Universidad de Valladolid
E.T.S. DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA INFORMÁTICA

ATENCIÓN AVANZADA EN EMERGENCIAS CON HCIS

AUTOR: Jesús Javier Fernández Hebrero
TUTORES: Benjamín Sahelices Fernández
Alberto Rivas Camacho

Agradecimientos:

A mis padres y hermana porque gracias a su apoyo y consejos, he llegado a realizar una de mis grandes metas lo cual constituye la herencia más valiosa que pudiera recibir.

A Raquel por su ayuda, confianza y apoyo en esta etapa de mi vida.

A mi tutor Alberto Rivas por la formación que he recibido de él de manera desinteresada.

Especialmente a mi abuelo como un testimonio de eterno agradecimiento por el gran amor y la confianza que siempre me brindó, gracias por darme la fuerza para irme superando.

Resumen

HP-HCIS (Health Care Information System) es una solución de HP orientada a la gestión de la información de una organización de salud, abarcando de forma integrada los distintos centros de atención primaria, hospitales y otros dispositivos asistenciales que la componen.

Se ha desarrollado una aplicación móvil en Android para que el personal de una ambulancia pueda disponer de la información clínica de un paciente escaneando y reconociendo su tarjeta sanitaria, así como registrar su llegada al área de urgencias y los cuidados que recibe. Para el reconocimiento de la tarjeta sanitaria se partido de un desarrollo ya existente que utiliza la cámara de un smartphone (con OpenCV).

CONTENIDO

1. INTRODUCCIÓN

1.1. Visión general

Las tecnologías de la salud son una amplia gama de productos destinados al cuidado de la salud de los ciudadanos, en una u otra forma, se utilizan para diagnosticar, vigilar o tratar cada enfermedad que afecta a los seres humanos.

Las mejoras tecnológicas en este sector están mejorando la calidad administrativa y el cuidado de los pacientes a través de un diagnostico precoz. Dispositivos tecnológicos, aplicaciones móviles y programas informáticos revolucionan las estrategias de los sistemas sanitarios. No sólo mejoran la adherencia de los pacientes a los tratamientos, sino que reducen el gasto y aumentan la calidad de vida.

El presente documento expone el desarrollo de una aplicación Android que ofrece información relevante de un paciente escaneando su tarjeta sanitaria y la integración de un sistema dentro de la aplicación web HCIS que permitirá al responsable pertinente llevar a cabo la tarea administrativa para la llegada del paciente al centro sanitario.

Este proyecto está ligado con las mejoras citadas anteriormente, y puede formar un papel vital en la atención de un paciente en estado crítico, ya que mantiene conectados los dispositivos móviles con HCIS.

1.2. Objetivos

Para el desarrollo de este proyecto se han establecido los siguientes objetivos:

- Estudio previo, analizando exhaustivamente la aplicación móvil de la que partimos y cómo funciona la aplicación HCIS, observando que requisitos cumple y cuales debemos de establecer tanto funcionales como no funcionales.
- Búsqueda de las tecnologías que se usarán para este proyecto.
- Estudio de otras aplicaciones similares (si las hubiera) para mejorar las prestaciones, experiencia y usabilidad.

- Familiarización con los entornos de desarrollo y estudio de las tecnologías que se utilizan en este proyecto.
- Realizar un diseño inicial y visual de la aplicación Android uniendo todas las funcionalidades encontradas durante el análisis de la misma.
- Elaborar un prototipado en espiral que permita hacernos una idea de la interfaz de la aplicación.
- Conexión de ambas aplicaciones con el servidor que proporcione los datos.

1.3. Motivaciones personales.

Desde mi niñez siempre me ha apasionado el mundo de la medicina y con los años me he interesado por los avances tecnológicos que han apareciendo en esta ciencia.

Con el auge hoy en día de las aplicaciones móviles y mis ganas de aprender a desarrollar este tipo de aplicaciones del que he partido con escasos conocimiento, comienzan a surgir en mí las ganas de seleccionar este proyecto cómo el primer candidato a su desarrollo.

He considerado también, la buena oportunidad de establecer un vínculo con una gran empresa como Hewlett Packard y conocer cómo trabajan este tipo de empresas.

Otra de mis motivaciones ha sido, las ganas de enfrentarme a algo nuevo y desconocido, con el fin de llevarlo a cabo con éxito.

Y por último afianzar mis conocimientos adquiridos en la carrera y mejorarlos.

1.4. Estructura del documento

En este apartado se expone de forma detallada la estructura del documento. Consta de ocho capítulos en los que se describe el proyecto persé.

Al final del proyecto hay un anexo que contiene información complementaria relativa al proyecto.

- Capítulo 1. Introducción:
- Capítulo 2. Contexto tecnológico :
- Capítulo 3. Gestión del proyecto:
- Capítulo 4. Análisis del sistema:
- Capítulo 5. Diseño:
- Capítulo 6. Implementación:
- Capítulo 7. Pruebas:
- Capítulo 8. Conclusiones:

2. CONTEXTO TECNOLÓGICO

2.1. Introducción

A continuación se expone de manera detallada el contexto del proyecto, se comentarán las tecnologías, entornos de desarrollo y software utilizado para el desarrollo del mismo.

2.2. Contexto de Desarrollo

En este apartado describiremos las tecnologías destacadas de este proyecto utilizadas en ambas aplicaciones.

2.2.1. Android

Android es un sistema operativo basado en el núcleo Linux. Fue diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes o tablets. Inicialmente fue desarrollado por Android Inc., empresa que Google respaldó económicamente y más tarde, en 2005, compró

Android fue presentado en 2007 junto la fundación del Open Handset Alliance para avanzar en los estándares abiertos de los dispositivos móviles.

Las aplicaciones se desarrollan habitualmente en el lenguaje Java con Android Software Development Kit (Android SDK), pero están disponibles otras herramientas de desarrollo, incluyendo un Kit de Desarrollo Nativo para aplicaciones o extensiones en C o C++.

2.2.1.1. Características

Estas son algunas de las principales características de Android:

- Código abierto.
- Núcleo basado en el Kernel de Linux.
- Adaptable a muchas pantallas y resoluciones.
- Utiliza SQLite para el almacenamiento de datos.
- Ofrece diferentes formas de mensajería.
- Navegador web basado en WebKit incluido.
- Soporte de Java y muchos formatos multimedia.

- Soporte de HTML, HTML5, Adobe Flash Player, etc.
- Incluye un emulador de dispositivos, herramientas para depuración de memoria y análisis del rendimiento del software.
- Catálogo de aplicaciones gratuitas o pagas en el que pueden ser descargadas e instaladas (Google Play).
- Google Talk para realizar videollamadas.
- Multitarea real de aplicaciones.

2.2.1.2. Arquitectura

La arquitectura está compuesta por cinco amplios componentes que se detallan brevemente a continuación:

- **Aplicaciones:** incluyen un cliente de correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros. Todas ellas escritas en Java.

- **Marco de trabajo de aplicaciones:** los desarrolladores tienen acceso completo a los mismos APIs del framework usados por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades.

- **Bibliotecas:** incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema.

- **Runtime de Android:** incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual Dalvik.

- **Núcleo Linux:** Android depende de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores. También actúa como capa de abstracción entre el hardware y el resto de la pila de software.

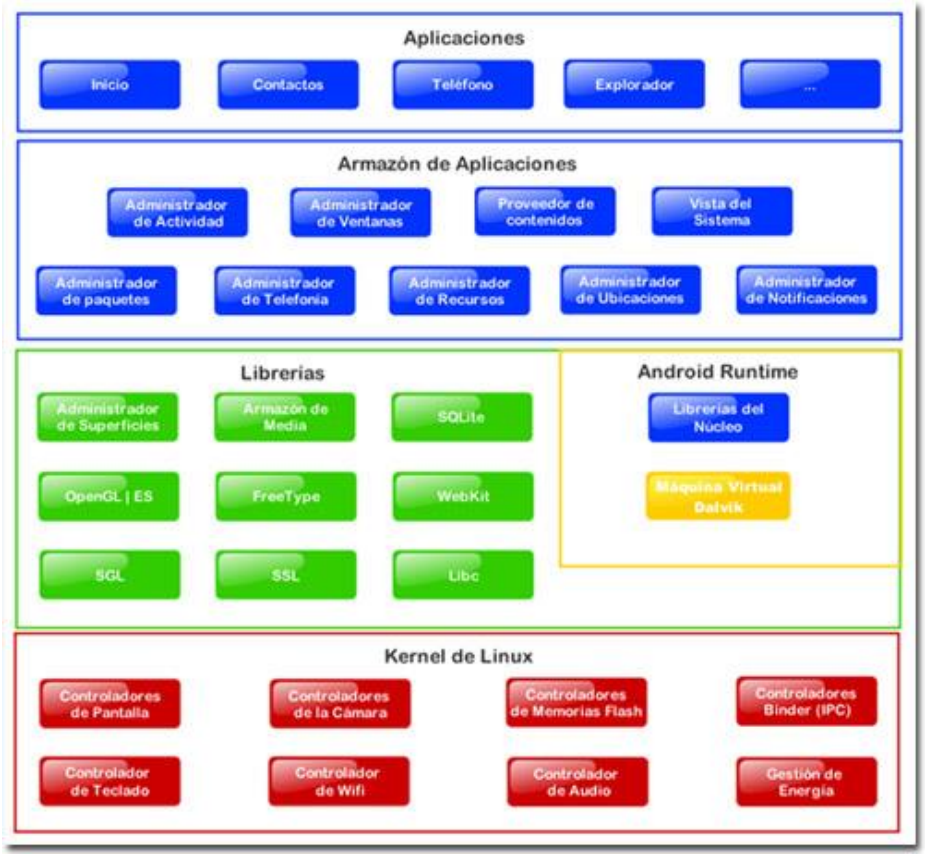


Ilustración 1 Arquitectura de un sistema Android

2.2.2. Servicio SOAP

SOAP (siglas de Simple Object Access Protocol) es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

Este protocolo está basado en XML y se conforma de tres partes:

- **Sobre (envelope):** el cual define qué hay en el mensaje y cómo procesarlo.
- **Conjunto de reglas de codificación** para expresar instancias de tipos de datos.
- **La Convención** para representar llamadas a procedimientos y respuestas.

2.2.2.1. Características principales

El protocolo SOAP tiene tres características principales:

- **Extensibilidad** seguridad y WS-routing son extensiones aplicadas en el desarrollo.
- **Neutralidad** SOAP puede ser utilizado sobre cualquier protocolo de transporte como HTTP, SMTP, TCP o JMS.
- **Independencia** SOAP permite cualquier modelo de programación.

2.2.3. OCR (Optical Character Recognition)

Reconocimiento Óptico de Caracteres, o ROC, es una tecnología que le permite convertir diferentes tipos de documentos, tales como documentos en papel escaneados, PDF archivos o imágenes captadas por una cámara digital en datos con opción de búsqueda y funcionalidad de editar.

El software que utiliza este tipo de tecnologías inspecciona la imagen pixel a pixel, buscando formas que coincidan con los rasgos de los caracteres. En función del nivel de complejidad o grado de desarrollo del software, éste buscará coincidencias con los caracteres y fuentes disponibles en el programa, o tratará de identificar los caracteres a través del análisis de sus características, de forma que el reconocimiento de los mismos no se limite exclusivamente a un determinado número de fuentes.

2.2.3.1 Ventajas e inconvenientes del OCR

Esta son las ventajas que puede tener la aplicación del OCR:

- Búsqueda y recuperación de documentos

1. La aplicación del OCR permite realizar búsquedas de texto libre sobre la totalidad del documento.

2. En el proceso de creación de los metadatos, el OCR se puede utilizar para generar índices de palabras clave del texto reconocido de forma automática.

-Explotación de los documentos

3. El OCR permite convertir el texto de los documentos digitalizados a formatos editables.
4. Aunque el OCR no es una herramienta para hacer los documentos accesibles para personas con discapacidades visuales, su aplicación combinada con otras tecnologías permite que el texto resultante se sintetice en líneas de braille o archivos de audio.

-Perspectiva económica

5. Ahorro de tiempo respecto a la inserción manual de datos (el OCR puede alcanzar una velocidad de lectura de hasta 1.200 caracteres por segundo).
6. El almacenamiento en formato de texto puede suponer un ahorro de espacio respecto del almacenamiento como imagen (el archivo de texto necesita aproximadamente 1/3 del espacio que ocupa la imagen).

¿Cuáles son los inconvenientes del OCR?:

Estos son algunos de roblemas en relación con el OCR:

1. Carencia de conocimiento y expertos en las instituciones.
2. Elevado coste de generar texto electrónico (no confundir con imagen digital) con todas sus funciones (este proceso puede realizarse tecleando el texto o a través de OCR y posterior revisión y corrección del texto).
3. Nivel de efectividad insatisfactorio del OCR en el reconocimiento de documentos históricos, anteriores al inicio de la edición industrial de libros a mediados del siglo XIX.

2.2.4. Spring e Hibernate

Spring es un framework para el desarrollo de aplicaciones y contenedor de inversión de control, de código abierto para la plataforma Java.2

Basándose en ficheros xml o anotaciones es el encargado de construir todos los objetos que la aplicación va a utilizar. De esta manera al ser Spring el encargado de inicializar todos los objetos, es también el responsable de asegurarnos que se integran de la forma correcta.

Spring comprende diversos módulos que proveen un rango de servicios:

- **Contenedor de inversión de control:** permite la configuración de los componentes de aplicación y la administración del ciclo de vida de los objetos Java, se lleva a cabo principalmente a través de la inyección de dependencias.
- **Acceso a datos:** se trabaja con RDBMS en la plataforma java, usando Java Database Connectivity y herramientas de Mapeo objeto relacional con bases de datos NoSQL.
- **Gestión de transacciones:** unifica distintas APIs de gestión y coordina las transacciones para los objetos Java.
- **Modelo Vista Controlador:** Un framework basado en HTTP y servlets, que provee herramientas para la extensión y personalización de aplicaciones web y servicios web REST.
- **Autenticación y Autorización:** procesos de seguridad configurables que soportan un rango de estándares, protocolos, herramientas y prácticas.

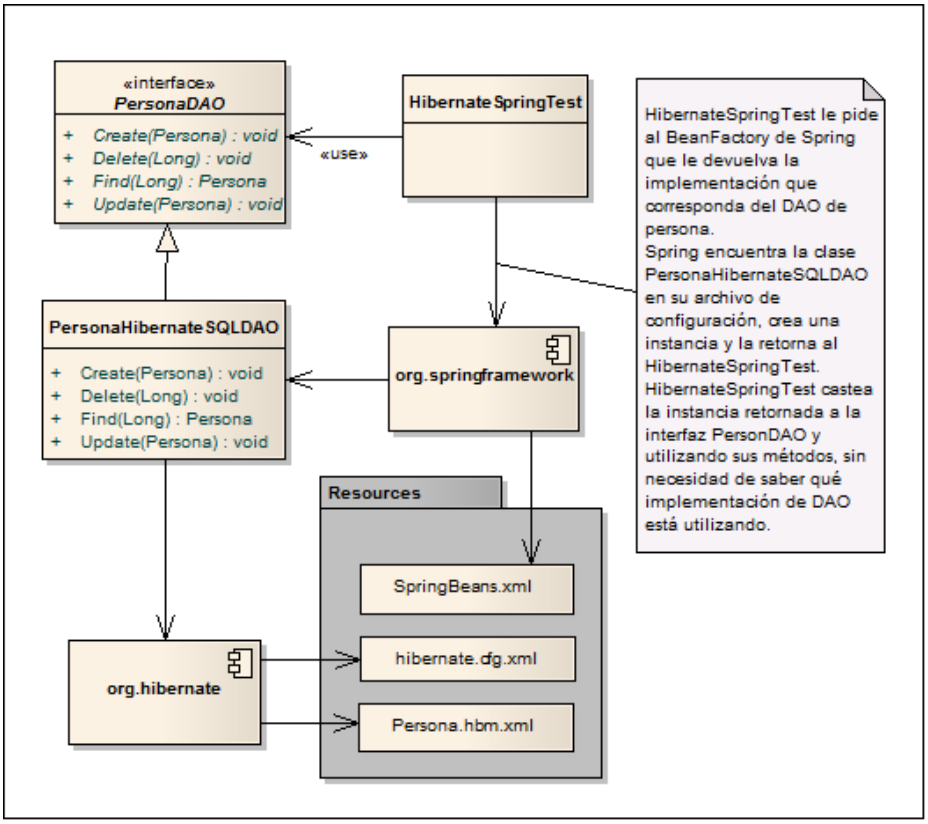


Ilustración 2 Diagrama Spring Hibernate

Hibernate es una herramienta de Mapeo objeto-relacional (ORM) para la plataforma Java (y disponible también para .Net con el nombre de NHibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los “beans” u objetos de las entidades que permiten establecer estas relaciones.

