UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

TRABAJO FIN DE GRADO

Diseño y desarrollo de una aplicación de gestión empresarial para una PYME

Carlos Medina Peñalver

Enero 2016

Diseño y desarrollo de una aplicación de gestión empresarial para una PYME

AUTOR: Carlos Medina Peñalver

TUTOR: Jack Mario Mingo Postiglioni

PONENTE: Juan De Lara Jaramillo

Dpto. Ingeniería Informática

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Fecha: Enero 2016

Resumen

En este trabajo se realiza el diseño y desarrollo de una aplicación de gestión con el objetivo de automatizar las tareas rutinarias de una PYME. Este sistema debe garantizar la gestión de manera eficiente de todos y cada uno de los procesos que actualmente se llevan a cabo en la empresa, así como facilitar la interacción con el usuario.

Este trabajo se ha desarrollado atendiendo a las necesidades y características de un cliente en concreto, la empresa Hidrosierra Sport S.L.

Hidrosierra Sport S.L. es una empresa que ofrece servicios en piscinas, desde el diseño de la piscina al asesoramiento en el tratamiento del agua. Servicios de obras, calderas, reparación de las mismas o servicios de mantenimiento en comunidades son otras actividades de la empresa. Así mismo dispone de un servicio integral de mantenimiento de piscinas de comunidad o ayuntamientos y del personal técnico cualificado, técnicos, socorristas, etc.

Para llevar a cabo el desarrollo de la aplicación se ha seguido una metodología ágil, concretamente SCRUM. Sin embargo, teniendo en cuenta que el trabajo lo realiza un solo desarrollador es necesario hacer algunas adaptaciones de esta metodología para equipos unipersonales, siendo también un objetivo del trabajo comprobar si este tipo de nuevas metodologías pueden suponer ventajas en aplicaciones de pequeño y medio tamaño frente a las metodologías tradicionales. El desarrollo ágil presenta una mejor disposición para adaptarse a los cambios en el entorno, cuestión muy importante en una aplicación de estas características, donde el cliente en muchas ocasiones no puede o no es capaz de especificar con el suficiente detalle sus procesos de negocio de forma efectiva. Además, en este tipo de metodologías el cliente tiene un papel muy importante durante todo el desarrollo, siendo a veces parte activa del mismo.

Las funcionalidades que se han implementado en la aplicación son las encargadas de gestionar los clientes propios de la empresa, proveedores, facturación, presupuestos, albaranes, trabajadores, partes de trabajo, tarifas, socorristas y comunidades, quedando abierta la posibilidad de añadir en el futuro las funciones de inventario y contabilidad. De esta manera pasan a estar automatizadas más del 90% de las funciones operativas de la empresa.

Palabras Clave

Software para PYME, Metodología Ágil, SCRUM, Herramientas de Gestión.

Abstract

This paper describes the design and development processes of a management application in order to automate routine tasks of an SME (Small and Medium-size Enterprise). This system should ensure the efficient management of each and every process that are currently performed in the company and facilitate interaction with the user.

This work has been developed according to the needs and characteristics of a particular customer, the company Hidrosierra Sport S.L.

Hidrosierra Sport S.L. is a company that offers services in pools, from design to consultancy pool water treatment. Services works, boilers and maintenance services in communities are activities of the company. Also it offers full service community pool maintenance or municipalities and qualified technical personnel, technicians, lifeguards, etc.

To perform, the application development has followed an agile methodology, SCRUM specifically. However, the work is done by a single developer, therefore is necessary to make some adjustments to the methodology for single-person teams, it is also an objective of the work, check if new methodologies can bring advantages in small and medium applications size front traditional methodologies. Agile development is more easily adapted to changes in the environment, very important in an application of these characteristics, where customers often can not or are not able to specify in sufficient detail their business processes in an effective way. Moreover, in this type of methodology the customer has a very important role throughout the development process, sometimes being active part of it.

The features that have been implemented in the application are responsible for managing customers, suppliers, billing, budgets, invoices, workers, job reports, fees, lifeguards and communities, leaving open the possibility of adding in the future inventory functions and accounting. Thus the company will have the 90% of its operational functions automated.

Key words

Software for PYMES, Agile methodologies, SCRUM, Management Application.

Glosario

API (*Application**P**rogramming **I**nterface*):** Conjunto de funciones y procedimientos (métodos en la terminología de la programación orientada objetos) que ofrece cierta biblioteca para ser utilizados por otro software como una capa de abstracción.

AWT (Abstract **W**indows**T**oolkit**):** Conjunto de herramientas de gráficos, interfaz de usuario y sistema de ventanas independiente de la plataforma original de Java.

ERP (*Enterprise **R**esource**P**lanning*): P**l**anificación de **R**ecursos de la **E**mpresa. Se trata de un software que facilita la integración entre las distintas áreas de una empresa.

GUI (***G**raphical**U**ser**I**nterface*)**:** Programa informático que actúa de interfaz de usuario, utilizando un conjunto de imágenes y objetos gráficos para representar información y acciones disponibles en la interface.

JAVA SWING: Paquete de librerías que proporciona herramientas o facilidades al usuario para construir interfaces gráficas.

MySQL: Sistema de gestión de base de datos de libre distribución.

NAS (*Network **A**ccess **S**erver*):** Dispositivo de almacenamiento al que se accede desde los equipos a través de la red.

NETBEANS: Entorno de programación.

PYME ***(P**equeña **Y M**ediana **E**mpresa)*:** Empresa mercantil, industrial, etc., compuesta por un número reducido de trabajadores, y con un moderado volumen de facturación.

SCRUM: Marco de trabajo para la gestión y desarrollo de software basada en un proceso iterativo e incremental, utilizado comúnmente en entornos basados en el desarrollo ágil de software.

SPRINT: Período en el cual se lleva a cabo el trabajo en sí, dentro de la metodología SCRUM. Suele tener una duración entre 2 y 6 semanas.

WAMP (*Windows **A**pache **M**ysql**P**HP*):** Acrónimo que describe un sistema de infraestructura de Internet.

Índice de Contenidos

1 Introducción 1

1.1 Marco del proyecto 1

1.2 Motivación 1

1.3 Objetivo 2

1.4 Estructura del documento 3

2 Estado del arte 5

2.1 Introducción 5

2.2 Herramientas estándar para PYMES 5

2.2.1 Enterprise Resource Planning (ERP) 5

2.3 Herramientas a medida para PYMES 6

2.4 Ventajas y desventajas del uso de herramientas a medida vs estándar. 7

2.4.1 Características del software a medida 7

2.4.2 Características del software comercial o estándar 7

2.5 Opiniones y conclusiones 7

3 Definición del Proyecto 9

3.1 Metodología 9

3.1.1 Introducción a la metodología 9

3.1.2 SCRUM 9

3.2 Herramientas y tecnologías utilizadas en el desarrollo 12

- Netbeans IDE 8.1 12
- WAMP 13

3.3 Definición del proyecto 13

3.3.1 Funcionalidades del sistema 14

3.3.2 Alcance del sistema 14

4 Desarrollo del Sistema con SCRUM 15

4.1 Análisis de requisitos 15

4.1.1 Diagrama de casos de uso 15

4.1.2 Catálogo de requisitos 17

4.2 Product Backlog 23

4.3 Desarrollo de los Sprints 24

4.3.1 Sprint 1. Clientes, proveedores y trabajadores. 24

4.3.2 Sprint 2. Socorristas y comunidades 26

4.3.3 Sprint 3. Presupuestos y facturas 27

4.3.4 Sprint 4. Tarifas y tipo de tarifas. 29

4.3.5 Sprint 5. Albaranes y partes de trabajo. 30

4.3.6 Sprint 6. Administrador 31

4.4 Valoración del desarrollo de los sprints 32

4.5 Diagrama de clases de la aplicación 32

4.6 Distribución de carpetas 35

4.7 Diagrama de secuencia 36

4.8 Diseño del modelo de la aplicación 36

4.8.1 Modelo 36

4.8.2 Vista 37

4.8.3 Controlador 37

4.8.4 Ejemplo del flujo de MVC en la aplicación. Insertando un nuevo cliente… 37

4.9 Arquitectura de la aplicación 37

4.9.1 Cliente 38

4.9.2 Servidor 38

4.9.3 Funcionamiento cliente/servidor de la aplicación 38

5 Detalles de la Implementación 39

5.1 Base de datos 39

5.2 Interfaz de usuario 40

5.3 Implementación de los modelos más destacados 45

5.3.1 Diseño de reportes con JasperReports 45

6 Validación y Verificación 47

6.1 Introducción 47

6.2 Estrategias de validación y verificación utilizadas 47

6.2.1 Desarrollo de las pruebas 47

7 Evaluación 53

7.1 Evaluación de los usuarios 53

8 Conclusiones y Líneas Futuras 55

8.1 Conclusiones 55

8.2 Trabajo futuro 56

9 Referencias 57

Anexo A: Esquema Relacional ..... A

Anexo B: Tablas de la Base de Datos ..... B

Anexo C: Ejemplo de creación de un Query Reporte ..... J

Anexo D: Ejemplo de reporte de un presupuesto ..... K

Anexo E: Desarrollo de pruebas atendiendo a los requisitos L

Anexo F: Diagramas de Secuencia. Inserción de un cliente ..... N

Anexo G: Distribución de las carpetas ..... O

ÍNDICE DE FIGURAS

FIGURA 1: CICLO DE VIDA DE LA METODOLOGÍA ÁGIL SCRUM 10

FIGURA 2: DIAGRAMA DE CASOS DE USO DE LA APLICACIÓN 16

FIGURA 3: DIAGRAMA DE CASOS DE USO PARA CLIENTES, PROVEEDORES Y TRABAJADORES. 25

FIGURA 4: DIAGRAMA DE CASOS DE USO PARA SOCORRISTAS Y COMUNIDADES 26

FIGURA 5: DIAGRAMA DE CASOS DE USO PARA PRESUPUESTOS Y FACTURAS 28

FIGURA 6: DIAGRAMA DE CASOS DE USO PARA TARIFAS Y TIPOS DE TARIFAS. 29

FIGURA 7: DIAGRAMA DE CASOS DE USO PARA ALBARANES Y PARTES DE TRABAJO 31

FIGURA 8: DIAGRAMA DE CASOS DE USO PARA ADMINISTRADOR 32

FIGURA 9: DIAGRAMA DE CLASES. PAQUETE VISTA GENERAL. 33

FIGURA 10: DIAGRAMA DE CLASES. EJEMPLO CLASE CLIENTE. 34

FIGURA 11: DIAGRAMA DE CLASES. MODELO 35

FIGURA 12: DIAGRAMA DE CLASES. CONTROLADOR 35

FIGURA 13: ESTRUCTURA MODELO-VISTA-CONTROLADOR 36

FIGURA 14: ESQUEMA DE LA ARQUITECTURA CLIENTE-SERVIDOR 38

FIGURA 15: PÁGINA PRINCIPAL DE LA APLICACIÓN 41

FIGURA 16: EJEMPLO DE JINTERNALFRAME DENTRO DE UN JDESKTOPPANEL. 42

FIGURA 17: LISTADO DE CLIENTES 42

FIGURA 18: INSERCIÓN DE UN NUEVO CLIENTE. 43

FIGURA 19: BÚSQUEDA DE CLIENTES. 43

FIGURA 20: VISUALIZAR DATOS DE UN CLIENTE 44

FIGURA 21: MODIFICAR DATOS DE UN CLIENTE. 44

FIGURA 22: EJEMPLO DE CREACIÓN REPORTE. 45

FIGURA 23: CONTROL DE FORMULARIO. AÑADIR NUEVO CLIENTE. 49

FIGURA 24: CONTROL DE FORMULARIO. BÚSQUEDA DE CLIENTES 50

FIGURA 25: CONTROL DE FORMULARIO. ERROR AL INTRODUCIR DATOS EN EL PRESUPUESTO. 51

FIGURA 26: CONTROL DE FORMULARIO. ERROR AL CREAR PRESUPUESTO. 51

FIGURA 27: ESQUEMA RELACIONAL DE LA BASE DE DATOS ..... A

FIGURA 28: EJEMPLO DE CREACIÓN DE UN QUERY REPORTE. .....J

Índice de Tablas

TABLA 1: EJEMPLO DE FORMATO PARA LAS HISTORIAS DE USUARIO 10

TABLA 2: HISTORIA DE USUARIO PARA GESTIÓN DE USUARIOS. 18

TABLA 3: HISTORIA DE USUARIO PARA GESTIÓN DE CLIENTES. 19

TABLA 4: HISTORIA DE USUARIO PARA GESTIÓN DE PROVEEDORES. 19

TABLA 5: HISTORIA DE USUARIO PARA GESTIÓN DE TRABAJADORES. 19

TABLA 6: HISTORIA DE USUARIO PARA GESTIÓN DE SOCORRISTAS. 20

TABLA 7: HISTORIA DE USUARIO PARA GESTIÓN DE COMUNIDADES. 20

TABLA 8: HISTORIA DE USUARIO PARA GESTIÓN DE PRESUPUESTOS. 20

TABLA 9: HISTORIA DE USUARIO PARA GESTIÓN DE FACTURAS. 21

TABLA 10: HISTORIA DE USUARIO PARA GESTIÓN DE TARIFAS. 21

TABLA 11: HISTORIA DE USUARIO PARA GESTIÓN DE TIPOS DE TARIFAS. 21

TABLA 12: HISTORIA DE USUARIO PARA GESTIÓN DE ALBARANES. 21

TABLA 13: HISTORIA DE USUARIO PARA GESTIÓN DE PARTES DE TRABAJO. 22

TABLA 14: PRODUCT BACKLOG 23

TABLA 15: SPRINT BACKLOG INICIAL DEL SPRINT 1 24

TABLA 16: SPRINT BACKLOG FINAL DEL SPRINT 1 25

TABLA 17: SPRINT BACKLOG INICIAL DEL SPRINT 2 26

TABLA 18: SPRINT BACKLOG FINAL DEL SPRINT 2 27

TABLA 19: SPRINT BACKLOG INICIAL DEL SPRINT 3 27

TABLA 20: SPRINT BACKLOG FINAL DEL SPRINT 3 28

TABLA 21: SPRINT BACKLOG INICIAL DEL SPRINT 4..... 29

TABLA 22: SPRINT BACKLOG FINAL DEL SPRINT 4 30

TABLA 23: SPRINT BACKLOG INICIAL DEL SPRINT 5..... 30

TABLA 24: SPRINT BACKLOG FINAL DEL SPRINT 5 31

TABLA 25: SPRINT BACKLOG INICIAL DEL SPRINT 6..... 31

TABLA 26: SPRINT BACKLOG FINAL DEL SPRINT 6 32

TABLA 27: TABLA DE CLIENTES .....B

TABLA 28: TABLA DE TARIFAS .....B

TABLA 29: TABLA DE LOS TIPOS DE TARIFAS C

TABLA 30: TABLA DE PROVEEDORES C

TABLA 31: TABLA DE SOCORRISTAS ..... D

TABLA 32: TABLA QUE RELACIONA SOCORRISTAS Y COMUNIDADES ..... D

TABLA 33: TABLA DE COMUNIDADES .....E

TABLA 34: TABLA DE PRESUPUESTOS .....E

TABLA 35: TABLA DE PRESUPUESTO ENTRADAS..... F

TABLA 36: TABLA DE TRABAJADORES ..... F

TABLA 37: TABLA DE PARTES DE TRABAJO..... G

TABLA 38: TABLA DE INFORMACIÓN DE UN OPERARIO PARA UN PARTE ..... G

TABLA 39: TABLA DE FACTURAS ..... H

TABLA 40: TABLA QUE RELACIONA FACTURA Y ALBARÁN ..... H

TABLA 41: TABLA DE ALBARANES.....I

TABLA 42: TABLA ALBARÁN ENTRADAS I

TABLA 43: TABLA DE USUARIOS. I

1 Introducción

1.1 *Marco del proyecto*

Este Trabajo de Fin de Grado (TFG) tiene como finalidad la creación de un sistema de gestión empresarial para satisfacer las necesidades de un cliente, en este caso la empresa Hidrosierra Sport S.L. Este sistema de gestión ofrece funciones que permitirán a los usuarios gestionar la facturación, clientes, proveedores, socorristas, comunidades, partes de trabajo realizados y los datos de todos los trabajadores de la empresa, con el fin de unificar y ordenar toda la información en un solo lugar, de este modo cualquier suceso queda a la vista de forma inmediata, posibilitando la toma de decisiones de una manera rápida y segura.

El sistema automatizará los procesos que a día de hoy se controlan de forma manual, asegurando una mejora en la calidad del trabajo y en el desarrollo de los procesos, a la vez que se consiguen reducir los tiempos a la hora de tratar la información.

1.2 *Motivación*

Un proyecto de implantación de un sistema de gestión en una pequeña y mediana empresa (PYME) normalmente suele ser un proceso largo y complejo, ya que implica el desarrollo de un nuevo sistema de información, al tiempo que supone rediseñar los esquemas relativos a la forma de trabajo utilizada en dicha empresa.

En la mayoría de los casos es el proyecto de tecnología más grande y transcendente dentro de la PYME. Es por eso que resulta de un gran atractivo para cualquier desarrollador de software poder enfrentarse a un reto de estas características, poniendo a prueba todos sus conocimientos y habilidades con el único fin de satisfacer las necesidades del cliente. Además, en este caso el cliente no trabaja actualmente con ningún tipo de aplicación y los procesos se realizan completamente a mano, mediante fichas y cuadernos o usando hojas de cálculo.

Hoy en día existen una gran cantidad de herramientas de gestión en el mercado, ya sean genéricas o a medida.

- **Software a medida:** Se trata de un sistema creado especialmente para la empresa prácticamente desde cero. El desarrollo y la implantación de un software de estas características suele ser más largo que una solución estándar, y además es mucho más costoso. Sin embargo, tiene la gran ventaja de que la aplicación resultante estará adaptada perfectamente a las necesidades de la empresa.

- **Software genérico (estándar o modular):** Muchas empresas eligen esta solución simplemente porque no pueden permitirse una solución a medida. Estas aplicaciones son mucho más baratas y más rápidas de implantar, y además se pueden pedir tantos módulos como se necesite. Se puede realizar un solo pedido para un módulo de almacén, compra, pedidos, facturación, etc. Alternativamente, en vez de módulos, se pueden pedir herramientas genéricas desarrolladas para sectores específicos, como por ejemplo construcción, restaurantes, bodegas, etc.

Con el software genérico podemos encontrarnos con un problema, ya que en la mayoría de los casos es muy difícil conocer y seleccionar las herramientas de gestión más adecuadas, debido a que el enfoque generalizado de estas herramientas conduce a dos situaciones:

1. Por un lado ofrecen una funcionalidad mayor respecto a la que necesita un cliente.
2. Por otro, carecen de alguna funcionalidad específica que necesita el cliente.

Por estos motivos se ha decidido en este caso crear un software a medida para que el sistema desarrollado se ajuste completamente a las necesidades de la empresa Hidrosierra Sport S.L.

Una vez elegido un desarrollo a medida del software se plantea la necesidad de emplear una metodología que pueda acortar el periodo de desarrollo, al tiempo que permita la evolución continua del proyecto, teniendo en cuenta las características cambiantes del entorno en el que se desarrolla la actividad empresarial de Hidrosierra Sport S.L. Por este motivo se adopta la decisión de realizar el proyecto bajo las directrices genéricas del desarrollo ágil, particularmente las propuestas por la metodología SCRUM, teniendo en cuenta que al tratarse de un desarrollo unipersonal deben modificarse y adaptarse algunos de los planteamientos de esta metodología.

1.3 *Objetivo*

El principal objetivo de este TFG es analizar, diseñar y desarrollar una aplicación de gestión e implantarla en un entorno real. En este caso en la empresa Hidrosierra Sport S.L.

Cabe destacar que el completo desarrollo del proyecto se realizará siguiendo un patrón de metodología ágil llamado SCRUM, adaptando totalmente este mecanismo a una sola persona, cambiando el enfoque de los diferentes roles.

La aplicación debe ser capaz de gestionar de manera eficiente todos y cada uno de los procesos descritos por el cliente. Para ello se ha de llevar a cabo una serie de tareas, empezando por conocer el funcionamiento de los procesos que se van a automatizar y reunir tanto la información necesaria que utiliza actualmente la empresa como la información que haya que añadir al nuevo sistema.

El análisis de los requisitos funcionales se realiza a través de diversas reuniones con los empleados de la empresa Hidrosierra Sport S.L. Este paso es muy importante porque es necesario que el modelo de datos y procesos sea totalmente completo, ya que serán la base de toda la funcionalidad del sistema. Se debe analizar toda la información y estudiar su viabilidad para poder realizar un diseño y una planificación que prevea y permita posibles ampliaciones futuras. Se desarrollará la aplicación siguiendo la planificación establecida y por último se debe verificar que todo funciona correctamente realizando un plan de pruebas adecuado.

1.4 *Estructura del documento*

Este documento se estructura de la siguiente manera:

En la sección 2 se lleva a cabo un breve análisis del estado del arte en lo referente a software de gestión empresarial, en el cual se muestra qué tipos de herramientas existen hoy en día en el mercado y se proporcionan algunos ejemplos, para más tarde evaluar las ventajas y desventajas de los distintos tipos de aplicaciones relacionadas con esta clase de proyectos.

En la sección 3 se definen la metodología y las herramientas utilizadas para la realización de la aplicación.

En la sección 4 se detallan las fases del análisis y diseño de la aplicación especificando el catálogo de requisitos, los casos de uso, el diagrama de clases de la aplicación y el diseño del modelo utilizado.

