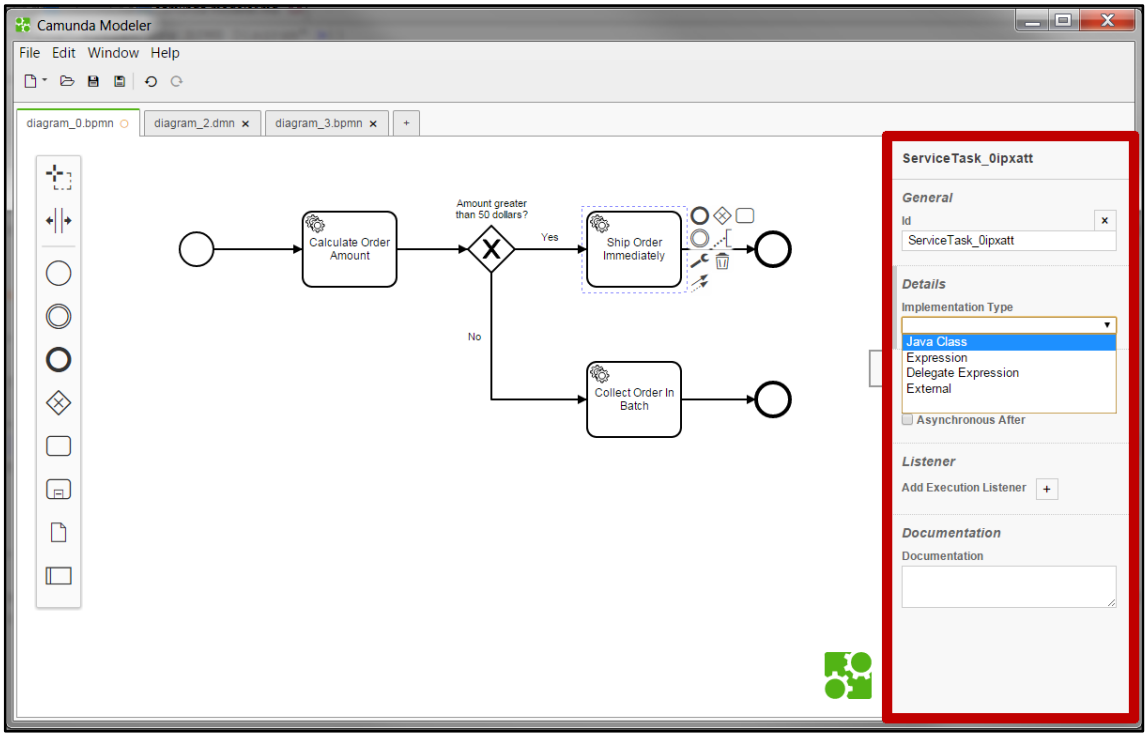


Figura 5.1. Interfaz de Camunda Modeler.

- **Camunda Workflow Engine**

Capaz de ejecutar la mayoría de los símbolos definidos en BPMN 2.0, el motor de procesos de Camunda automatiza diagramas de proceso para orquestación de servicios, flujos de tareas humanas o ambas.

Además, es posible acceder a él mediante REST para iniciar instancias, completar tareas y demás funciones. Este motor aporta, a grandes rasgos, una gran escalabilidad y rendimiento que se combinan con la persistencia de los resultados y que se ajusta perfectamente a este proyecto.

Se puede acceder a esta herramienta de forma gratuita desde la página web oficial de Camunda.

- **StarUML**

StarUML, en su versión 3.1.0 es el software que se ha utilizado para crear los diagramas de casos de uso del apartado 2.3 referente a las funciones internas de los trabajadores de la empresa.

StarUML es compatible con UML 2.x y permite crear diagramas de clase, objeto, casos de uso, componentes, despliegue, secuencia, etc. Además, es compatible con diagramas de entidad-relación, de flujo de datos y de secuencia de flujo.

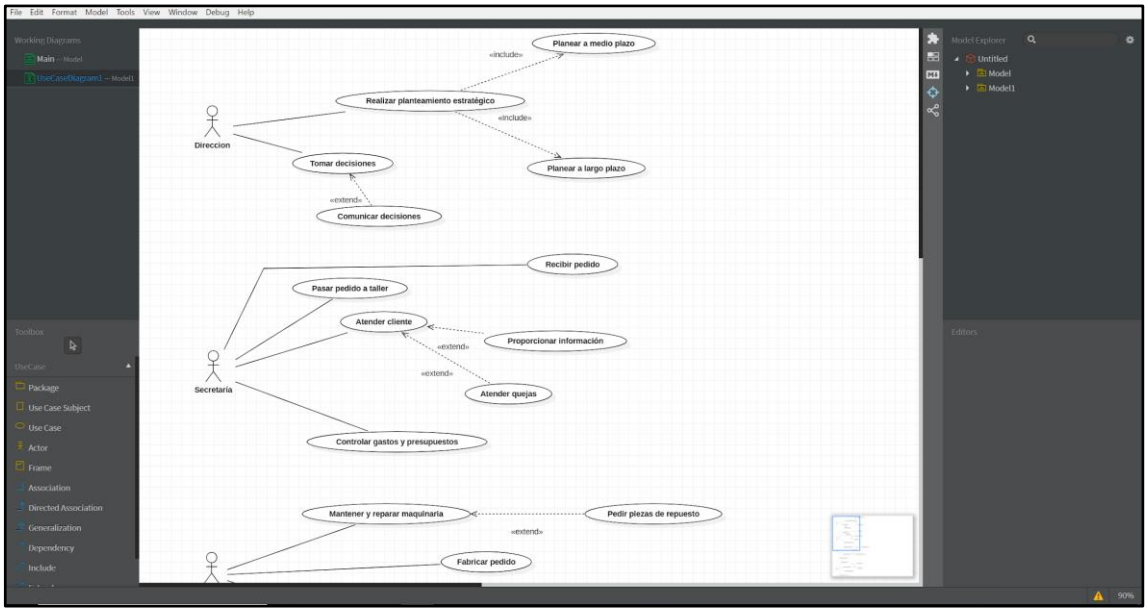


Figura 5.2. Interfaz de StarUML.

- **Microsoft Office Word**

Además de generar documentos de texto, Microsoft Office Word tiene otras herramientas que permiten la creación de diferentes modelos o esquemas. Uno de estos esquemas creado con esta aplicación es la figura 2, comentada anteriormente, que nos permite tener una idea de la estructura organizacional de la empresa.

- **MockFlow**

MockFlow es una herramienta de modelado que permite crear plantillas "a mano alzada" de las páginas de un sitio web. Su uso es muy intuitivo y fácil de entender y aprender a usar.

Se ha utilizado el editor web de MockFlow, que permite realizar las mismas funciones que la aplicación de la que disponen.

5.2. Herramientas de apoyo

- **MySQL**

MySQL es el sistema de gestión de base de datos relacional de código abierto más popular del mundo, junto con Oracle y Microsoft SQL.

En concreto, en el proyecto se ha utilizado su versión 5.7, lanzada a mediados de 2013, debido a que versiones posteriores generan fallos con el sistema de Camunda. Pese a esto, se pueden utilizar otras herramientas de desarrollo incluidas en el paquete de MySQL Community referentes a versiones posteriores, como MySQL Workbench.

Del paquete MySQL Community se han utilizado los módulos MySQL Workbench, del que se hablará posteriormente, MySQL Connectors, que permite acceder a la base de datos mediante código C# e InnoDB, que es el mecanismo de almacenamiento de datos propio de MySQL.

- **Servidor de aplicaciones Tomcat**

Desarrollado por integrantes de Apache, Tomcat *"funciona como un contenedor de servlets"*. A partir de su versión 4.0, Tomcat utiliza el contenedor Catalina.

En concreto, en el proyecto se ha utilizado la versión 7.11.0 correspondiente a Camunda BPM Tomcat, y la versión Apache Tomcat 9.0.12, debido a incompatibilidades de sus versiones más recientes con el repositorio Maven.

- **Azure DevOps**

"El término DevOps, compuesto por dev (desarrollo) y ops (operaciones), da nombre a una práctica de desarrollo de software que unifica el desarrollo y las operaciones de TI.

DevOps incluye prácticas principales, como planeamiento y seguimiento, desarrollo, compilación y pruebas, entrega, supervisión y operaciones. Estas prácticas, junto con las herramientas y tecnologías de DevOps, permiten automatizar el ciclo de vida de las aplicaciones. Los procesos que solían ser manuales y lentos para los equipos, como actualizar el código o aprovisionar un nuevo entorno, se pueden hacer de forma rápida y continua cuando se utilizan herramientas y prácticas de DevOps. Además, es más fácil cumplir las normas de seguridad y confiabilidad, porque estas consideraciones están integradas en el proceso."

Se ha utilizado en el proyecto principalmente como repositorio del código de la interfaz, y para tener un control de las versiones a medida que se iban añadiendo contenidos a la misma o resolviendo bugs (fallos inesperados del programa).

- **GitHub**

GitHub es un repositorio que utiliza el control de versiones de Git. Se ha utilizado en el proyecto para alojar la última versión y algún otro archivo necesario en la instalación por parte del usuario.

5.3. Herramientas de desarrollo

- **Visual Studio**

Visual Studio es un IDE compatible con gran cantidad de lenguajes de programación como C, C#, F#, Java, etc. Permite crear sitios, aplicaciones web y servicios web compatibles con .NET. Durante el proyecto se ha utilizado Microsoft Visual Studio Enterprise 2017 en su versión 15.9.6 acompañado de Microsoft .NET Framework en su versión 4.7. Esta versión ha sido adquirida a través de la intranet, aunque es posible acceder al software de Visual Studio en su versión Community sin coste adicional.