Hibernate está diseñado para ser flexible en cuanto al esquema de tablas utilizado, para poder adaptarse a su uso sobre una base de datos ya existente

Hibernate ofrece también un lenguaje de consulta de datos llamado HQL (Hibernate Query Language), al mismo tiempo que una API para construir las consultas programáticamente (conocida como "criteria").

2.3. Lenguajes de programación

Para la elaboración de este proyecto se han utilizado diversos lenguajes de programación que se detallan a continuación

2.3.1. Java

Un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems. El lenguaje Java está estructurado por paquetes, espacios de nombres de Java compuestos por clases.

Estas son algunas de las características de java

- Encapsulación, herencia y polimorfismo.
- Fuertemente tipado.
- Gestión automática de la memoria (recogida de basura).
- Soporte para concurrencia (multihilo).
- Gestión de excepciones.
- Constructores independientes de la arquitectura del procesador.

2.3.2. XML

XML proviene de eXtensible Markup Language ("Lenguaje de Marcas Extensible"). Se trata de un metalenguaje (un lenguaje que se utiliza para decir algo acerca de otro) extensible de etiquetas que fue desarrollado por el Word Wide Web Consortium (W3C), una sociedad mercantil internacional que elabora recomendaciones para la World Wide Web.

Se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo, etc.

2.3.3. JavaScript

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, está basado en prototipos, es imperativo, débilmente tipado y dinámico.

2.3.4. JSP

Es una tecnología que ayuda a los desarrolladores de software a crear páginas web dinámicas basadas en HTML, XML, entre otros tipos de documentos. JSP es similar a PHP, pero usa el lenguaje de programación Java.

JSP hereda la portabilidad de Java, y es posible ejecutar las aplicaciones en múltiples plataformas sin cambios.

2.3.5. SQL

Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

Una de sus características es el manejo del álgebra y el cálculo relacional que permiten efectuar consultas con el fin de recuperar, de forma sencilla, información de bases de datos, así como hacer cambios en ellas.

2.3.6. HTML

Es un estándar que sirve de referencia para la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, entre otros.

2.3.7. UML

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group).

Es un lenguaje gráfico que sirve para diseñar, construir, documentar un sistema.

2.3.8. C++

Es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al lenguaje de programación C mecanismos que permiten la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.

2.4. Herramientas

A continuación se detallarán las herramientas esenciales utilizadas en el proyecto:

2.4.1. Android SDK

El kit de desarrollo software (SDK) es un conjunto de herramientas de desarrollo de software que le permite al desarrollador de software crear aplicaciones para un sistema concreto.

Es algo tan sencillo como una interfaz de programación de aplicaciones o API creada para permitir el uso de cierto lenguaje de programación.

Android SDK está desarrollado por Google y entre las herramientas que nos ofrece se encuentran un depurador de código, librerías de compatibilidad de versiones, un emulador de dispositivos Android (ADV Manager) y diversas herramientas para la interacción con otro tipo de dispositivos.

2.4.2. Eclipse

Es un programa que permite al desarrollador de software, proporciona herramientas para la gestión de espacios de trabajo, escribir, desplegar, ejecutar y depurar aplicaciones.

Algunas características de este popular entorno de desarrollo son:

- **Perspectivas, editores y vistas:** en Eclipse el concepto de trabajo está basado en las perspectivas, que no es otra cosa que una preconfiguración de ventanas y editores, relacionadas entre sí, y

que nos permiten trabajar en un determinado entorno de trabajo de forma óptima.

- **Gestión de proyectos**: el desarrollo sobre Eclipse se basa en los proyectos, que son el conjunto de recursos relacionados entre sí, como puede ser el código fuente, documentación, ficheros configuración, árbol de directorios,… El IDE nos proporcionará asistentes y ayudas para la creación de proyectos. Por ejemplo, cuando creamos uno, se abre la perspectiva adecuada al tipo de proyecto que estemos creando, con la colección de vistas, editores y ventanas preconfigurada por defecto.

- **Depurador de código**: se incluye un potente depurador, de uso fácil e intuitivo, y que visualmente nos ayuda a mejorar nuestro código. Para ello sólo debemos ejecutar el programa en modo depuración (con un simple botón). De nuevo, tenemos una perspectiva específica para la depuración de código, la perspectiva depuración, donde se muestra de forma ordenada toda la información necesaria para realizar dicha tarea.

- **Extensa colección de plug-ins:** están disponibles en una gran cantidad, unos publicados por Eclipse, otros por terceros. Al haber sido un estándar de facto durante tanto tiempo (no el único estándar, pero sí uno de ellos), la colección disponible es muy grande. Los hay gratuitos, de pago, bajo distintas licencias, pero casi para cualquier cosa que nos imaginemos tenemos el plug-in adecuado.

Para este proyecto se ha utilizado eclipse para la aplicación móvil desarrollada en Android.

2.4.3. Intellij IDEA

Es un entorno de desarrollo integrado (IDE) para el desarrollo de programas informáticos. Fue desarrollado por JetBrains.

Proporciona unas utilidades similares al entorno de desarrollo descrito en el anterior punto, ofrece análisis del código, compilación, ejecución, debugging, control de versiones, detección de duplicaciones, análisis de dependencias y soporte para plugins

Se ha utilizado este software para la parte de desarrollo de HCIS.

2.4.4. TOAD for Oracle

TOAD es una aplicación informática de desarrollo SQL y administración de base de datos, considerada una herramienta útil para los Administradores de bases de datos.

Se ejecuta en todas las plataformas Windows de 32 bits, incluidas Windows 95, 98, NT, 2000, y XP.

2.4.5. Virtual Box

Es un software de virtualización, está desarrollado por Oracle Corporation. Gracias a este programa es posible instalar múltiples sistemas operativos, conocidos como sistemas operativos invitados, sobre otro sistema operativo comúnmente conocido como anfitrión.

2.4.6. Oracle 11G

Es un sistema de gestión de base de datos objeto-relacional, desarrollado por Oracle Corporation.

Oracle se caracteriza por ser un sistema de gestión bastante completo destacando el soporte de transacciones completo, estabilidad, escalabilidad y soporte multiplataforma.

2.4.7. Balsamiq Mockups

Es una herramienta para realizar prototipos, tanto de aplicaciones web como móviles, gracias a esta herramienta podemos ver el esqueleto general visual de nuestro proyecto.

2.4.8. Jboss

Es un servidor de aplicaciones Java EE de código abierto implementado en Java puro.

Ofrece una plataforma de alto rendimiento para aplicaciones de e-business. Combinando una arquitectura orientada a servicios, con una licencia GNU de código abierto, JBoss AS puede ser descargado, utilizado, incrustado y distribuido sin restricciones por la licencia.

2.4.9. Apache Tomcat

Tomcat es un contenedor web con soporte de servlets y JSPs. Tomcat no es un servidor de aplicaciones, como JBoss o JOnAS. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor web Apache.

Tomcat puede funcionar como servidor web por sí mismo.

2.4.10. Dropbox

Es un sistema de control de versiones distribuido propietario desarrollado por la empresa española Códice Software. Como objetivos fundamentales, Plastic trata de dar un mayor soporte al desarrollo paralelo, creación de ramas, integración (*merge*) de ramas, seguridad y desarrollo distribuido.

2.4.11. Astah Community

Editor UML para la realización de diagramas de diseño y de análisis. La versión Community es de licencia gratuita y proporciona todas las herramientas necesarias en nuestro caso.

Se trata de una herramienta intuitiva y de fácil manejo, algo que facilita el desarrollo de los diagramas presentes en este documento.

2.4.12. Dropbox

Dropbox se trata de una herramienta de almacenamiento y compartición de archivos en la nube. Mediante un usuario y una contraseña podemos

almacenar nuestros ficheros en Internet. Resulta de gran utilidad como copia de seguridad.

3. GESTIÓN DEL PROYECTO

3.1. Introducción

En este apartado se describirá el modelo de proceso elegido para el desarrollo del presente proyecto, se analizarán los posibles riesgos a los que nuestro sistema está expuesto y se detallarán aspectos relacionados con la planificación del proyecto y los recursos de los que se dispone.

3.2. Modelo de proceso

En informática dividimos los proyectos en fases facilitar su gestión, mejorar el control y mantenerlos alineados con los objetivos.

Estas fases componen el ciclo de vida del proyecto, Según define el PMBOK

"El ciclo de vida del proyecto es un conjunto de fases del mismo, generalmente secuenciales y en ocasiones superpuestas, cuyo nombre y número se determinan por las necesidades de gestión y control de la organización u organizaciones que participan en el proyecto, la naturaleza propia del proyecto y su área de aplicación."

"El ciclo de vida proporciona el marco de referencia básico para dirigir el proyecto".

Los proyectos independientemente de su tamaño y complejidad están compuestos por 4 elementos.

- Inicio
- Planificación
- Ejecución
- Cierre del proyecto

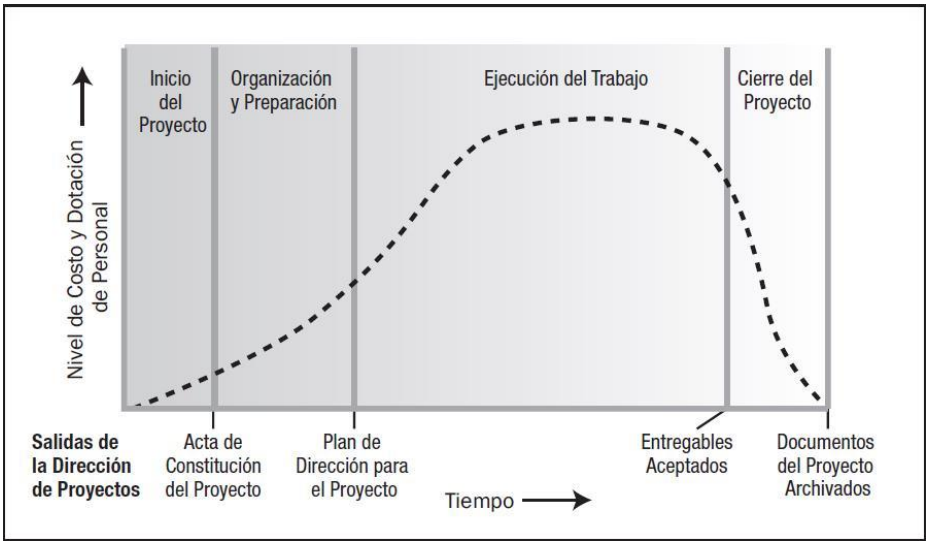


Ilustración 3 Modelo de proceso

Para el desarrollo de este proyecto se ha elegido un desarrollo en espiral, que es un modelo de ciclo de vida del software utilizado de manera muy general en la ingeniería de software.

El ciclo de Vida en Espiral tiene en cuenta fuertemente el riesgo que aparece a la hora de desarrollar software. Para ello, se comienza mirando las posibles alternativas de desarrollo, se opta por la de riesgo más asumible y se hace un ciclo de la espiral.

Con este modelo los requisitos pueden ir variando a lo largo que avanzamos en el desarrollo, podemos considerarlo como una ventaja puesto que inicialmente es difícil obtener un documento con los requisitos totalmente definidos.

3.3. Gestión de Riesgos

La gestión de riesgos es parte integral del desarrollo de un proyecto, es un proceso que se considera fundamental para la toma de decisiones tiene como propósito disminuir la probabilidad y el impacto de los eventos adversos para los objetivos de un proyecto.

¿Qué se entiende por riesgo en un proyecto?

Se define como: *Un evento o condición que, si ocurre, tiene un efecto sobre los objetivos del proyecto.*

Los riesgos pueden ser positivos o negativos, los negativos influyen negativamente sobre alguno o varios objetivos del proyecto.

A continuación se detallan los riesgos conforme a una serie de aspectos que se han tenido en cuenta:

- **Nombre y descripción:** Nombre y descripción breve del riesgo a tratar.
- **Probabilidad:** Es la probabilidad de ocurrencia del fallo. Se evalúa en una escala desde una probabilidad de ocurrencia baja, hasta una muy alta o catastrófica.
- **Impacto o gravedad:** descripción de los posibles efectos sobre el proyecto y su magnitud. Se valora en una escala desde insignificante hasta un impacto catastrófico.
- **Riesgo**: resultado de multiplicar el impacto por la probabilidad de ocurrencia.
- **Estrategia de prevención:** Acciones o medidas a llevar a cabo en el proyecto para evitar que el riesgo aparezca o en caso de que este ocurra, minimizar su efecto en la medida de lo posible. Estas medidas son aplicadas antes de que el riesgo se transforme en un hecho.
- **Plan de contingencia:** Acciones o medidas a llevar a cabo en el proyecto cuando el riesgo ha ocurrido.

Las siguientes tablas permiten valorar el efecto que pueden tener los riesgos en nuestro proyecto, se han utilizado los aspectos descritos anteriormente.

Nº Riesgo	1
Nombre	Planificación irreal u optimista
Descripción	Planificación que no se ajusta a la realidad ni a los recursos de los que se dispone.
Probabilidad	Media
Impacto	Medio
Riesgo	Medio
Estrategia de prevención	Realizar una planificación los más realista posible
Plan de contingencia	Reorganización de trabajo y análisis de recursos

Nº Riesgo	2
Nombre	Pérdida de información
Descripción	Perdida de los datos necesarios útiles para el desarrollo exitoso del proyecto
Probabilidad	Baja
Impacto	Alto
Riesgo	Medio
Estrategia de prevención	Copias de seguridad actualizadas
Plan de contingencia	Examinar que datos hemos perdido y volver a elaborar el trabajo perdido

Nº Riesgo	3
Nombre	Avería de equipos
Descripción	Inutilización de algún dispositivo necesario para el desarrollo del proyecto
Probabilidad	Baja
Impacto	Medio
Riesgo	Medio
Estrategia de prevención	Tener en cuenta en la planificación
Plan de contingencia	Reemplazar dispositivo averiado por otro funcional

Nº Riesgo	4
Nombre	Cambios imprevistos en requisitos
Descripción	Gran cambio en los requisitos puede retrasar otras tareas el proyecto
Probabilidad	Media
Impacto	Alto
Riesgo	Medio
Estrategia de prevención	Mejora de planificación temporal
Plan de contingencia	Trabajar en los nuevos cambios

Nº Riesgo	5
Nombre	Falta de experiencia en las tecnologías a utilizar

Descripción	Falta de conocimiento de los lenguajes de programación y tecnologías a usar en el sistema para su desarrollo
Probabilidad	Media
Impacto	Media
Riesgo	Medio
Estrategia de prevención	Realizar una búsqueda previa y aprendizaje sobre los lenguajes y tecnologías a utilizar
Plan de contingencia	Comunicación con el tutor y estudio de las tecnologías que producen este riesgo

Nº Riesgo	6
Nombre	Falta de comunicación con el tutor
Descripción	Fallo en la comunicación con el tutor de prácticas.
Probabilidad	Media
Impacto	Media
Riesgo	Medio
Estrategia de prevención	Mantener reuniones periódicas, establecer un canal de comunicación y Trabajar para mantener un buen clima con el tutor
Plan de contingencia	Avisar a la universidad de la situación

Nº Riesgo	7
Nombre	Fallos en el sofware
Descripción	Mal funcionamiento del software utilizado en el proyecto debido a un problema externo como puede ser amenazas en forma de virus, desarrollo incorrecto del software utilizado, incompatibilidad de versiones.
Probabilidad	Media
Impacto	Alto
Riesgo	Medio
Estrategia de prevención	Disponibilidad de herramientas similares para su utilización que permitan realizar las tareas afectadas
Plan de contingencia	Utilizar programas equiparables

Nº Riesgo	8
Nombre	Diseño inadecuado

Descripción	Diseño erróneo y desarrollo incorrecto de la funcionalidad e interfaces del sistema
Probabilidad	Media
Impacto	Media
Riesgo	Medio
Estrategia de prevención	Estudiar y plantear detenidamente las posibles opciones para elaborar un diseño y funcionalidades sin errores.
Plan de contingencia	Rediseño de las partes que sean posibles

3.4. Planificación

En este punto se presenta una planificación temporal del proyecto. A pesar de tener en cuenta los riesgos, la planificación ha sufrido alguna ligera modificación a lo largo del desarrollo del proyecto.