En la sección 5 se desarrollará la implementación de la aplicación. En este apartado se describe la arquitectura, el desarrollo de la base de datos, así como la explicación de algunos de los módulos más destacables.

En la sección 6 se detalla la verificación y validación de la aplicación mediante un conjunto de pruebas realizadas sobre el sistema.

En la sección 7 se evalúa el resultado final.

En la sección 8 se exponen las conclusiones obtenidas tras el desarrollo del proyecto y se mencionan las posibles líneas de trabajo futuro.

2 Estado del arte

2.1 *Introducción*

Hoy en día, prácticamente todas las PYMES requieren de herramientas automatizadas para poder gestionar el trabajo de una forma sencilla, ágil y eficiente. Se necesita además que estas herramientas ofrezcan buenos resultados con el fin de mejorar el rendimiento de la empresa. Es evidente que esta situación ha causado una gran competitividad entre las empresas de desarrollo de software, como suministradoras de productos de estas características, las cuales están creando aplicaciones para conseguir mayor cuota de mercado intentando aunar todos los procesos y actividades fundamentales dentro de las empresas, con independencia del sector, en busca de su producto estrella.

La consecuencia de esta situación es el deseo por parte de las PYMES por obtener herramientas o paquetes de software empresarial, con el que los directivos o propietarios de estas PYMES puedan tener incorporadas todas las áreas de la compañía, permitiéndoles obtener el máximo control y centralización de la información, para así poder tomar las mejores decisiones en sus procesos y estrategias de negocios, al tiempo que se facilita la labor diaria de los procesos operacionales.

Como ya se ha comentado, hay diferentes tipos de herramientas de gestión. Por un lado están las herramientas hechas a medida y por otro las herramientas genéricas o estándar. Dentro de las genéricas existen diferentes tipos, pero nos vamos a centrar en las llamadas Enterprise Resource Planning (ERP), que son las que más se asemejan al tipo de herramienta a desarrollar en este trabajo.

2.2 *Herramientas estándar para PYMES*

2.2.1 Enterprise Resource Planning (ERP)

Los sistemas ERP son herramientas de gestión empresarial que se caracterizan por modelar y automatizar los procesos de negocio dentro de una empresa. ERP puede ofrecer, por un lado, herramientas que solo ofrezcan módulos individuales, es decir, producción, ventas, inventario, contabilidad, etc. Y por otro lado, herramientas que incluyan todos los módulos que la empresa piensa que son imprescindibles para la totalidad de sus áreas de negocio. Este tipo de software es desarrollado generalmente por empresas del sector informático, ajenas a negocios concretos, que crean este tipo de software para satisfacer las demandas de empresas de cualquier tamaño.

2.2.1.1 Ejemplos de Sistemas ERP

- Open ERP

Se trata de un sistema de gestión de código abierto bastante completo. Ofrece gran variedad de módulos de manera estándar, tales como: gestión de compraventa, gestión de proyectos, sistema de gestión de almacenes, manufactura, contabilidad analítica y financiera, puntos de venta, gestión de activos, gestión de recursos humanos, gestión de inventario, ayuda técnica, campañas de marketing o flujos de trabajo. [1]

- Blue ERP

Es una herramienta de código abierto que se distribuye de forma gratuita y está centrada en las PYMES. Sus principales módulos ERP son: contabilidad, facturación, gestión de pedidos y ventas, inventario y análisis de ventas por usuario.

- Sage ERP

Sage ERP x3 es una herramienta de gestión que se caracteriza por su funcionalidad y fácil implementación. Se trata de un sistema escalable y se pueden seguir añadiendo funcionalidades a medida que la empresa vaya creciendo. Sus principales módulos son: finanzas, gestión del stock, compras, ventas y fabricación. Además, puede proporcionar acceso a través de cualquier dispositivo móvil, Tablet y Smartphone. [2]

- Kubbos ERP

Kubbos es otra aplicación de gestión empresarial que ofrece módulos como: facturación, CRM, compras, ventas y almacén. [3]

2.3 *Herramientas a medida para PYMES*

El software a medida es aquel que como su propio nombre indica se diseña de manera específica para el usuario que lo contrata, es decir, para una empresa concreta y teniendo en cuenta su particular forma de trabajar. Por tanto, este software busca satisfacer todas las demandas específicas y adaptarse lo mejor posible a lo que una empresa realmente necesita. [4]

2.4 *Ventajas y desventajas del uso de herramientas a medida vs estándar.*

2.4.1 Características del software a medida

1. La adaptación a la empresa es plena, ya que se realiza un análisis, un diseño y posteriormente un desarrollo orientado a las necesidades de la propia empresa.
2. El aprendizaje por parte de los usuarios para usar la aplicación es mínimo, debido a que se realizan generalmente numerosas reuniones con los mismos durante el desarrollo.
3. El Software está totalmente optimizado, evitando procesos redundantes. Además, normalmente se utilizaran el 100% de las funcionalidades.
4. Es necesaria una gran implicación de los usuarios de la empresa a la hora de probar y verificar que el software funciona correctamente, y cumple con los requisitos iniciales.
5. El tiempo de desarrollo puede ser alto dependiendo de la dificultad de la aplicación a desarrollar.
6. El coste a menudo es más elevado que las soluciones estándar.
7. El software a medida puede ser ampliado si hay nuevas necesidades en la empresa y se ha tenido en cuenta esta circunstancia durante su desarrollo.

2.4.2 Características del software comercial o estándar

1. La adaptación a la empresa no es plena. Tendría que darse la casualidad de que haya en el mercado un software que implemente justo las funciones que requiere la empresa.
2. El aprendizaje por parte de los empleados para usar la aplicación es normalmente elevado, debido a que para los usuarios se trata de un software completamente nuevo, en el que no han participado en su desarrollo.
3. La implicación por parte de los usuarios será mínima, como consecuencia de no participardirectamente en el desarrollo de la aplicación.
4. El tiempo de implantación de la herramienta dependerá del tiempo que necesiten los usuarios en aprender a usar dicha aplicación.
5. El coste a menudo será menor que en las herramientas hechas a medida.
6. El software estándar no podrá ser ampliado tan fácilmente a no ser que la empresa desarrolladora saque al mercado nuevos módulos que se ajusten a las nuevas necesidades de la empresa.

2.5 *Opiniones y conclusiones*

Tras analizar distintos tipos de herramientas de gestión que existen actualmente, y valorar las ventajas y desventajas de las aplicaciones a medida y de los ERP, se ha considerado oportuno para el desarrollo de este proyecto la creación de una aplicación a medida por los siguientes motivos.

- Se necesita una herramienta que, además de aunar varios de los módulos que suelen venir integrados en los sistemas ERP, como son los de inventario, facturación, ventas, clientes o proveedores, proporcione módulos especiales para cumplir con los requisitos específicos del cliente. Estos módulos son los de gestión de socorristas y comunidades. Respecto al punto uno de las características del software a medida, esto implica que se necesita una herramienta con adaptación plena a la empresa.
- Se pretende aprovechar la implicación de los empleados en el desarrollo de la aplicación, que verificaran en los momentos oportunos el correcto funcionamiento de cada nuevo módulo.
- Uno de los puntos que hay que valorar a la hora de elegir el tipo de desarrollo es el coste del proyecto. Aunque lo normal es que el software a medida sea más costoso, en este caso, al tratarse de un trabajo académico el coste para la empresa se reduce considerablemente. Por lo tanto, debido a esta circunstancia es menos costoso realizar un software a medida para la empresa Hidrosierra Sport S.L.
- Aparte de la funcionalidad especial que necesita la aplicación y de su coste, otro punto a favor de un diseño a medida es el apartado de la interfaz gráfica, que debe diseñarse al detalle con la información propia del cliente. Esta personalización de los aspectos visuales es más fácil de conseguir con una aplicación a medida.

En definitiva, se considera de especial importancia que el software a medida se adapta totalmente a las necesidades particulares de la empresa. También el aprendizaje del sistema es en este caso menor que el de utilizar un ERP, por lo que el tiempo de implantación se reduce ya que no se necesita un periodo de formación prolongado. La razón es que el uso de una metodología como SCRUM facilita mucho la interacción del propio cliente a lo largo de todo el proceso de desarrollo.

3 Definición del Proyecto

3.1 *Metodología*

3.1.1 Introducción a la metodología

Una metodología es un conjunto de métodos, técnicas y herramientas que se utilizan en una determinada actividad con el objetivo de formalizarla y optimizarla. Determina qué pasos hay que seguir y como realizarlos para finalizar una tarea. Si se aplica este concepto al desarrollo de software, la metodología define qué hacer, cómo y cuándo hacerlo durante todo el periodo de desarrollo y mantenimiento del proyecto.

Para este proyecto se ha considerado apropiado utilizar una metodología basada en los denominados desarrollos ágiles, debido a algunas de sus principales características:

- La máxima prioridad es la satisfacción del cliente a través de entregas continuas donde el cliente puede ir comprobando el avance del proyecto.
- Se permiten cambios en los requisitos en cualquier fase del desarrollo.
- Se realizan de forma frecuente entregas de software.
- Tanto clientes como desarrolladores deben trabajar juntos de forma regular durante la realización del proyecto.

Dentro de estas metodologías ágiles se propone el uso de SCRUM.

3.1.2 SCRUM

SCRUM es un tipo de metodología ágil que se caracteriza por determinar un conjunto de prácticas y de roles, que de una forma iterativa abarcan todo el proceso de desarrollo de un proyecto. [4]

3.1.2.1 Proceso de desarrollo

Se basa en el concepto de sprint, que son iteraciones cuya duración suele estar comprendida entre 2 y 4 semanas y donde el resultado de cada sprint se convierte en una versión del software totalmente entregable al cliente. Estas versiones corresponden a un incremento o a parte de un incremento del producto final. La figura 1 muestra de forma esquemática el ciclo de vida de la metodología SCRUM.

Figura 1: Ciclo de vida de la metodología ágil SCRUM

Los elementos principales de la metodología SCRUM son los siguientes:

Product Backlog: Incluye el conjunto de requisitos que definen el trabajo a realizar. Representa la definición del proyecto y se divide en objetivos, cada uno aportando valor de negocio incremental. Normalmente los objetivos se especifican como *Historias de Usuario*. Una historia de usuario es un requisito de negocio planteado desde el punto de vista del cliente. A continuación se muestra un ejemplo del formato típico de las historias de usuario aplicado al proyecto desarrollado.

Historia de Usuario		
ID: HU-1	**Nombre: Gestionar Clientes**	**Usuario: Empleado**
Prioridad: Alta	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 40**
Descripción: Como empleado quiero tener la posibilidad de crear nuevos clientes, modificar sus datos actuales, buscar información y visualizar sus datos, así como borrarlos del sistema, con el objetivo de automatizar la gestión de los clientes.		
Observaciones: Esta operación solo la realiza un usuario con privilegios de empleado.		

Tabla 1: Ejemplo de formato para las historias de usuario

Sprint Planning: Es una reunión liderada por el Scrum Master, en la cual el Product Owner (ambos roles se explican a continuación en el apartado de roles) identifica los aspectos más importantes y con más prioridad del Product Backlog y los comunica al equipo de desarrollo mediante el Sprint Backlog. En esta reunión el equipo planifica el sprint y determina las tareas que serán necesarias para satisfacer cada requisito o historia de usuario seleccionado del Product Backlog, planteando el tablero de tareas inicial.

Sprint Backlog: Contiene las historias de usuario que se han seleccionado para llevar a cabo durante el sprint. El equipo organiza la forma de trabajar según le parezca y lleva a cabo las tareas que tienen que realizar en el sprint para cumplir con las historias de usuario.

Daily Sprint Meeting: Es una reunión diaria, que se celebra con el fin de trabajar de manera coordinada. Cada miembro del equipo expone lo que ha hecho el día anterior y lo que le toca hacer ese día para así tener constancia de todo lo que se va terminando o de lo que hay que modificar.

Demostración y retrospectiva: Estas actividades ocurren al final de cada sprint. El equipo muestra una versión de la funcionalidad del producto terminada para poder continuar con el siguiente incremento. Posteriormente se produce la retrospectiva, donde el equipo estudia si el trabajo se ha realizado con éxito o bien existen partes mejorables que habrá que tratar.

3.1.2.2 Roles

Product Owner: Representa al cliente. Es la persona que conoce el proyecto y el que dirige e incorpora las tareas prioritarias que hay que realizar en el **Product Backlog.**

Scrum Master: Se refiere a la persona que se encarga de guiar al grupo para cumplir las metas, asegurando que los problemas, obstáculos e impedimentos se resuelven de manera satisfactoria. No siempre el **Scrum Master** es el encargado de resolver esos problemas, sino el que se preocupa de que el equipo pueda solucionarlos.

Scrum Team: Constituye el equipo de desarrollo, es decir, el conjunto de personas encargadas de desarrollar e implementar las funcionalidades descritas por el Product Owner en cada sprint.

3.1.2.3 Beneficios de la metodología SCRUM

- Uno de los grandes beneficios sin duda es la gran flexibilidad que SCRUM ofrece a la hora de poder realizar posibles cambios requeridos por el cliente, o por diversas causas, al finalizar cada sprint. Durante el sprint los cambios no son permitidos.
- Otra ventaja considerable es que el cliente puede comenzar a utilizar las funcionalidades más importantes del proyecto antes de que el mismo esté totalmente terminado, gracias a las entregas que se van realizando en cada sprint. Como se comentó, el resultado del sprint son incrementos o partes de incrementos del producto final.

- SCRUM asegura el cumplimiento de las expectativas de manera regular, ya que en cada demostración que se le enseña al cliente, el Product Owner comprueba que los requisitos del sprint se han cumplido.

3.1.2.4 ***Cambios en la metodología SCRUM para adaptarla a un equipo de desarrollo unipersonal***

Al tratarse en este caso de un proyecto realizado por un solo desarrollador es necesario realizar una serie de adaptaciones de la metodología SCRUM, debido a que determinados roles, actuaciones o productos no tienen sentido en esta situación. Las modificaciones son las siguientes:

- Los roles quedan reducidos al único desarrollador, que hace las veces de Scrum Master y Scrum Team. El papel de Product Onwer se reparte entre empleados de la empresa Hidrosierra Sport S.L. y el propio desarrollador.
- Los Daily Sprint Meeting son sustituidos por revisiones en las que se analiza lo que se ha realizado actualmente y lo que falta por realizar. La periodicidad de estas revisiones no está determinada previamente pero se garantiza un mínimo de una a la semana.
- La reunión de demostración se mantiene, con la particularidad de que el desarrollador único muestra el resultado del sprint al cliente.
- La reunión de retrospectiva se sustituye por una revisión en la que se analizan los puntos fuertes y débiles del sprint.
- El Product Backlog se mantiene respecto a su configuración estándar.
- Como solo hay desarrollador, el concepto de asignación de tareas carece de sentido, por lo que el Tablero de Tareas puede eliminarse, ya que la información de las tareas aparece en el Sprint Backlog. Se opta por añadir una columna al Sprint Backlog que indique el estado de realización de la tarea.
- En la metodología SCRUM tradicional suele utilizarse también un **Diagrama de Quemado**, que representa gráficamente la cantidad de trabajo que le falta al equipo para completar las tareas del sprint en curso. Este diagrama no se ha considerado necesario en este caso porque su función es equivalente a la revisión que se lleva a cabo como sustitución de los Daily Sprint Meeting.

3.2 *Herramientas y tecnologías utilizadas en el desarrollo*

Se enumeran a continuación las herramientas y tecnologías principalmente utilizadas durante el desarrollo del proyecto.

Netbeans IDE 8.1

Netbeans es un entorno de desarrollo integrado (IDE) libre y gratuito. Está escrito en JAVA pero sirve para cualquier tipo de lenguaje. En este caso sí se utiliza como lenguaje de programación JAVA, que contiene clases para interactuar con el API de

Netbeans. Encierra muchas funcionalidades, de las cuales la prueba y la depuración son algunas de las más importantes. [4]

Java Swing

Java Swing es un paquete propio del lenguaje Java que se instala en Netbeans y que permite dotar de una interfaz gráfica de usuario a las aplicaciones, incrementando su interactividad y riqueza visual. La herramienta Swing es la evolución de AWT (AbstractWindowToolkit), que como Swing, es un conjunto de librerías que permiten crear interfaces. [5]

WAMP

WAMP constituye un entorno de desarrollo Web que permite tener un servidor propio de forma local en un ordenador personal. Las principales características por las que se ha utilizado en este proyecto son: [6]

- Manejo de Bases de Datos como MySQL.
- Software libre.
- Permite el manejo sencillo de las Bases de Datos gracias PHPMyAdmin.

MySQL

MySQL es un Sistema Gestor de Bases de Datos (Database Management System) para bases de datos relacionales. MySQL utiliza tablas para almacenar y organizar la información. Destaca por su gran adaptación con los diferentes entornos de desarrollo, por lo que permite interactuar con los lenguajes de programación más utilizados como PHP y JAVA. Además, MySQL es un sistema de código abierto totalmente gratuito que ha hecho que sea una de las herramientas más utilizada por los desarrolladores. [7]

PHPMyAdmin

PHPMyAdmin es una herramienta que sirve para administrar Bases de Datos MySQL. Es un programa de libre distribución escrito en PHP. Se pueden crear, modificar y eliminar tanto las tablas como los campos de dichas tablas, dentro de la Base de Datos. PHPMyAdmin ofrece la posibilidad de obtener el diagrama relacional de la Base de Datos implementada. Es una herramienta muy completa que accede a todas las funciones de la base de datos MySQL a través de una interfaz web muy sencilla e intuitiva. [8]

3.3 *Definición del proyecto*

El sistema a desarrollar para la empresa Hidrosierra Sport S.L. es esencialmente una aplicación de gestión empresarial para PYMES. Incluye funciones típicas como la gestión de clientes, proveedores, trabajadores propios de la empresa, facturas, albaranes y partes de trabajo. También implementa funciones específicas de la empresa, como el tratamiento de comunidades o socorristas. El objetivo del sistema es automatizar todas estas actividades,

que actualmente se llevan a cabo de forma manual o semi-automática, mediante sencillas hojas de cálculo.

3.3.1 Funcionalidades del sistema

A continuación se describen brevemente las principales funciones del proyecto.

- **Gestión de clientes, proveedores y trabajadores**: La aplicación debe permitir la gestión integral de los clientes, los proveedores y los trabajadores propios de HidroSierra Sport S.L. Las operaciones de gestión incluyen el alta, baja, modificación y visualización de la información pertinente a cada uno de estos colectivos.
- **Gestión de facturas y presupuestos**: La emisión de facturas y presupuestos son tareas cotidianas de cualquier empresa, que deben gestionarse a través de la aplicación, permitiendo su creación, modificación, visualización e impresión.
- **Gestión de albaranes y partes de trabajo**: De manera similar a las facturas y los presupuestos, la gestión de los albaranes y los partes de trabajo permite el tratamiento apropiado de estos elementos.
- **Gestión de tarifas y tipos de tarifa**: La empresa Hidrosierra Sport S.L. mantiene distintos tipos de tarifa, cada una con su tarifa correspondiente. Estos elementos son utilizados por los empleados para elaborar los presupuestos y las facturas, por lo que la aplicación permite su creación, modificación y borrado.
- **Gestión de cuentas de empleados**: Esta sencilla funcionalidad permite crear y mantener información sobre las cuentas que se crean para los empleados que utilizan la aplicación.
- **Gestión de socorristas y comunidades**: Los socorristas son personal de la empresa Hidrosierra Sport S.L. con características específicas, que pueden etar asociados a comunidades. Ambos elementos forman parte del sistema y son gestionados a través del mismo.

3.3.2 Alcance del sistema

El sistema a desarrollar está pensado para una empresa de tipo PYME. De manera específica su utilización se dirige hacia empleados concretos de la empresa, con funciones definidas sobre las operaciones que pueden llevar a cabo. El propósito del sistema es facilitar las tareas cotidianas y habituales de la empresa, al tiempo que permite el registro y mantenimiento fiable y seguro de toda la información empresarial.

4 Desarrollo del Sistema con SCRUM

4.1 *Análisis de requisitos*

En este apartado se describe la etapa de análisis de requisitos. Dicha etapa es de suma importancia, probablemente una de las más importantes a la hora de realizar un proyecto.

El análisis de requisitos trata de recoger y describir detalladamente los requisitos de funcionalidad y de calidad de servicio del producto que se va a desarrollar.

Un requisito "*es una característica del sistema o una descripción de algo que el sistema es capaz de hacer con el objeto de satisfacer el propósito del sistema*", es decir, los requisitos son lo que el cliente espera que haga el sistema. Es por eso que es de tal relevancia entender y recoger bien los requisitos por parte del cliente, ya que es donde se detalla el verdadero fin del sistema.

Por cuestiones metodológicas propias del desarrollo orientado a objetos que se sigue en este proyecto, se van a definir a través de Diagramas de Casos de Uso las diferentes funcionalidades o acciones que puede realizar el usuario en el sistema.

4.1.1 Diagrama de casos de uso

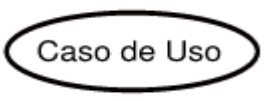
Un diagrama de casos de uso representa el modo en el que un usuario (actor) interactúa con el sistema. Además, muestra la forma y orden relativos a cómo los elementos se relacionan entre sí.