Mediante esta herramienta se han creado las páginas de la interfaz, así como cada elemento que las compone y se ha manejado cada botón o evento que el usuario puede accionar. También se han implementado clases intermediarias entre la base de datos y la aplicación. Todo ello implementado mediante C# y ASPX.

- **MySQL Workbench**

MySQL Workbench es una herramienta visual unificada para arquitectos, desarrolladores y DBA de bases de datos. Proporciona modelado de datos, desarrollo SQL y herramientas de configuración para el servidor incluyendo administración de usuarios, copias de seguridad, etc. Además, es una herramienta disponible para Windows, Linux y MacOS.

En el proyecto se ha utilizado la versión 8.0.x, con la cual se ha creado la base de datos de la empresa mediante querys (cadenas de consulta a la base de datos).

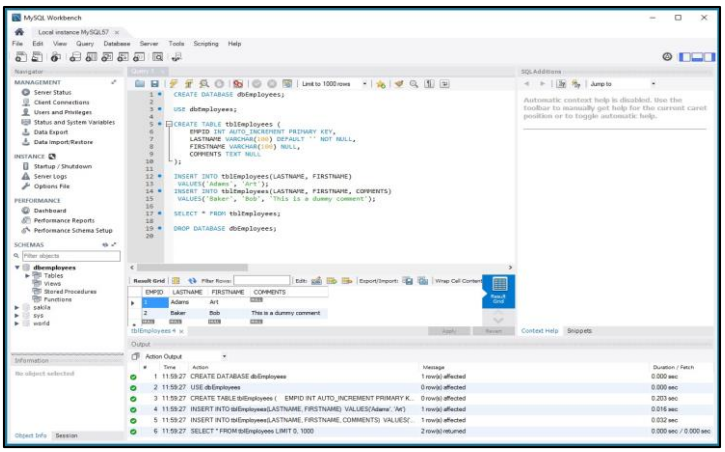


Figura 5.3. Interfaz de MySQL Workbench.

5.4. Lenguajes utilizados

- **C# con ASP.Net**

Derivado de C y C++, C# es un lenguaje OOP (Oriented Object Programming) desarrollado por Microsoft e incluido en su plataforma .NET. Aun así, pese a estar incluído en .NET, es un lenguaje de programación independiente. El núcleo del proyecto se basa en la combinación de C# y ASP.NET, ya que aquello con lo que el usuario interacciona son formularios web ASP, cuyos controles son manejados con C#, con los que posteriormente se crean accesos a la base de datos y se modifica la misma.

- **SQL**

Del inglés SQL (Structured Query Language), es un lenguaje específico diseñado para administrar, recuperar e insertar información de SGBD (Sistemas de Gestión de Bases de Datos) mediante el uso de consultas basadas en el álgebra relacional. En SQL se definen dos grandes bloques: DDL (Data Definition Language) y DML (Data Manipulation Language).

En cuanto a DDL, en el proyecto se ha creado mediante la herramienta MySQL Workbench y el lenguaje SQL, la base de datos correspondiente a "Empresa", que contiene las tablas de Clientes, Pedidos, Líneas y Productos. Estas tablas son modificadas mediante DML a medida que se usa la aplicación web.

- **JSON**

Del inglés JSON (JavaScript Object Notation), es un formato ligero de intercambio de datos. Es sencillo de comprender su estructura tanto como para humanos como para máquinas. Se basa en dos estructuras: una colección de tuplas de nombre/valor y una lista ordenada de valores. En el proyecto se utiliza para comunicar los servicios REST con el código utilizado.

- **HTML**

Del inglés HTML (HyperText Markup Language), es un lenguaje usado para crear páginas web. Es un estándar de la W3C (World Wide Web Consortium) utilizado por todos los navegadores para la visualización de páginas web.

Se ha utilizado en el proyecto para posicionar los elementos ASP, así como de base para otros elementos visuales como la barra de menú, autogenerada en Bootstrap y que aparece en todas las páginas de la web.

- **CSS**

Del inglés CSS (Cascading Style Sheets), es un tipo de lenguaje empleado para crear presentaciones vistosas de un código en lenguaje marcado. Es ampliamente utilizado en la gran mayoría de webs.

Se ha utilizado en el código autogenerado junto a la barra de menú, así como con algún pequeño retoque del mismo y del posicionamiento de los objetos ASP.

6. Servicios Web Rest y Camunda

6.1. REST

REST (*REpresentational State Transfer*) es un estilo arquitectónico que se basa en mensajes de petición y respuesta transferidos entre clientes y servidores, donde ninguno de los nodos que participan en la comunicación mantienen el estado de las sesiones precedentes, es decir no se almacena ningún dato en el servidor durante el procesamiento de cada petición. El estado de la sesión se almacena en el lado del cliente.

Para la comunicación en sí se utiliza HTTP (*Hypertext Transfer Protocol*) que es el protocolo a nivel de aplicación que define las operaciones de transferencia entre clientes y servidores. En este protocolo, métodos tales como GET, POST, PUT y DELETE son operaciones sobre recursos. El protocolo elimina la necesidad de utilizar nombre de operaciones especificas a la aplicación que se está utilizando como "crearPedido", "obtenerEstatus", "actualizarDatos", etc.

Los recursos son los bloques básicos ([6.3]) que permiten la construcción de sistemas basados en la Web. Un recurso es cualquier entidad que queremos exponer en la Web, desde un documento o un video clip a un proceso de negocio o un dispositivo. Desde el punto de vista del cliente, un recurso es cualquier entidad con la que un consumidor puede interactuará mientras progresa a algún objetivo prefijado. Un recurso ([6.2]) puede verse también como un objeto con datos asociados, y relaciones con otros recursos y un conjunto de métodos que operan sobre él. Es similar a una instancia de un objeto en un lenguaje orientado a objetos, pero los únicos métodos que pueden aplicarse son los asociados al estándar HTTP (GET, POST, PUT, DELETE).

Los recursos se identifican mediante una URI (*Uniform Resource Indentifier*) que es un conjunto de caracteres que identifica inequívocamente un recurso lógico o físico. Hay dos tipos de URIs: URN *(Uniform Resource Name*) y URL (*Uniform Resource Locator*)

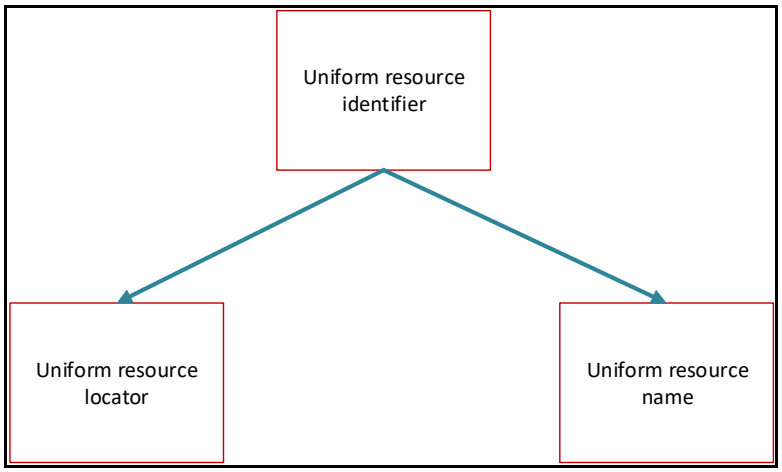


Figura 6.1. Especialización de URI en URL y URN.

Un URN es una URI que identifica a un recurso por un nombre dentro de un espacio de nombres particular. Por ejemplo, el ISBN de un libro identifica unívocamente a ese libro, pero no proporciona información referente a dónde puede ser encontrado el mismo.

Un URL es una URI que especifica la forma de actuación sobre la representación de un recurso, es decir especifica el método de acceso y la ubicación en la red. Por ejemplo, la URL http://ejemplo.org/wiki/Pagina_Principal se refiere a un recurso identificado mediante /wiki/Pagina_Principal cuya representación es en la forma de HTML se obtiene mediante el protocolo http de una red cuyo nombre de dominio es "ejemplo.org".

6.2. RESTful API

Una API RESTful (también conocida como servicio Web RESTful) es un servicio web implementado usando el protocolo HTTP y los principios de la arquitectura REST. Es una colección de recursos que emplean métodos HTTP (GET, PUT, POST, DELETE) para interactuar con el servidor.

La colección de recursos se representa de una forma estandarizada (habitualmente en XML) que puede ser cualquier estándar de hipertexto.

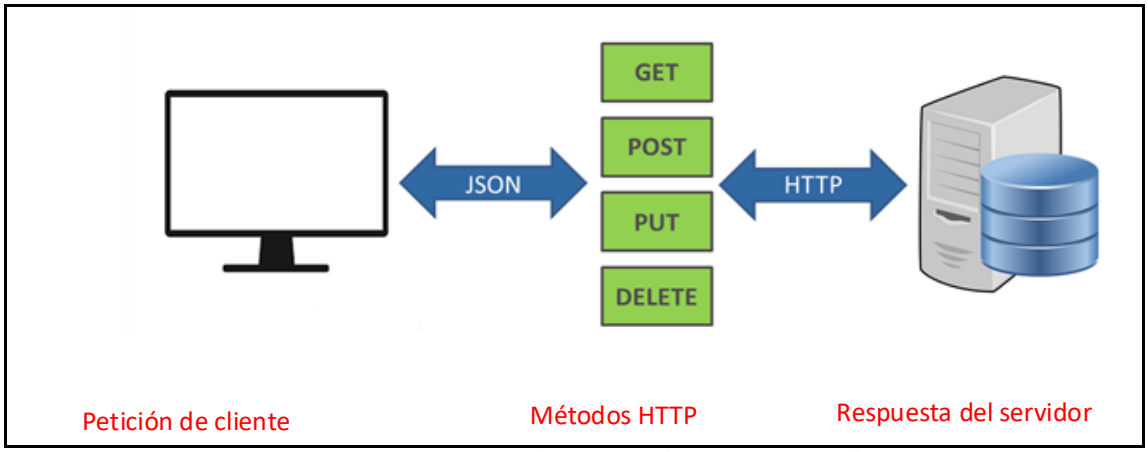


Figura 6.2. Comunicación entre clientes y servidores.