La duración de nuestro proyecto dura 300 horas, en la siguiente imagen puede verse como se han repartido las horas en el proyecto.

	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	**Fase inicial**	**60 días**	**lun 10/11/14**	**lun 02/02/15**	
2	Documentación del sistema	6 días	lun 10/11/14	lun 17/11/14	
3	Documentación de las tecnologías	5 días	mar 18/11/14	lun 24/11/14	2
4	Planificación del proyecto	2 días	mar 25/11/14	mié 26/11/14	3
5	Instalación del entorno	25 días	jue 27/11/14	mié 31/12/14	4
6	Análisis y diseño del sistema	20 días	lun 05/01/15	vie 30/01/15	5
7	Reunión fin de ciclo	0 días	lun 02/02/15	lun 02/02/15	6
8	**Implementación**	**80 días**	**lun 02/02/15**	**vie 22/05/15**	**2**
9	Servicio web	12 días	lun 02/02/15	mar 17/02/15	7
10	Aplicación móvil	35 días	lun 16/02/15	vie 03/04/15	
11	Reunión	0 días	vie 17/04/15	vie 17/04/15	
12	Aplicación web	35 días	lun 06/04/15	vie 22/05/15	10
13	Reunión fin de ciclo	0 días	mié 20/05/15	mié 20/05/15	
14	**Pruebas**	**5 días**	**lun 25/05/15**	**vie 29/05/15**	**12**
15	Pruebas del sistema	5 días	lun 25/05/15	vie 29/05/15	
16	Reunión fin de ciclo	0 días	vie 29/05/15	vie 29/05/15	15
17	**Cierre de proyecto**	**20 días**	**lun 06/07/15**	**vie 31/07/15**	**15**
18	Cierre de la documentación	15 días	lun 06/07/15	vie 24/07/15	
19	Revisión de la documentación	5 días	lun 27/07/15	vie 31/07/15	18

Ilustración 4 Planificación inicial

En la siguiente imagen podemos ver la planificación inicial en un diagrama de Gantt:

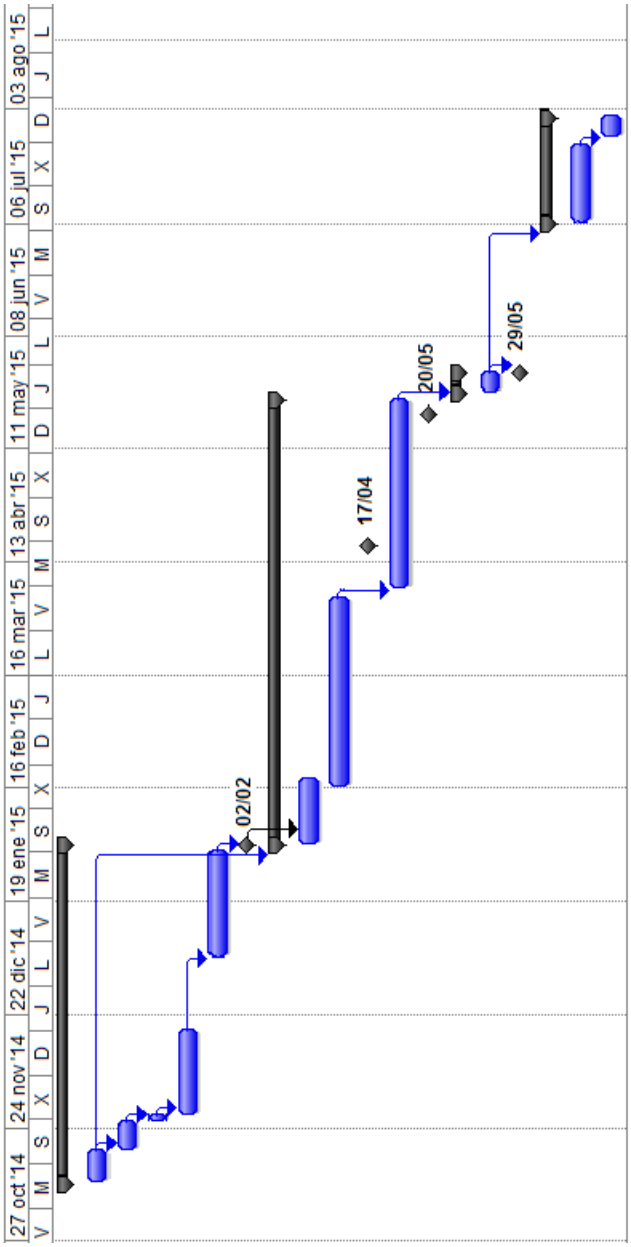


Ilustración 5 Diagrama de Gantt

Nos encontramos con una planificación de 135 días, respetando los periodos de exámenes. Se ha establecido una media de trabajo de 2,5 horas diarias, teniendo en cuenta:

- Horas lectivas en la Universidad de Valladolid.
- Horas empleadas en prácticas en empresa.

Teniendo en cuenta todo tenemos un total de 413 horas, como podemos ver es una planificación muy poco optimista, de cara a poder abordar los posibles imprevistos que vayan surgiendo durante el desarrollo de éste.

3.5. Recursos

A continuación se describen los recursos de los que se ha dispuesto para el desarrollo del proyecto.

3.5.1. Recursos Hardware

Los dispositivos de los que se ha dispuesto en el desarrollo del proyecto son los siguientes:

Portátil HP Pavilion g6, con procesador Intel i5 (2.40GHz, 3MB (L3) Cache) y 6GB de Memoria RAM DDR3.
Smartphone Samsung Galaxy Grand Neo Android 4.4 (KitKat) Procesador Quad Core 1.2 GHz 1 GB RAM
Sony Xperia E3 Android 4.4 (KitKat) Procesador Qualcomm MSM8926-2 Snapdragon 400 quad-core 1.2GHz 1 GB RAM

Los tres dispositivos han sido utilizados durante todo el desarrollo del proyecto, cabe mencionar que los dispositivos móviles han sido utilizados para la comunicación con el tutor

3.5.2. Recursos Software

Los recursos software de los que se ha dispuesto se mencionan en la siguiente tabla junto a su propósito.

Herramienta	**Proposito**
Windows 8 Proffesional	Sistema operativo (Soporte)
Mockups	Diseño del sistema
Astah community	Análisis y diseño del sistema
Eclipse IDE luna	Implementación
Intellij IDEA	Implementación
Toad	implementación
Virtual Box (win XP)	Implementación

3.5.3. Recursos Humanos

Para el desarrollo del sistema serán necesarios los perfiles mostrados en la siguiente tabla

Recurso	**Propósito**
Analista	Análisis y diseño del sistema
Programador	Implementación del sistema

La asignación de recursos no será constante a lo largo de todo el proyecto sino que cada uno realizará su trabajo en el momento que le corresponda, estipulado en la planificación inicial.

3.6. Costes y presupuesto

A continuación se presentan los costes presentes en el proyecto y el presupuesto de la aplicación para poder estimar un precio.

Para realizar dicha estimación de los costes se tienen en cuenta los siguientes factores:

- Los costes del hardware y software utilizado, incluyendo su mantenimiento y amortización.

- Horas de trabajo, donde se incluyen los gastos derivados de los recursos humanos utilizados, como nivel de profesionalidad o seguridad social del individuo.

- Costes derivados: como gastos de luz, internet o desplazamientos.

La parte de software en este proyecto no presenta costes para el desarrollo del proyecto, puesto que se ha utilizado software libre sin ningún tipo de licencia.

Las horas de trabajo de los recursos humanos presentan las siguientes tarifas:

- Analista: 25 euros/hora.
- Programador: 15 euros/hora.

En la siguiente tabla se muestra una estimación de todos los costes en euros:

Concepto	**Cantidad**	**Precio €**
Herramientas de desarrollo	7	0
Herramientas de Hardware		
Ordenador HP	1	500
Samsung Galaxy Grand Neo	1	30
Sony Xperia E3	1	30
	Subtotal	560
Recursos humanos		
Analista/diseñador	240h	6000
Programador de aplicaciones	255h	3825
	Subtotal	9825
Otros costes (luz, internet, etc)		60
TOTAL		**10445**

3.6.1. Presupuesto final

Teniendo en cuentas los gastos estimados en el apartado anterior y que la aplicación móvil estará totalmente libre de publicidad, se establece el siguiente presupuesto final para el desarrollo del proyecto. Se han estimado unos beneficios del 30%, por lo que el precio final del proyecto ascendería a 12578,5€ euros totales, sin incluir mantenimiento de la aplicación.

Este presupuesto incluiría todos los gastos totales el primer año, sin incluir el mantenimiento de la aplicación.

4. ANÁLISIS DEL SISTEMA

4.1. Introducción

En este capítulo se exponen cuáles son los requisitos y a continuación su descomposición en casos de uso, con el fin de identificar las características y las funciones del sistema.

4.2. Especificación de los requisitos

En este punto se hará una descripción completa del comportamiento del sistema a desarrollar separando entre los dos subsistemas, se incluirán en él los requisitos funcionales y no funcionales, junto con el conjunto de casos de uso del sistema en cuestión.

4.2.1. Requisitos funcionales aplicación móvil

RF1. El sistema será capaz de permitir al usuario su autenticación para acceder a la aplicación mediante nombre de usuario y contraseña.

RF2. La aplicación será capaz de permitir del usuario mantener la sesión iniciada y cerrar sesión.

RF3. La aplicación será capaz de escanear la tarjeta de la seguridad social.

RF4. El sistema advertirá al usuario si el dispositivo tiene el WIFI o datos móviles activados.

RF5. Será capaz de avisar al usuario si ha habido escaneo exitoso.

RF6. El sistema devolverá la siguiente información del paciente: Nombre, NSS, Dirección, número de historia clínica (NHC), teléfono de un contacto, parentesco de ese contacto, alergias, hábitos, Enfermedades Abiertas y observaciones relevantes.

RF7. El sistema será capaz de sincronizar con la base de datos, el motivo, el tratamiento, observaciones y prioridad.

RF8. El sistema avisará al usuario si los datos se han sincronizado exitosamente o no.

RF9. El sistema permitirá llamar al contacto de ese paciente.

4.2.2. Requisitos funcionales HCIS

RF11. El sistema mostrará los pacientes actuales en el área de urgencias.

RF12. El sistema permitirá ver los pacientes introducidos desde la aplicación móvil.

RF13. HCIS permitirá seleccionar entre los pacientes.

RF14. El sistema permitirá ver información más detallada de los pacientes introducidos desde la aplicación móvil.

RF15. El sistema permitirá al usuario preparar la llegada a urgencias del paciente.

RF16. El sistema actualizará automáticamente las listas de pacientes.

4.2.3. Requisitos no funcionales aplicación móvil

RNF1. La aplicación deberá ser soportada por dispositivos móviles que funcionen con Android 3.0 (Honeycomb) en adelante.

RNF2. La aplicación deberá adaptarse a dispositivos con pantallas de tamaños desde 3 hasta 10 pulgadas.

RNF3. La conexión a la base de datos se realizará mediante un servicio web.

RNF4. El servicio web será implementado por la arquitectura SOAP.

RNF5. El dispositivo deberá soportar la conexión a internet.

RNF6. La aplicación será extensible.

RNF7. La aplicación deberá ser implementada en el lenguaje de programación JAVA.

RNF8. La aplicación deberá ser intuitiva y fácil de manejar.

RNF9. La aplicación deberá ser fiable y robusta.

4.2.4. Requisitos no funcionales HCIS

RNF1. El sistema será implementando en el lenguaje JAVA mediante la tecnología JSP.

RNF2. El sistema será implementando utilizando el framework de desarrollo Spring MVC.

RNF3. El sistema se comunicará con la base de datos mediante la herramienta Hibernate.

RNF4. El sistema debe ser intuitivo y rápido.

RNF5. El sistema HCIS deberá ser fiable y seguro

4.3. Modelado de Casos de Uso

En este apartado se define el comportamiento de nuestro sistema. Definiremos los usuarios que utilizará el sistema y después el comportamiento del mismo.

4.3.1. Identificación de los actores

Un actor del sistema es quien interactúa o hace uso del sistema.

En nuestro caso tenemos dos actores bien definidos, Enfermero Ambulancia que será quien interactúe con la aplicación móvil y por otro lado, el responsable urgencias que será quien interactúe con HCIS

4.3.2. Casos de Uso Aplicación móvil

Para la identificación de los casos de uso a continuación presentaremos un diagrama de casos de uso de la aplicación móvil y una descripción de éstos.

4.3.2.1 Diagrama de Casos de uso

Mediante los diagramas de caso de uso se pueden ver las funcionalidades que ofrece el sistema y que los usuarios pueden realizar. En este diagrama se muestra el límite del sistema, representado.

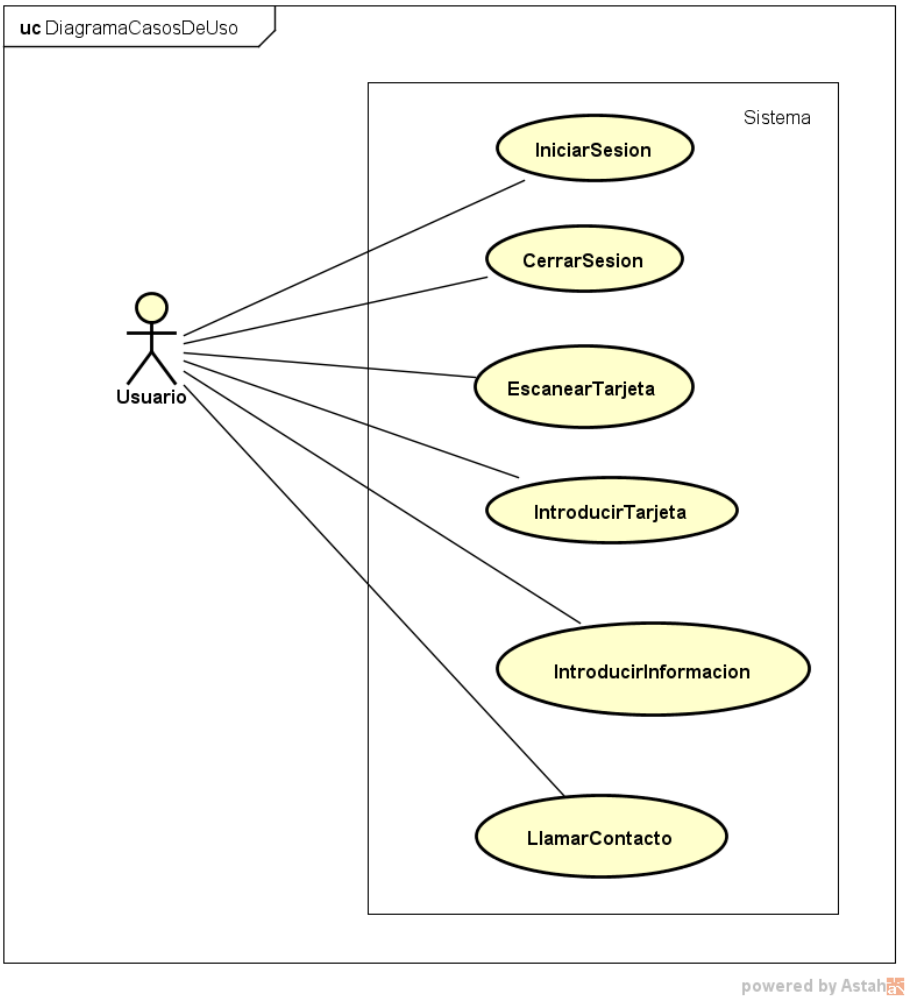


Ilustración 6 Diagrama de Casos de Uso Aplicación Móvil

4.3.2.2 Descripción de Casos de uso

A continuación se describirán los casos de uso del diagrama anterior. En cada caso se incluirá la secuencia asociada.