Un diagrama de casos de uso se compone de los siguientes elementos (roles):

- Actor
- Casos de uso
- Relaciones de uso

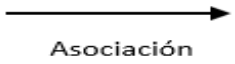
Actor: Representa un rol que juega una persona, un dispositivo u otro sistema que interactúe con el sistema a desarrollar. Se representan con el siguiente símbolo:

Caso de Uso: Representa la acción que realiza el actor. Se representan mediante un óvalo:

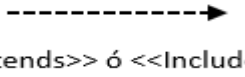


Relaciones: Representan los tipos de interacción o asociación entre elementos. Las principales son:

- **Asociación**: Indica la relación entre un actor o un caso de uso y otro caso de uso. Se representa con una flecha simple.



- **Dependencia o Instanciación:** Se produce cuando un caso de uso depende de otro. Se representa con una flecha punteada.



  - ❖ *Include*: Indica que el primer caso (caso de uso base) incluye al segundo (caso de uso incluido). Equivale a expresar que el segundo caso es parte esencial del primer caso, es decir, sin el segundo el primero no podría funcionar.
  - ❖ *Extend*: Se utiliza cuando un caso de uso base incorpora el comportamiento de otro caso de uso y "extiende" su funcionamiento.

A continuación se muestra el Diagrama de Casos de Uso general de la aplicación. Es importante destacar que la creación de este diagrama se ha hecho con anterioridad al comienzo de los sprints, concretamente durante la fase de educción de requisitos que se expondrá en el siguiente apartado. El motivo es que se necesitaba una descripción que incluyera las principales funciones del sistema, con el fin de clarificar y ordenar el conjunto de subsistemas o módulos que lo conforman.

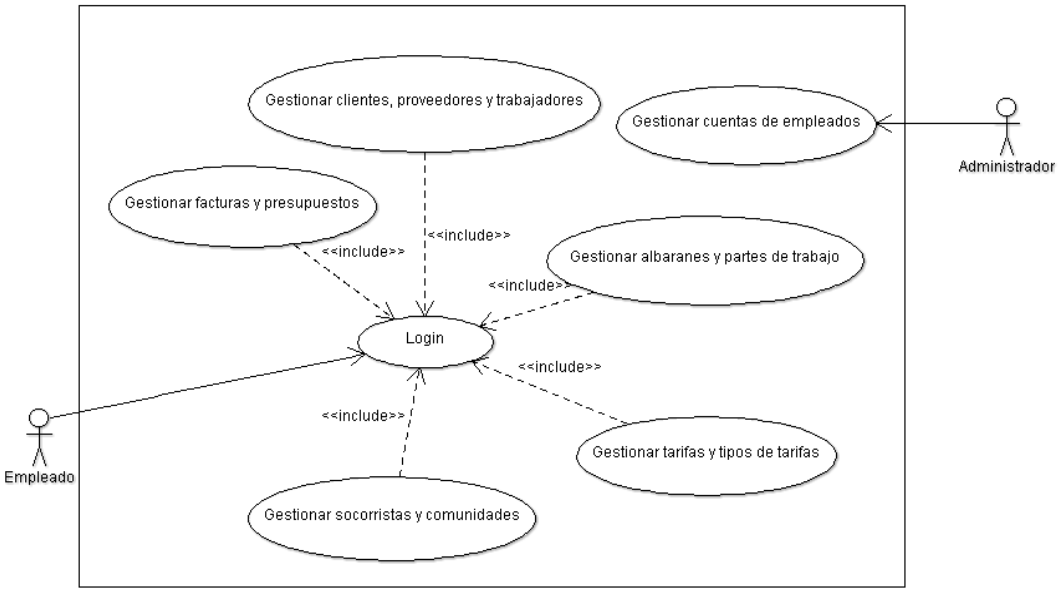


Figura 2: Diagrama de Casos de Uso de la Aplicación

En el diagrama se puede observar que para poder realizar cualquier operación primero el empleado deberá identificarse (login) en el sistema.

4.1.2 Catálogo de requisitos

En la metodología SCRUM, los requisitos se recogen bajo el formato de Historias de Usuario, que son funcionalidades del negocio escritas desde la perspectiva del cliente. Estos requisitos funcionales son complementados por otro tipo de requisitos, que generalmente se denominan requisitos no funcionales. Ambos tipos de requisitos se resumen a continuación.

- **Funcionales:** especifican las características útiles del sistema. Se refieren a las acciones fundamentales que tienen que tener lugar durante la ejecución de la aplicación.
- **No funcionales:** pueden ser de diferentes tipos.
  - Operacionales (recuperación, backups, etc.)
  - Seguridad (niveles de acceso, protección, etc.)
  - Disponibilidad (acceso a la información, etc.)
  - Mantenibilidad y portabilidad (valido para Windows, Linux, etc.)
  - Recursos (memoria, almacenamiento, etc.)
  - Rendimiento (tiempos de respuesta, nº de usuarios, etc.)
  - Interfaz y usabilidad (tamaños de los elementos en pantalla, etc.)
  - Fiabilidad y verificación (situaciones anómalas de error, etc.)

4.1.2.1 Requisitos Funcionales

Los requisitos funcionales constituyen la base a partir de la que se describen las Historias de Usuario del Product Backlog. Se obtienen a partir de sucesivas entrevistas con los usuarios de la aplicación, es decir, los empleados de la empresa Hidrosierra Sport S.L. A continuación se enumeran los requisitos funcionales educidos durante la fase de análisis. Es importante indicar que SCRUM permite la adaptación y creación de los requisitos entre Sprints, lo que indudablemente es una ventaja en cuanto a flexibilidad a los cambios, frente a metodologías tradicionales. Siguiendo la metodología SCRUM se exponen los requisitos como Historias de Usuario.

REQUISITOS DE GESTIÓN DE USUARIOS

Historias de Usuario		
ID: HU-1	**Nombre: Gestión de usuarios**	**Usuario: Administrador**
Prioridad: Baja	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 4**
Descripción: Como administrador quiero tener la posibilidad de crear usuarios con contraseña y poder cambiar las contraseñas, con el objetivo de que los empleados puedan acceder al sistema. Para cualquier operación el usuario deberá autentificarse en primer lugar.		

Tabla 2: Historia de usuario para gestión de usuarios.

REQUISITOS DE GESTIÓN DEL EMPLEADO

Los requisitos de gestión del empleado son los que permiten el acceso a los datos del sistema, así como la posibilidad de operar con ellos. Un empleado puede: insertar nuevos datos, visualizar los datos ya existentes, modificar los datos y eliminarlos. Todas las operaciones que se describen en esta sección solo puede realizarlas un usuario con el rol de empleado.

Desde el menú principal, el empleado puede acceder a cada una de las entidades: **clientes**, **proveedores**, **facturación**, **presupuestos**, **albaranes**, **tarifas**, **partes de trabajo**, **trabajadores**, **socorristas** y **comunidades**. Una vez el que el empleado ha seleccionado alguna de las entidades anteriores ya puede operar con los datos correspondientes.

A continuación se explican a grandes rasgos los requisitos para insertar nuevos datos, buscar, modificar y eliminar de forma generalizada, debido a que en todas las entidades se opera de la misma manera.

- **Insertar Nueva entidad**
  - El empleado debe introducir todos los datos requeridos para dar de alta la nueva entidad, estos datos son validados por un control de formulario, mostrando un error si no se han introducido bien los datos y un aviso si se ha dado de alta correctamente.
- **Buscar y filtrar entidad**
  - El empleado puede buscar entidades gracias al método de filtrado. Estos filtros permiten al empleado seleccionar varios valores de una lista para facilitar la búsqueda. Además, el empleado puede realizar una búsqueda utilizando un solo filtro o todos los filtros disponibles, ya que dependiendo de la entidad existen varios tipos de filtro. Alguno de estos filtros son: Nombre, Dni, Fecha, etc.
- **Modificar entidad**

- El empleado puede actualizar los datos que desee de una entidad ya guardada en la base de datos y que previamente ha buscado en la lista de entidades.

Eliminar entidad

- El empleado puede eliminar los datos que desee de una entidad de la búsqueda realizada. Antes de eliminar, se solicitará confirmación al empleado.

Una vez explicadas las funciones genéricas se procede a la identificación de cada una de las Historias de Usuario de manera individual.

Historia de Usuario		
ID: HU-2	**Nombre: Gestionar clientes**	**Usuario: Empleado**
Prioridad: Alta	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 40**
Descripción: Como empleado quiero tener la posibilidad de crear nuevos clientes, modificar los datos de los clientes actuales, buscar información de los clientes y visualizar sus datos, con el objetivo de automatizar la gestión de los clientes.		

Tabla 3: Historia de usuario para gestión de clientes.

Historia de Usuario		
ID: HU-3	**Nombre: Gestionar proveedores**	**Usuario: Empleado**
Prioridad: Alta	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 30**
Descripción: Como empleado quiero tener la posibilidad de crear nuevos proveedores, modificar los datos de los proveedores actuales, buscar información de los proveedores, visualizar sus datos y poder eliminarlos del sistema, con el objetivo de automatizar la gestión de los proveedores.		

Tabla 4: Historia de usuario para gestión de proveedores.

Historia de Usuario		
ID: HU-4	**Nombre: Gestionar trabajadores**	**Usuario: Empleado**
Prioridad: Alta	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 30**
Descripción: Como empleado quiero tener la posibilidad de crear nuevos trabajadores, modificar los datos de los trabajadores actuales, buscar información de los trabajadores y visualizar sus datos, con el objetivo de automatizar la gestión de los trabajadores.		

Tabla 5: Historia de usuario para gestión de trabajadores.

Historia de Usuario		
ID: HU-5	Nombre: Gestionar socorristas	Usuario: Empleado
Prioridad: Alta	Riesgo en desarrollo: Bajo	Estimación en horas: 35
Descripción: Como empleado quiero tener la posibilidad de crear nuevos socorristas, modificar los datos de los socorristas actuales, buscar información de los socorristas, así como poder añadir en que comunidades trabaja o ha trabajado un socorrista, con el objetivo de automatizar la gestión de los socorristas.		

Tabla 6: Historia de usuario para gestión de socorristas.

Historia de Usuario		
ID: HU-6	Nombre: Gestionar comunidades	Usuario: Empleado
Prioridad: Alta	Riesgo en desarrollo: Bajo	Estimación en horas: 35
Descripción: Como empleado quiero tener la posibilidad de crear nuevas comunidades, modificar los datos de las comunidades actuales, buscar información de dichas comunidades y visualizar sus datos, con el objetivo de automatizar la gestión de las comunidades.		

Tabla 7: Historia de usuario para gestión de comunidades.

Historia de Usuario		
ID: HU-7	Nombre: Gestionar presupuestos	Usuario: Empleado
Prioridad: Alta	Riesgo en desarrollo: Bajo	Estimación en horas: 30
Descripción: Como empleado quiero tener la posibilidad de crear nuevos presupuestos, modificar los datos de los presupuestos actuales, buscar información de los presupuestos, así como poder imprimirlos y guardarlos en PDF, con el objetivo de automatizar la gestión de los presupuestos.		

Tabla 8: Historia de usuario para gestión de presupuestos.

Historia de Usuario		
ID: HU-8	**Nombre: Gestionar facturas**	**Usuario: Empleado**
Prioridad: Alta	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 30**
Descripción: Como empleado quiero tener la posibilidad de crear nuevas facturas, modificar los datos de las facturas actuales, buscar información de las facturas, así como poder imprimirlas y guardarlas en PDF, con el objetivo de automatizar la gestión de las facturas.		

Tabla 9: Historia de usuario para gestión de facturas.

Historia de Usuario		
ID: HU-9	**Nombre: Gestionar tarifas**	**Usuario: Empleado**
Prioridad: Media	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 25**
Descripción: Como empleado quiero tener la posibilidad de crear nuevas tarifas, modificar los datos de las tarifas actuales, buscar tarifas, y eliminar tarifas, con el objetivo de automatizar la gestión de las tarifas.		

Tabla 10: Historia de usuario para gestión de tarifas.

Historia de Usuario		
ID: HU-10	**Nombre: Gestionar tipos de tarifa**	**Usuario: Empleado**
Prioridad: Media	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 25**
Descripción: Como empleado quiero tener la posibilidad de crear nuevos tipos de tarifas, modificar los datos de los tipos actuales, buscar tipos, y eliminar tipos, con el objetivo de automatizar la gestión de los tipos de tarifas.		

Tabla 11: Historia de usuario para gestión de tipos de tarifas.

Historia de Usuario		
ID: HU-11	**Nombre: Gestionar albaranes**	**Usuario: Empleado**
Prioridad: Media	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 30**
Descripción: Como empleado quiero tener la posibilidad de crear nuevos albaranes, modificar los datos de los albaranes actuales, buscar información de los albaranes, y modificar la información de los albaranes, con el objetivo de automatizar la gestión de los albaranes.		

Tabla 12: Historia de usuario para gestión de albaranes.

Historia de Usuario		
ID: HU-12	**Nombre: Gestionar partes de trabajo**	**Usuario: Empleado**
Prioridad: Media	**Riesgo en desarrollo: Bajo**	**Estimación en horas: 30**
Descripción: Como empleado quiero tener la posibilidad de crear nuevos partes de trabajo, modificar los datos de los partes actuales, buscar partes, y poder modificar los partes, con el objetivo de automatizar la gestión de los partes de trabajo.		

Tabla 13: Historia de usuario para gestión de partes de trabajo.

4.1.2.2 Requisitos no funcionales

SEGURIDAD

- **[RNF-1] Seguridad para la privacidad de los datos.**
  El sistema solicitará hacer login con un usuario y contraseña a los empleados de la empresa. Esta contraseña la proporciona el administrador a los empleados. De esta forma se evita que personas ajenas puedan tener acceso a los datos de la aplicación.

DISPONIBILIDAD

- **[RNF-2] Disponibilidad.**
  Se garantizará el acceso de los empleados autorizados en el sistema a la información, para obtener los datos deseados en el momento que lo consideren oportuno.

INTERFAZ Y USABILIDAD

- **[RNF-3] Introducción de datos.**
  La inserción de los datos se realizará de tal forma que un empleado no pueda cometer errores a la hora de guardar la información en la base de datos, mediante una serie de controles de formulario.
- **[RNF-4] Navegación y uso.**
  Se creará una interfaz sencilla e intuitiva que facilite al empleado interactuar con el sistema con cierta facilidad y comodidad.

RECURSOS

- **[RNF-5] Volumen de datos.**
  No debe existir limite a la hora de crear clientes, proveedores, trabajadores, facturas, albaranes ni en el resto de entidades que utiliza el sistema.
- **[RNF-6] Bases de datos.**
  El sistema debe disponer de una Base de Datos capaz de albergar la variedad de datos requeridos por el cliente.

RENDIMIENTO

- **[RNF-7] Acceso a la información.**
  Se garantizará al empleado tiempos de respuesta del orden de segundos, a la hora de realizar búsqueda de información, inserción, modificación o eliminación de entidades en el sistema.

MANTENIMIENTO Y PORTABILIDAD

- **[RNF-8] Mantenimiento.**
  El sistema debe ser flexible, lo que quiere decir que estará estructurado de tal forma que se permita la introducción de nueva funcionalidad realizando solo pequeñas modificaciones en la aplicación.

OPERACIONALES

- **[RNF-9] Backup.**
  Regularmente se realizarán copias de seguridad de los datos de la Base de datos. Inicialmente se realizarán backups diariamente.

4.2 *Product Backlog*

Una vez que se han identificado con el cliente las funcionalidades principales y se han plasmado en forma de Historias de Usuario, se desarrolla el Product Backlog, que contiene esta información en un formato útil para su desarrollo de manera organizada. En la tabla 14 se muestra el Product Backlog utilizado. La división de la aplicación en Sprints se ha hecho en base a las prioridades del cliente indicadas en las Historias de Usuario. En caso de Historias de Usuario con igual prioridad el cliente establece un orden en la categoría asignando un valor numérico que aparece en la columna "Valor".

Historias de Usuario	Entidades	Sprint	Prioridad	Valor
HU-2 HU-3 HU-4	Clientes Proveedores Trabajadores	1	Alta	10
HU-5 HU-6	Socorristas Comunidades	2	Alta	9
HU-7 HU-8	Presupuestos Facturas	3	Alta	8
HU-9 HU-10	Tarifas Tipos de Tarifas	4	Media	7
HU-11 HU-12	Albaranes Partes de Trabajo	5	Media	6
HU-1	Administrador	6	Baja	5

Tabla 14: Product Backlog

Esencialmente, cada sprint recoge todos los requisitos funcionales correspondientes a un módulo o subsistema de la aplicación: clientes, proveedores, trabajadores, etc.

4.3 *Desarrollo de los Sprints*

Al iniciar cada sprint se seleccionan las historias de usuario a partir de la lista priorizada del Product Backlog. Para llevar a cabo cada Historia de Usuario se deben realizar una serie de tareas. Estas tareas recogen actividades de análisis, diseño, implementación y pruebas. Algunas de ellas abarcan más de un sprint, como es el caso del Diagrama de Clases, que es genérico para la aplicación y no puede catalogarse como propio de un módulo concreto. El contenido del Diagrama de Clases se detallará en un apartado posterior. Cabe destacar que en el transcurso de un sprint no se pueden cambiar los requisitos funcionales, pero la metodología sí permite adaptaciones entre los sprints. Por razones de espacio no se muestra la situación de todos los tableros de tareas, que como se recordará se encuentran unificados con el Sprint Backlog. Solo se presentan la situación inicial y final de los mismos en cada sprint.

4.3.1 Sprint 1. Clientes, proveedores y trabajadores.

En el Sprint 1 se desarrolla la gestión de los clientes, proveedores y trabajadores de la empresa Hidrosierra Sport S.L. El Sprint Backlog inicial correspondiente al primer sprint se muestra en la tabla 15.

Historia de Usuario	Tarea	Estado
HU-2, HU-3, HU-4	T-1:Diagrama de casos de uso	Pendiente
Global	T-2: Diagrama de clases	Pendiente
HU-2	T-3: Insertar nuevo cliente T-4: Buscar y filtrar cliente T-5: Modificar cliente	Pendiente Pendiente Pendiente
HU-3	T-6: Insertar nuevo proveedor T-7: Buscar proveedor T-8: Modificar proveedor T-9: Eliminar proveedor	Pendiente Pendiente Pendiente Pendiente
HU-4	T-10: Insertar nuevo trabajador T-11: Buscar y filtrar trabajador T-12: Modificar trabajador	Pendiente Pendiente Pendiente

Tabla 15: Sprint Backlog inicial del Sprint 1

En la figura 3 se muestra el Diagrama de Casos de Uso (T-1) para la gestión de clientes, proveedores y trabajadores. Como se puede observar en la figura, trabajadores y clientes no se pueden eliminar ya que es necesario tener un historial al completo de los operarios y clientes en la empresa.

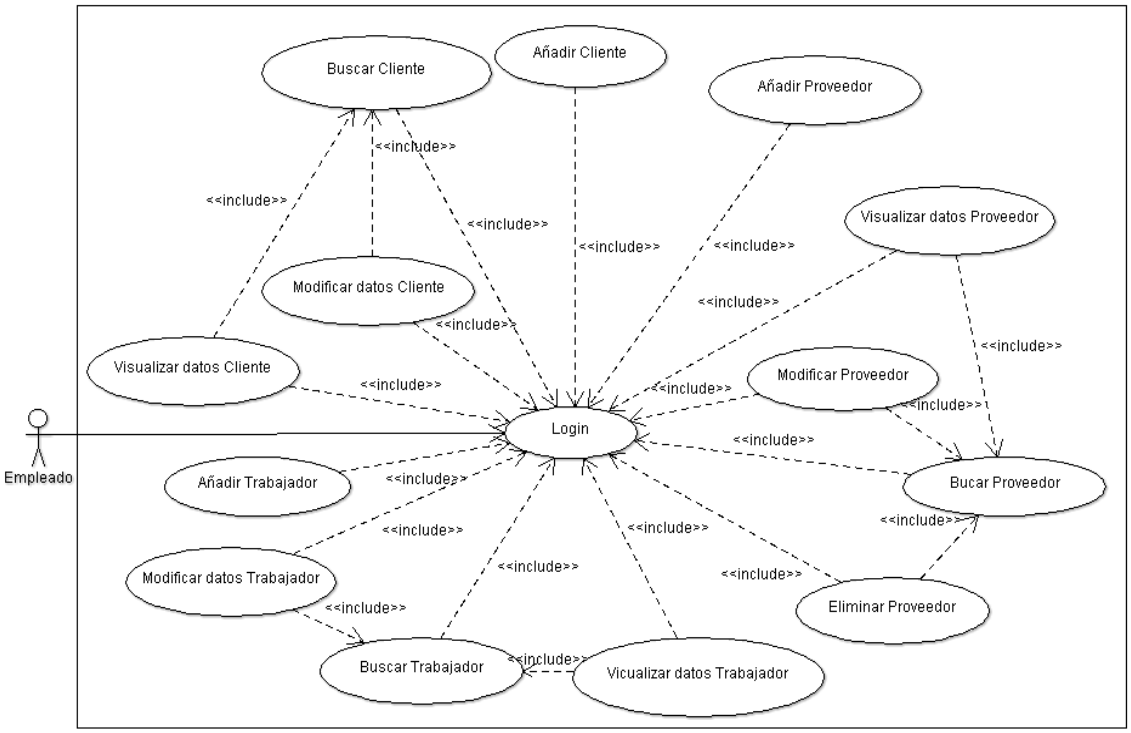


Figura 3: Diagrama de casos de uso para clientes, proveedores y trabajadores.