La información de intercambio entre clientes y servidores se retorna habitualmente en JSON dado que puede ser leído por cualquier lenguaje de programación

6.3. Camunda BPMN Workflow Engine

Como se ha comentado previamente, el motor BPMN de Camunda, nos permite automatizar diagramas de proceso compatibles hasta la versión 2.0. En este caso, se utilizará la versión Camunda BPM Tomcat 7.11, en su versión Community, que integra

los motores, además de Tasklist y Cockpit básico. La distribución incluye Camunda como una parte de Apache Tomcat.

Para arrancar el motor, simplemente hay que ejecutar start-camunda.bat, si se realiza desde una distribución Windows, o start-camunda.sh, si se realiza desde una distribución Linux o MacOS.

Una vez ejecutado, se ha de esperar unos instantes a que arranque completamente el motor. Cuando esto ocurra, se podrá acceder a la página del motor a través de la dirección "http://localhost:8080/camunda-welcome/index.html".

Esta es la primera aproximación e instalación previa a la configuración.

6.4. Camunda REST API

El motor de Camunda está escrito en Java y necesita una máquina virtual Java para poder ser ejecutado, de manera adicional Camunda proporciona también una API Rest que permite que código cliente escrito en cualquier lenguaje pueda comunicarse con el motor mediante el intercambio de mensajes en formato JSON

Una aplicación creada por un desarrollador puede interactuar con el motor de acuerdo al esquema de la siguiente figura:

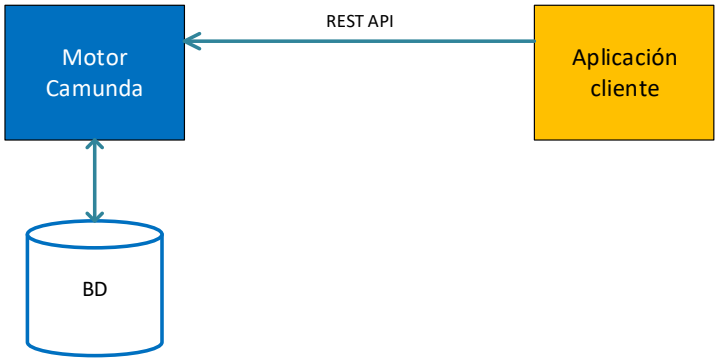


Figura 6.3. Aplicación interactuando con el API REST *de Camunda.*

La llamada a la API Rest es relativamente fácil desde cualquier lenguaje de programación. En JavaScript se utiliza simplemente JQuery y en C# debe incluirse System.Net.Http y Newtonsoft.Json. Las llamadas pueden ser excesivamente verbosas pudiéndose ocultar los detalles Rest mediante alguna librería cliente.

En la actualidad existen una serie de librerías disponibles:

- JavaScript: Esta librería es soportada por Camunda, accesible en https://github.com/camunda/camunda-external-task-client-js
- Java: También soportada por Camunda, accesible en https://github.com/camunda/camunda-external-task-client-java

- C#: Existen varias librerias para utilizar código cliente en C#: https://github.com/berndruecker/camunda-dot-net-showcase y https://github.com/salajlan/camundacsharpclient. Ambas iniciativas están en un estado inactivo y pueden considerarse como un buen punto de inicio para completarlas. Más avanzada se encuentra la librería Camunda.Api.Client https://github.com/jlucansky/Camunda.Api.Client utilizada para la versión 7.9 de Camunda.

Las dos primeras librerías forman parte del producto Camunda, en cada versión estas son actualizadas.

Para interactuar con el motor de procesos, como acabamos de mencionar, se proporciona una API Rest que invoca los servicios Web RESTful. Algunas de las operaciones de la API que hemos utilizado serán comentadas a continuación. La URL de los servicios que vamos a utilizar se encuentra en:

http://localhost:8080/engine-rest/

6.4.1 Obtener el Id de un proceso desplegado

Una vez desplegado el proceso en Apache Tomcat, mediante el nombre del mismo obtendremos el id del despliegue, para ello se utiliza la siguiente operación:

GET /deployment

Donde utilizaremos el nombre del proceso desplegado para obtener su Id. Esta Id permite posteriormente manipular la instancia del proceso.

GET /deployment?name=NombreDelDespliegue

La invocación devuelve la siguiente información en formato JSON:


```
[
  {
    "id": "someId",
    "name": "deploymentName",
    "source": "process application",
    "tenantId": null,
    "deploymentTime": "2019-04-23T13:42:43.000+0200"
  }
]
```


Figura 6.4. Información e id de un proceso.

6.4.2 Instanciar proceso

Una vez obtenido el id de despliegue de un proceso en el apartado anterior podemos instanciar el proceso. Para ello se utiliza el servicio

POST /process-definition/{id}/start

Donde id es el id del proceso que quiere ser instanciado. Si en la instanciación se necesitan variables éstas deben ser proporcionadas en el cuerpo de la petición HTTP.

La ejecución de la operación retorna en el formato JSON los datos de la nueva instancia creada.

6.4.3 Instanciar proceso con un mensaje de recepción

Al igual que cualquier otro mensaje que se capture en el proceso el evento inicial de mensaje se trata como una correlación. El servicio que se emplea es

POST /message

El cuerpo de la petición contiene un objeto JSON entre otras cosas con la información de la instancia que recibirá el mensaje, las variables de proceso necesarias. Para detalles adicionales consultar el manual de Camunda.

6.4.4 Determinar el camino en las puertas de datos

Para determinar el camino que tomará el proceso frente a una puerta basada en datos (exclusiva o inclusiva) se debe acceder a las variables de proceso ya que estas determinan el camino a seguir en tiempo de ejecución. Para ello se utiliza el servicio

GET /execution/{id}/localVariables/{NombreVar}

Donde id es el identificador del proceso que queremos consultar y NombreVar es el nombre de la variable.

7. Implementación

7.1. Estructura del proyecto (Cliente)

El proyecto se ha estructurado a medida que se ha ido codificando. De esta forma, en cuanto a cliente se refiere, podemos encontrar las páginas web en sí, que incluyen el fichero ASPX, que es dónde se insertan los objetos con los que interactúa el cliente, el fichero ASPX.CS, que es el fichero dónde se controla el comportamiento de estos objetos si es necesario, y finalmente un ASP.DESIGNER.CS, que no ha sido modificado en ningún caso. Las páginas con las que cuenta el proyecto son:

- Default.aspx

Es la página principal del proyecto. Al iniciar se accede mediante esta página y permite añadir líneas (tuplas de producto y número de elementos del producto seleccionado) a la cesta, ver la cesta si ya hay insertados elementos, finalizar la compra si ya hay insertados elementos, iniciar sesión si no está iniciada, cerrar sesión si hay una sesión iniciada y ver pedidos previos si hay una sesión iniciada con pedidos previos realizados.

Figura 7.1.1. Página Default.

- Cesta.aspx

Es la página dónde se puede ver la cesta de productos que se han añadido desde la página default. Desde aquí se pueden eliminar las líneas seleccionadas mediante un CheckBox, acceder a finalizar el pedido o volver a la página default.

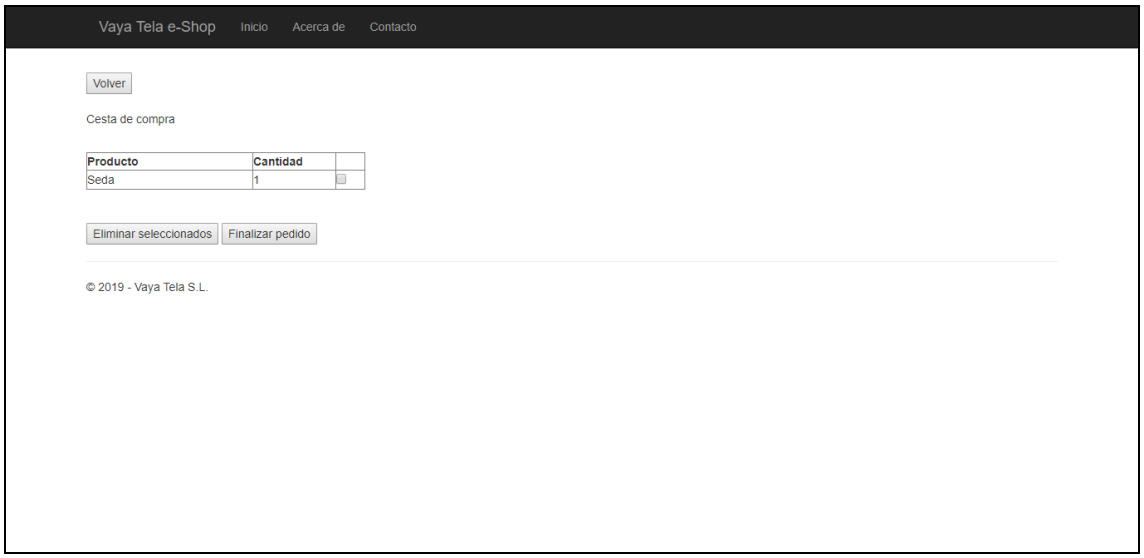


Figura 7.1.2. Página Cesta.