ID: CU_1	**Caso de Uso: IniciarSesion**	
Breve descripción:	El usuario decide iniciar sesión en el sistema	
Actores principales:	Usuario	
Precondiciones:	Aplicación ha sido iniciada	
Flujo principal	**Paso**	**Acción**
	1	El caso de uso comienza cuando el usuario decide iniciar sesión.
	2	El sistema comprueba que el dispositivo tiene conexión a internet consulta las preferencias de sesión, consulta en base de datos usuario y contraseña.
	3	El caso de uso finaliza cuando el usuario puede autenticarse en el sistema
Postcondiciones:	El usuario queda autenticado en el sistema	
Flujos alternativos	2.1 Si en las preferencias está marcada la sesión accedemos al sistema directamente. 2.2 Si los campos están vacíos el sistema avisa al usuario. 2.3 Si los datos son erróneos el sistema avisa al usuario 2.4 Si no se dispone de conexión a internet se le avisa al usuario	

ID: CU_2	**Caso de Uso: CerrarSesion**	
Breve descripción:	El usuario decide cerrar sesión en el sistema	
Actores principales:	Usuario	
Precondiciones:	Usuario ha iniciado sesión en el sistema	
Flujo principal	**Paso**	**Acción**
	1	El caso de uso comienza cuando el usuario decide cerar sesión.
	2	El sistema elimina los parámetros correspondientes al inicio de sesión.
Postcondiciones:	El usuario pierde permisos utilizar aplicación	
Flujos alternativos	No existen posibles flujos alternativos.	

ID: CU_3	**Caso de Uso: EscanearTarjeta**	
Breve descripción:	El usuario escanea la tarjeta del paciente	
Actores principales:	Usuario	
Precondiciones:	Usuario ha iniciado sesión en el sistema	
Flujo principal	**Paso**	**Acción**
	1	El caso de uso comienza cuando el usuario gira el móvil para comenzar con el escaneo.
	2	El sistema escanea la tarjeta y consulta el número de tarjeta en base de datos y avisa al usuario.
	3	El usuario voltea el móvil de nuevo.
	4	El sistema muestra la información asociada a ese número de tarjeta
Postcondiciones:	El usuario tiene acceso a la información	
Flujos alternativos	No existen posibles flujos alternativos.	

ID: CU_4	**Caso de Uso: IntroducirInformacion**	
Breve descripción:	El usuario introduce el motivo, tratamiento, prioridad y observaciones del suceso	
Actores principales:	Usuario	
Precondiciones:	Usuario ha iniciado sesión en el sistema y el sistema ha mostrado la información.	
Flujo principal	**Paso**	**Acción**
	1	El caso de uso comienza cuando el usuario introduce la información del suceso.
	2	El sistema comprueba que los datos introducidos son válidos y los datos son sincronizados.
	3	El caso de uso acaba cuando el usuario recibe el aviso de sincronización de datos
Postcondiciones:	La información relevante ha sido sincronizada	
Flujos alternativos	2.1 Si los campos están vacíos, el sistema avisa al usuario	

ID: CU_5	**Caso de Uso: LlamarContacto**	
Breve descripción:	El usuario decide llamar al contacto del paciente	
Actores principales:	Usuario	
Precondiciones:	Usuario ha iniciado sesión en el sistema y el sistema ha mostrado la información.	
Flujo principal	**Paso**	**Acción**
	1	El caso de uso comienza cuando el usuario desea llamar al contacto del paciente.
	2	El sistema llama al sistema operativo y ejecuta la llamada.
	3	El caso de uso acaba cuando la llamada finaliza sea exitosa o no y devuelve el control al sistema
Postcondiciones:	No existen postcondiciones	
Flujos alternativos	No existen posibles flujos alternativos.	

ID: CU_6	Caso de Uso: IntroducirTarjeta	
Breve descripción:	El usuario introduce el número de la tarjeta del paciente mediante teclado	
Actores principales:	Usuario	
Precondiciones:	Usuario ha iniciado sesión en el sistema.	
Flujo principal	**Paso**	**Acción**
	1	El caso de uso comienza cuando el usuario introduce el número de seguridad social.
	2	El sistema busca el número de tarjeta y lo consulta en base de datos.
	3	El sistema muestra la información asociada a ese número de tarjeta.
Postcondiciones:	El usuario tiene acceso a la información	
Flujos alternativos	No existen posibles flujos alternativos.	

4.3.3. Casos de Uso HCIS

Para la identificación de los casos de uso a continuación presentaremos un diagrama de casos de uso de la aplicación web HCIS y una descripción de éstos.

4.3.3.1. Diagrama de Casos de Uso

A continuación se muestra el diagrama de casos de uso correspondiente a la aplicación web, está compuesto por cinco casos de uso, pero solo se detallará uno de ellos puesto que los otros ya eran parte del sistema antes de empezar con el desarrollo del proyecto.

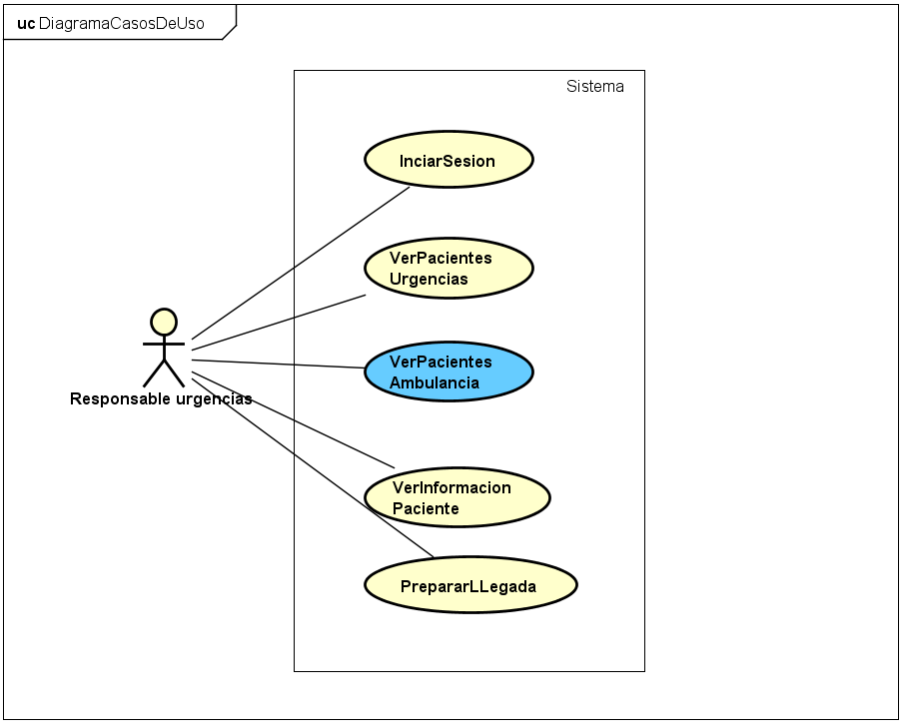


Ilustración 7 Diagrama de Casos de Uso HCIS

4.3.3.2. Descripción de Casos de Uso

A continuación se describirán los casos de uso del diagrama anterior. En cada caso se incluirá la secuencia asociada.

ID: CU_1	**Caso de Uso: VerPacientesAmbulancia**	
Breve descripción:	El usuario accede a ver los pacientes actualmente en ambulancias en dirección al hospital (urgencias).	
Actores principales:	Responsable urgencias	
Precondiciones:	El usuario debe de estar autenticado en el sistema.	
Flujo principal	**Paso**	**Acción**
	1	El caso de uso comienza cuando el usuario accede al servicio de ambulancias perteneciente a urgencias.

	2 3	El sistema recibe la petición del usuario y devuelve una vista en una nueva ventana con una lista con los pacientes en dirección al hospital viajando en ambulancia. El caso de uso finaliza cuando el usuario puede ver los pacientes en la vista de ambulancias urgencias.
Postcondiciones:	No existen Postcondiciones	
Flujos alternativos	No existen posibles flujos alternativos.	

4.4. Modelo de dominio

El modelo de dominio proporciona una perspectiva conceptual, objetos del dominio o clases conceptuales, asociaciones entre clases conceptuales y atributos de las clases conceptuales.

En la siguiente figura se muestra de manera esquemática el conjunto de clases y conceptos de la aplicación móvil que se tendrá en cuenta para la fase de diseño y posterior implementación del software.

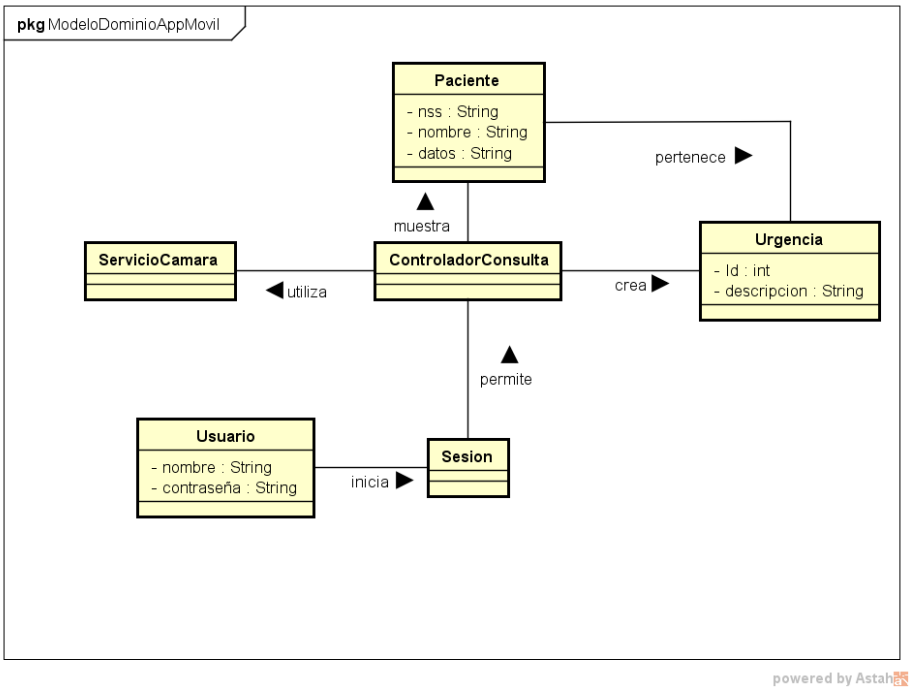


Ilustración 8 Modelo de Dominio Aplicación móvil

En esta figura se muestran para la aplicación HCIS

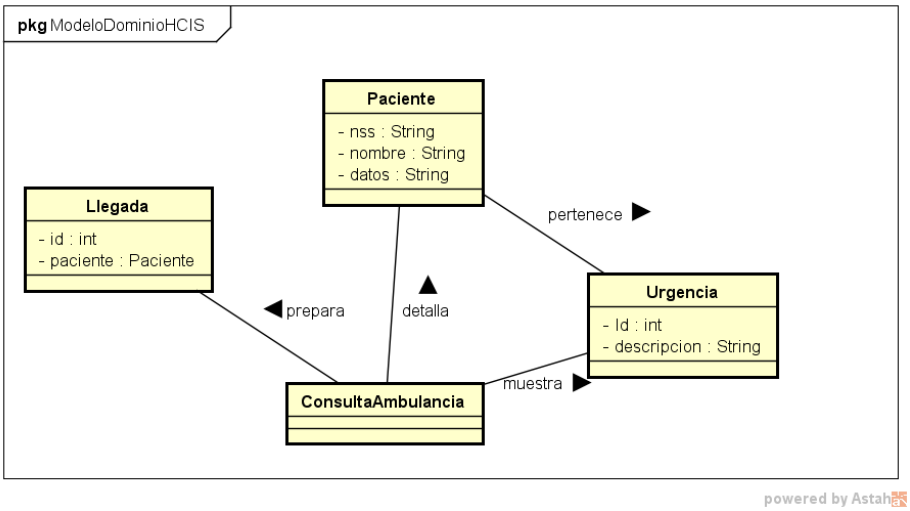


Ilustración 9 Modelo de Dominio HCIS

4.5 Prototipos

Con el fin de facilitar la fase de diseño e implementación. Se han realizado prototipos para la parte de la aplicación.

Los prototipos vieron el visto bueno por parte del tutor de prácticas y formaron un papel fundamental.

1. **Pantalla principal:** En un primer lugar no se contempló la idea de llevar a cabo el control de acceso, la pantalla principal propuesta en el prototipo directamente permitía acceder a la aplicación, sin ningún tipo de restricción.

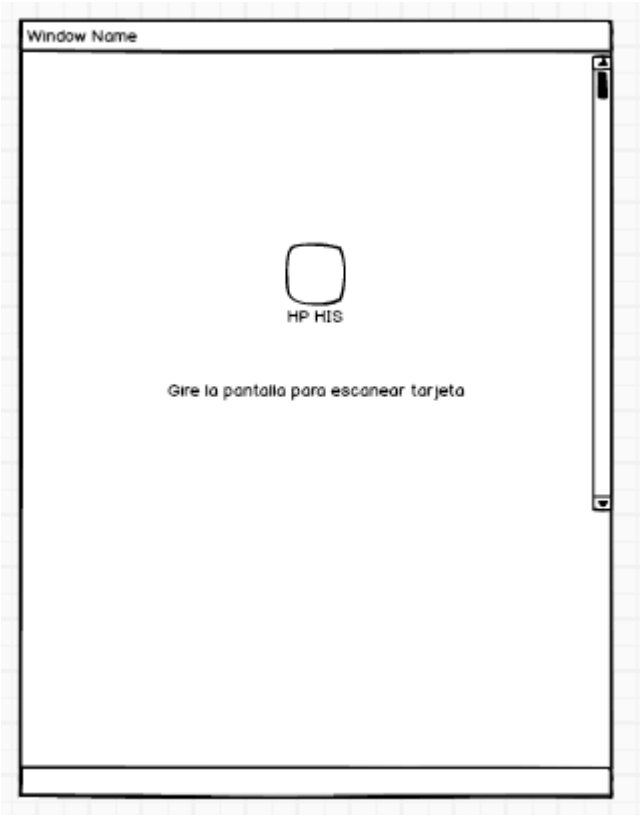


Ilustración 10 Imagen Prototipo Pantalla Principal

2. **Escanear tarjeta:** Esta pantalla es la que se corresponde con el escaneo de la tarjeta, y ha resultado muy representativa

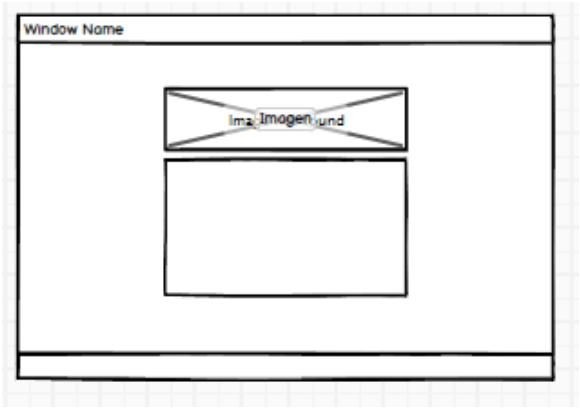


Ilustración 11 Imagen Prototipo Escaneo

3. **Ver información del paciente:** esta ventana se corresponde con la información que devuelve el sistema después de haber escaneado la tarjeta.