El conjunto de tareas T-3 a T-12 incluye como se ha indicado antes el diseño, la codificación de las clases y la realización de las pruebas pertinentes. Los detalles sobre la implementación y las pruebas se comentarán en otros capítulos. La situación al finalizar el sprint se muestra en la tabla 16. Dos tareas no se pudieron completar en el sprint.

Historia de Usuario	Tarea	Estado
HU-2, HU-3, HU-4	T-1:Diagrama de casos de uso	Terminado
Global	T-2: Diagrama de clases	En proceso
HU-2	T-3: Insertar nuevo cliente	Terminado
	T-4: Buscar y filtrar cliente	Terminado
	T-5: Modificar cliente	Terminado
HU-3	T-6: Insertar nuevo proveedor	Terminado
	T-7: Buscar proveedor	Terminado
	T-8: Modificar proveedor	Terminado
	T-9: Eliminar proveedor	Terminado
HU-4	T-10: Insertar nuevo trabajador	Terminado
	T-11: Buscar y filtrar trabajador	En proceso
	T-12: Modificar trabajador	En proceso

Tabla 16: Sprint Backlog final del Sprint 1

El sprint termina con una revisión del trabajo realizado para comprobar que se han cumplido las tareas comprometidas y con una demostración al cliente de las funcionalidades desarrolladas y completadas. En la evaluación se detecta un error en la estimación de la duración del sprint, que se planificó para 100 horas (alrededor de 3 semanas de trabajo efectivo) teniendo en cuenta la estimación de cada Historia de Usuario incluida en el sprint. El hecho de ser éste el primer sprint y el arranque del proyecto puede

ser la causa del retraso final. Se espera que en posteriores sprints este problema pueda resolverse, ya que se dispondrá de material de partida sobre el que trabajar y los módulos son muy parecidos entre ellos.

4.3.2 Sprint 2. Socorristas y comunidades

En el Sprint 2 se desarrolla la gestión de los socorristas y las comunidades. El Sprint Backlog inicial correspondiente al segundo sprint se muestra en la tabla 17. Como en el caso de los clientes, un socorrista y una comunidad no pueden ser eliminados del sistema por los empleados, ya que también es necesario tener guardado en la base de datos el historial de las comunidades y todos los trabajadores que han trabajado en ellas.

Historia de Usuario	Tarea	Estado
HU-5, HU-6	T-13: Diagrama de casos de uso	Pendiente
Global	T-2: Diagrama de clases	En proceso
HU-4	T-11: Buscar y filtrar trabajador T-12: Modificar trabajador	En proceso En proceso
HU-5	T-14: Insertar nuevo socorrista T-15: Buscar socorrista T-16: Modificar socorrista	Pendiente Pendiente Pendiente
HU-6	T-17: Insertar nueva comunidad T-18: Buscar comunidad T-19: Modificar comunidad	Pendiente Pendiente Pendiente

Tabla 17: Sprint Backlog inicial del Sprint 2

En la figura 4 se muestra el Diagrama de Casos de Uso (T-13) para la gestión de socorristas y comunidades.

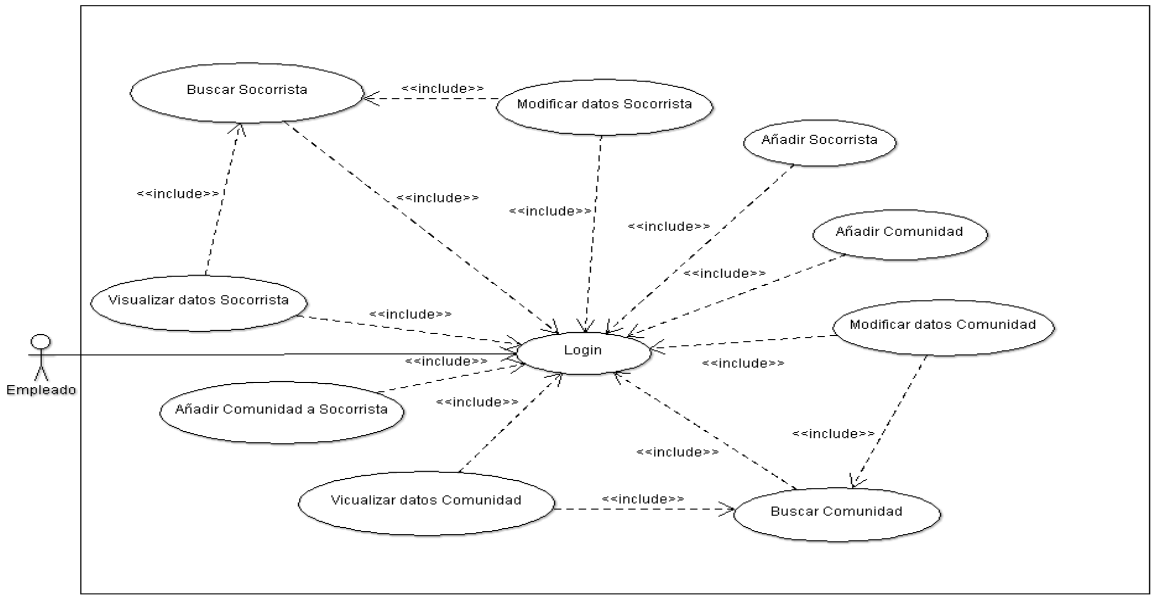


Figura 4: Diagrama de casos de uso para socorristas y comunidades

El conjunto de tareas T-14 a T-19 incluye el diseño, la codificación de las clases y la realización de las pruebas pertinentes. Las tareas T-11 y T-12 quedaron pendientes en el sprint anterior. Los detalles sobre la implementación y las pruebas se comentarán en otros capítulos. La situación al finalizar el sprint se muestra en la tabla 18.

Historia de Usuario	Tarea	Estado
HU-5, HU-6	T-13: Diagrama de casos de uso	Terminado
Global	T-2: Diagrama de clases	Terminado
HU-4	T-11: Buscar y filtrar trabajador	Terminado
	T-12: Modificar trabajador	Terminado
HU-5	T-14: Insertar nuevo socorrista	Terminado
	T-15: Buscar socorrista	Terminado
	T-16: Modificar socorrista	Terminado
HU-6	T-17: Insertar nueva comunidad	Terminado
	T-18: Buscar comunidad	Terminado
	T-19: Modificar comunidad	Terminado

Tabla 18: Sprint Backlog final del Sprint 2

El sprint termina con una revisión del trabajo realizado para comprobar que se han cumplido las tareas comprometidas y con una demostración al cliente de las funcionalidades desarrolladas y completadas. Como puede verse, en este sprint se completan las tareas pendientes del sprint anterior. La evaluación es positiva en este caso y se comprueba que, como se esperaba, el hecho de que los módulos se parezcan en su funcionalidad y que se disponga de material facilita el desarrollo de las nuevas funciones.

4.3.3 Sprint 3. Presupuestos y facturas

En el Sprint 3 se desarrolla la gestión de los presupuestos y las facturas. El Sprint Backlog inicial correspondiente al tercer sprint se muestra en la tabla 19. Como ocurría en casos anteriores los presupuestos y las facturas no pueden ser eliminados de la base de datos por los empleados.

Historia de Usuario	Tarea	Estado
HU-7, HU-8	T-20: Diagrama de casos de uso	Pendiente
HU-7	T-21: Insertar nuevo presupuesto	Pendiente
	T-22: Buscar presupuesto	Pendiente
	T-23: Modificar presupuesto	Pendiente
	T-24: Imprimir presupuesto	Pendiente
	T-25: Guardar presupuesto en PDF	Pendiente
HU-8	T-26: Insertar nueva factura	Pendiente
	T-27: Buscar factura	Pendiente
	T-28: Modificar factura	Pendiente
	T-29: Imprimir factura	Pendiente
	T-30: Guardar factura en PDF	Pendiente

Tabla 19: Sprint Backlog inicial del Sprint 3

Cabe destacar que en estos dos módulos se crean funcionalidades nuevas que no contienen los demás módulos: la impresión y la creación de documentos PDF. En la figura 5 se muestra el Diagrama de Casos de Uso (T-20) para la gestión de presupuestos y facturas. El conjunto de tareas T-21 a T-30 incluye como se ha indicado antes el diseño, la codificación de las clases y la realización de las pruebas pertinentes. Los detalles sobre la implementación y las pruebas se comentarán en otros capítulos.

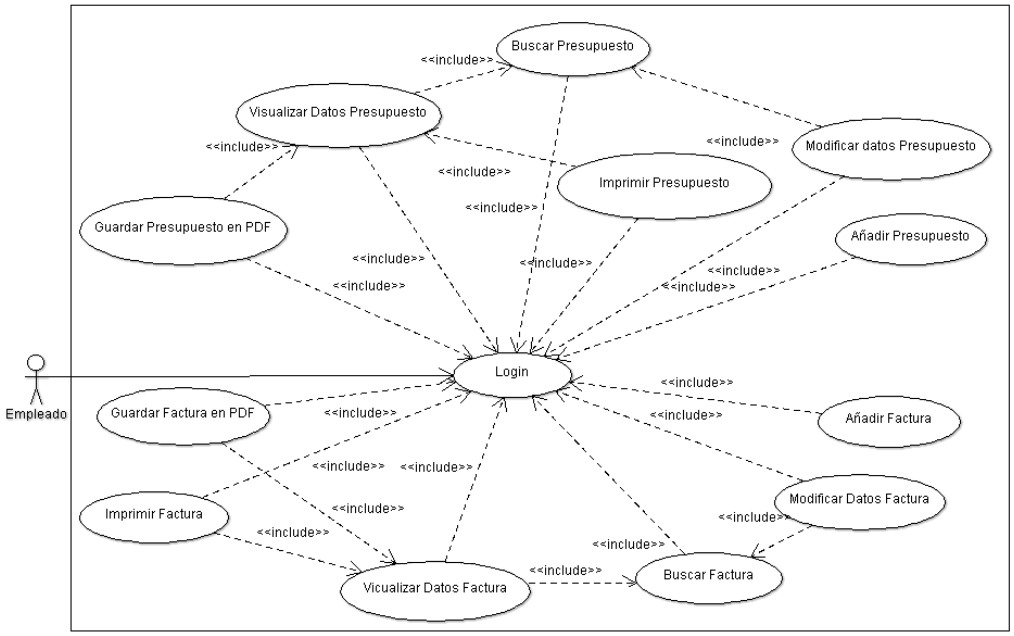


Figura 5: Diagrama de casos de uso para presupuestos y facturas.

La situación al finalizar el sprint se muestra en la tabla 20.

Historia de Usuario	Tarea	Estado
HU-7, HU-8	T-20: Diagrama de casos de uso	Terminado
HU-7	T-21: Insertar nuevo presupuesto	Terminado
	T-22: Buscar presupuesto	Terminado
	T-23: Modificar presupuesto	Terminado
	T-24: Imprimir presupuesto	Terminado
	T-25: Guardar presupuesto en PDF	Terminado
HU-8	T-26: Insertar nueva factura	Terminado
	T-27: Buscar factura	Terminado
	T-28: Modificar factura	Terminado
	T-29: Imprimir factura	Terminado
	T-30: Guardar factura en PDF	Terminado

Tabla 20: Sprint Backlog final del Sprint 3

El sprint termina con una revisión del trabajo realizado para comprobar que se han cumplido las tareas comprometidas y con una demostración al cliente de las funcionalidades desarrolladas y completadas.

4.3.4 Sprint 4. Tarifas y tipo de tarifas.

En el cuarto sprint se desarrolla la gestión de las tarifas y los tipos de tarifa. A diferencia de otras clases, éstas sí se pueden eliminar. En este caso, las tarifas y sus tipos se usan por el empleado como información sobre qué tarifas hay, que tipos y cuál es el valor de dicha tarifa, por lo que en cualquier momento una tarifa o un tipo pueden ser eliminados del sistema. El Sprint Backlog inicial se muestra en la tabla 21.

Historia de Usuario	Tarea	Estado
HU-9, HU-10	T-31: Diagrama de casos de uso	Pendiente
HU-9	T-32: Insertar nueva tarifa	Pendiente
	T-33: Buscar tarifa	Pendiente
	T-34: Modificar tarifa	Pendiente
	T-35: Eliminar tarifa	Pendiente
HU-10	T-36: Insertar nuevo tipo de tarifa	Pendiente
	T-37: Modificar tipo de tarifa	Pendiente
	T-38: Eliminar tipo de tarifa	Pendiente

Tabla 21: Sprint Backlog inicial del Sprint 4

En la figura 6 se muestra el Diagrama de Casos de Uso (T-31) para la gestión de las tarifas y los tipos de tarifa.

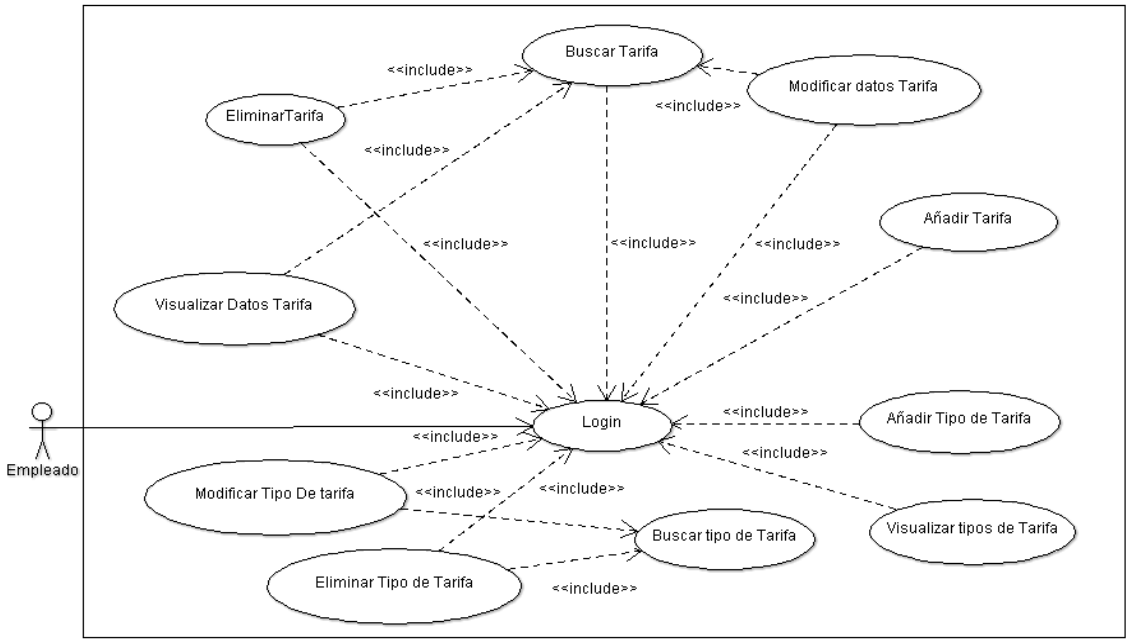


Figura 6: Diagrama de casos de uso para tarifas y tipos de tarifas.

El conjunto de tareas T-32 a T-38 incluye el diseño, la codificación de las clases y la realización de las pruebas pertinentes. Los detalles sobre la implementación y las pruebas

se comentarán en otros capítulos. El sprint termina con una revisión del trabajo realizado para comprobar que se han cumplido las tareas comprometidas y con una demostración al cliente de las funcionalidades desarrolladas y completadas. La situación final del sprint se refleja en la tabla 22.

Historia de Usuario	Tarea	Estado
HU-9, HU-10	T-31: Diagrama de casos de uso	Terminado
HU-9	T-32: Insertar nueva tarifa	Terminado
	T-33: Buscar tarifa	Terminado
	T-34: Modificar tarifa	Terminado
	T-35: Eliminar tarifa	Terminado
HU-10	T-36: Insertar nuevo tipo de tarifa	Terminado
	T-37: Modificar tipo de tarifa	Terminado
	T-38: Eliminar tipo de tarifa	Terminado

Tabla 22: Sprint Backlog final del Sprint 4

4.3.5 Sprint 5. Albaranes y partes de trabajo.

En el quinto sprint se desarrolla la gestión de los albaranes y los partes de trabajo. Estas dos clases tampoco se pueden eliminar ya que es muy importante tener el historial de los albaranes y los partes de trabajo realizados por los operarios. La tabla 23 muestra la configuración inicial del Sprint Backlog.

Historia de Usuario	Tarea	Estado
HU-11, HU-12	T-39: Diagrama de casos de uso	Pendiente
HU-11	T-40: Insertar nuevo albarán	Pendiente
	T-41: Buscar albarán	Pendiente
	T-42: Modificar albarán	Pendiente
HU-12	T-43: Insertar nuevo parte de trabajo	Pendiente
	T-44: Buscar parte de trabajo	Pendiente
	T-45: Modificar parte de trabajo	Pendiente

Tabla 23: Sprint Backlog inicial del Sprint 5

En la figura 7 se muestra el Diagrama de Casos de Uso (T-39) para la gestión de albaranes y partes de trabajo. El conjunto de tareas T-40 a T-45 incluye el diseño, la codificación de las clases y la realización de las pruebas pertinentes. Los detalles sobre la implementación y las pruebas se comentarán en otros capítulos. El sprint termina con una revisión del trabajo realizado para comprobar que se han cumplido las tareas comprometidas y con una demostración al cliente de las funcionalidades desarrolladas y completadas. La situación final del sprint se refleja en la tabla 24.

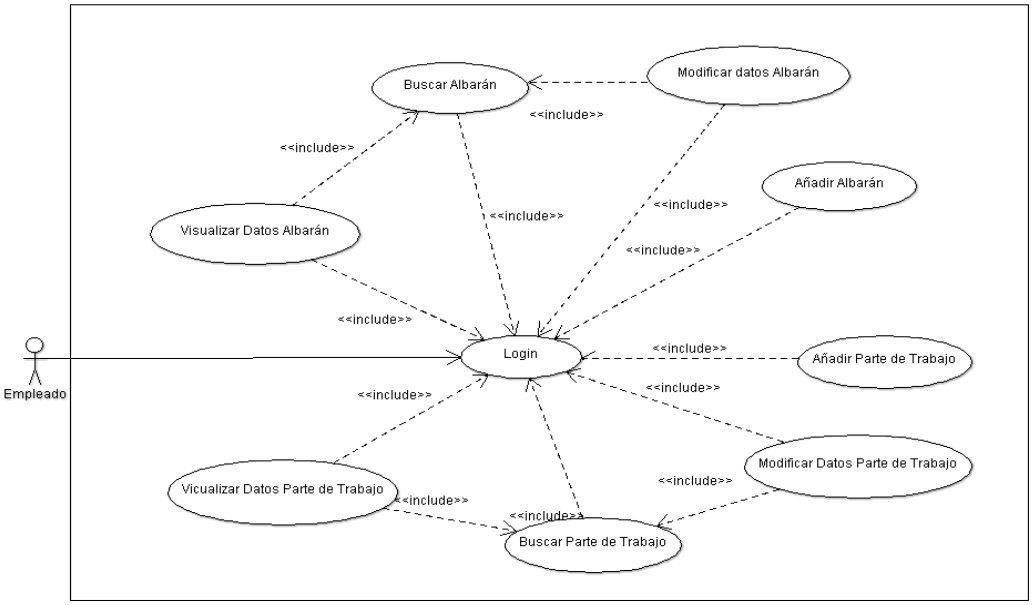


Figura 7: Diagrama de casos de uso para albaranes y partes de trabajo

Historia de Usuario	Tarea	Estado
HU-11, HU-12	T-39: Diagrama de casos de uso	Terminado
HU-11	T-40: Insertar nuevo albarán T-41: Buscar albarán T-42: Modificar albarán	Terminado Terminado Terminado
HU-12	T-43: Insertar nuevo parte de trabajo T-44: Buscar parte de trabajo T-45: Modificar parte de trabajo	Terminado Terminado Terminado

Tabla 24: Sprint Backlog final del Sprint 5

4.3.6 Sprint 6. Administrador

En el último sprint se desarrolla la gestión del administrador. Este módulo es muy sencillo ya que solo se encarga de crear la contraseña de acceso a la aplicación de los usuarios. El Sprint Backlog inicial para este sprint se muestra en la tabla 25.

Historia de Usuario	Tarea	Estado
HU-1	T-46: Diagrama de casos de uso T-47: Crear usuario T-48: Cambiar contraseña	Pendiente Pendiente Pendiente

Tabla 25: Sprint Backlog inicial del Sprint 6

En la figura 8 se muestra el Diagrama de Casos de Uso (T-46) para la gestión de albaranes y partes. Las tareas T-47 y T-48 incluyen el diseño, la codificación y las pruebas oportunas. Al final del sprint se realiza la revisión de tareas y la demostración al cliente.

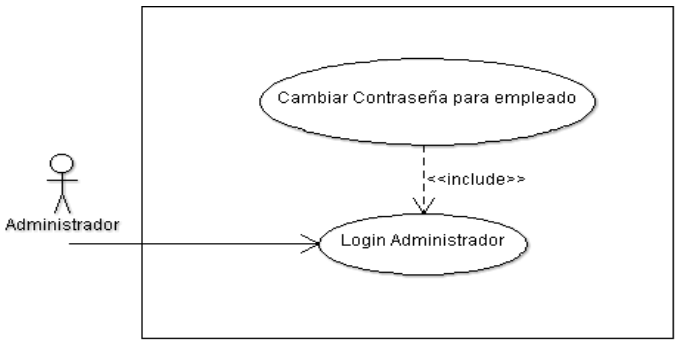


Figura 8: Diagrama de casos de uso para administrador

La tabla 26 muestra la situación final del sprint.