- Finalizar.aspx

Es la página dónde se finaliza el pedido. Aquí se puede ver el subtotal de coste de los productos seleccionados en la cesta y el coste total del pedido, introducir los datos de pago y confirmar el pedido o volver al inicio. Sólo se puede acceder a esta página si se ha iniciado una sesión y si la cesta contiene algún elemento.

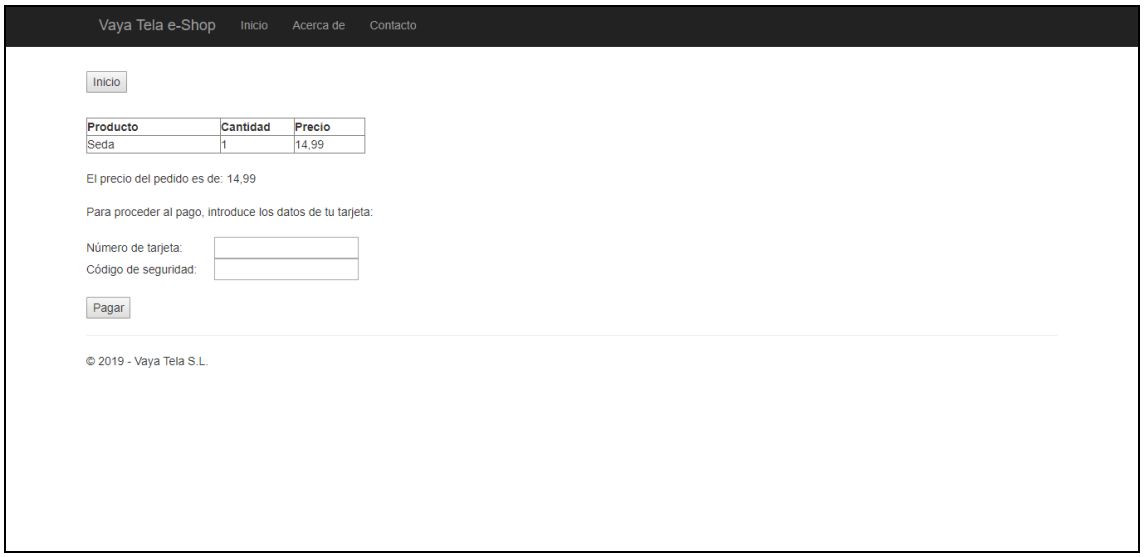


Figura 7.1.3. Página Finalizar.

- Sesion.aspx

Permite introducir los datos de usuario si se ha registrado previamente, acceder a la pestaña de registro o volver a la página principal. Una vez la sesión esté activa no se puede acceder a esta página y habrá que cerrar la sesión actual previamente.

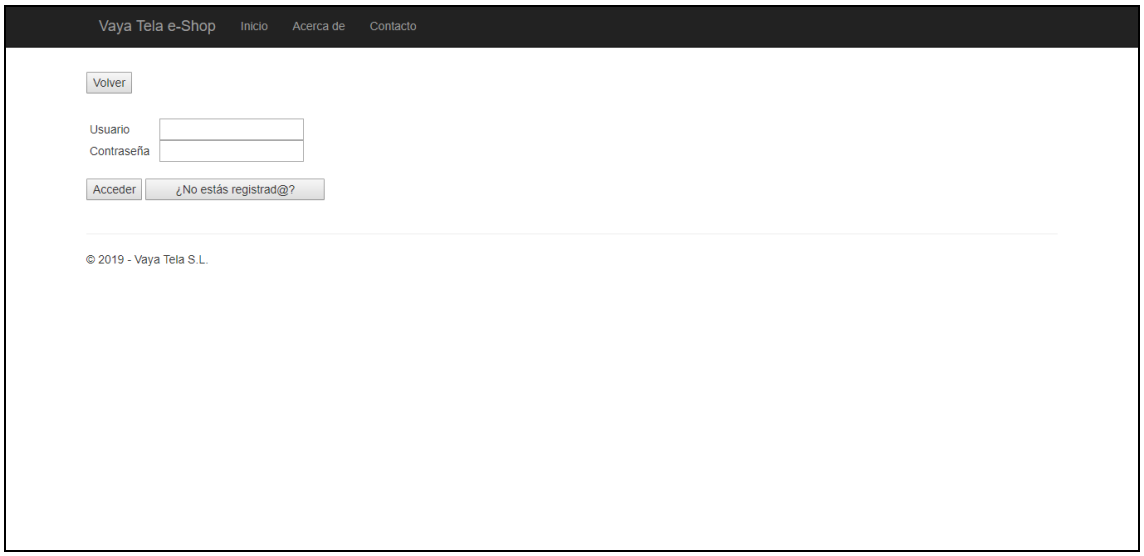


Figura 7.1.4. Página Sesion.

- Registrarse.aspx

Contiene todos los campos que ha de introducir un usuario para registrarse. Una vez todos se han rellenado, se puede clicar el botón y se guardará el usuario en la base de datos. También se puede volver a la pestaña de iniciar sesión.

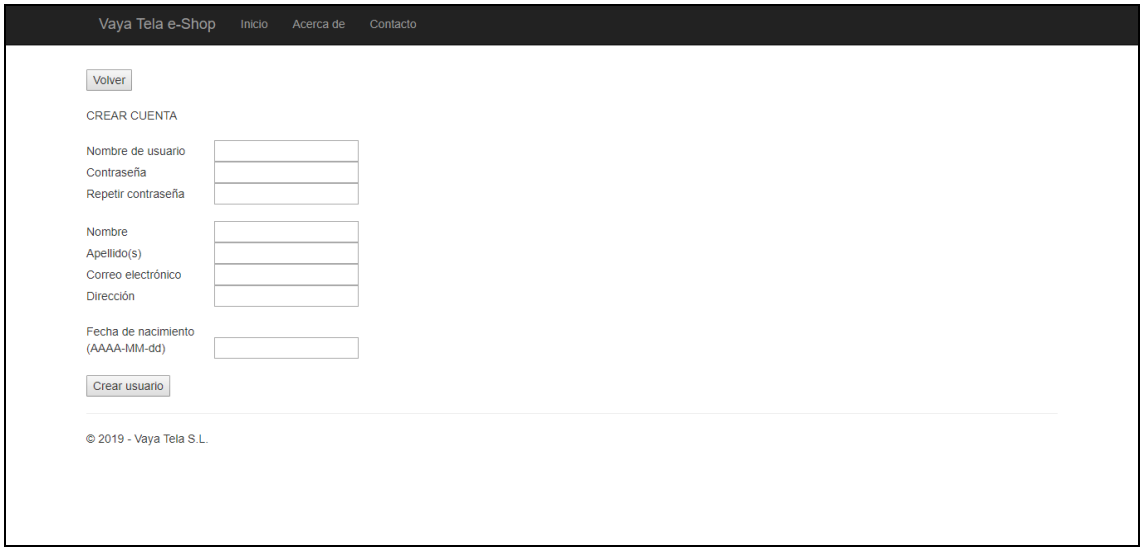


Figura 7.1.5. Página Registrarse.

- Pedidos.aspx

Contiene un listado de todos los pedidos realizados por un usuario en concreto. Además, permite cancelar un pedido dependiendo de una restricción de tiempo determinada, desde que se realizó el pedido.

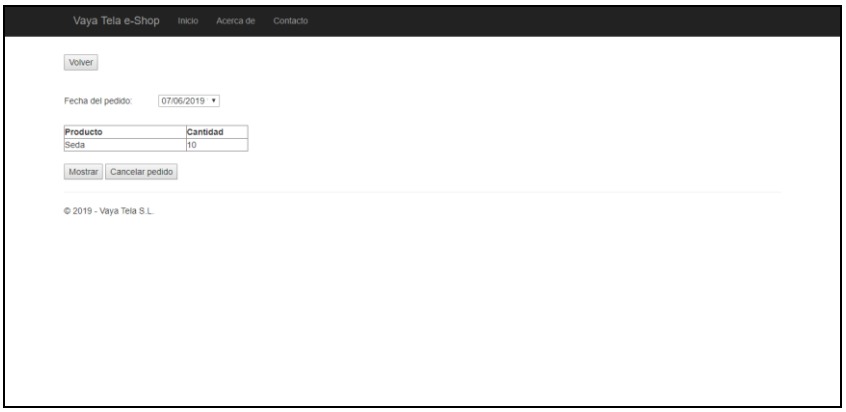


Figura 7.1.6. Página Pedidos.

- Contact.aspx

Contiene información sobre cómo contactar con la empresa, incluyendo un número de teléfono y un correo electrónico, así como la dirección de la empresa.

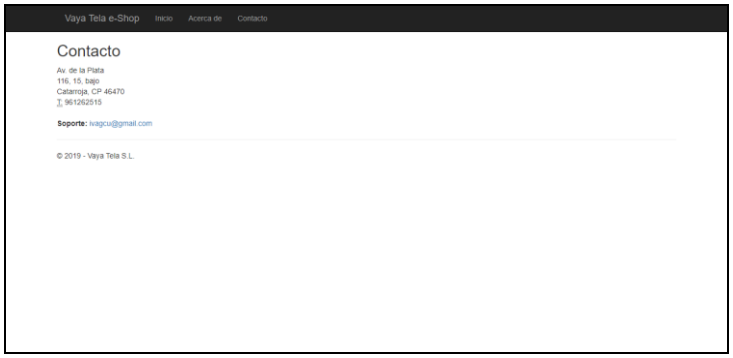


Figura 7.1.7. Página Contact.

- About.aspx

Contiene información sobre la empresa. Es una página meramente informativa.

Figura 7.1.8. Página About.