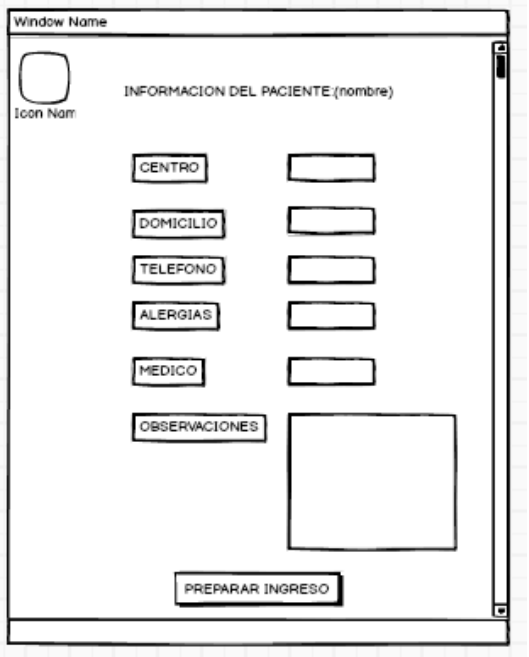


Ilustración 12 Imagen Prototipo Ver Información

4. **Introducir información del suceso:** esta intuitiva pantalla sirve para que el usuario pueda introducir de manera rápida la información necesaria.

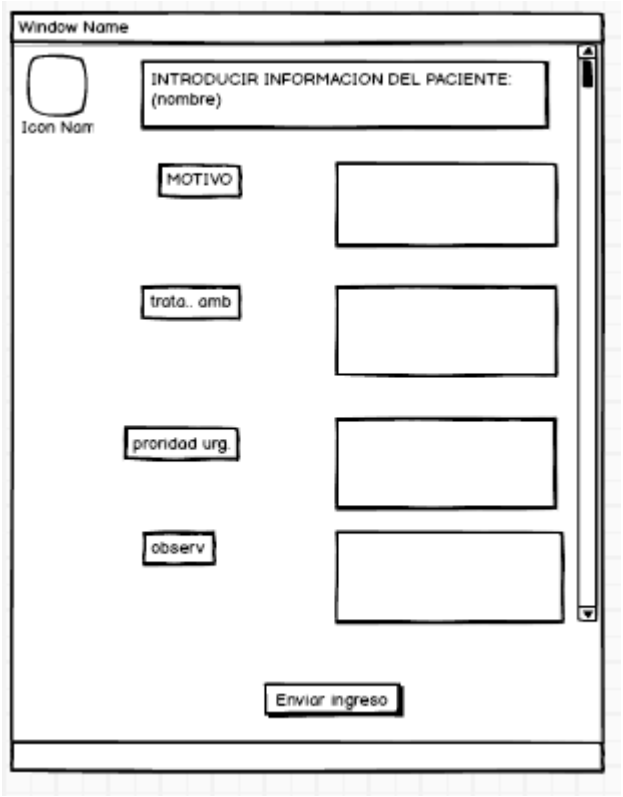


Ilustración 13 Imagen Prototipo Introducir Información

5. **Respuesta del sistema:** esta pantalla se pensó para avisar al usuario de si los datos han sido sincronizados correctamente o no. Se decidió sustituirla por un Toast.

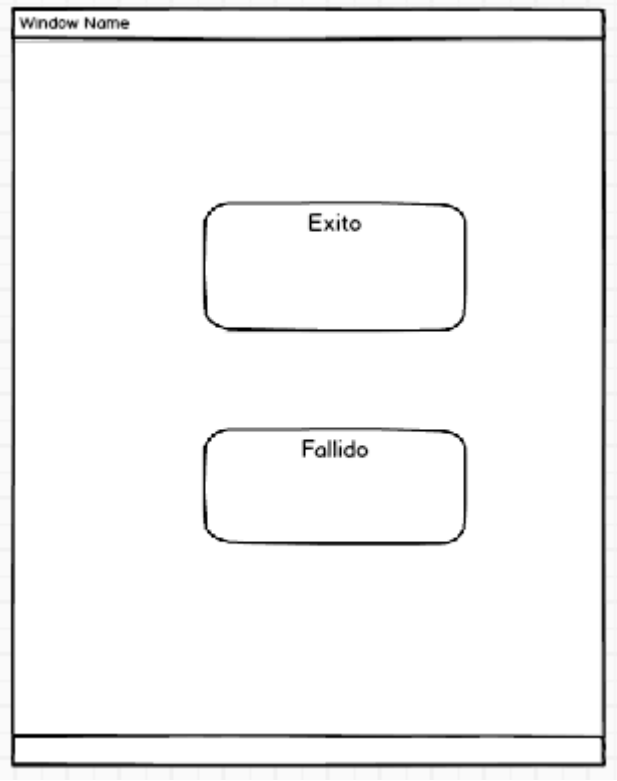


Ilustración 14 Imagen Prototipo Respuesta

Si nos fijamos en el resultado final del producto podemos ver que el número de pantallas ha cambiado y se ha añadido alguna funcionalidad nueva con el avance del proyecto como llamar al usuario del contacto y el control de acceso.

5. DISEÑO DEL SISTEMA

5.1. Introducción

A continuación se detallarán las decisiones de diseño de la aplicación, es decir la arquitectura física del sistema, la persistencia de datos, el uso de diversos patrones y la interfaz gráfica de usuario.

5.2. Decisiones de diseño

En el presente proyecto se ha decidido para estructurar el sistema, emplear un modelo Cliente-Servidor.

En la siguiente figura podemos ver una distribución de las principales partes de la aplicación.

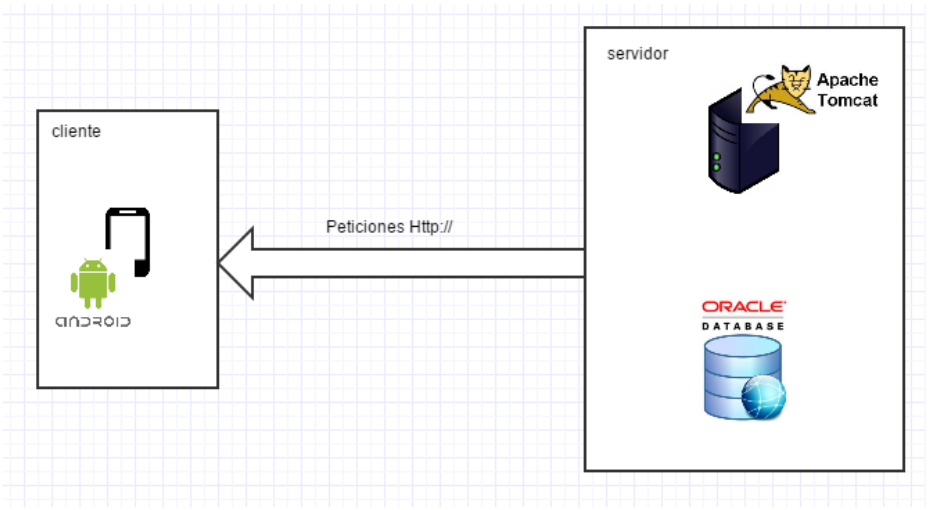


Ilustración 15 Arquitectura Aplicación Android

Este sistema se divide en dos partes que se comunicarán entre sí:

- El servidor está compuesto por un servidor web y una base de datos Oracle, atenderá las peticiones y recogerá los datos para proporcionar al cliente la información solicitada.

- El cliente está compuesto por una aplicación Android, que será la encargada de realizar peticiones al servidor para obtener los datos necesarios, que más adelante se mostrarán al usuario.

La parte de HCIS presenta una estructura similar a la aplicación móvil.

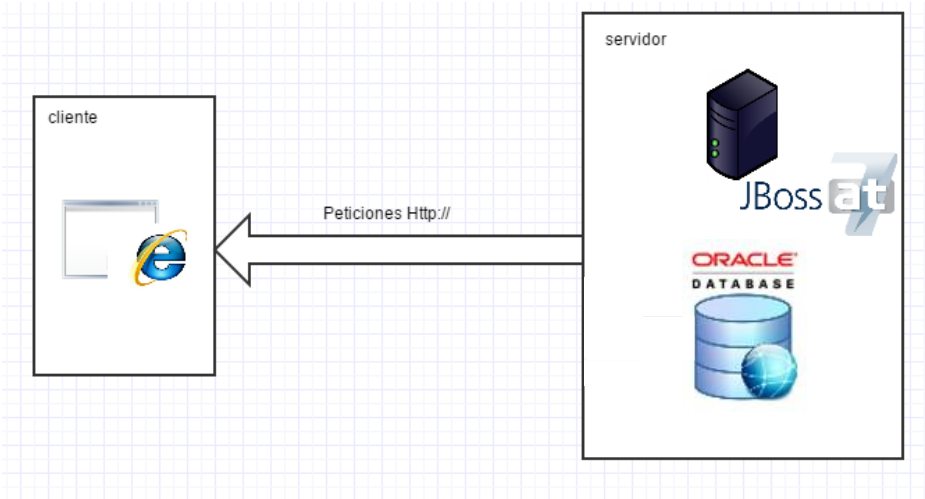


Ilustración 16 Arquitectura HCIS

5.2.1. Topología del sistema

Anteriormente decíamos que los sistemas se dividen en dos componentes: aplicación Android como cliente y un servidor atendiendo las peticiones, ambos comunicados entre sí.

En la siguiente imagen podemos ver la arquitectura física de la aplicación móvil.

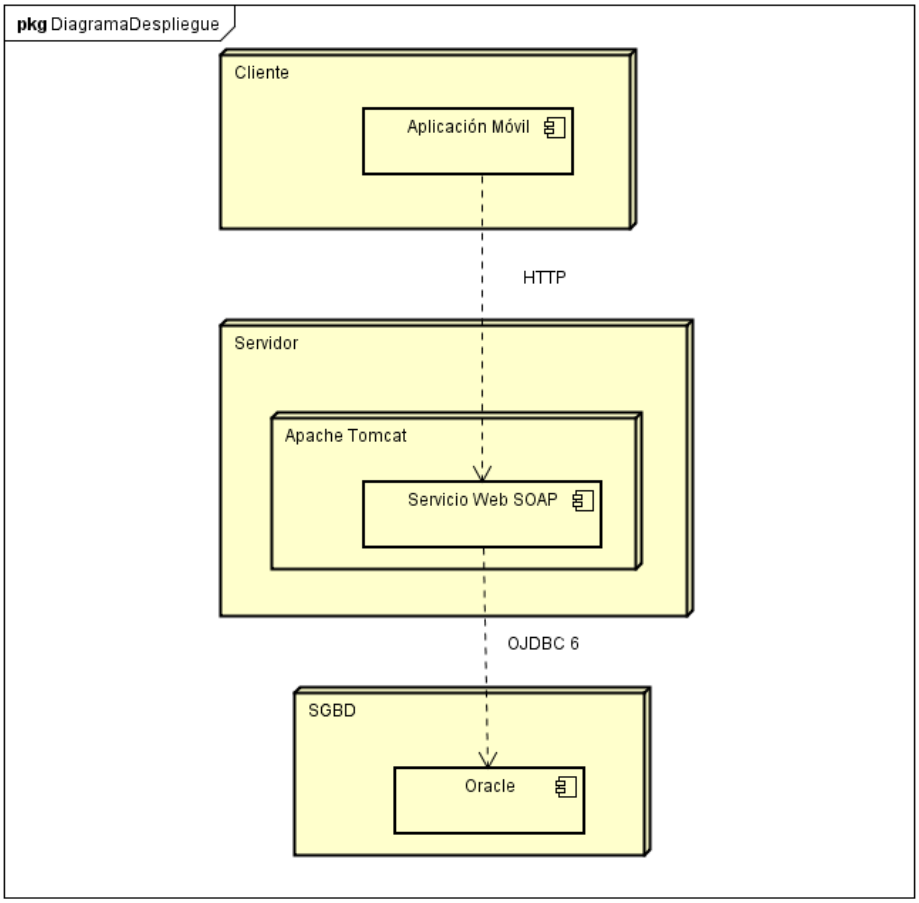


Ilustración 17 Diagrama de Despliegue Apliación móvil

La arquitectura lógica del sistema está constituido por tres componentes:

- En el cliente nos encontramos con la aplicación Android.
- En el lado del servidor nos encontramos con el servidor web que se comunicará con la base de datos y responderá las peticiones de la aplicación móvil. El servidor será un apache Tomcat. A pesar de que Apache Tomcat no es un servidor de aplicaciones puede utilizarse como servidor web por sí mismo
- El sistema gestor de base de datos será el encargado del almacenamiento de los datos, está constituido por una base de datos de tipo Oracle 11G. Más adelante se hablará sobre ello.

En la siguiente imagen podemos ver la arquitectura física de HCIS:

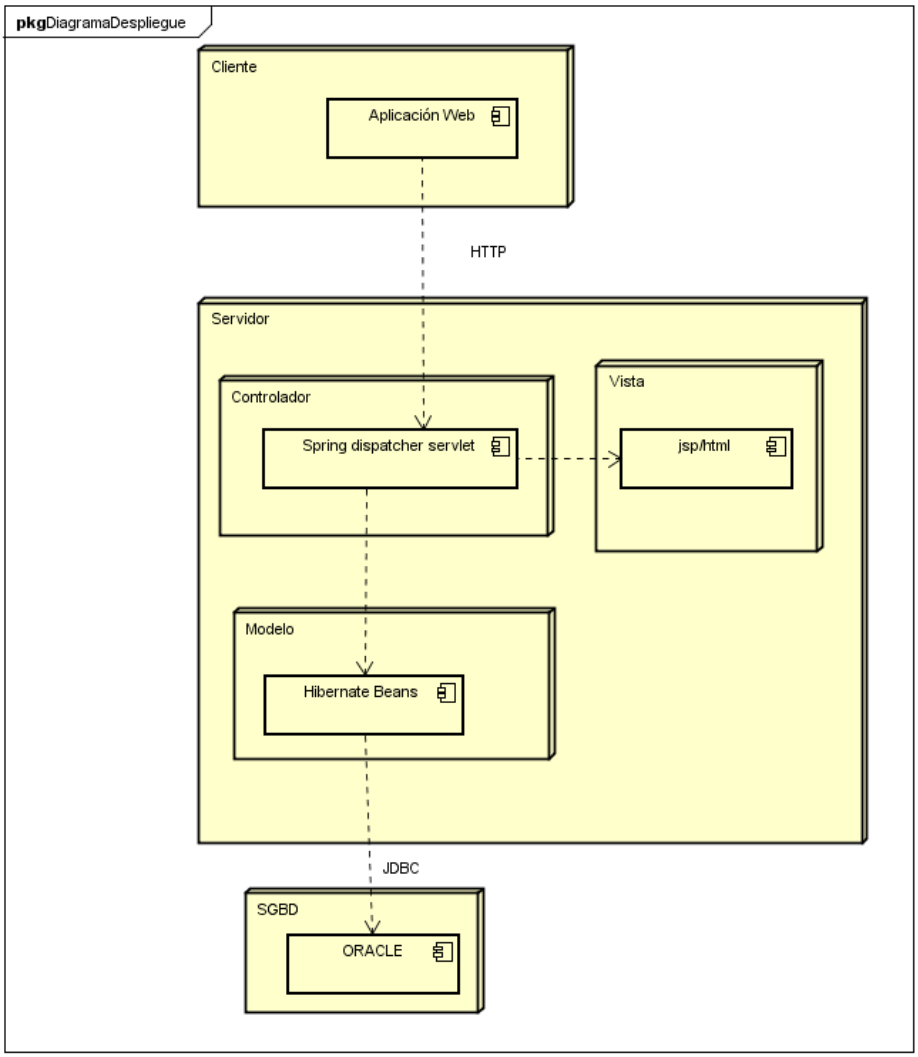


Ilustración 18 Diagrama de Despligue HCIS

En HCIS tenemos el cliente, servidor y el sistema gestor de base de datos.