Historia de Usuario	Tarea	Estado
HU-1	T-46: Diagrama de casos de uso T-47: Crear usuario T-48: Cambiar contraseña	Terminado Terminado Terminado

Tabla 26: Sprint Backlog final del Sprint 6

4.4 *Valoración del desarrollo de los sprints*

El trabajo realizado durante los sprints ha resultado bastante positivo a la hora de desarrollar la aplicación, ya que permite tener controlado en todo momento el proyecto. En el primer sprint se produjo un problema en la estimación ya que finalmente no se pudieron completar todas las tareas que formaban las Historias de Usuario de este sprint, pero la situación pudo arreglarse en los sprints posteriores. Evidentemente, la aplicación tiene muchos módulos, pero funcionalmente son muy parecidos, por lo que una vez que se dispone de un material base se favorece el desarrollo de otros incrementos. Por otro lado, la división en sprints dedicados a módulos concretos permite que el cliente pueda obtener versiones completas de la aplicación que son completamente operativas.

4.5 *Diagrama de clases de la aplicación*

El diagrama de clases se ha dividido en varias partes. Primero se mostrará el paquete Vista, el cual contiene las entidades principales que componen la interfaz de usuario.

La figura 9 muestra el diagrama de clases UML que representa la estructura general del paquete Vista. Cabe destacar que el diagrama no muestra todos los atributos y métodos que contiene.

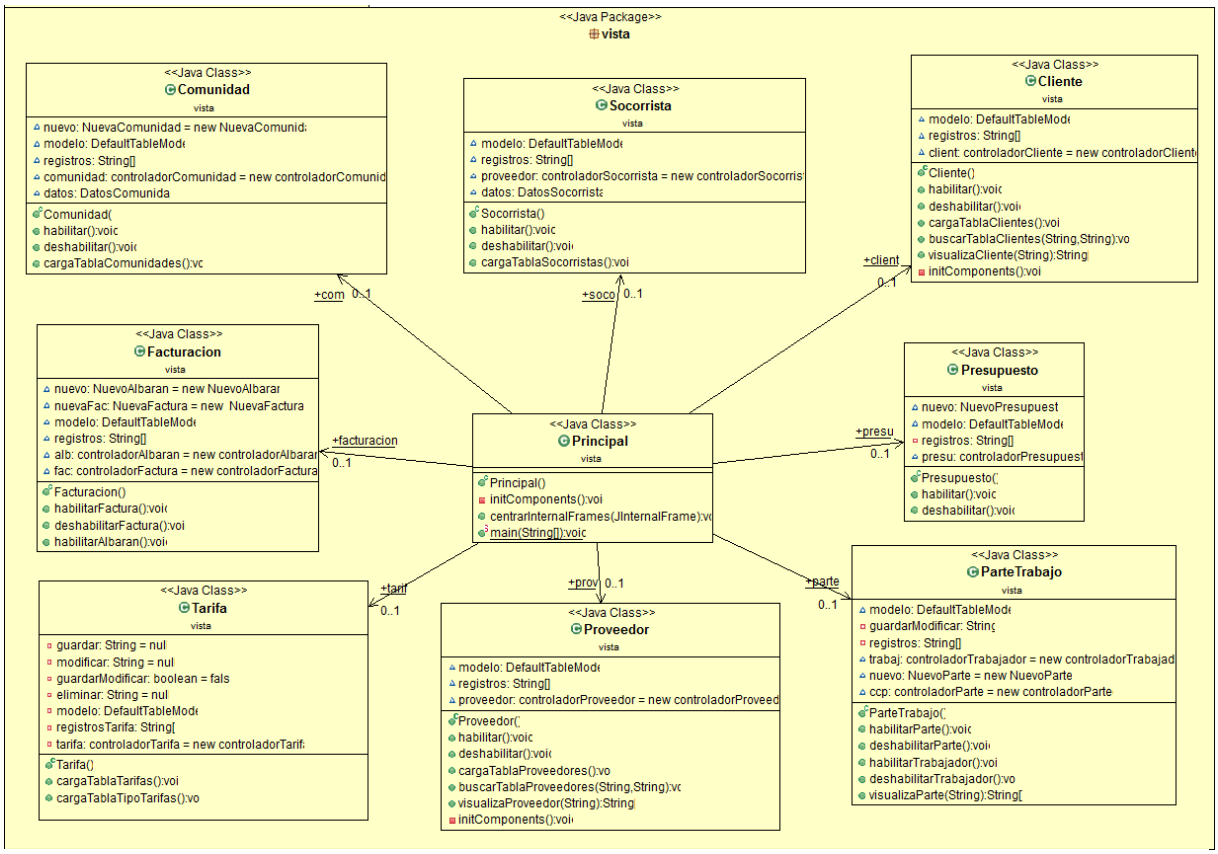


Figura 9: Diagrama de clases. Paquete Vista general.

La clase Principal es la que incluye el menú principal de la aplicación y es la encargada de crear cada objeto (entidad) que contiene, en este caso los objetos con los que se relaciona. Cada objeto que puede crear la clase principal está relacionado con objetos de su misma "clase".

En la figura 10 se puede observar un ejemplo de una de las entidades, la entidad cliente, la cual se relaciona con las clases que permiten visualizar y modificar los datos de cliente (DatosCliente) y dar de alta a un nuevo cliente (NuevoCliente).

La clase BuscarCliente es utilizada por las clases de nueva factura, nuevo presupuesto, nuevo parte de trabajo y nuevo albarán a la hora de añadir un cliente. En el caso de que este cliente no se encuentre en la base de datos, existe la posibilidad de crear uno nuevo.

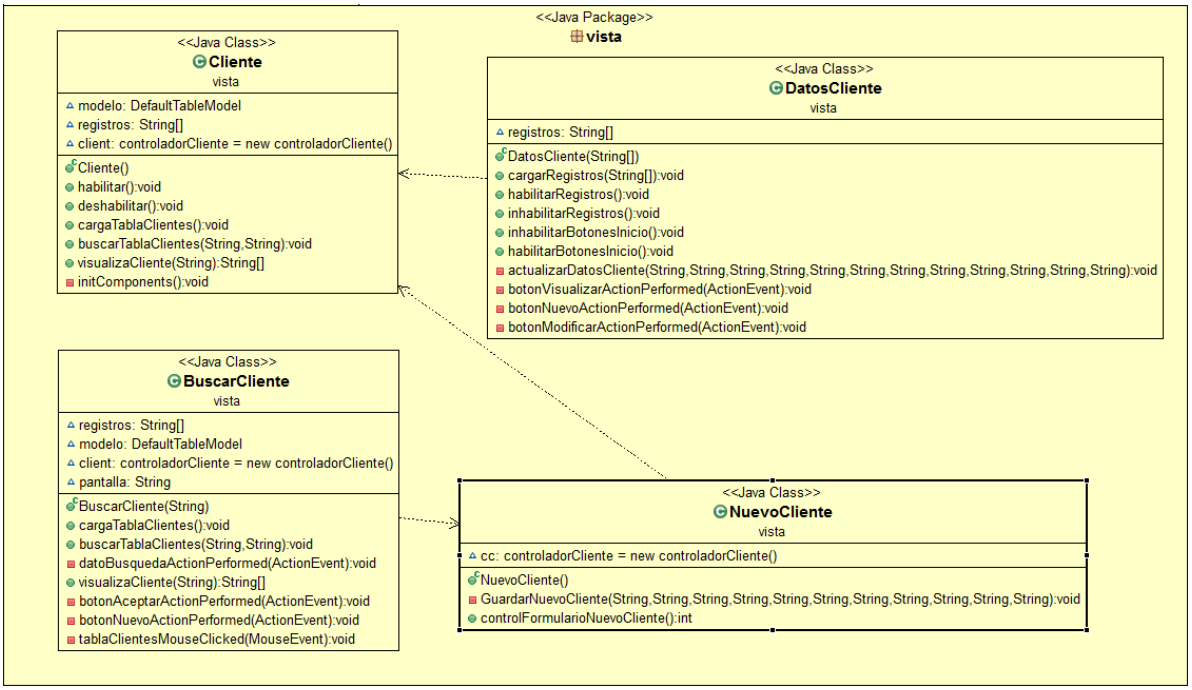


Figura 10: Diagrama de clases. Ejemplo clase cliente.

Como se explica posteriormente, se ha utilizado el modelo MVC para el diseño de la aplicación, por lo que cada clase está conectada con su propio Modelo y su propio Controlador, con los cuales la aplicación interactúa con la base de datos y con los datos que utiliza el usuario.

En las figuras 11 y 12 se muestra el diagrama de clases de los paquetes Modelo y Controlador donde se encuentran las clases antes mencionadas. Al igual que en la figura 9 no se muestran todos los atributos y métodos y en estos diagramas solo se muestran algunas de las clases.

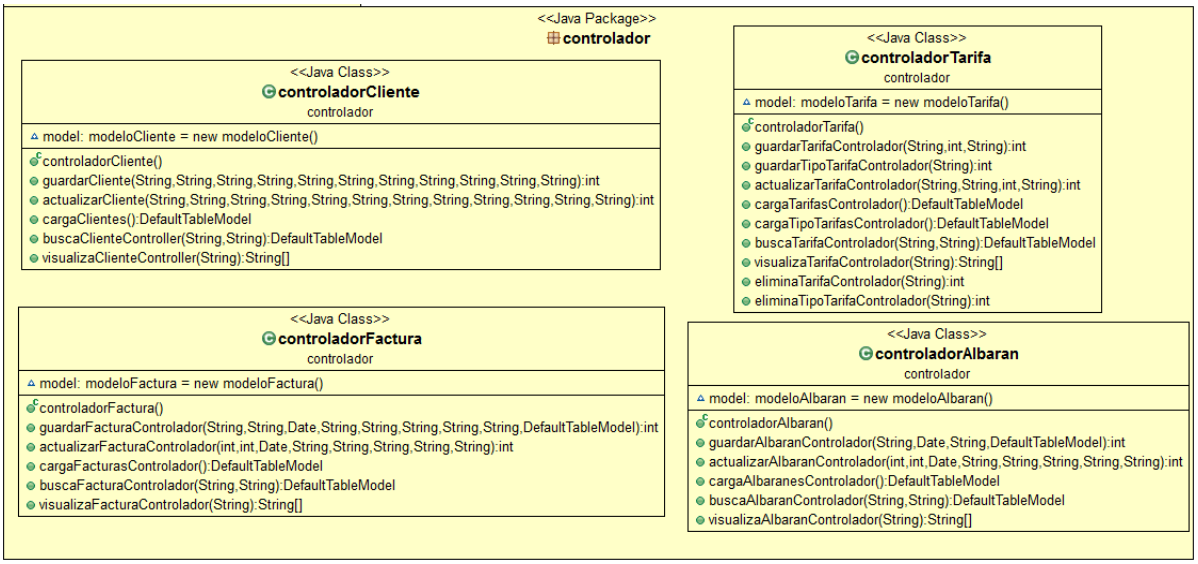


Figura 12: Diagrama de clases. Controlador

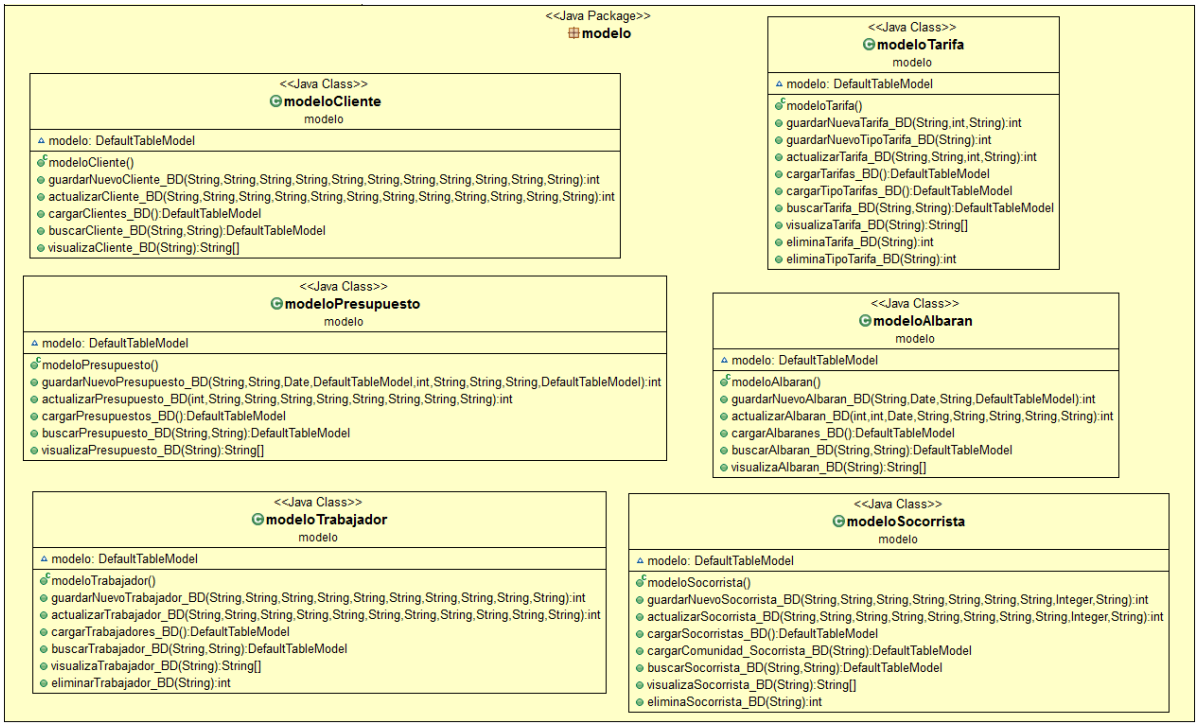


Figura 11: Diagrama de clases. Modelo

4.6 *Distribución de carpetas*

En el Anexo G se muestra como se han organizado las carpetas dentro del proyecto. Se pueden diferenciar los distintos paquetes existentes: reportes, modelo, controlador y vista.

4.7 *Diagrama de secuencia*

Para clarificar la secuencia de acciones que internamente realiza el sistema y que van desde la Vista hasta el Modelo y viceversa, se muestra en el Anexo F el diagrama de secuencia relativo a la inserción de un nuevo cliente, el cual se empleará como ejemplo.

4.8 *Diseño del modelo de la aplicación*

El modelo utilizado para el diseño de la aplicación es el Modelo-Vista-Controlador (MVC) [9]. Como su nombre indica, la aplicación está claramente separada en tres niveles: el Modelo, la Vista y el Controlador. La finalidad de este modelo es separar la interfaz de usuario (Vista), el tratamiento de los datos (Modelo) y la lógica que lleva a cabo la aplicación (Controlador). Esta división facilita que la aplicación permita poder actualizarse y mantenerse con gran facilidad, ya que si lo que se pretende es cambiar la apariencia de la aplicación solo se modificará el nivel de la Vista, si se quiere cambiar el almacenamiento de los datos se actualizará el nivel del Modelo y si lo que se quiere es cambiar el código para mejorar el rendimiento u otras características se modificaría el nivel del Controlador. La figura 13 muestra el modelo MVC y los siguientes apartados especifican las funciones de cada módulo.

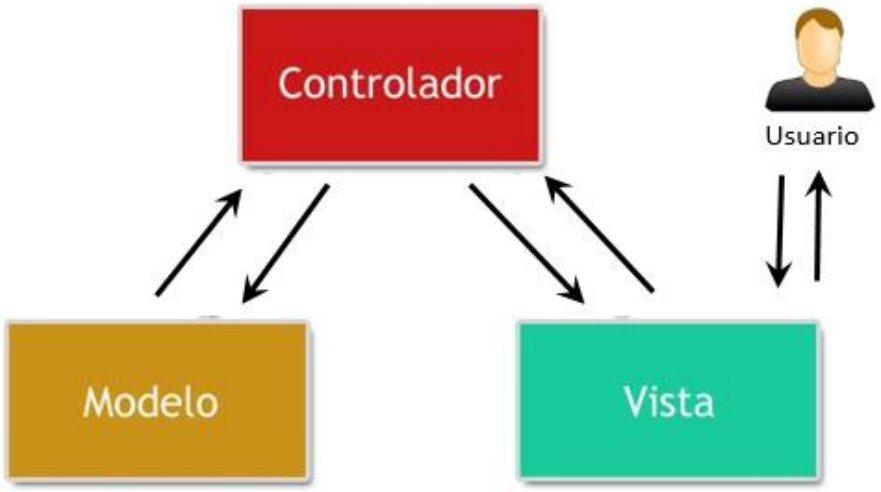


Figura 13: Estructura Modelo-Vista-Controlador

4.8.1 Modelo

El modelo es donde se gestiona toda la información con la que trabaja la aplicación. Se encarga de añadir, eliminar, modificar y buscar los datos oportunos. Es el lugar donde se guardan los datos de la vista o donde se solicitan los datos que se quieren enviar a la vista.

4.8.2 Vista

La vista es la parte visible de la aplicación, el módulo gracias al cual el usuario interactúa con los datos y es el elemento que permite al controlador extraer dichos datos para luego tratarlos.

4.8.3 Controlador

El controlador es el componente encargado de responder a los eventos que genera el usuario. Recibe los datos de la Vista, los trata y conecta con el Modelo para pasarle dichos datos. A su vez el controlador mandará a la Vista los datos para que sean mostrados al usuario.

4.8.4 Ejemplo del flujo de MVC en la aplicación. Insertando un nuevo cliente.

1. El empleado, a través de la interfaz (Vista) para la entidad cliente, inserta los datos de un nuevo cliente y a continuación pulsa guardar.
2. El punto de entrada de la aplicación recibe la información de la entidad y la acción solicitada (mediante la interfaz).
3. El Controlador correspondiente a la entidad cliente trata la información y se encarga de enviar los datos recogidos y tratados al Modelo.
4. El Modelo también correspondiente con la entidad cliente, se encarga de interactuar con la base de datos para guardar la información introducida por el empleado.
5. Por último, el Controlador vuelve a recoger los datos que ha tratado el Modelo y los envía a la Vista para actualizar la interfaz y que se muestre el nuevo cliente añadido.

4.9 *Arquitectura de la aplicación*

La arquitectura propuesta para la realización y cumplimiento de los requisitos y funcionalidades del cliente corresponde al modelo Cliente-Servidor. La interacción Cliente-Servidor es el soporte de la mayor parte de la comunicación por redes [10] [11]. Esta arquitectura consiste básicamente en:

- Un cliente trabajando en su ordenador local se conecta con un servidor para pedirle cierta información.
- El servidor devuelve la información solicitada al cliente.

Como se puede observar en la Figura 14 un único servidor puede servir a muchos clientes, y esto ahorra a cada uno de ellos el problema de tener información almacenada localmente. Además, lo habitual es que la mayoría del trabajo complicado y pesado lo realice el servidor, mientras que los clientes solo se ocupan de interactuar con el usuario.

Figura 14: Esquema de la arquitectura cliente-servidor

La arquitectura para el sistema Hidrosierra Sport S.L. está formada por:

- Varios clientes (instancias de la aplicación en los ordenadores de los empleados) que envían peticiones.
- Un servidor NAS que recibe y responde a las peticiones de los clientes. En el servidor se aloja una Base de Datos relacional en la que se almacenan los datos de la aplicación.

4.9.1 Cliente

En este caso, la aplicación, que se encuentra alojada en el ordenador del usuario, actúa como cliente realizando peticiones en forma de consultas y actualizaciones a una Base de Datos. Es importante saber que la ubicación de los datos es totalmente transparente para el cliente. El proceso del cliente que interactúa con el usuario recibe el nombre de Front-End.

4.9.2 Servidor

La Base de Datos está alojada en un servidor. Concretamente, la arquitectura para el sistema Hidrosierra Sport S.L cuenta para esta tarea con un NAS. La función del servidor será atender y procesar las distintas peticiones que realizan los clientes. Este proceso recibe el nombre de Back-End.

4.9.3 Funcionamiento cliente/servidor de la aplicación

La idea general es que el Front-End sea el responsable de recoger los datos de entrada del usuario, en este caso mediante los formularios de datos, y de transformarlos ajustándolos a las especificaciones que demanda el Back-End para poder procesarlos, devolviendo generalmente una respuesta que el Front-End recibe, interpreta y muestra al usuario de una forma que éste lo pueda entender.

5 Detalles de la Implementación

5.1 *Base de datos*

En este apartado del capítulo, primero se va a explicar el esquema relacional de la Base de Datos y después se mostrará con detalle cada tabla que compone la Base de Datos y los datos que se almacenan en ella.

La implementación de la base de datos trata de satisfacer los siguientes objetivos:

- Independencia Física.
  - Los usuarios pueden acceder a los datos sin ningún tipo de problema ante cualquier cambio en el almacenamiento físico.
- Independencia Lógica.
  - Los usuarios pueden operar con los datos de la aplicación añadiendo, modificando y borrando objetos de la base de datos, sin que estas operaciones interfieran con otros usuarios que pueden acceder a subconjuntos parciales de los mismos (vistas).
- Flexibilidad.
  - Poder presentar a los empleados los datos en la forma que ellos prefieren.
- Uniformidad.
  - Las estructuras lógicas de los datos presentan un aspecto semejante (tablas), lo que facilita la manipulación de la Base de Datos.
- Sencillez.
  - Según las características anteriores y mediante un lenguaje sencillo, como SQL, se produce como resultado que el Modelo de Datos Relacional sea fácil de comprender y de utilizar.