7.2. Estructura del proyecto (Clases intermediarias)

Este fragmento del código es el que se encarga de realizar la comunicación con la base de datos desde la aplicación. Esta comunicación se ha realizado mediante una clase simple C# que utiliza el conector MySQL llamado MySQL Connect/NET, que proporciona la posibilidad de realizar consultas a la base de datos.

Las clases intermediarias son:

- Querys.cs

Genera una serie de consultas a partir de unos parámetros de entrada y llama a la clase Control.cs para ejecutarlas.

- Control.cs

Es la clase encargada de abrir y cerrar conexiones con la base de datos, así como de ejecutar las consultas que se le pasen desde Querys.cs.

7.3. Estructura del proyecto (Base de datos del cliente)

Como se ha comentado previamente, la base de datos se ha realizado mediante MySQL. Para ello se ha realizado la creación de las siguientes tablas:

- Clientes

```
CREATE TABLE `clientes` (
  `IdCliente` int(11) NOT NULL AUTO_INCREMENT,
  `Nombre_Usuario` varchar(50) NOT NULL,
  `ContraseÃ±a` varchar(50) NOT NULL,
  `Nombre` varchar(50) NOT NULL,
  `Apellidos` varchar(70) NOT NULL,
  `Email` varchar(50) NOT NULL,
  `Direccion` varchar(100) NOT NULL,
  `Fecha_Nacimiento` date DEFAULT NULL,
  PRIMARY KEY (`IdCliente`),
  UNIQUE KEY `Nombre_Usuario` (`Nombre_Usuario`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=latin1;
```

Figura 7.2.1. Script de creación de la tabla "Clientes".

- Líneas

```
CREATE TABLE `lineas` (
  `IdLinea` int(11) NOT NULL AUTO_INCREMENT,
  `Nombre_Producto` varchar(50) NOT NULL,
  `Cantidad` int(11) NOT NULL,
  `IdPedido` int(11) NOT NULL,
  `IdProducto` int(11) DEFAULT NULL,
  PRIMARY KEY (`IdLinea`),
  KEY `IdProducto` (`IdProducto`),
  KEY `lineas_ibfk_1` (`IdPedido`),
  CONSTRAINT `lineas_ibfk_1` FOREIGN KEY (`IdPedido`)
REFERENCES `pedidos` (`IdPedido`) ON DELETE CASCADE,
  CONSTRAINT `lineas_ibfk_2` FOREIGN KEY (`IdProducto`)
REFERENCES `productos` (`IdProducto`)
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=latin1;
```

Figura 7.2.2. Script de creación de la tabla "Lineas".

- Pedidos

```
CREATE TABLE `pedidos` (
  `IdPedido` int(11) NOT NULL AUTO_INCREMENT,
  `Nombre_Usuario` varchar(50) NOT NULL,
  `Fecha_Pedido` datetime NOT NULL,
  PRIMARY KEY (`IdPedido`),
  KEY `Nombre_Usuario` (`Nombre_Usuario`),
  CONSTRAINT `pedidos_ibfk_1` FOREIGN KEY (`Nombre_Usuario`)
REFERENCES `clientes` (`Nombre_Usuario`)
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=latin1;
```

Figura 7.2.3. Script de creación de la tabla "Pedidos".

```
CREATE TABLE `productos` (
  `IdProducto` int(11) NOT NULL AUTO_INCREMENT,
  `Nombre` varchar(50) NOT NULL,
  `Precio_Unitario` double NOT NULL,
  `Stock` int(11) NOT NULL DEFAULT '999',
  PRIMARY KEY (`IdProducto`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=latin1;
```

Figura 7.2.4. Script de creación de la tabla "Productos".

7.4. Estructura del proyecto (Cliente ASP.NET de Camunda)

Crearemos una aplicación Web en ASP que se encargará por una parte de visualizar las páginas del sitio Web y por otra de realizar llamadas al API rest de Camunda para realizar los cambios de estado dentro del proceso. El proceso de compra se describió en el capítulo 3, figuras 3.2 hasta 3.6.

A continuación, reproducimos un fragmento del mismo:

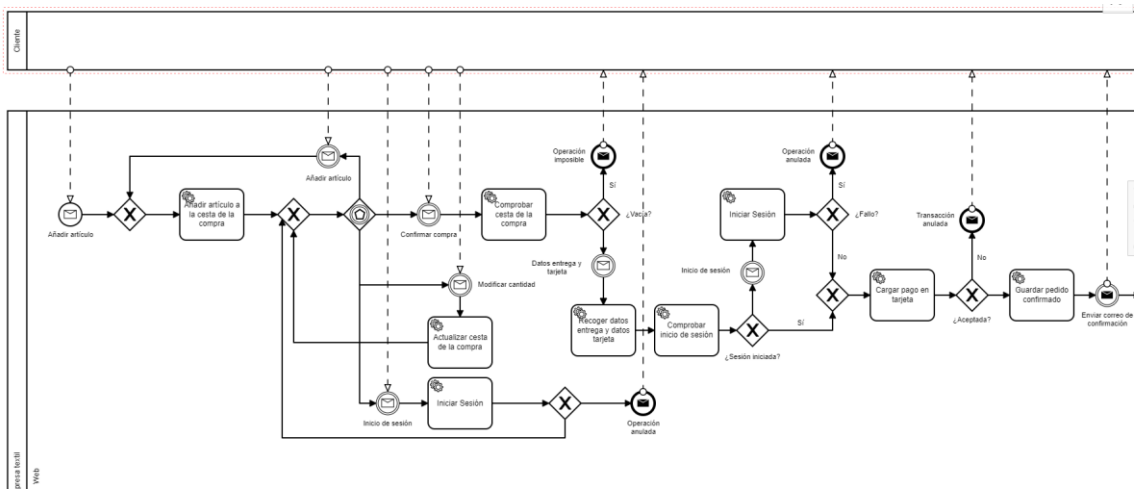


Figura 7.3.1. Fragmento del proceso de compra del sitio web.

Desde el código cliente se envía el mensaje inicial con los datos del artículo seleccionado, estos datos se capturan de la interfaz de usuario de la aplicación Web y se emplean para instanciar el proceso mediante un mensaje en C# que emplea el RESTApi del motor de procesos.

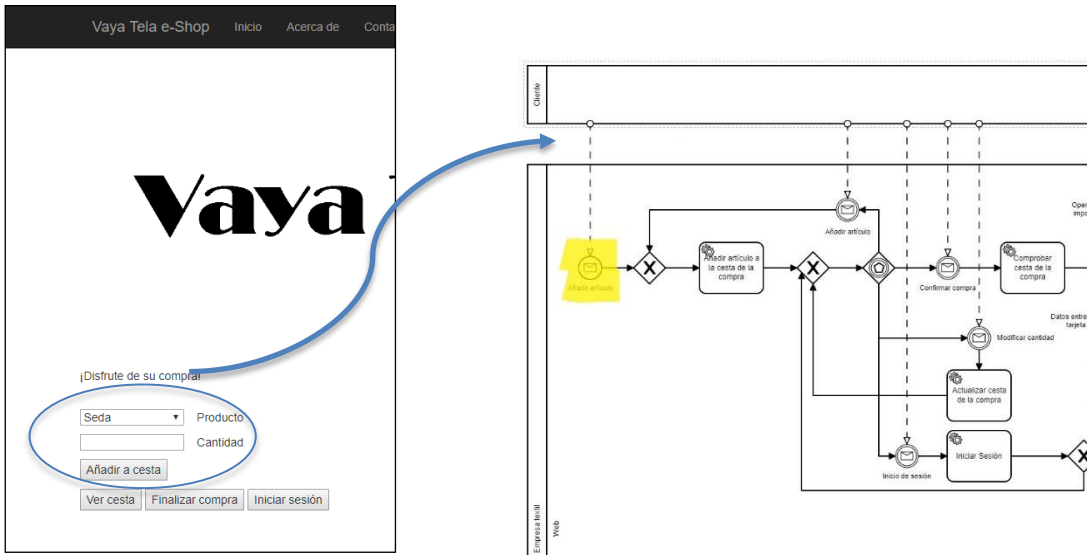


Figura 7.3.2. Instanciación del proceso al pulsar añadir a cesta.

El token, representado en amarillo en la figura anterior, progresa hasta la puerta basada en datos y ejecuta a tarea automática *Añadir artículo a cesta de la compra* (Figura 7.3.3). En ese punto se queda a la espera de la captura de un mensaje. Estos mensajes se generan desde la interfaz de usuario y están asociados a los diferentes botones de la misma, vea la figura 7.3.2

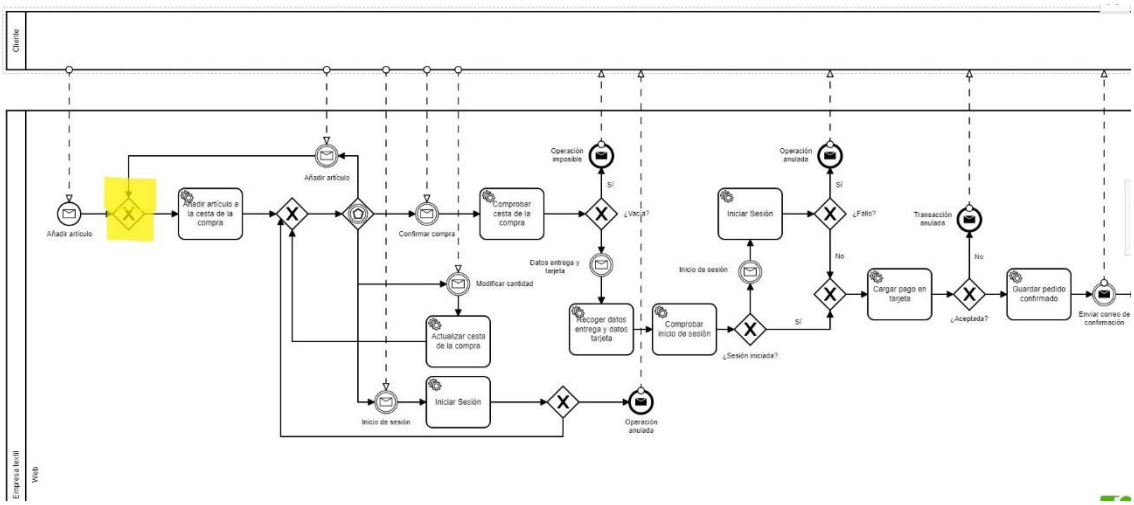


Figura 7.3.3. Fragmento del proceso de compra del sitio web, proceso activo, 2.