- En la parte del cliente tenemos la aplicación web que solicitará datos al controlador Spring.
- En el servidor tenemos controlador de Spring este será el encargado de atender las peticiones del cliente, a grandes rasgos contiene una serie de servlets identificados unívocamente con una

vista, que será la que se le devuelva al usuario. En el modelo tenemos Hibernate, que permite manipular los datos en la base de datos operando sobre objetos (POJO).

- En el sistema gestor de base de datos, tenemos la base de datos Oracle 11G.

5.2.2. Comunicación entre aplicación móvil y base datos.

La comunicación con la base de datos, forma un papel fundamental en nuestro proyecto, pues hace de punto de unión entre las dos aplicaciones puesto que es común para la aplicación móvil y HCIS.

La comunicación se realizará mediante servicio web que hará de intermediario para obtener los datos que se requieran.

Para la parte móvil el servicio encargado de recuperar los datos será de tipo SOAP, el servicio retornará un documento XML con la información solicitada, que será parseado mediante DOM.

El servicio web estará implementado en el lenguaje de programación Java debido al tipo de servidor sobre el que estará publicado.

5.2.3. Persistencia de datos.

La persistencia de datos será gestionada por un sistema gestor de base de datos de Oracle. La base de datos proporcionada por Hewlett-Packard almacena todos los datos de los hospitales asociados al sistema HCIS.

Estos datos se almacenarán en un servidor remoto al que la aplicación deberá conectarse.

5.4.3.1. Datos a almacenar

Dentro de la base de datos que se nos proporciona, se ha tomado la decisión de crear una tabla EPISODIO_AMBULANCIA, donde se almacenará la información importante relativa al suceso que acontece la emergencia, dicha información se describe a continuación:

- Codigo_pac: identificador de la clave primaria de la tabla, servirá para localizar datos de un paciente en otras tablas.
- Tarjeta: número de tarjeta de la seguridad social del paciente que está siendo atendido.
- Prioridad: nivel de prioridad.
- Tratamientoamb: tratamiento que se le está dando al paciente en la ambulancia de camino al hospital.
- Fech_env: fecha en la que se han enviado los datos.
- Hora_env: hora exacta a la que se han enviado los datos.

Está información será utilizada por HCIS y servirá para preparar la llegada del paciente a urgencias antes de que esté llegue.

A continuación se muestra un diagrama modelo relacional de las tablas que hemos usado en este proyecto en el que podemos observar de manera visual la estructura y de qué manera conectan estas tablas.

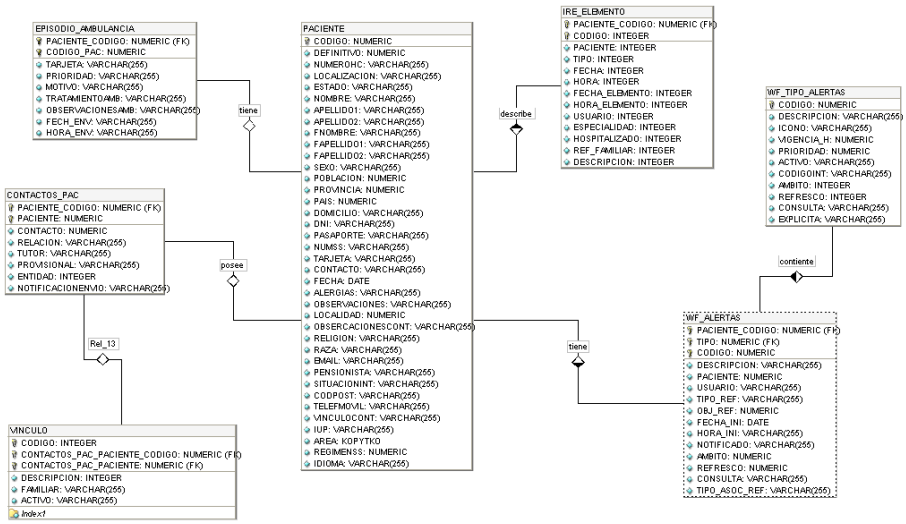


Ilustración 19 Esquema de Tablas del Sistema

5.3. Patrones de diseño

En este punto se detallará los patrones de diseño utilizados. Éstos nos proporcionan soluciones generales para problemas concretos.

5.3.1. MVC (Aplicación Android y HCIS)

Tanto en la aplicación móvil como en la aplicación web utilizamos el patrón de diseño modelo vista controlador (MVC).

Este tipo de diseño separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones

En la siguiente figura podemos ver su comportamiento:

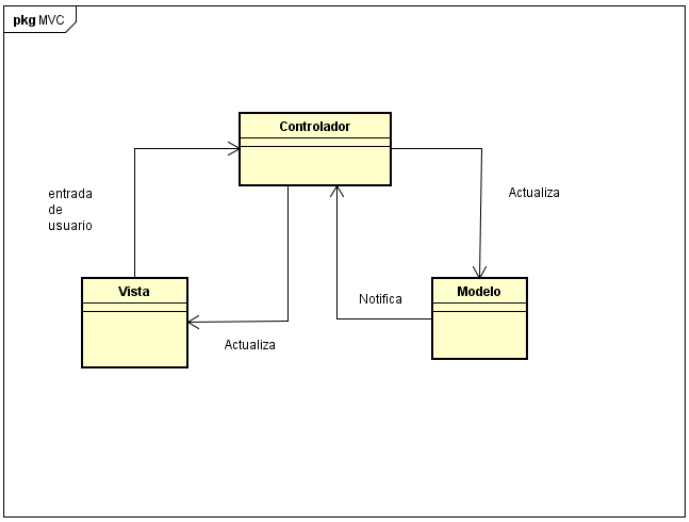


Ilustración 20 Patrón MVC

Nos encontramos con tres componentes con funciones bien diferenciadas entre sí:

- Modelo: son las representaciones de la información con la que trabaja la aplicación. El modelo se creará a partir de los ficheros XML recuperados mediante el servicio web.
- Vista: es la interfaz con la interactúa el usuario. Las vistas se implementan mediante XML.
- Controlador: Los controladores son propiamente las actividades y clase creadas en Java. Manejan la información y la despliegan para que la vista pueda mostrarla.

5.3.2. DAO (HCIS)

Es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo.

En la siguiente imagen podemos ver la estructura del patron dao, mas adelante veremos como está implantado en nuestro proyecto

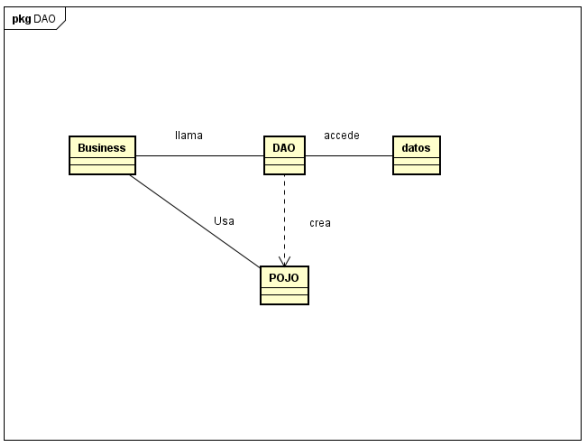


Ilustración 21 Patrón DAO

En nuestro proyecto tenemos la siguiente estructura:

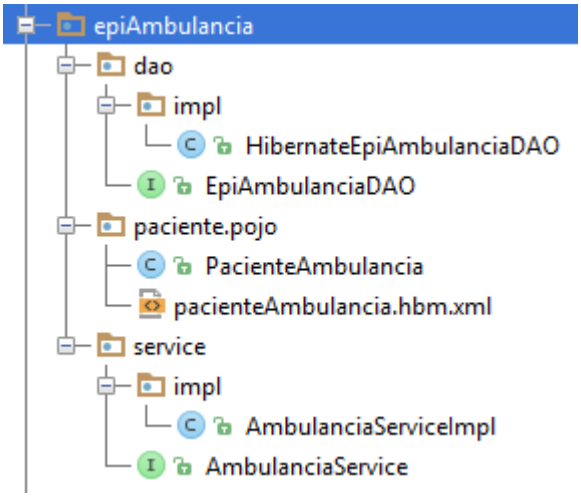


Ilustración 22 Estructura DAO en proyecto

Estos 4 componentes se definen de la siguiente manera:

- DAO: en nuestro caso tenemos EpiAmbulaciaDAO como interfaz que define los métodos que implementa HibernateEpiAmbulanciaDAO, El servicio llamará a esta interfaz y el DAO accederá a los datos y creará objetos (POJO) que podrán ser usados por la aplicación.
- POJO: son objetos creados por el DAO, en nuestro caso PacienteAmbulancia, que contiene los atributos del paciente.
- Business: será el servicio que se encargue de comunicarse con la interfaz DAO y utilizar los objetos.
- Datos: es la base de datos en nuestro caso Oracle 11G.

5.4 Interfaz de Usuario

Nuestra aplicación debe presumir de poseer una interfaz simple y de fácil manejo, con el fin de garantizar la rapidez del servicio.

Se han tenido en cuenta los siguientes aspectos:

- Facilidad y comprensión: facilidad de uso e intuición para que no quepan dudas de su utilización.

- Unidireccional: esta característica permite va a permitir al usuario no perderse por actividades de la aplicación, Algunas aplicaciones presentan tantas bifurcaciones resulta complicado saber en qué punto de la aplicación nos encontramos.
- Diseño responsivo: el diseño de la aplicación deberá adaptarse a diferentes tamaños de pantallas.
- Diseño corporativo: La aplicación tendrá los colores corporativos de la empresa Hewlett-Packard.

5.5. Diagrama de paquetes

En este punto se mostrarán los diagramas de paquetes de la aplicación móvil y HCIS.

5.5.1. Diagrama de Paquetes aplicación móvil

En la siguiente figura se muestra el sistema dividido en paquetes con el objetivo de facilitar la compresión del mismo.

Tenemos tres paquetes fundamentales escaneo que abarca toda la parte del escaneo y procesamiento de la imagen, GestionSesion que abarca todo el tema de la sesión en la aplicación móvil y el paquete sincronización que estará vinculado a la sincronización con la base de datos.

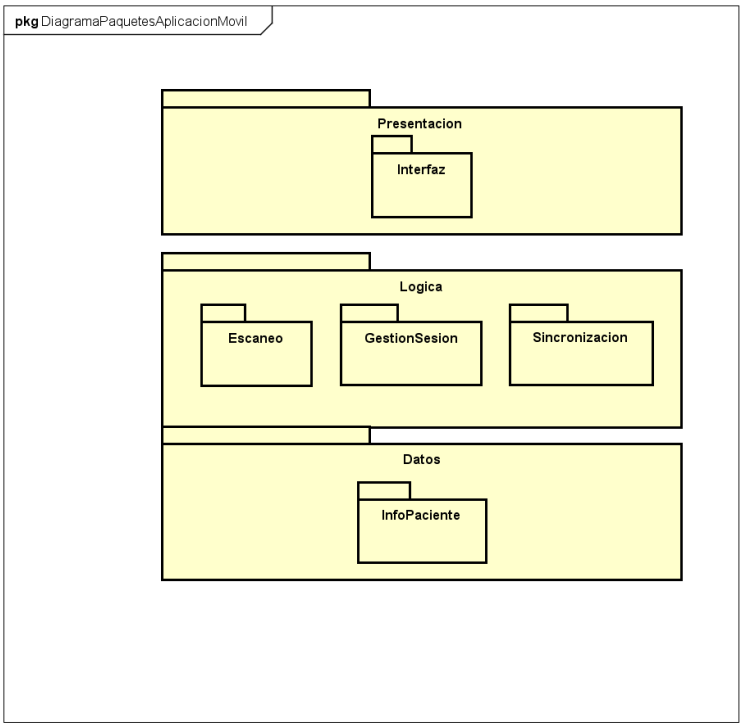


Ilustración 23 Diagrama de Paquetes Aplicación móvil

5.5.2. Diagrama de Paquetes HCIS

Este es el diagrama de paquetes de HCIS, en el que podemos ver que tenemos un único paquete que es el de Urgencias, más adelante se verán las clases que hay dentro de este paquete.

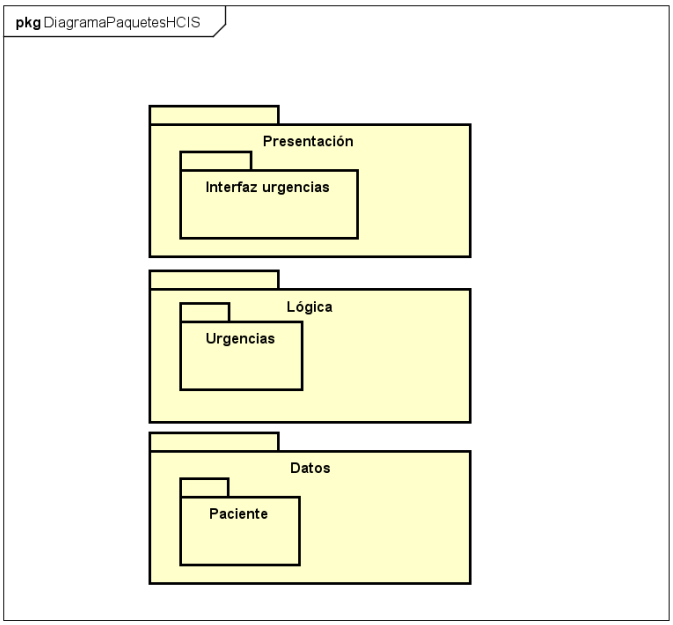


Ilustración 24 Diagrama de Paquetes HCIS

5.6. Diagrama de secuencia

En este apartado se presentan los diagramas de secuencia para el diseño realizado.

5.6.1 Diagramas de secuencia Aplicación móvil

Diagrama de secuencia	Figura
IniciarSesion	25
CerrarSesion	26
IntroducirTarjeta	27
EscanearTarjeta	28
RealizarLLamada	29