La Base de Datos para la aplicación Hidrosierra Sport S.L. contiene 18 tablas que almacenan toda la información de la empresa. En el Anexo A se muestra el esquema relacional de la base de datos en detalle, incluyendo todos los campos que contienen las tablas así como las claves primarias y foráneas. En todas las tablas se ha creado un campo identificador que va a ser siempre único como clave primaria. Este identificador, aunque es totalmente transparente para el usuario que utiliza la aplicación a la hora de operar con los datos, sirve para saber el número de partes, facturas, albaranes y presupuestos que va utilizando la empresa, ya que siempre es único y auto incrementable cada vez que se inserta una entidad.

Se utilizan tablas de relación N a M, o muchos a muchos. Estas clases de relaciones ocurren cuando una fila de la primera tabla puede estar relacionada con varias filas de la segunda tabla y una fila de la segunda tabla puede estarlo con varias filas de la primera. Un ejemplo de este tipo lo tenemos en la relación entre la tabla socorrista y la tabla comunidad porque, dada una comunidad en particular, se deben conocer todos los socorristas que han trabajado en ella, y, viceversa, dado un socorrista, se tiene que tener constancia de las comunidades en las que ha trabajado. En el Anexo A se muestra el Esquema Relacional y en el Anexo B se muestran todas las tablas de la aplicación y se indican con detalle los campos que las componen indicando el nombre de la tabla, una descripción de la misma y la lista de sus atributos (campos).

5.2 *Interfaz de usuario*

Para desarrollar la interfaz se ha utilizado la herramienta Java Swing. Java Swing es un paquete de librerías que proporciona herramientas o facilidades al usuario para construir GUI`s o interfaces gráficas, con las que el usuario final que utilizará la aplicación interactuará con el sistema. Con Java Swing hay muchas posibilidades para estructurar el proyecto, según las necesidades. La jerarquía de componentes se muestra en la figura 15.

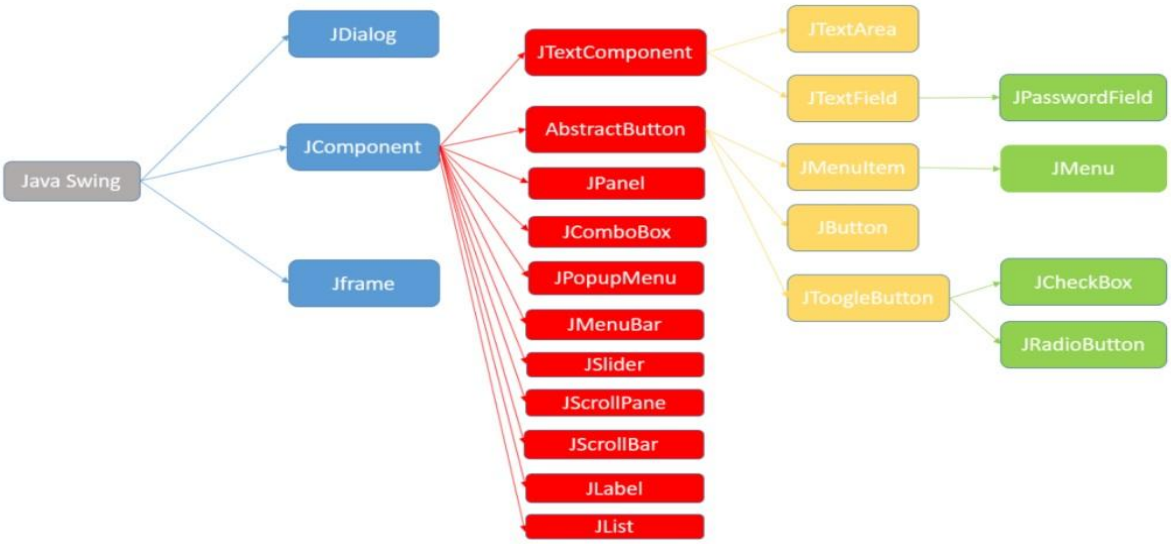


Figura 15: Componentes Swing

Para comenzar a desarrollar la interfaz debemos saber que se necesita un contenedor como base donde poder introducir todos los componentes que se quieren mostrar por pantalla. Por cada pantalla se ha creado un *JFrame* que se usa como contenedor y dentro de él se añaden los paneles, botones, cajas de texto, áreas de texto, etc.

Se compone de un *Jframe* (toda la pantalla) y de un gran *Jpanel* dentro del *Jframe* que se divide en dos partes. Una de ellas es la parte superior, donde se encuentran los *Jbutton* que conducen a los módulos que componen la aplicación. La segunda es el *JdesktopPanel*,

el cual tiene integrado el logo de la empresa y es donde irán apareciendo los diferentes *InternalJFrame* (las ventanas) que se ejecutan dentro de un *JdesktopPanel*.

Tanto *JFrame* como *Jpanel* se refieren a contenedores. Cada contenedor tiene asociado un objeto que es el que se encarga de ordenar los componentes. Este objeto se llama *Layout*, que es una clase que decide en qué posición van los botones y el resto de componentes, cuáles van alineados, cuales se hacen grandes al agrandar una ventana, etc.

En el lenguaje Java existen varios tipos de *layouts* disponibles, pero el utilizado aquí es el *GroupLayout*. El componente *GroupLayout* es de los más utilizados ya que permite establecer de una manera precisa la posición y el tamaño de cada uno de los componentes dentro del contenedor. Esta clase necesita de los métodos *setHorizontalGroup* y *setVerticalGroup* para definir la posición y el tamaño de los componentes.

Con la herramienta IDE NetBeans la forma de posicionar los componentes se resuelve de manera más o menos sencilla, ya que esta herramienta permite al usuario arrastrar e introducir los componentes dentro de las pantallas a través de una vista previa para el desarrollo de la interface.

La página principal de Hidrosierra Sport S.L. se estructura como muestra la figura 16.

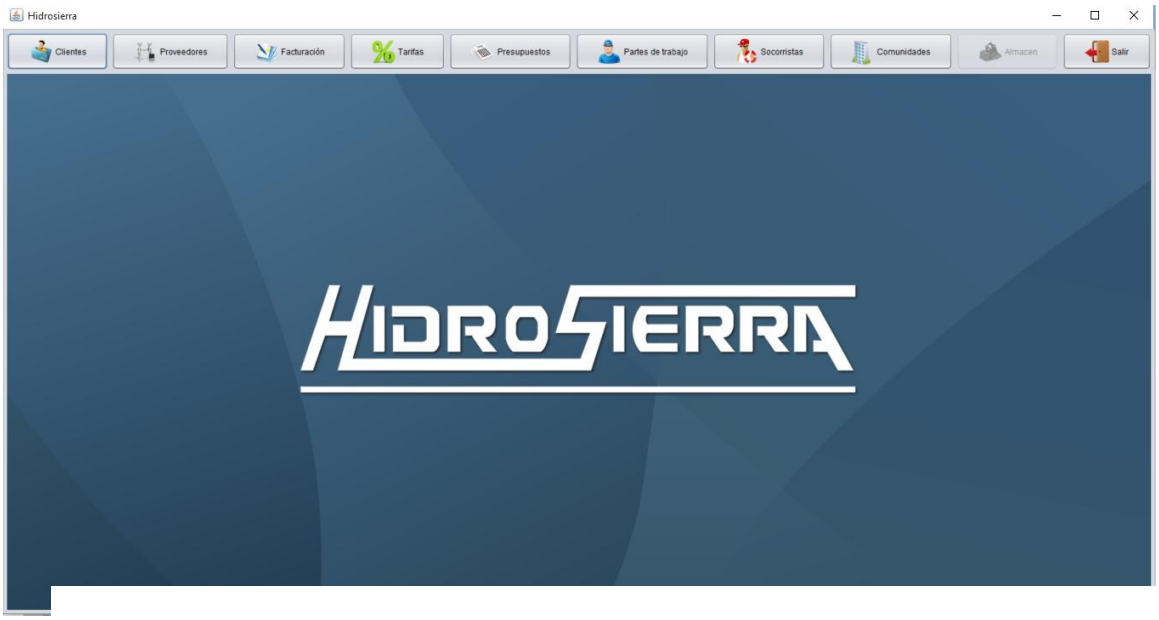


Figura 15: Página principal de la aplicación

En la figura 17 se muestra el ejemplo del módulo Tarifas (*JInternalFrame*) donde se puede ver cómo está integrado dentro de un *JdesktopPanel*.

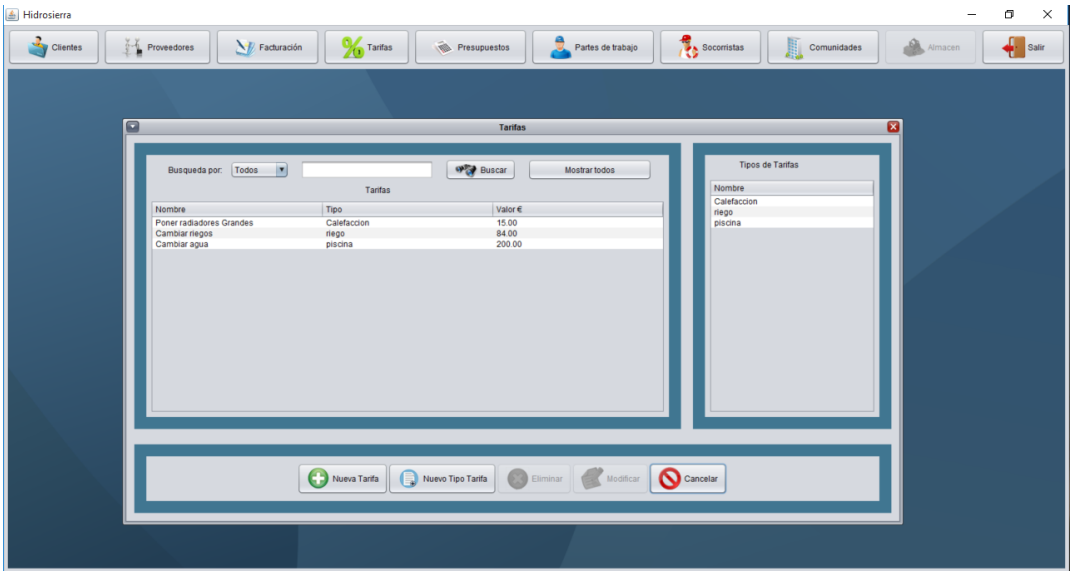


Figura 16: Ejemplo de JInternalFrame dentro de un JDesktopPanel.

Tras mostrar la estructura de la página principal de la aplicación y la inclusión de un *JInternalFrame* en un *JDesktopPanel* se incluyen algunas otras pantallas. Se ha elegido mostrar sólo el módulo de los clientes debido a que los demás módulos son muy similares a la hora de trabajar con listados, buscar datos, introducir datos y visualizar y modificar dichos datos.

Modulo Clientes: Listado de los clientes.

En la figura 18 se muestra la ventana principal de los clientes. En ella aparece una lista de todos los clientes que se almacenan en la Base de Datos ordenados alfabéticamente. A través de esta pantalla se podrá insertar un nuevo cliente, buscar un cliente o visualizar los datos de un cliente pulsando sobre un cliente y dando al botón visualizar o clicando dos veces en un cliente.

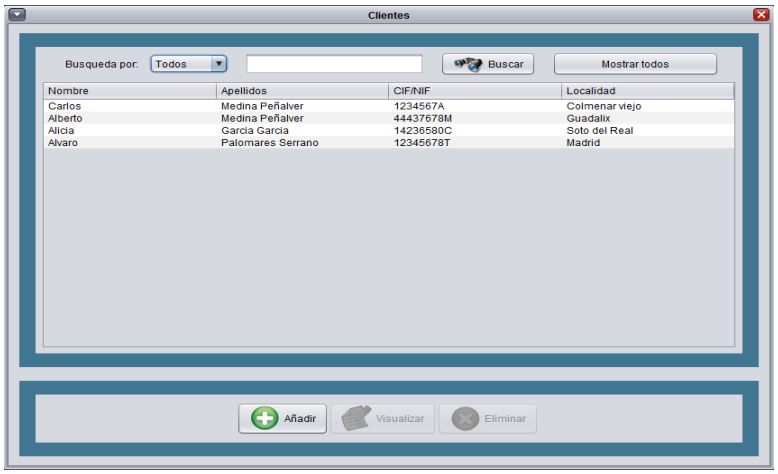


Figura 17: Listado de clientes

Modulo Clientes: Añadir nuevo cliente.

La siguiente figura muestra la pantalla para la inserción de un nuevo cliente. Se debe seleccionar mediante un *checkbox* si el cliente es una persona o una empresa, para diferenciar algunos datos a introducir en la Base de Datos. Lo más destacable es que los campos obligatorios están marcados con un asterisco rojo. En el caso de no seleccionar el *checkbox* o de dejar en blanco alguno de los campos obligatorios saltaría un mensaje informando al usuario.

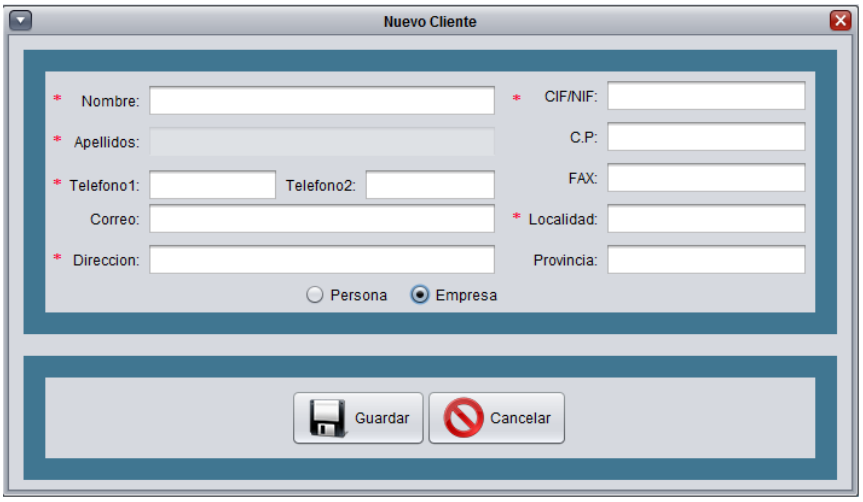


Figura 18: Inserción de un nuevo cliente.

Modulo Clientes: Búsqueda de clientes.

La opción de búsqueda de clientes es muy sencilla. Ofrece la posibilidad de buscar por diferentes filtros: nombre, apellidos, cif/nif, localidad o por todos los campos. En este caso se muestra en la figura 20 una búsqueda en la que sin ningún filtro, introduciendo en el campo de búsqueda un apellido y pulsando *enter*, o pulsando el botón *buscar*, el sistema encuentra dos personas con dicho apellido.

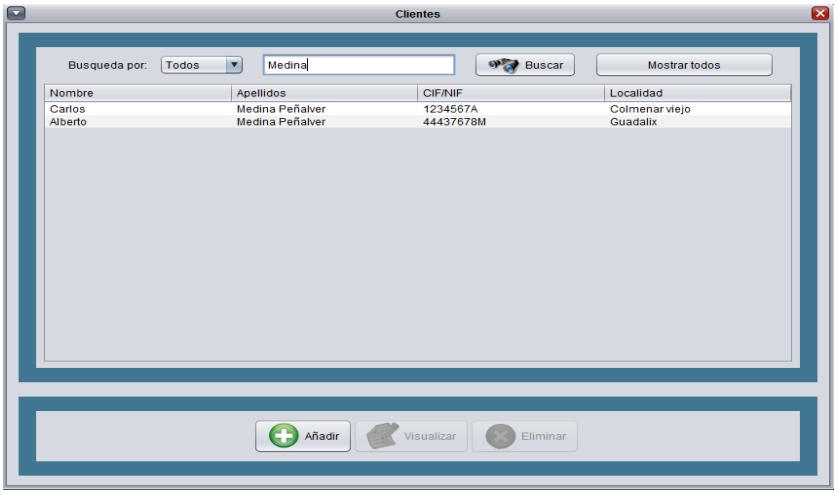


Figura 19: Búsqueda de clientes.

Modulo Clientes: Visualizar datos de clientes.

La siguiente figura muestra una ventana con los datos del cliente elegido. Estos datos no se podrán editar. Para editarlos se proporciona la opción de modificar los datos.

Figura 20: Visualizar datos de un cliente

Modulo Clientes: Modificar datos de clientes.

Esta opción permite modificar los datos del cliente, siempre respetando los campos obligatorios. Para actualizar los datos hay que pulsar el botón guardar.

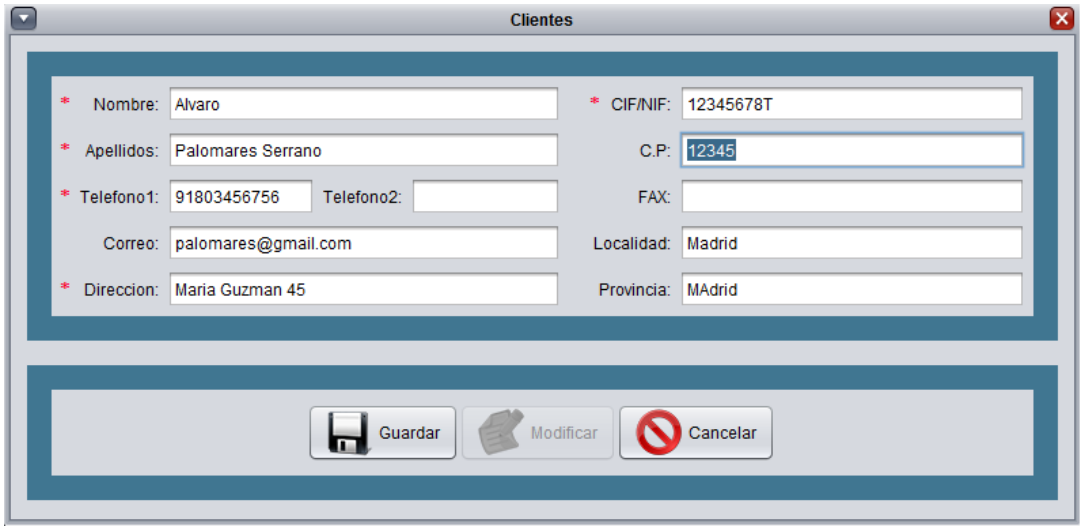


Figura 21: Modificar datos de un cliente.

5.3 *Implementación de los modelos más destacados*

5.3.1 Diseño de reportes con JasperReports

Para la creación de las facturas, presupuestos y albaranes se ha utilizado la herramienta *JasperReports*. JasperReports es un API de código abierto que permite desarrollar reportes tanto en web como en aplicaciones de escritorio en Java. Ofrece la posibilidad de generar reportes en formatos PDF, XML, HTML, CSV, XLS, RTF, y TXT con solo unas cuantas líneas de código. También se pueden guardar estos reportes en un archivo o mostrarlos directamente con un visor siempre y cuando se esté usando una aplicación Swing.

Por otro lado, JasperReports proporciona un editor con el que permite crear la parte visual de los reportes (posición de los textos, imágenes, tipos de letra, estilos, etc.). Este componente se denomina *iReport*.

Para poder utilizar JasperReports ha sido necesario instalar varios plugins en la herramienta IDE NetBeans. En la figura 23 se muestra como se utiliza JasperReports una vez instalado en NetBeans. Gracias al editor *iReport* se puede ir creando el reporte de una manera muy sencilla, arrastrando los elementos directamente desde la barra de paletas a nuestro diseño. Una vez se tiene el diseño, el siguiente paso es elegir los datos que se pretenden mostrar en el reporte. Como se puede observar, en la parte de la izquierda de la figura 23, en el *Report Inspector* tenemos los campos que queremos tras haber realizado la *Query* correspondiente para mostrar campos. El propio editor es el que ofrece la posibilidad de realizar una Query, ya sea escribiendo la consulta o seleccionando los campos que se desean de una manera visual, encargándose la herramienta de crear la consulta.

En el Anexo C se muestra un ejemplo sobre cómo se realiza una Query para mostrar los campos que necesita un reporte. Mientras que en el anexo D se muestra un ejemplo de un presupuesto realizado con JasperReports.

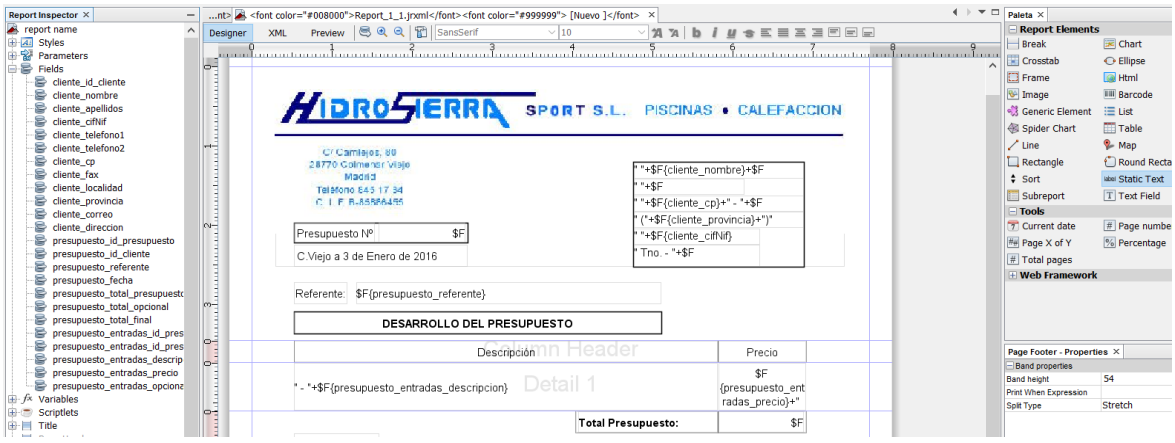


Figura 22: Ejemplo de creación reporte.