Un posible camino de ejecución se muestra en amarillo en la figura 7.3.4, después de introducir un artículo en la cesta de la compra el usuario puede utilizar la opción de editar o modificar cesta y poner a 0 la cantidad del artículo. Posteriormente puede utilizar *Confirmar compra*. Como la cesta está vacía se produce una salida del proceso mediante el envío de un mensaje al usuario. Obviamente el usuario puede en otra interacción formalizar completamente un pedido.

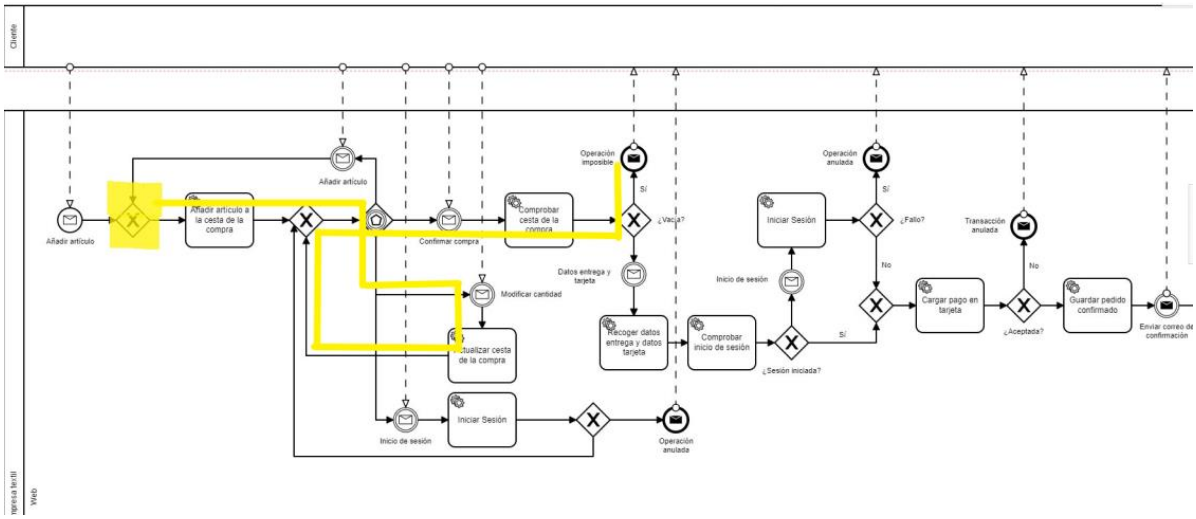


Figura 7.3.4. Fragmento del proceso de compra del sitio web, token de ejecución.

Hay que destacar finalmente que el código cliente que invoca la RestAPI del motor está escrito en C# mientras que la tareas automáticas del modelo invocan a clases Java.

8. Conclusiones

8.1. Revisión de objetivos

Con respecto a los objetivos, se puede decir que se han cumplido todos los puntos, ya que se ha ido especificando todo a lo largo de esta memoria y los objetivos han sido la piedra angular del proyecto.

Se han cumplido incluso los de "evaluar el funcionamiento del servicio de venta online" y "extraer conclusiones en cuanto al uso de implementar automatización y optimización de procesos" ya que en cuanto a la evaluación del servicio de venta online, se puede concretar que funciona, y que realmente quien ha de evaluar si es eficiente o no, es la propia empresa, y en cuanto a la extracción de conclusiones, se puede afirmar que esta implementación mejora enormemente la situación anterior de la empresa, donde los pedidos se recepcionaban a teléfono y se anotaban a mano.

8.2. Conclusiones generales

Se ha creado partiendo de cero una aplicación web compatible con sistemas que dispongan del Framework .NET que sirve para gestionar pedidos realizados en línea, con una base de datos de apoyo dónde se almacena toda esta información, y utilizando variables de sesión para mantener los datos en una determinada conexión concreta. Además, se ha enlazado esta aplicación que podría trabajar de forma aislada, con un modelo de flujo de procesos referente a la empresa tratada, y mediante el motor de Camunda, se ha implementado la solución apoyándose en una API Restful que comunica la implementación del código con el modelo de procesos.

En cuanto a la implementación y creación del proyecto, pese que se ha tenido que realizar investigación sobre los diferentes lenguajes usados, no ha habido mayor problema en la creación del mismo. El único punto que ha sido algo más dificultoso ha sido la creación de la API Restful. Por otro lado, aunque no ha habido gran dificultad, se ha echado en falta la ayuda de un diseñador gráfico o de tener más conocimientos de CSS para que la aplicación web resultase más vistosa.

Se ha de tener muy en cuenta la ayuda externa que he podido encontrar en páginas de intercambio de opinión y foros de ayuda de programadores como son StackOverflow y StackExchange. La consulta de estas páginas durante la codificación ha resuelto una gran cantidad de dudas a lo largo de la misma y ha ayudado a avanzar más rápidamente y a comprender algunos conceptos que resultaban algo difusos.

A grandes rasgos, la creación de esta aplicación web de venta online ha resultado ser una experiencia muy fructífera y gratificante, que ha servido para aprender a desarrollar un programa completamente desde cero y con cierto grado de complejidad.

Además, ha servido como una gran prueba personal para comprobar el uso que tienen todos los conocimientos, (o al menos gran parte de ellos), adquiridos durante el grado.

8.3. Previsiones de futuro

Tras finalizar esta versión inicial de aplicación web dirigida a la venta online se puede deducir que el funcionamiento de aplicaciones de estas características para generar un cambio tecnológico en las empresas tradicionales es un gran avance, y la implantación en ellas debería ser casi obligatoria.

En cuanto a la aplicación en sí, se ha de marcar una pauta de mejoras y actualizaciones que se irán definiendo con el paso del tiempo, pese a que algunas de ellas pueden ser:

1. Mejora estética.
    a. Menú más simple
    b. Visualizado de elementos seleccionados
    c. Icono de sesión iniciada
2. Incorporación de redes sociales de la empresa en la aplicación.
3. Añadir sección de últimas noticias de la empresa.
    a. Llegada de nuevos productos
    b. Cancelación de pedidos
    c. Agotamiento de existencias
4. Adaptar la aplicación correctamente al uso en dispositivos móviles y tabletas.

9. Relación con los estudios cursados

9.1. Conceptos teóricos

Tras realizar este proyecto, es obvio que hay una serie de conceptos del marco teórico que se han impartido y enseñado durante el grado que han servido durante la confección del mismo. Por este motivo, y teniendo en cuenta la relación que el proyecto tiene con muchas de las asignaturas impartidas durante estos cuatro cursos, se quiere citar algunas de las asignaturas que más han influenciado el trabajo realizado y el por qué:

- **Análisis de requisitos de negocio**

Es el punto de partida del proyecto. Sin conocer la forma y uso de los modelos de procesos, así como la existencia de otras técnicas de modelado para definir diferentes aspectos de una empresa, difícilmente se hubiera escogido realizar un proyecto como este.

- **Comportamiento organizativo y gestión del cambio**

Esta asignatura ha aportado muchos conceptos teóricos sobre cómo funcionan las organizaciones y cómo estas cambian durante el tiempo. Por ello, estos conceptos han servido de base teórica para el cambio que se ha deseado implementar y para tener en cuenta los factores que influyen en estos cambios.

- **Diseño y gestión de bases de datos**

Pese a que la base de datos creada no es del todo compleja, sin los conceptos impartidos en esta asignatura, hubiera resultado mucho más dificultosa la realización de la base de datos con la que la empresa trabaja.

- **Interfaces persona computador**

Gracias a cursar esta asignatura se han comprendido muchos conceptos relacionados con cómo debe ser una interfaz y, pese a que la interfaz actual no esté completamente pulida, cumple muchos de los aspectos que debe cumplir y resulta sencillo su uso.

- **Programación**

La programación es el punto base del grado. Sin la programación, este proyecto no hubiera sido posible, y en esta asignatura, pese a que no se enseñó a programar directamente, se enseñó el correcto uso de las diferentes estructuras, dependiendo del contexto. De esta forma, el código generado es eficiente y está debidamente comentado.

9.2. Competencias requeridas

Para la realización de este proyecto se han requerido unas competencias que, pese a no impartirse directamente, se han trabajado a lo largo del grado, de una forma u otra, dependiendo de cuáles fueran éstas y del plan de cada asignatura.

En este caso, las competencias más importantes para crear este proyecto han sido las siguientes:

- **Análisis y resolución de problemas**

Desde el inicio del proyecto se tiene en mente resolver el problema de disminuir la carga de trabajo que tienen los empleados que recepcionan pedidos y, además, realizar este cometido de una forma eficiente. Y para ello, primero se ha tenido que hacer un pequeño estudio de la empresa para conocer sus características y el público al que está enfocada.