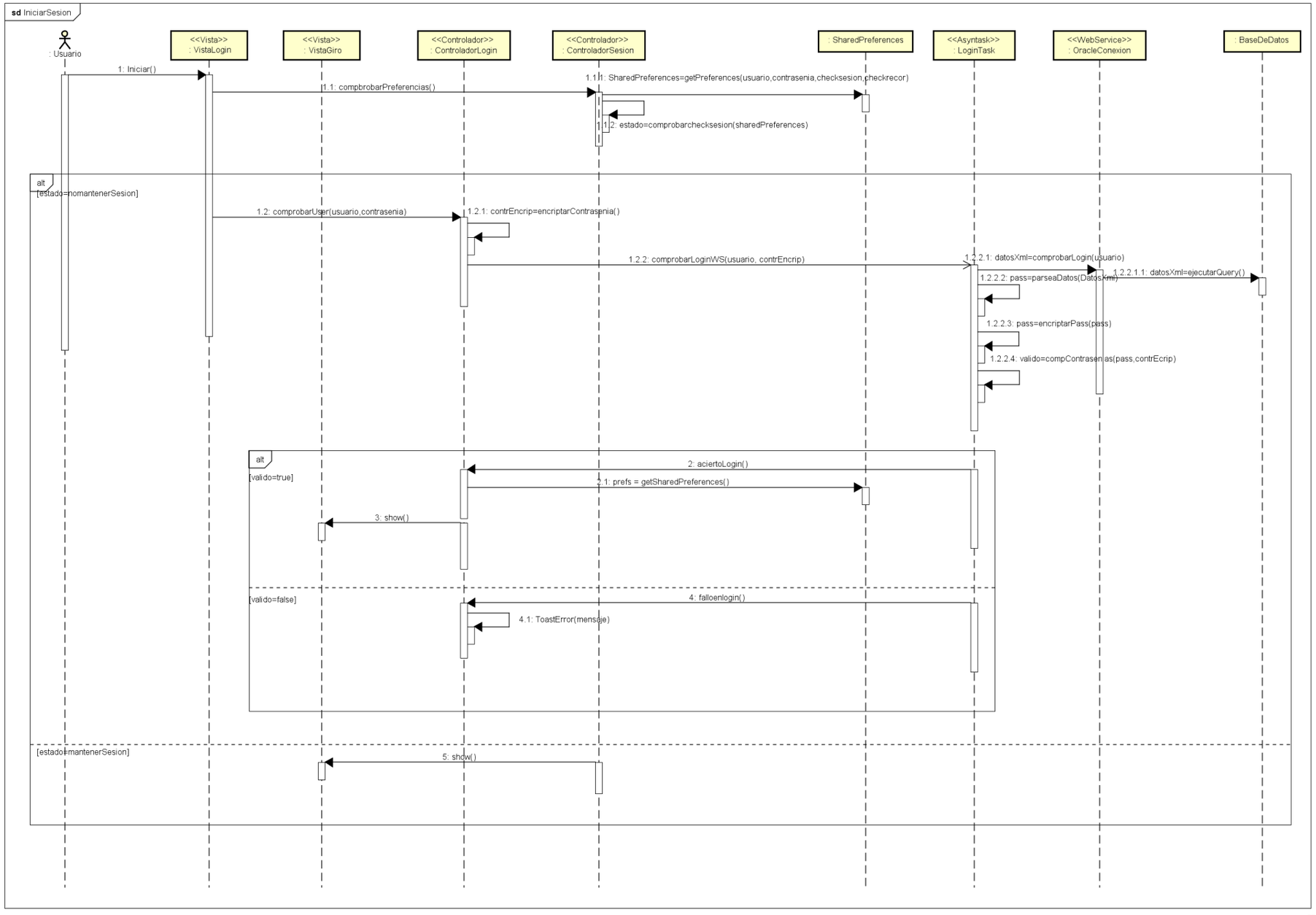


Ilustración 25 Diagrama de Secuencia IniciarSesion

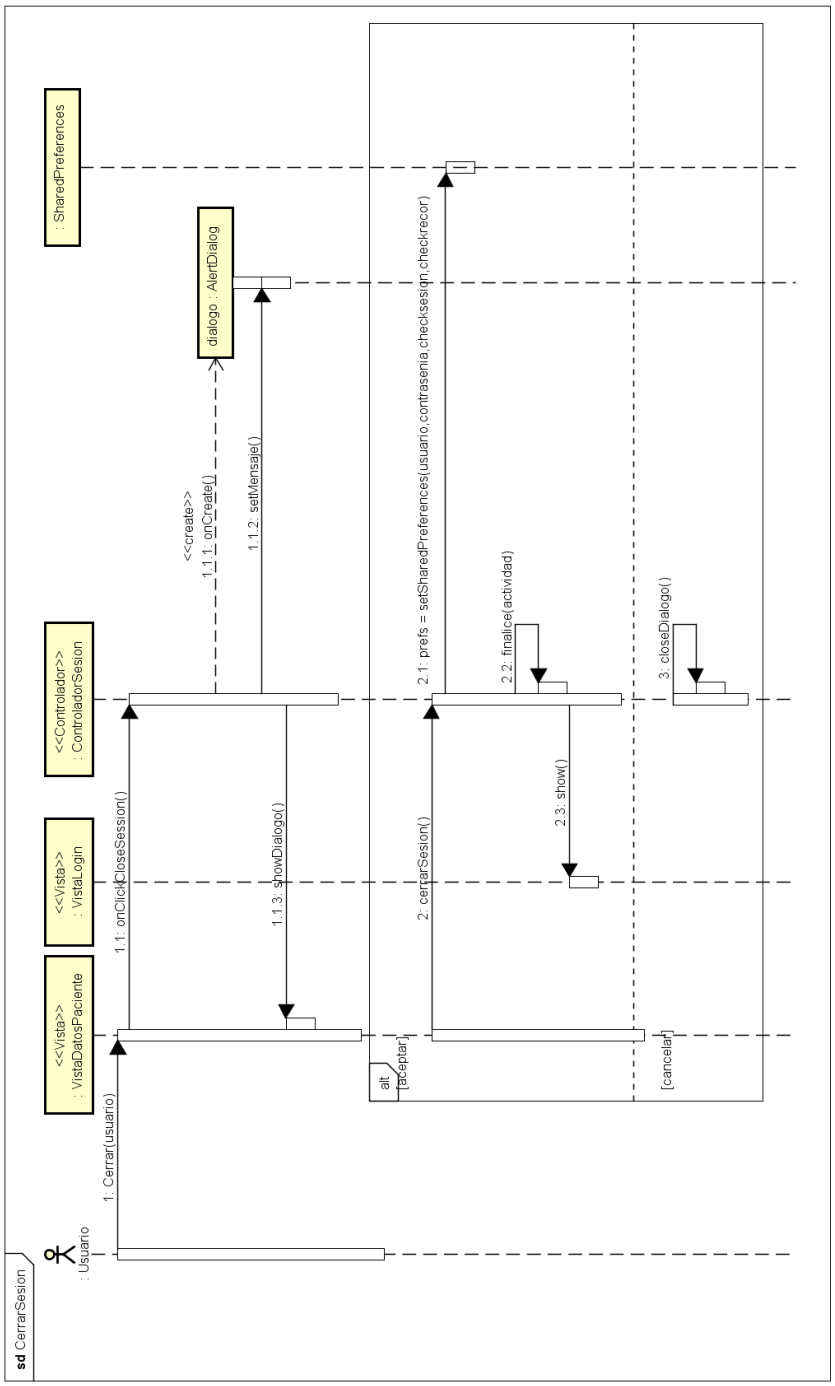


Ilustración 26 Diagrama de Secuencia CerraSesion

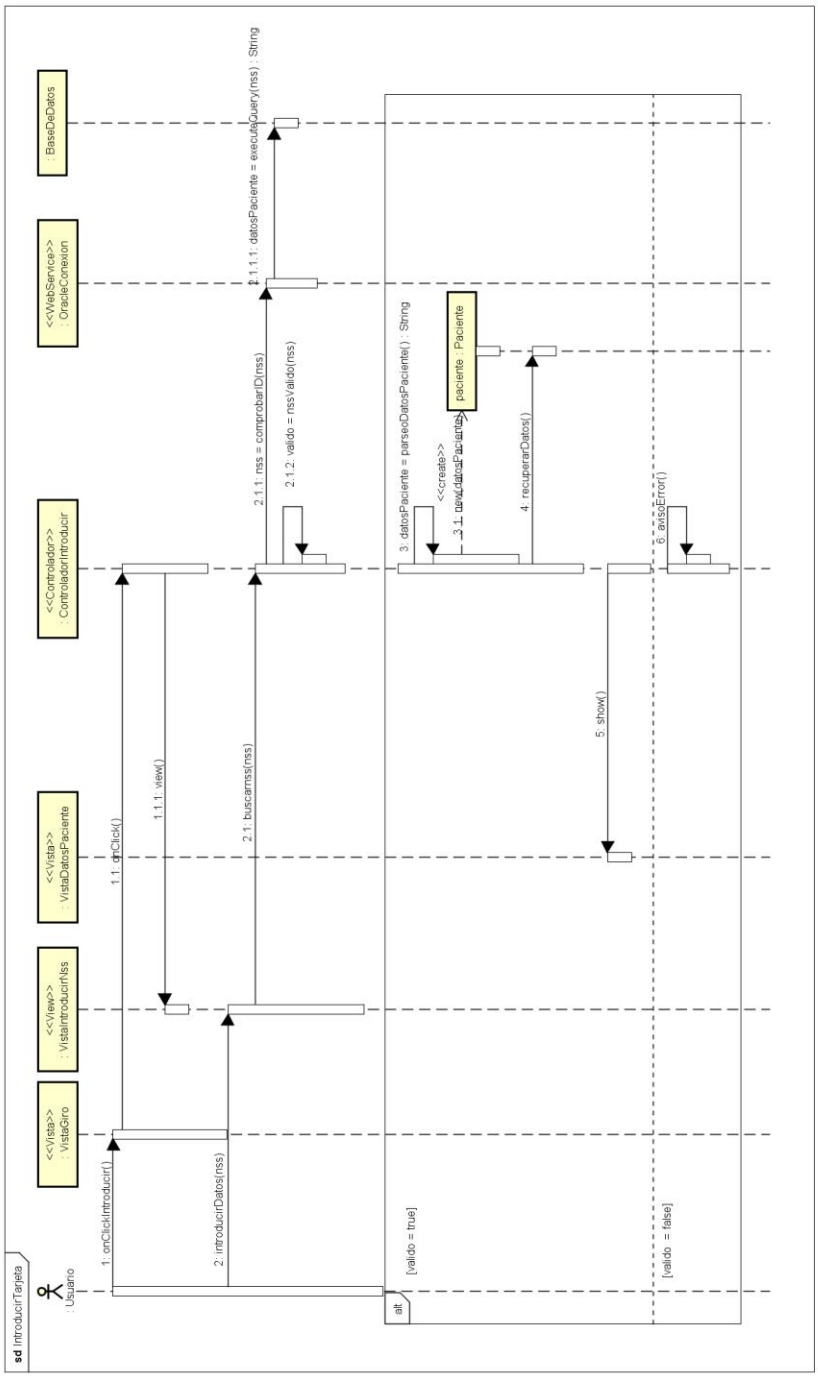


Ilustración 27 Diagrama de Secuencia IntroducirTarjeta

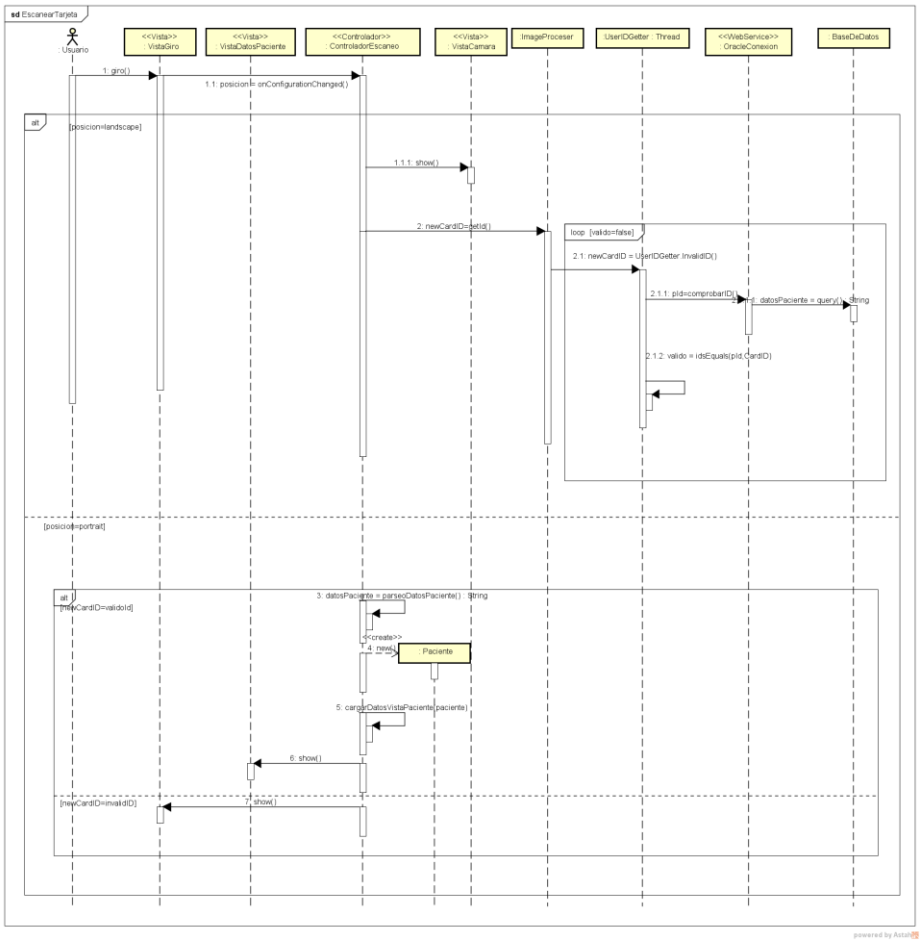


Ilustración 28 Diagrama de Secuencia EscanearTarjeta

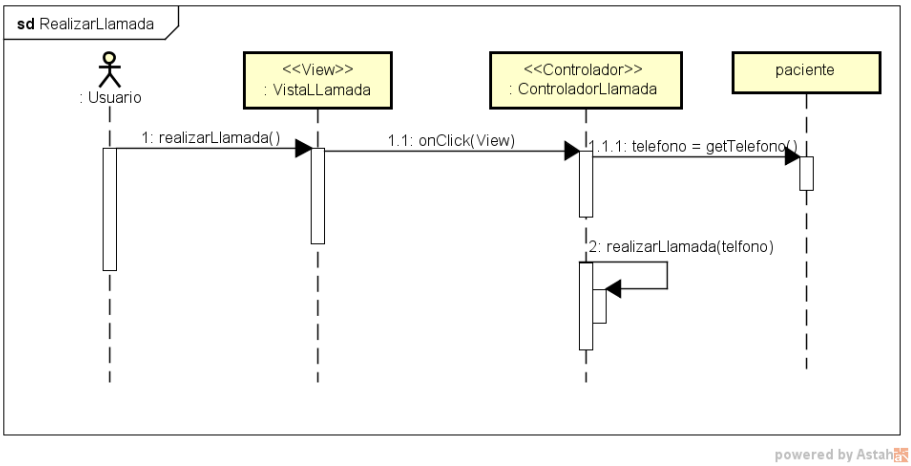


Ilustración 29 Diagrama de Secuencia RealizarLLamada

5.6.2. Diagrama de secuencia HCIS

A continuación se muestra el diagrama de secuencia para el caso de uso ConsultaAmbulancia.

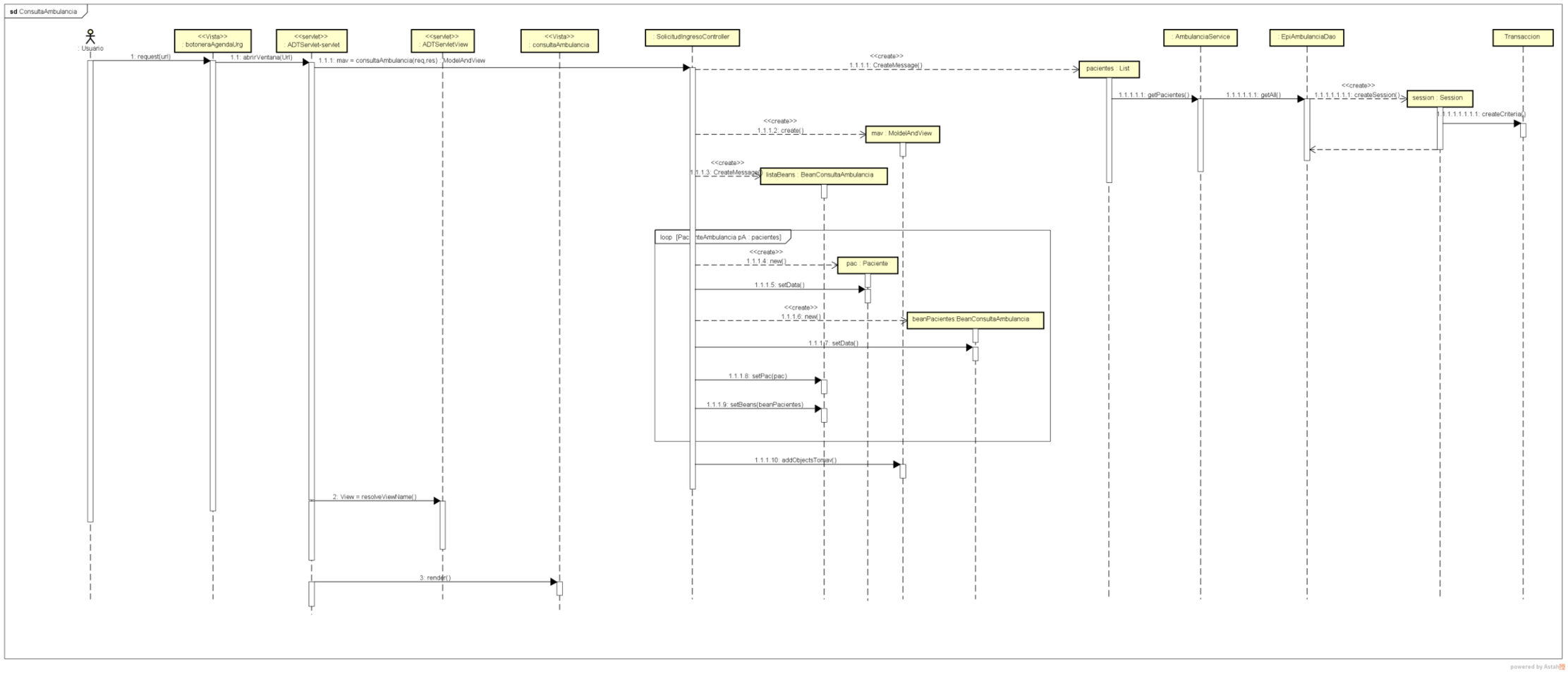


Ilustración 30 Diagrama de Secuencia ConsultaAmbulancia

5.7. Diagramas de Clases de diseño

En este punto se muestran los diagramas de clases de diseño de la aplicación móvil y de HCIS