6 Validación y Verificación

6.1 *Introducción*

La validación y la verificación involucran un conjunto de procedimientos, técnicas y herramientas que se utilizan para asegurar que un producto software cumple con los requisitos especificados al inicio del proceso de desarrollo (validación) y que es conforme a las especificaciones y normas internas de garantía de calidad (verificación).

6.2 *Estrategias de validación y verificación utilizadas*

Como se ha explicado, la finalidad de la validación y la verificación consiste en comprobar si se ha implementado correctamente, tanto la funcionalidad cómo la propia aplicación. Para ello se definen una serie de pruebas que comprenden, las pruebas unitarias de caja blanca, pruebas unitarias de caja negra, pruebas de integración, pruebas de interfaz de usuario, pruebas de compatibilidad y pruebas de validación.

El objetivo de todas estas pruebas es encontrar el mayor número de errores posibles. Cabe destacar, que gracias a la metodología SCRUM utilizada, el usuario ha tenido una participación muy activa durante el proceso y esto ha supuesto que muchas de las pruebas se hayan ido realizando contando con su colaboración como parte de los sprints. Además, la utilización de esta metodología supone entregar funcionalidades completas al terminar el sprint, lo que facilita la realimentación continua y progresiva del usuario final.

6.2.1 Desarrollo de las pruebas

A continuación se detallan las técnicas que se han utilizado para realizar las pruebas de verificación y validación. Se especifica cómo se han realizado dichas pruebas y los resultados obtenidos. En el Anexo E se detallan las pruebas realizadas atendiendo a los requisitos.

6.2.1.1 Pruebas estáticas: inspección de código

Las llamadas pruebas de inspección de código son las primeras pruebas que se realizan durante el desarrollo. Consisten en la lectura del código implementado con el fin de localizar algún error. Este tipo de pruebas se ha realizado en todos los módulos, concretamente en aquellos que se encargan de recibir los datos del usuario, tratarlos y guardarlos en la base de datos comprobando que sean correctos.

Algunas preguntas típicas en la comprobación de código son las siguientes:

- ¿Las llamadas a funciones y métodos tienen el número correcto de parámetros?
- ¿Concuerdan los tipos de los parámetros formales y reales?
- ¿Están los parámetros en el orden adecuado?
- ¿Se utilizan los resultados de las funciones?
- ¿Existen funciones o procedimientos no invocados?

6.2.1.2 Pruebas unitarias de caja negra

Las pruebas de caja negra son aquellas que se realizan sobre los componentes de la aplicación sin tener en cuenta la estructura interna del programa, solo teniendo en cuenta la entrada y la salida. Su objetivo es probar los requisitos funcionales. Las pruebas se han realizado sobre la conexión de la base de datos, la creación de facturas, presupuestos, albaranes, clientes, trabajadores, socorristas, comunidades, proveedores y tarifas.

Para llevar a cabo estas pruebas se han definido casos de pruebas válidas e inválidas para cada método a probar, es decir, se han definido tanto casos en los que la salida esperada era la correcta como en el caso contrario.

6.2.1.3 Pruebas unitarias de caja blanca

Las pruebas de caja blanca son aquellas que se ejecutan teniendo en cuenta la estructura interna de cada método. Estas pruebas tienen como objetivo la ejecución de todas las posibilidades existentes en el código. En este caso se han realizado pruebas de caja blanca sobre los algoritmos encargados de la lectura de los datos y sobre los encargados de tratarlos y enviarlos a la base de datos, con el fin de minimizar el número de errores que puedan surgir a la hora de introducir información en la aplicación.

6.2.1.4 Pruebas de integración

Las pruebas de integración se realizan sobre todos los componentes implementados para comprobar su correcto acoplamiento. Estas pruebas sirven para comprobar, por ejemplo, que todas las clases están bien relacionadas y que se puede acceder desde una clase a los métodos de otra clase. Lo dicho tiene sentido para clases que tengan algún tipo de relación.

6.2.1.5 Pruebas de interfaz de usuario

Las pruebas de interfaz se realizan mediante la verificación y validación por parte del usuario. Como ya se ha mencionado antes, el usuario tiene un papel muy relevante en el desarrollo de la aplicación cuando se emplean metodologías ágiles y desde el principio se realizan pruebas de contenido contando con su participación, es decir, en cada entrega el usuario debe determinar si la interfaz o los cambios en la interfaz incluyen todo aquello que necesita. Entre otras cosas se comprueba si se muestran en la interfaz todos los datos

requeridos de una forma intuitiva y visual, además de los todos los elementos como son las tablas, los botones, etc.

Se deben comprobar todas las pantallas de la aplicación con el objetivo de valorar si la implementación es la deseada por el usuario. Esta labor se facilita mucho con la metodología SCRUM ya que el sprint supone entregar un incremento con funcionalidad reducida, lo que implica un menor número de componentes a validar en cada ocasión.

6.2.1.6 Pruebas de compatibilidad

La aplicación debe ser diseñada y ajustada para utilizar cualquier tipo de pantalla y resolución, por lo que se realizan pruebas con diferentes resoluciones de pantalla para comprobar que la aplicación es capaz de ajustarse a cualquier tipo de resolución.

6.2.1.7 Pruebas de validación

Para la validación de la aplicación se comprueba cada uno de los requisitos desde el punto de vista del usuario final, en este caso los empleados de la empresa Hidrosierra Sport S.L. que van a utilizar la aplicación. Este proceso se realiza primero haciendo comprobaciones por parte del desarrollador de la aplicación, y después contando con la colaboración de los empleados.

La validación se basa en comprobar requisito a requisito la aplicación. Como ya se ha comentado, esta validación se ha ido realizando en cada sprint. Al finalizar el sprint los empleados de la empresa comprueban el sistema para ver que cumple con los requerimientos acordados para el incremento tratado en el sprint.

A continuación se muestra algún ejemplo de validación hecha por los usuarios en los módulos de clientes, presupuestos y login para comprobar la inserción de los datos en la aplicación. En la figura 24 se muestra las pruebas realizadas al introducir datos a la hora de crear un nuevo cliente. Es obligatorio rellenar los campos requeridos, ya que en caso contrario la aplicación mostrará un mensaje informando al usuario de que tiene que rellenar dichos campos, tal y como se muestra en la figura.

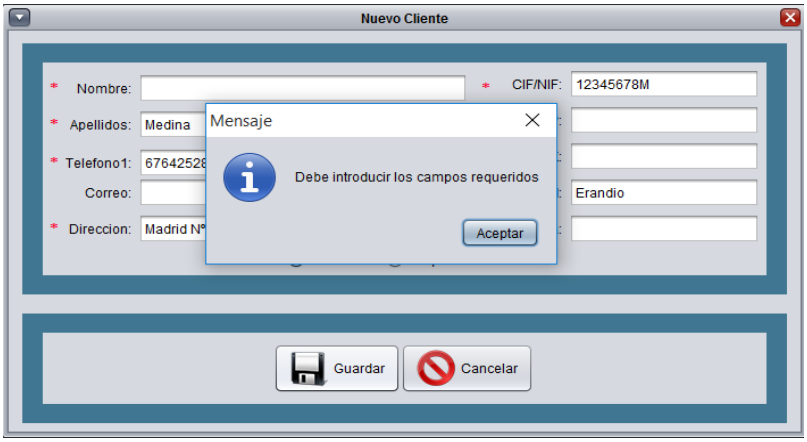


Figura 23: Control de formulario. Añadir nuevo cliente.

En la siguiente prueba se comprueba la respuesta del sistema cuando se busca un cliente y éste no se encuentra en la Base de Datos. El sistema informa al usuario que no ha encontrado resultados.

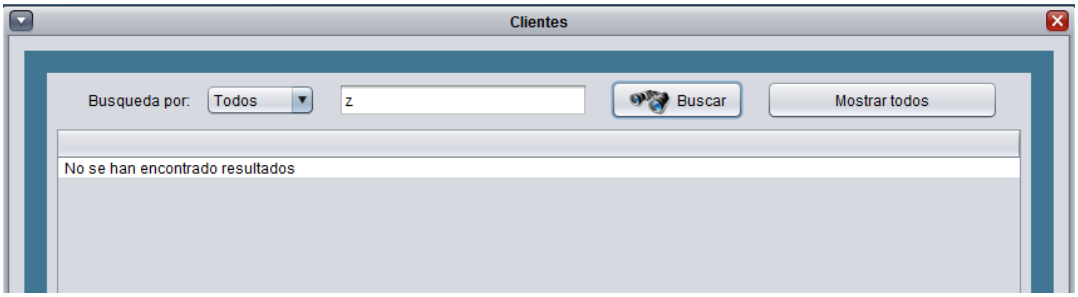


Figura 24: Control de formulario. Búsqueda de clientes.

Se verifica el control de los usuarios para entrar en la aplicación. El administrador crea una contraseña para los empleados. En este caso no es necesario un nombre y contraseña para cada empleado ya que no se va a llevar ningún tipo de control o historial. Cuando un usuario introduce mal la contraseña, el sistema avisa con un mensaje para que el usuario vuelva a introducirla. Las figuras 26 y 27 muestran los resultados

Figura 26: Login.

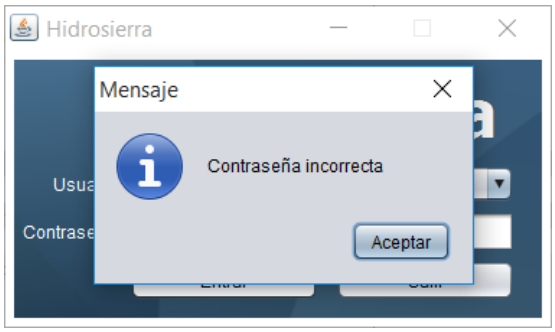


Figura 27: Error contraseña.

En el módulo de presupuestos se realizan verificaciones similares respecto a los otros módulos. En este caso se añade un control en el formulario para incluir un cliente, debido a que esta condición es necesaria para la creación del presupuesto. Además es obligatorio rellenar la tabla del desarrollo del presupuesto y todos sus campos. Estas situaciones se representan en las figuras 28 y 29.

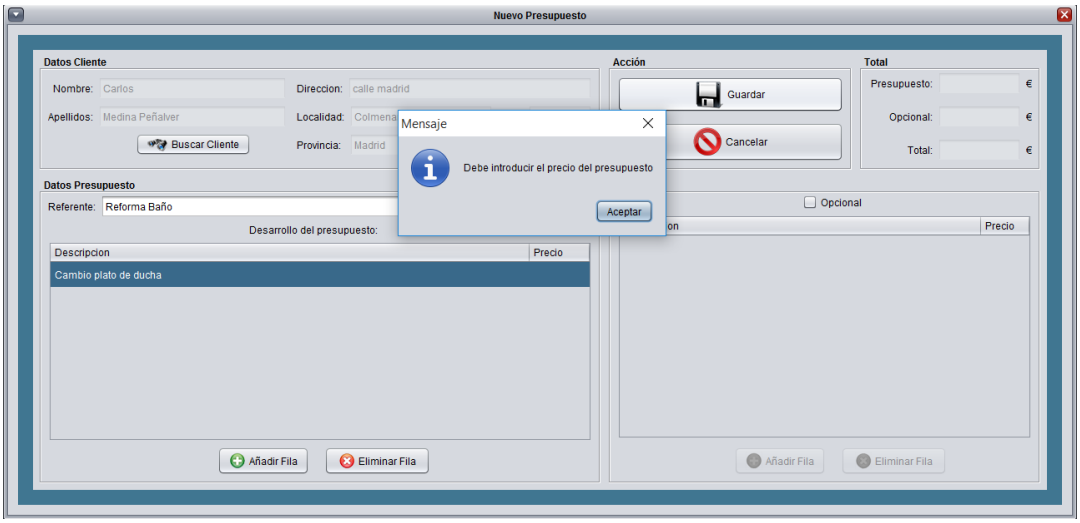


Figura 25: Control de formulario. Error al introducir datos en el presupuesto.

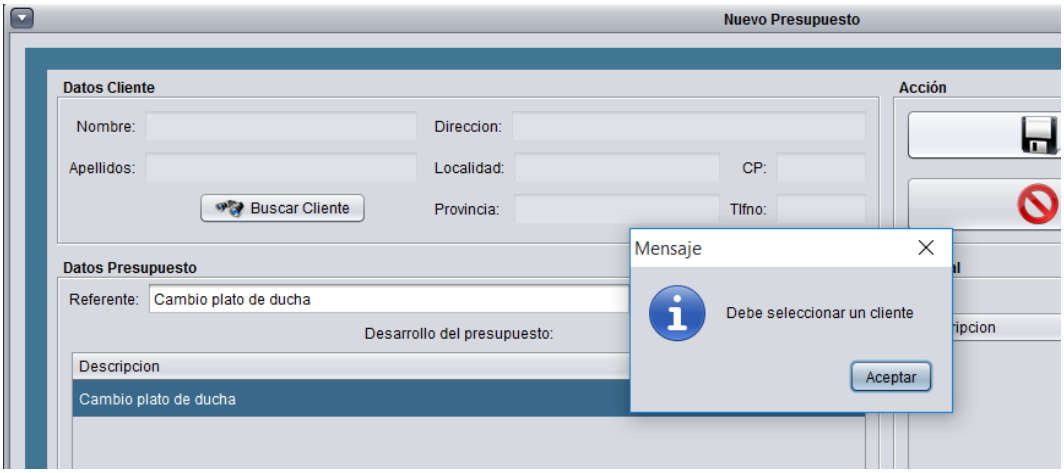


Figura 26: Control de formulario. Error al crear presupuesto.

7 Evaluación

7.1 *Evaluación de los usuarios*

Este proyecto ha sido evaluado por distintos tipos de usuarios tras su implementación y validación. Por una parte han sido los propios empleados de la empresa Hidrosierra Sport S.L., y por otra parte, han participado tanto usuarios que tenían conocimientos de informática como algunos usuarios que no tenían dichos conocimientos.

No se ha elaborado ningún manual de usuario porque el tipo de usuario hacia el que está destinada la aplicación ha trabajado durante todo el proyecto en cada iteración junto al desarrollador, lo cual es característico de las metodologías ágiles, y por tanto conoce perfectamente el uso de la aplicación. Por este motivo los empleados, tras la evaluación final, han quedado muy satisfechos con el resultado.

En líneas generales, el resto de usuarios que han probado la aplicación han destacado que cumple con los requisitos exigidos y con lo que una persona ajena al proyecto espera encontrarse cuando se enfrenta por primera vez a la aplicación. Han valorado de forma positiva el diseño de la interfaz de usuario y la gran facilidad que se ofrece al usuario para interactuar con la aplicación. De manera sencilla e intuitiva realizaron pruebas como inserción de nuevos clientes, visualización de sus datos, pruebas con las facturas, presupuestos, etc. En el ámbito de la metodología cabe destacar, como anécdota de la gran capacidad de adaptación de SCRUM, que en uno de los sprint hubo algún usuario que informó sobre el funcionamiento de la tecla cancelar, alegando inicialmente que no entendía bien su uso o que al menos a ese usuario no le parecía del todo intuitiva. El botón cancelar, en un principio estaba diseñado para limpiar cualquier operación que se hubiera quedado a medio ejecutar o que al final no se deseara ejecutar, como por ejemplo a la hora de modificar los datos de alguna entidad.Teniendo en cuenta que los empleados son los que tienen que estar familiarizados con la interfaz, esta tiene que ser totalmente diseñada según sus preferencias, por lo que se procedió a cambiar el uso del botón cancelar dándole la misma funcionalidad que el botón X para cerrar una ventana.

8 Conclusiones y Líneas Futuras

8.1 *Conclusiones*

Este Trabajo de Fin de Grado ha consistido en el análisis, diseño y desarrollo de una aplicación de gestión empresarial para una PYME. En su realización se han podido poner en práctica todas las técnicas y métodos aprendidos durante la carrera, mediante la realización de un proceso de desarrollo software completo. Se han cubierto especialmente algunos procesos de ingeniería del software, tecnología de bases de datos y programación en el lenguaje Java.

La principal aportación metodológica del trabajo ha sido la aplicación de una metodología muy actual, como es la metodología SCRUM, perteneciente a las denominadas metodologías ágiles. Este tipo de metodologías establecen un marco que favorece la flexibilidad del proceso de desarrollo y la rápida adaptación a cambios en el entorno, problemas muy frecuentes en metodologías tradicionales. En el caso particular de esta aplicación, el uso de SCRUM ha supuesto las siguientes ventajas:

1. Se ha conseguido una división sencilla de los módulos o subsistemas que componen la aplicación, debido a que la descomposición del producto en incrementos e iteraciones facilita la visión de cada componente. Además, el cliente determina en todo momento los incrementos que considera prioritarios.
2. Los elementos propios de SCRUM, como el Product Backlog, el Sprint Backlog o el Tablero de Tareas (fusionado aquí con el anterior), permiten tener siempre claro qué es lo que se ha hecho y lo que falta por hacer, ayudando a mejorar la organización del proyecto.
3. A pesar de tratarse de un proyecto con un solo desarrollador, para el que la metodología no está pensada inicialmente, la adaptación de su filosofía y proceso han contribuido notablemente a que los problemas detectados al comienzo del proyecto, sobre todo con la estimación, se resolvieran satisfactoriamente en el primer sprint. A nivel personal se ha facilitado tener un control de lo que se ha hecho y lo que falta.
4. La incorporación del usuario que promueven las metodologías ágiles es muy importante de cara a resolver problemas en pequeña escala, es decir, el incremento de un sprint es de un tamaño más manejable que la aplicación completa, y el hecho de que el usuario se involucre en su revisión y participe en el sprint facilita la resolución de incidencias, al tiempo que se modela la aplicación al gusto del propio cliente.

Tras el desarrollo de este proyecto resulta muy satisfactorio mirar atrás y ver como desde una idea se ha llegado a la creación de la aplicación. Partiendo de pequeñas reuniones con el cliente con el único objetivo de intentar identificar sus necesidades, hasta un exhaustivo análisis y diseño que ha llevado al completo desarrollo de la herramienta. Es

por este motivo que se considera que se han cumplido con todos y cada uno de los objetivos que se marcaron al principio de este proyecto. El principal, poder satisfacer las necesidades de un cliente real y poder entregarle una herramienta que cumpla con su cometido y ayude a mejorar y facilitar todos los procesos que realiza la empresa. El secundario, la posibilidad de profundizar en una metodología que se encuentra actualmente en expansión.

8.2 *Trabajo futuro*

El trabajo que se espera realizar sobre la herramienta en el futuro se centraría en poder ampliar sus funcionalidades. Se procedería a la creación de un módulo Almacén, en el que se recogerían todos los datos del inventario, tanto los entrantes como los salientes, para tener un control de todos los productos. Los productos entrantes se refieren a los pedidos a proveedores para ser recibidos en el almacén. El modulo permitiría que un usuario inserte los datos más relevantes de los productos como números de serie, nombre, cantidad, precio por unidad etc. Los productos salientes son los que abandonan el almacén. Estas salidas serían registradas por un empleado en la aplicación, ya sea cuando un producto es vendido en la tienda o cuando un trabajador emita una orden sobre los productos que ha cogido del almacén para su trabajo. A día de hoy no ha sido posible la preparación de este módulo debido a que la empresa está pendiente de algunos cambios. Por ejemplo, actualmente el almacén está lejos de la tienda y se pretende trasladarlo justo donde está la tienda, por lo que así se podría tener a una persona encargada de registrar todos los cambios en el almacén. La idea de la empresa es poner en disposición un plan para estar en contacto con los trabajadores que entran y salen del almacén y se van llevando productos según los van necesitando, para que todos los movimientos queden de alguna manera registrados.

9 Referencias

[1] OpenERP, «OpenERP Spain,» 2015. [En línea]. Available: http://openerpspain.com/.

[2] SAGE, «Sage ERP,» 2015. [En línea]. Available: http://www.sage.es/software/erp.

[3] Kubbos, «Kubbos ERP,» 2015. [En línea]. Available: http://www.kubbos.com/.

[4] D. V. Álvarez, «Software a medida vs Software estandar,» 2015. [En línea]. Available: https://webprogramacion.com/366/blog-informatica-tecnologia/comparativa-software-a-medida-vs-software-comercial.aspx.

[5] Softeng, «Metodología Scrum,» 2015. [En línea]. Available: https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum.html.

[6] NetBeans, «netbeans.org,» 2015. [En línea]. Available: https://netbeans.org/index_es.html.

[7] Oracle, «Oracle Java Documentation,» 2015. [En línea]. Available: http://docs.oracle.com/javase/tutorial/uiswing/.

[8] Codegeando, «Codegeando Tutoriales,» 2015. [En línea]. Available: http://codegeando.blogspot.com.es/2013/03/php-wampserver-definicion-instalacion-y.html.

[9] MySQL, «MySQL,» 2015. [En línea]. Available: https://www.mysql.com.

[10] phpMyAdmin, «phpMyAdmin,» 2015. [En línea]. Available: http://www.phpmyadmin.net/. .

[11] M. A. Alvarez, «MVC,» 2015. [En línea]. Available: http://www.desarrolloweb.com/articulos/que-es-mvc.html.

[12] J. G. VALLE, «Arquitectura cliente servidor,» [En línea]. Available: http://www.monografias.com/trabajos24/arquitectura-cliente-servidor/arquitectura-cliente-servidor.shtml.