- **Innovación, creatividad y emprendimiento**

El hecho de proponer una mejora así para cualquier empresa supone un plus para la misma. Para llegar a una solución así, se requiere un cierto grado de creatividad y de saber interpretar lo que se requiere, a lo que siempre es recomendable innovar en algún aspecto.

- **Aprendizaje permanente**

Para realizar este proyecto se ha requerido seguir formándose en todos los aspectos. Aprender nuevos lenguajes de programación, usar nuevas herramientas, realizar modelos que nunca se habían realizado, etc.

- **Planificación y gestión del tiempo**

Quizá el punto más importante, ya que es fundamental organizarse correctamente para poder cumplir unos plazos determinados. Durante el proyecto se ha seguido una pauta que ha servido para gestionar correctamente el avance de este.

10. Bibliografía

10.1. Información de la empresa

[1.1] Empresa y cultura emprendedora: unidad 1, la empresa y su entorno. Último acceso 07/05/2019. URL: https://www.edebe.com/educacion/documentos/830343-0-529-830343_LA_EIE_CAS.pdf

[1.2] Factores internos y externos. Lo que influye en la actividad empresarial. Último acceso 08/05/2019. URL: https://www.pymesyautonomos.com/estrategia/factores-internos-factores-externos-lo-que-influye-en-la-actividad-empresarial

10.2. Información de BPMN

[2.1] Sitio oficial de BPMN. Último acceso 10/05/2019. URL: http://www.bpmn.org/

[2.2] Business Process Model And Notation 2.0 (2011). Último acceso 10/05/2019. URL: https://www.omg.org/spec/BPMN/2.0

[2.3] Edutechwiki: BPMN. Último acceso 10/05/2019. URL: http://edutechwiki.unige.ch/en/BPMN

[2.4] Definición de BPMI. Último acceso 10/05/2019. URL: https://searchcio.techtarget.com/definition/Business-Process-Management-Initiative-BPMI

[2.5] Aclaraciones sobre BPMI y OMG. Sitio oficial. Último acceso 10/05/2019. URL: https://www.omg.org/BPMI-OMG-Merger-FAQs.pdf

[2.6] Bruce Silver, BPMN Method and Style, Segunda edición, Cody-Cassidy Press, 2011. Disponible en: https://www.amazon.es/Bpmn-Method-Style-Implementers-Guide/dp/0982368119

[2.7] Thomas Allweyer, BPMN 2.0: Introduction to the Standard for Business Process Modeling, Primera edición, Books on Demand, 2016. Disponible en: https://www.amazon.co.uk/BPMN-2-0-Introduction-Standard-Business-ebook/dp/B01F3G2D5A

10.3. Información de herramientas de modelado

[3.1] Sitio oficial StarUML. Último acceso 12/05/2019. URL: http://staruml.io/

[3.2] Sitio oficial Camunda. Último acceso 20/06/2019. URL: https://camunda.com/

[3.3] Arquitectura de Camunda. Último acceso 26/06/2019. URL: https://docs.camunda.org/manual/7.11/introduction/architecture/

[3.4] Entorno de desarrollo online de MockFlow. Último acceso 08/08/2019. URL: https://mockflow.com/app/#Wireframe

10.4. Información de herramientas de apoyo

[4.1] Sitio oficial MySQL. Último acceso 03/06/2019. URL: https://www.mysql.com

[4.2] “Azure DevOps”. Último acceso 05/06/2019. URL: https://azure.microsoft.com/es-es/overview/what-is-devops/

[4.3] Sitio oficial Apache Tomcat. Último acceso 05/06/2019. URL: http://tomcat.apache.org/

10.5. Codificación

[5.1] Sitio oficial C#. Último acceso 05/06/2019. URL: https://docs.microsoft.com/es-es/dotnet/csharp/

[5.2] Brice-Arnaud Guérin, ASP.NET con C# en Visual Studio 2017. Diseño y desarrollo de aplicaciones Web, Primera edición, ENI, 2018. Disponible en: https://www.amazon.es/ASP-NET-Visual-Studio-desarrollo-aplicaciones/dp/2409013899/

[5.3] Sitio oficial JSON. Último acceso 15/06/2019. URL: https://www.json.org/

[5.4] StackOverflow para resolución de dudas. Último acceso 15/06/2019. URL: https://stackoverflow.com/

[5.5] StackExchange para resolución de dudas. Último acceso 15/06/2019. URL: https://stackexchange.com/

[5.6] Eric Godoc, Anne-Christine Bisson, SQL. Los fundamentos del lenguaje (con ejercicios corregidos), Segunda edición, ENI, 2018. Disponible en: https://www.amazon.es/SQL-fundamentos-lenguaje-ejercicios-corregidos/dp/2409014933/

[5.7] Sitio oficial W3C: HTML. Último acceso 11/06/2019. URL: https://www.w3.org/html/

[5.8] HTML en Wikipedia. Último acceso 11/06/2019. URL: https://es.wikipedia.org/wiki/HTML

[5.9] Jon Duckett, HTML and CSS: Design and Build Websites, Primera edición, John Wiley & Sons Inc, 2011. Disponible en: https://www.amazon.es/HTML-CSS-Design-Build-Websites/dp/1118008189

[5.10] Github de Bernd Rueker, cofundador de Camunda. Último acceso 26/06/2019. URL: https://github.com/berndruecker/camunda-csharp-showcase

10.6. Servicios REST

[6.1] Información general sobre REST API. Último acceso: 01/08/2019. URL: https://phpenthusiast.com/blog/what-is-rest-api

[6.2]Leon Richardson & Sam Ruby, Restful web services, Primera edición, O’Reilly, 2007. Disponible en: https://www.amazon.es/gp/offer-listing/0596529260/ref=tmm_pap_new_olp_sr?ie=UTF8&condition=new&qid=&sr=

[6.3] Jim Webber, REST in Practice, Primera edición, O’Reilly, 2010. Disponible en: https://www.amazon.es/REST-Practice-Hypermedia-Systems-Architecture/dp/0596805829

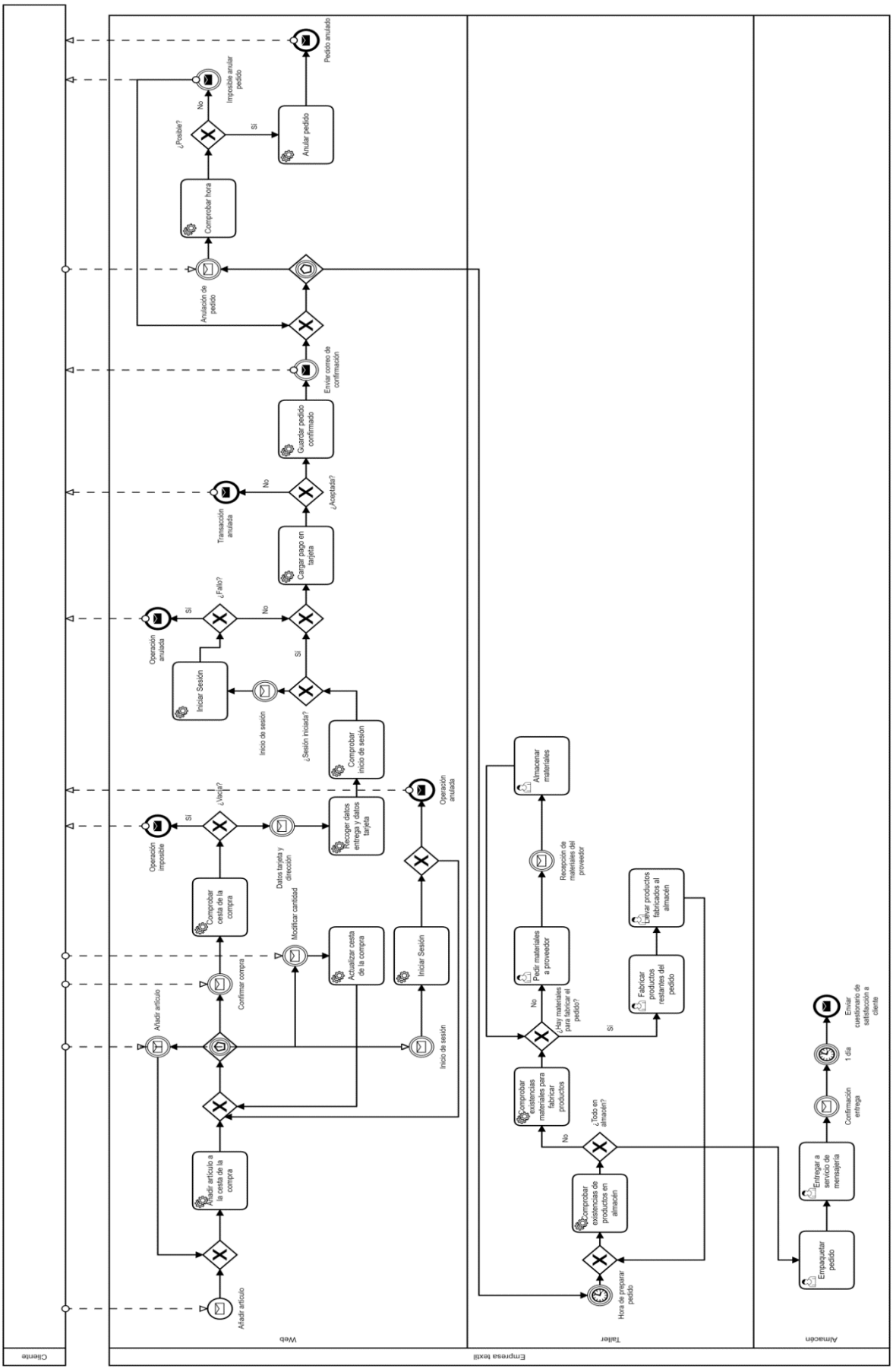
Anexos

Glosario de términos

	Término	Significado
1.	BPMN	Business Process Model and Notation
2.	B2B	Business 2 Business
3.	BPMI	Business Process Management Initiative
4.	E-commerce	Comercio electrónico o en web
5.	BPD	Business Process Diagram
6.	CMMN	Case Management Model and Notation
7.	IDE	Integrated Development Environment
8.	Bug	Cuando el programa no se comporta como debería o no se ajusta a las intenciones del programador
9.	OOP	Oriented Object Programming, en castellano Programación Orientada a Objetos
10.	Query	Interacción con una base de datos
11.	SQL	Structured Query Language
12.	SGBD	Sistemas de Gestión de Bases de Datos
13.	DDL	Data Definition Language
14.	DML	Data Manipulation Language
15.	JSON	JavaScript Object Notation
16.	HTML	HyperText Markup Language

17.	W3C	World Wide Web Consortium
18.	CSS	Cascading Style Sheets
19.	ASP	Active Server Pages
20.	API	Application Programming Interface
21.	REST	REpresentational State Transfer
22.	URI	Uniform Resource Identifier
23.	URN	Uniform Resource Name
24.	URL	Uniform Resource Locator
25.	OMG	Object Management Group

Modelo BPMN del caso de estudio



Manual de usuario

1. Requisitos mínimos del sistema

- Procesador: Intel Pentium E2180 2.0GHz
- RAM: 2GB
- Sistema operativo: Windows 10
- Tarjeta de vídeo: 64MB
- Espacio libre en disco: 500MB

2. Software necesario

- Navegador web (Google Chrome, Mozilla, Internet Explorer…)
- MySQL 5.7

3. Configuración previa

3.1. Base de datos de la aplicación

Una vez se tenga instalada una versión de MySQL, será necesario configurar la base de datos para el correcto funcionamiento de la aplicación web. Para ello, es necesario seguir los pasos siguientes:

a. Abrir consola
   i. mysql –u root –p
   ii. password > admin
   iii. CREATE DATABASE empresa;
   iv. Salir de la terminal
b. Acceder a MySQL Workbench
   i. Crear una nueva conexión:
      1. Hostname: localhost
      2. Username: root
c. Descargar el fichero DumpLimpio.sql desde: https://github.com/ivagcu/Vaya-Tela-Download
d. Abrir consola
   i. mysql –u root –p empresa < DumpLimpio.sql
   ii. Cerrar consola

3.2. Camunda BPM Tomcat y base de datos del motor

Será necesario descargar y configurar el motor de procesos de Camunda:

a. Acceder a https://camunda.com/download/
b. Descargar Camunda Community Platform en ZIP
c. Descomprimir el zip
d. Abrir terminal
   i. mysql –u root –p

ii. CREATE DATABASE IF NOT EXISTS camundabpm;
iii. GRANT ALL ON camundabpm.* TO camundabpm@'localhost' IDENTIFIED BY 'root';
iv. GRANT ALL ON camundabpm.* TO camundabpm@'localhost.localdomain' IDENTIFIED BY 'root';
v. GRANT ALL ON camundabpm.* TO camundabpm@'127.0.0.1' IDENTIFIED BY 'root';
vi. Cerrar consola

e. Abrir consola
   i. Acceder a DIRECTORIO_CAMUNDA/sql/créate
   ii. mysql camundabpm -u camundabpm -proot < mysql_engine_7.11.0.sql;
   iii. mysql camundabpm -u camundabpm -proot < mysql_identity_7.11.0.sql
f. Abrir archivo DIRECTORIO_CAMUNDA/SERVER/apache-tomcat-version/conf/server.xml
   i. Cambiar desde “<Resource” hasta “maxIdle=”20” />” por:

```
<Resource name="jdbc/ProcessEngine"
auth="Container"
type="javax.sql.DataSource"
factory="org.apache.tomcat.jdbc.pool.DataSourceFactory"
uniqueResourceName="process-engine"
driverClassName="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost:3306/camundabpm?useSSL=false"
username="camundabpm"
password="root"
maxPoolSize="20"
minPoolSize="5" />
```

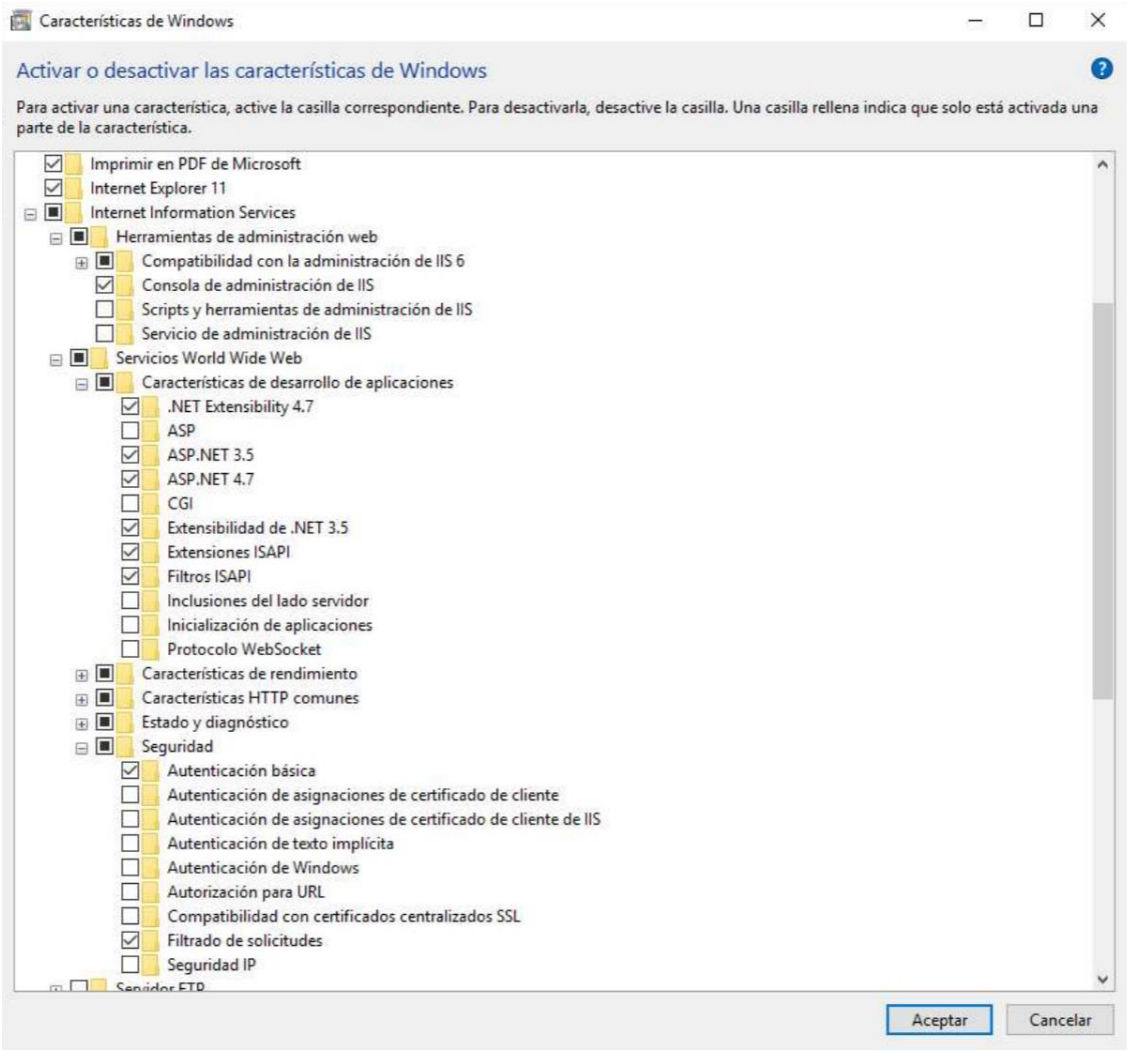
g. Copiar el driver mysql-connector-java-VERSION-bin.jar en: DIRECTORIO_CAMUNDA\server\apache-tomcat-VERSION\lib

3.3. Internet Information Services

Para poder desplegar la aplicación web desde el equipo es necesario realizar la configuración del IIS. Para ello es recomendable seguir los pasos siguientes:

a. Activar Internet Information Services
   i. Panel de Control -> Programas y características -> Activar o desactivar características de Windows

b. Activar las características tal como se muestra:



c. Clicar aceptar y esperar que finalice la instalación
d. Arrancar Administrador de Internet Information Services (IIS)
e. Acceder a la página principal de nuestro equipo
f. Clicar en IIS -> Autenticación
    i. Habilitar Autenticación anónima y Suplantación de ASP.NET

4. Instalación de la aplicación web

a. Descargar el zip de la aplicación web desde: https://github.com/ivagcu/Vaya-Tela-Download
b. Descomprimir zip
c. Copiar la carpeta “AplicacionWeb” a “C:\inetpub\wwwroot”
d. Acceder a IIS
    i. Convertir la carpeta de la web en aplicación:
        1. Clic derecho -> Convertir en aplicación
    ii. Cambiar al grupo de aplicaciones en modo clásico

1. Clic derecho a la carpeta desde IIS -> Administrar aplicación -> Configuración avanazada -> Grupo de aplicaciones -> Seleccionar “Classic .NET 4.5”

5. Acceso a la aplicación web

Primer método

a. Acceder a Internet Information Services (IIS)
b. Clicar en “AplicacionWeb”
c. Clicar en Administrar aplicación -> Examinar aplicación -> Examinar *:80 (http)

Segundo método

a. Abrir un navegador
b. Acceder a “localhost/AplicacionWeb/”