Anexo A: Esquema Relacional

Figura 27: Esquema relacional de la Base de Datos

Anexo B: Tablas de la Base de Datos

Nombre	cliente
Descripción	Almacena la información de los clientes.
Atributos	• id_cliente: identificador único del cliente. • nombre: nombre del cliente. • apellidos: apellidos del cliente. • cifNif: cif/nif del cliente según sea persona/empresa. • teléfono1: primer número de teléfono del cliente. • telefono2: segundo número teléfono del cliente • cp: código postal. • fax: número de fax del cliente. • localidad: localidad del cliente. • provincia: provincia del cliente. • correo: correo o e-mail del cliente. • dirección: dirección del cliente.

Tabla 27: Tabla de clientes

Nombre	tarifa
Descripción	Almacena la información de las tarifas.
Atributos	• id_tarifa: identificador único de la tarifa. • nombre: nombre de la tarifa. • id tipo: identificador único del tipo de tarifa asociado. • valor: valor en euros de la tarifa.

Tabla 28: Tabla de tarifas

Nombre	tipo tarifa
Descripción	Almacena la información de los tipos de tarifa existentes.
Atributos	• id tipo: identificador único de la tarifa. • nombre: nombre del tipo de la tarifa.

Tabla 29: Tabla de los tipos de tarifas

Nombre	proveedor
Descripción	Almacena la información de los proveedores.
Atributos	• id_proveedor: identificador único del proveedor. • nombre: nombre del proveedor. • cif: código de identificación fiscal del proveedor. • teléfono: número de teléfono del proveedor. • cp: código postal. • fax: número de fax del proveedor. • localidad: localidad del proveedor. • provincia: provincia del proveedor. • correo: correo o e-mail del proveedor. • dirección: dirección del proveedor. • nombre_comercial: nombre del comercial de contacto. • teléfono_comercial: teléfono del comercial de contacto. • correo_comercial: correo del comercial de contacto. • nombre_contabilidad: nombre del contable de contacto. • teléfono_contabilidad: teléfono del contable de contacto. • correo_contabilidad: correo del contable de contacto.

Tabla 30: Tabla de proveedores

Nombre	socorrista
Descripción	Almacena la información de los socorristas.
Atributos	• id socorrista: identificador único del socorrista. • nombre: nombre del socorrista. • apellidos: apellidos del socorrista. • dni: DNI del socorrista. • teléfono1: primer número de teléfono del socorrista. • telefono2: segundo número teléfono del socorrista. • correo: correo del socorrista • localidad: localidad del socorrista. • título: indica si tiene o no título de socorrista. • horas: correo o e-mail del socorrista. • fecha_inicio: fecha en la cual comienza a trabajar. • fecha_fin: fecha en la cual termina de trabajar.

Tabla 31: Tabla de socorristas

Nombre	socorrista_comunidad.
Descripción	Tabla que relaciona las tablas socorrista y comunidad.
Atributos	• id_socorrista: identificador único del socorrista. • id_comunidad: identificador único de la comunidad. • fecha_inicio: fecha en la que comienza a trabajar un socorrista en una comunidad. • fecha_fin: fecha en la que termina de trabajar un socorrista.

Tabla 32: Tabla que relaciona socorristas y comunidades

D

Nombre	comunidad
Descripción	Almacena la información de las comunidades.
Atributos	• id_comunidad: identificador único de la comunidad. • nombre: nombre de la comunidad. • localidad: localidad de la comunidad. • dirección: dirección de la comunidad. • nombre_admin: nombre del administrador de la comunidad. • teléfono_admin: teléfono del administrador de la comunidad. • correo_admin: correo del administrador de la comunidad. • nombre_presidente: nombre del presidente de la comunidad. • teléfono_presidente: teléfono del presidente de la comunidad. • correo_presidente: correo del presidente de la comunidad. • horario: horario en el que estará abierta la piscina. • fecha_apertura: fecha en la que abre la temporada de piscina. • fecha_cierre: fecha en la que se acaba la temporada de piscina.

Tabla 33: Tabla de comunidades

Nombre	presupuesto
Descripción	Almacena información de los presupuestos.
Atributos	• id_presupuesto: identificador único del presupuesto. • id_cliente: identificador único del cliente al que se le hace el presupuesto. • referente: nombre o pequeña descripción del presupuesto. • fecha: fecha en la que se realiza el presupuesto. • total_presupuesto: valor total en euros. • total_opcional: valor total del opcional en euros. • total_final: valor total final en euros.

Tabla 34: Tabla de presupuestos

E

Nombre	Presupuesto entradas
Descripción	Almacena información de los presupuestos.
Atributos	• id_presupuesto_entradas: identificador único de la entrada del presupuesto. • id_presupuesto: identificador único del presupuesto. • descripcion: descripción de la entrada del presupuesto. • precio: valor de la entrada del presupuesto. • opcional: valor que indica si la entrada es opcional o no.

Tabla 35: Tabla de presupuesto entradas

Nombre	trabajador
Descripción	Almacena la información de los trabajadores.
Atributos	• id_trabajador: identificador único del trabajador. • nombre: nombre del trabajador. • apellidos: apellidos del trabajador. • dni: documento nacional de identidad del trabajador. • cp: código postal. • telefono1: primer número de teléfono del trabajador. • telefono2: segundo teléfono del trabajador. • localidad: localidad del trabajador. • provincia: provincia del trabajador. • correo: correo o e-mail del trabajador • dirección: dirección del trabajador.

Tabla 36: Tabla de trabajadores

F

Nombre	parte_trabajo
Descripción	Almacena la información de los partes de trabajo.
Atributos	• id_parte: identificador único del parte. • id_cliente: identificador único del cliente al que se le realiza el trabajo. • id_trabajador: identificador único del trabajador que realiza el trabajo. • fecha_aviso: fecha en la que se recibe el aviso. • fecha_inicio: fecha de inicio del trabajo. • fecha_final: fecha de finalización del trabajo. • garantía: si tiene garantía o no. • montaje: si el trabajo ha sido montado por la empresa o no. • descripción_averia: descripción de la avería. • informe: informe sobre la realización del trabajo. • total_horas: total horas. • total_kms: total kilómetros. • valor_material: valor en euros del material utilizado. • impuesto: impuesto que se pueda aplicar. • total: total en euros.

Tabla 37: Tabla de partes de trabajo

Nombre	datos_operario_parte
Descripción	Almacena información especial del trabajador para el parte de trabajo.
Atributos	• id_parte: identificador único del parte. • id_trabajador: identificador único del trabajador. • fecha: día, mes y año en que se realiza el trabajo. • hora_salida: hora en la que va a realizar el trabajo. • hora_regreso: hora de regreso al finalizar el trabajo.

Tabla 38: Tabla de información de un operario para un parte

Nombre	factura
Descripción	Almacena información de las facturas.
Atributos	• id_factura: identificador único de la factura. • id_cliente: identificador único del cliente al que se le hace la factura. • fecha: fecha en la que se realiza la factura. • direccion_expedición: dirección a la que se expide la factura. • forma_envio: pequeña descripción de la forma de envió. • sub_total: suma de importes. • iva: tanto % de iva. • total_factura: total valor de la factura en euros.

Tabla 39: Tabla de facturas

Nombre	factura_entradas
Descripción	Almacena información adicional de la factura (albaranes, presupuesto, etc.)
Atributos	• id_factura_entrada: identificador único de la entrada. • id_factura: identificador único de la factura asociada al albarán. • id_ entrada: identificador único de la entrada (albarán, presupuesto, etc.) • descripción: descripción de la entrada.

Tabla 40: Tabla que relaciona factura y albarán

Nombre	albarán
Descripción	Almacena información de los albaranes.
Atributos	• id_albaran: identificador único del albarán. • id_cliente: identificador único del cliente al que se le hace el albarán. • fecha: fecha en la que se realiza el albarán. • importe: valor total en euros.

Tabla 41: Tabla de albaranes

Nombre	Albarán entradas
Descripción	Almacena información adicional de los albaranes.
Atributos	• id_albaran_entradas: identificador único de la entrada. • id_albaran: identificador único del albarán. • entrada_precio: fecha en la que se realiza el albarán. • entrada_cantidad: número de materiales. • entrada_precio_total: precio unitario en euros. • entrada_nombre: nombre o descripción.

Tabla 42: Tabla albarán entradas

Nombre	usuario
Descripción	Almacena información de los usuarios.
Atributos	• id_usuario: identificador único del usuario. • contraseña: contraseña que proporciona el administrador al usuario empleado.

Tabla 43: Tabla de usuarios.

Anexo C: Ejemplo de creación de un Query Reporte

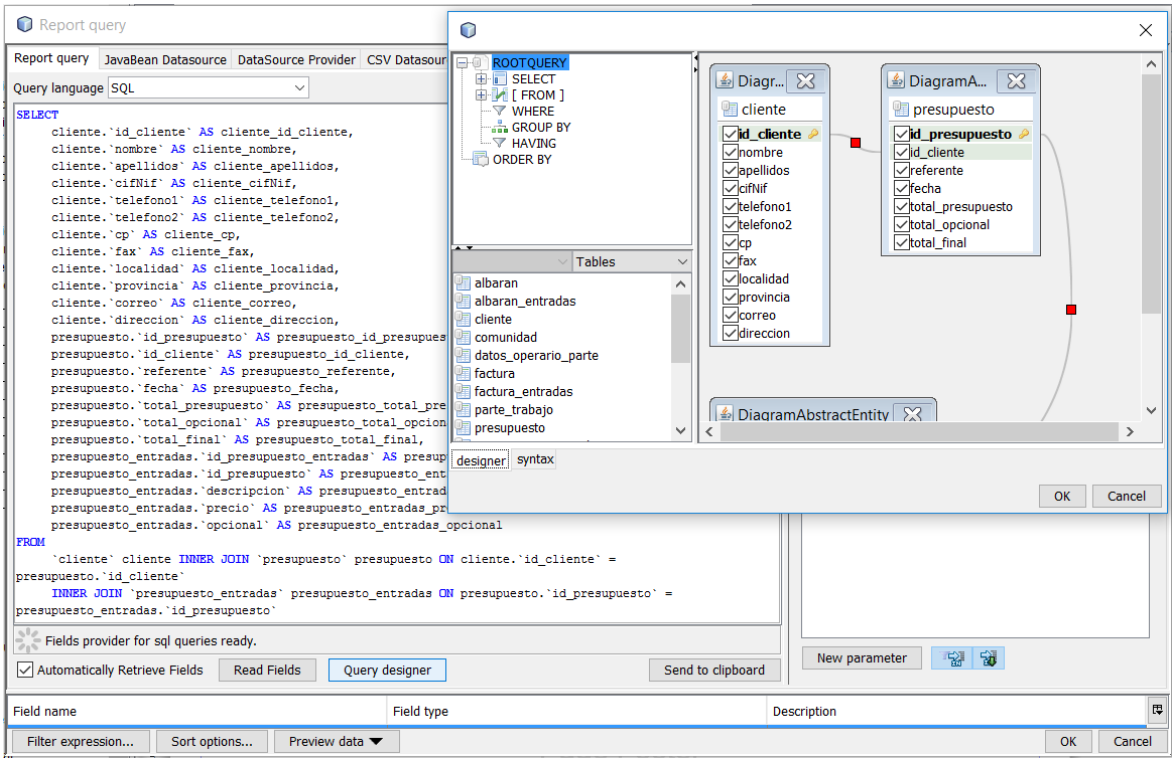


Figura 28: Ejemplo de creación de un Query Reporte.

Anexo D: Ejemplo de reporte de un presupuesto

HIDROSIERRA SPORT S.L. PISCINAS • CALEFACCION

C/ Carrilejos, 80
28770 Colmenar Viejo
Madrid
Teléfono 845 17 84
C. I. F. B-85886455

Alberto Medina Peñalver
calle andalucia
00003 - Guadalix
(Madrid)
44437678M
Tno. - 635123847

Presupuesto Nº 1

C.Viejo a 3 de Enero de 2016

Referente: Sustitucion de bañera por plato de ducha.

DESARROLLO DEL PRESUPUESTO

Descripción	Precio
- Levantar bañera existente y traslado a vertedero autorizado de escombro resultante.	150.0 €
- Impermeabilizacion de la zona en nueva ubicación de plato ducha.	300.0 €
- Suministro e instalación de plato de ducha modelo ATLAS de 1,80x0,75 blanco (MENORCA), con modificacion de desague actual, así como, válvula sifónica.	565.0 €
- Preparación de paramentos verticales de 1,80x0,75 - 0,70x0,70 para posterior alicatado.	236.0 €
- Alicatado de la zona ubicada en nuevo plato de ducha, con plaqueta modelo IRENE color marfil.	350.0 €
Total Presupuesto:	1601.0 €

Opcional.

Suministro e instalación de mampara con perfileria en color blanco, y mampara crilica, de 1,85x2,99 mts.	395 €
Total Opcional:	395.0

Figura 32: Ejemplo de reporte de un presupuesto

Anexo E: Desarrollo de pruebas atendiendo a los requisitos

En la tabla 44 se observa el listado de pruebas realizadas durante los sprints en el proyecto, indicando la salida esperada y la dificultad encontrada. Esta dificultad es evaluada de 1 a 10 en dicha prueba. Cabe destacar que aquí se muestran sólo algunas pruebas por motivos de espacio.

Entrada/Requisito	Salida esperada	Posible error	Nivel de dificultad
Compatibilidad	La visualización de la aplicación se adapta a diferentes pantallas y resoluciones.	Elementos visuales mal colocados. Errores de diseño.	6
Iniciar sesión en la aplicación	Acceso a la página principal del sistema.	Mensaje erróneo en caso de introducir datos erróneos.	1
Desarrollo y administración de la base de datos	La base de datos debe guardar correctamente todos los datos de las tablas configuradas. Esta función es muy importante porque de ella dependen gran parte de los listados de datos, la edición y la creación de nuevos registros en las tablas administradas.	Datos no guardados correctamente.	9
Cambiar contraseña de acceso	Se modifica la contraseña de los empleados por el administrador.	Visualiza un mensaje de error en caso de que la contraseña no se haya modificado.	1
Crear entidades y editar datos de la entidad creada.	La entidad aparece con todos sus datos en las tablas de la base de datos cuando el usuario la crea.	Datos no guardados correctamente.	10 (*)
Eliminación de entidades tales como proveedores, tarifas y tipo de tarifas	Los datos borrados deben desaparecer del sistema, tanto en la aplicación como en la base de datos.	Datos no borrados correctamente.	7

L

Consultar datos de una entidad.	La entidad aparece con todos sus datos extraídos de las tablas de la base de datos.	Algún campo vacío podría implicar que probablemente no se esté recogiendo bien de la base de datos.	1
Evitar información irrelevante	La información que se introduzca en los formularios es relevante	Que la aplicación contenga información que sea irrelevante o que rara vez sea de utilidad.	8
Mensajes de error claros y concisos	Dichos mensajes deben ser claros, indicando exactamente el problema	Problema de los usuarios a la hora de entender el error.	9
Menús	Los menús deben ser sencillos, con una estructura adecuada a las necesidades de la aplicación.	Problema de los usuarios a la hora de comprender la estructura de la aplicación.	7
Indicar los campos obligatorios en formularios	En los formularios de registro de entidades, los campos obligatorios se marcan mediante un asterisco en rojo.	Encontrar algún dato o campo que no sea necesariamente obligatorio o que tenga que serlo.	5
Descargar facturas o presupuestos en PDF	Se descarga un fichero en formato PDF con el reporte solicitado.	Problemas con los datos al mostrar los reportes descargados o que el reporte no se genere bien.	6
Impresión de facturas o presupuestos.	Los reportes de facturas y presupuestos deben crearse correctamente con los datos requeridos.	Mal diseño del reporte o que puedan faltar datos importantes.	9

Tabla 40: Tabla con algunas de las pruebas realizadas.

(*) En esta prueba se agrupa toda la gestión de creación y edición de todos los registros de las tablas administradas, como por ejemplo, la creación de clientes, proveedores, tarifas, presupuestos, etc.

Anexo F: Diagramas de Secuencia. Inserción de un cliente

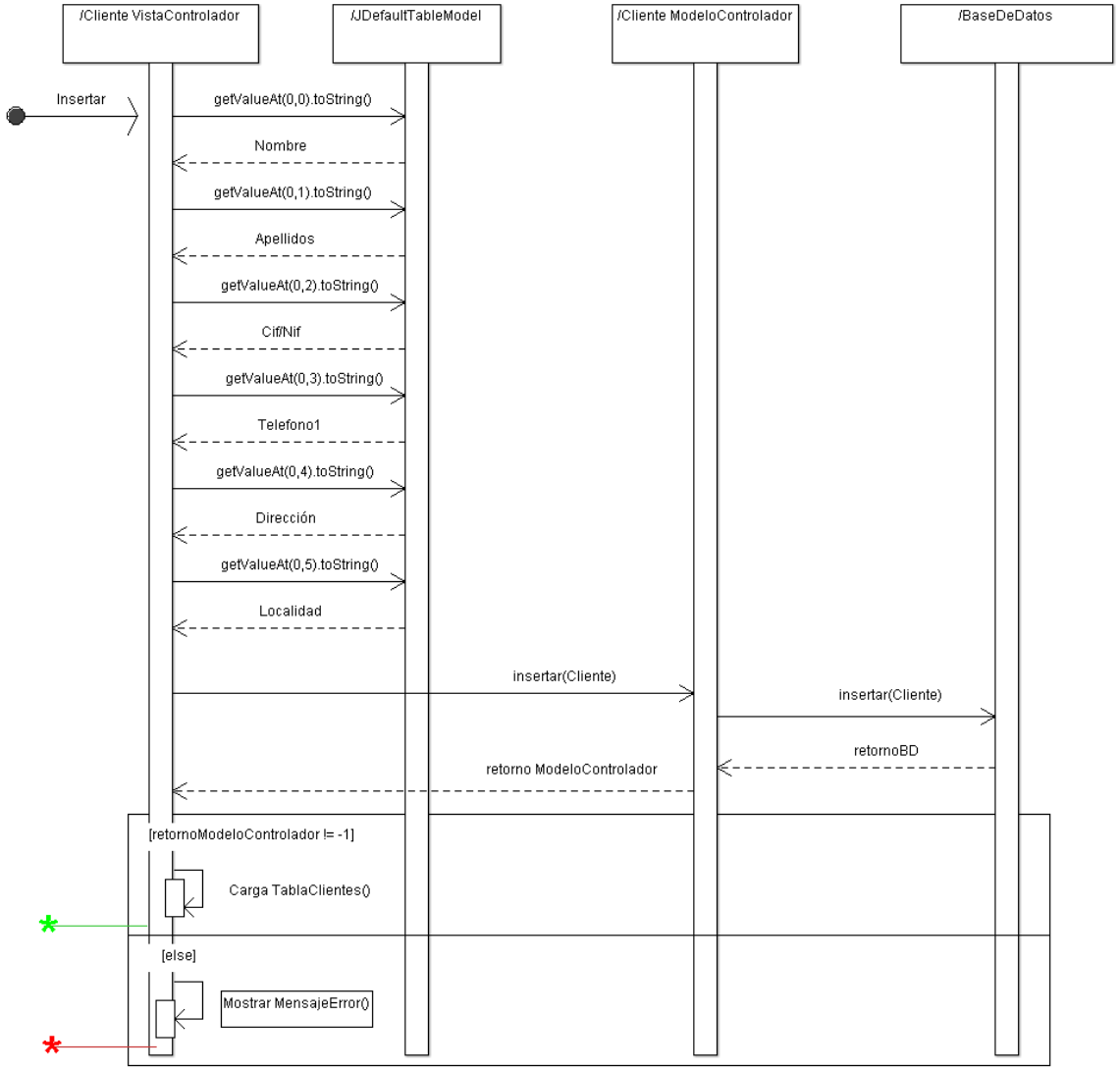


Figura 33: Diagrama de secuencia. Inserción de un cliente

Anexo G: Distribución de las carpetas

Es importante destacar que el paquete Vista se encuentra minimizado debido a la gran cantidad de clases que existen en él.

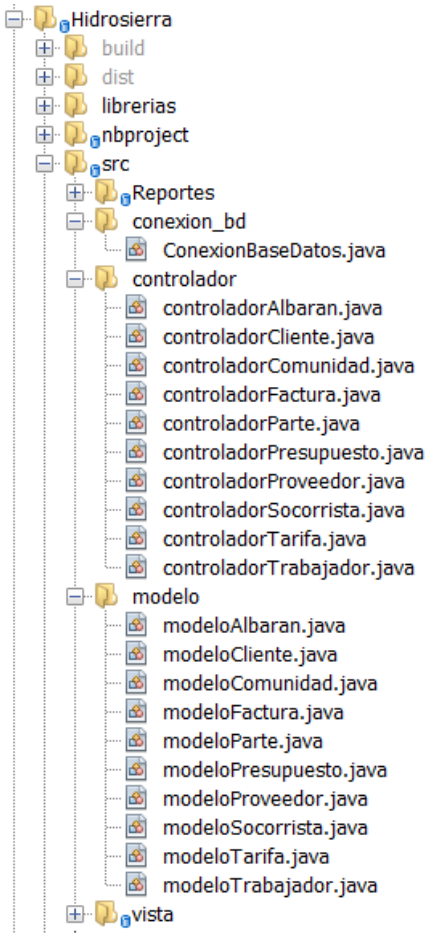


Figura 44: Distribución de las carpetas