5.7.1. Diagrama de clases de Aplicación móvil

Paquete	Figura
GestionSesion	31
Sincronizacion	32
Escaneo	33

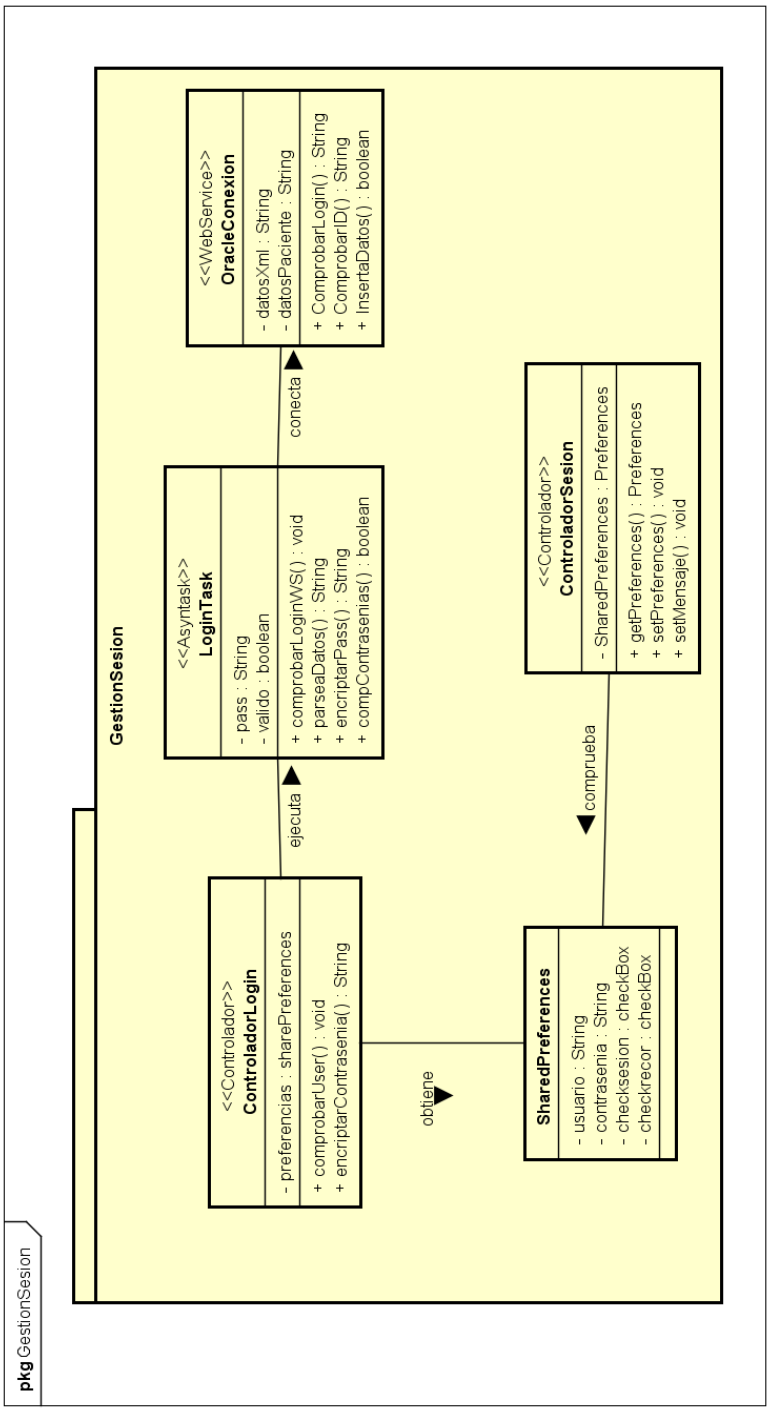


Ilustración 31 Diagrama de Clases paquete GestionSesion

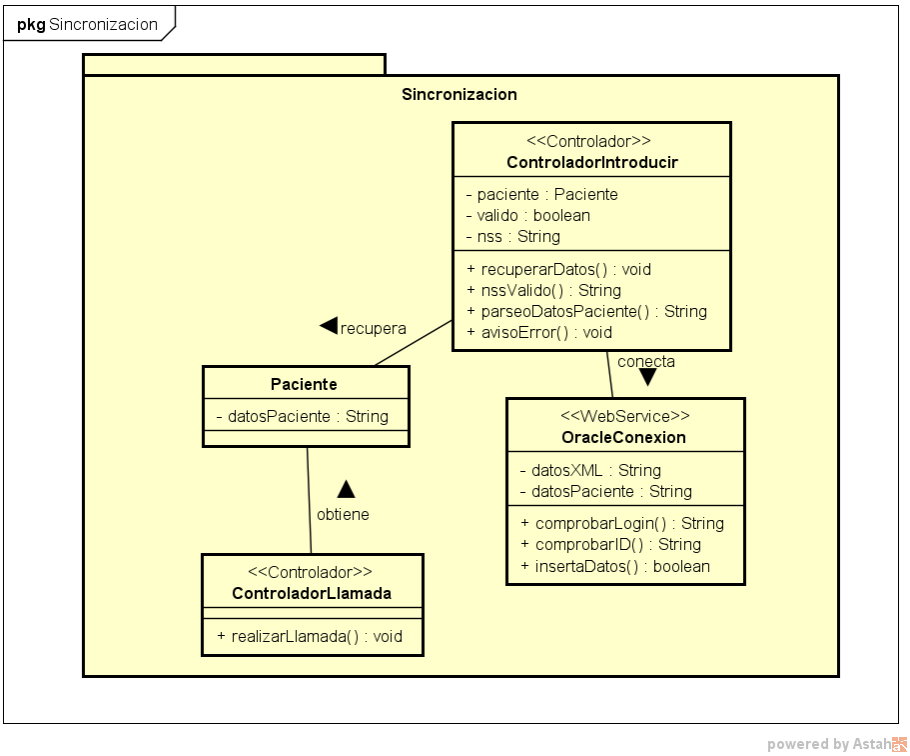


Ilustración 32 Diagrama de Clases paquete Sincronizacion

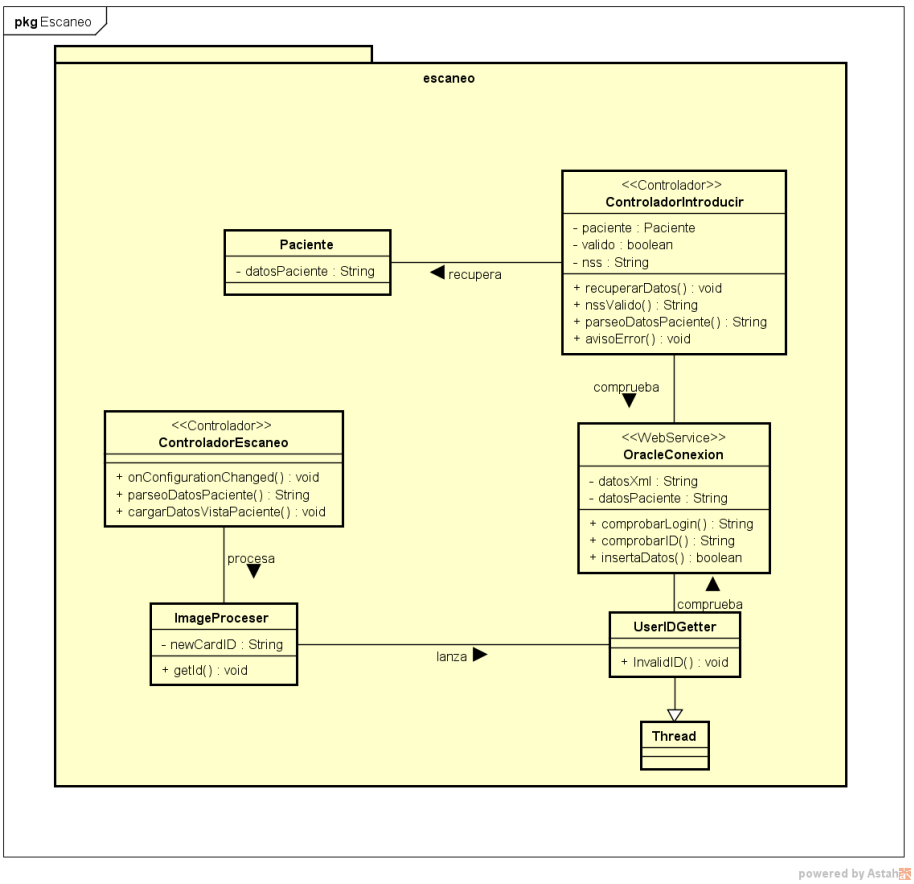


Ilustración 33 Diagrama de Clases paquete Escaneo

5.7.2. Diagrama de clases HCIS

En la siguiente ilustración podemos ver cómo están conectadas las clases dentro del paquete “Urgencias”.

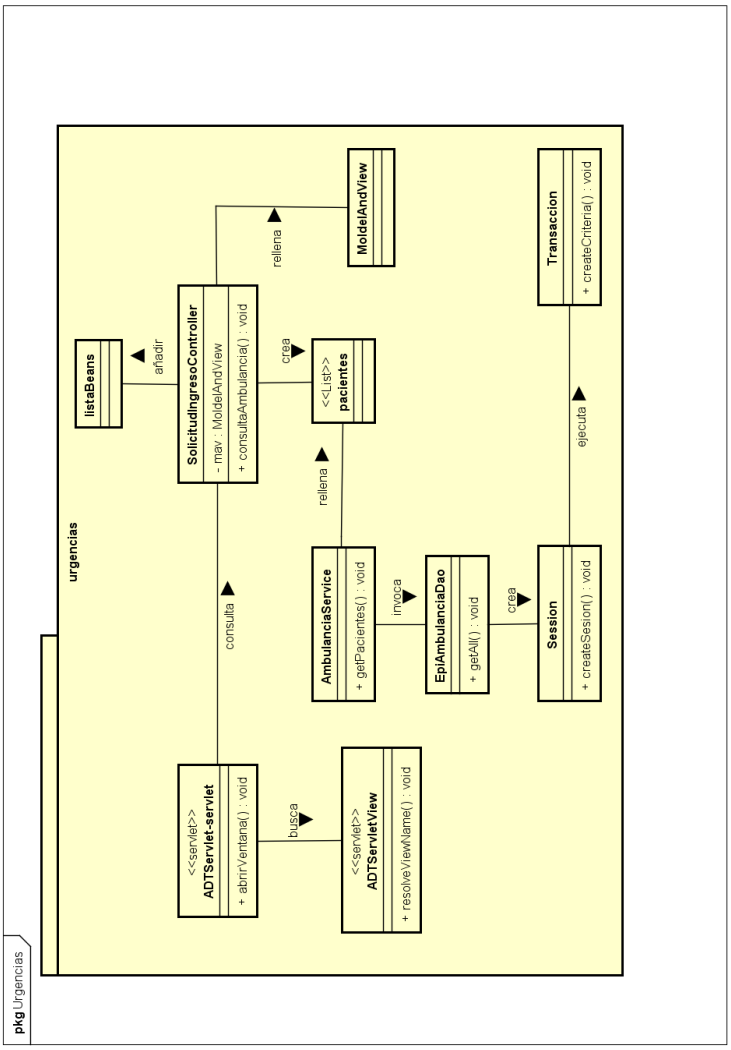


Ilustración 34 Diagrama de Clases paquete Urgencias

6. IMPLEMENTACIÓN

6.1. Introducción

En este capítulo se comentarán como se han llevado a cabo el desarrollo de las decisiones tomadas en el análisis y el diseño.

Los lenguajes utilizados para el desarrollo del sistema son Java, XML y JSON.

También se comentarán las librerías y APIS utilizadas.

6.2. Librerías y APIS utilizadas

Durante el desarrollo del sistema se han utilizado diversas librerías y frameworks disponibles para facilitar el trabajo al alumno.

6.2.1. Servicio Web

El servicio web ha sido implementado en JAVA, se han utilizado librerías que nos permitan leer o generar mensajes SOAP para la invocación de métodos remotos, como es el caso de la API JAX-WS

JAX-WS es una especificación estándar de Sun Microsystems, pero no todos los servidores de aplicaciones utilizan esta librería para gestionar los Servicios Web.

Se ha utilizado KSOAP2 que es una librería ligera y eficiente, debido a sus características resulta imprescindible para su utilización en entornos de aplicaciones móviles. Desde la página del proyecto hospedado en Google Code podemos bajarnos la última versión de la librería para incluir en nuestro proyecto de Eclipse.

6.2.2. Aplicación Android

Para el desarrollo de la aplicación Android se ha utilizado Java y C++, para el procesamiento y XML para el diseño de la interfaz gráfica.

Se ha utilizado OpenCV (Open source Computer Vision library) es una librería Open Source desarrollado por Intel para el campo de la Visión por Computador.

Escrita en un optimizado C/C++, se aprovecha también de multiprocesamiento.

Open CV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X y Windows. Así como también, podemos encontrar interfaces para C/C++, Java y Python.

- **FEATURES2D**: detectores de características, descriptores y comparadores.
- **OBJDETECT**: detección de objetos e instancias de las clases predefinidas, como por ejemplo: caras, ojos, gente, coche, etc.
- **HighGUI**: Interfaz gráfica de OpenCV y funciones que permiten importar imágenes y video.
- **ML**: Machine learning para clasificaciones estadísticas, cálculos regresión, clustering (algoritmos de agrupamiento), etc.
- **GPU**: algoritmos acelerados por hardware para distintos módulos de OpenCV.
- Otros módulos de ayuda…

TESSERACT es un motor OCR (Optical Character Recognition), libre bajo licencia Apache, escrito en C++, desarrollado inicialmente por HP (1985-1995) y actualmente por Google tras su liberación en 2005. Se ha utilizado para poder interpretar el texto de la imagen que se recoge con OpenCV.

Puede procesar inglés, francés, italiano, alemán, español, portugués y holandés, y puede ser entrenado para funcionar con otros idiomas.

Se ha utilizado NDK para poder compilar el código de OpenCV y Tesseract.

El NDK de Android es un conjunto de herramientas que permiten embeber código máquina nativo compilado en lenguajes C y/o C++. Las instrucciones nativas se ejecutan sin pasar por la máquina virtual, gracias a esto podríamos programar una aplicación 100% nativa. Es decir: incluyendo actividades codificadas completamente en C.

6.3. Seguridad en Android

La seguridad es un aspecto clave de todo sistema.

Android propone un esquema de seguridad que protege a los usuarios, sin la necesidad de imponer un sistema centralizado y controlado por una única empresa. La seguridad en Android se fundamenta en los siguientes tres pilares:

1. Android puede impedir que las aplicaciones tengan acceso directo al hardware o interfieran con recursos de otras aplicaciones.
2. Toda aplicación ha de ser firmada con un certificado digital que identifique a su autor. La firma digital también garantiza que el fichero de la aplicación no ha sido modificado.
3. Si queremos que una aplicación tenga acceso a partes del sistema que pueden comprometer la seguridad del sistema hemos de utilizar un modelo de permisos, de forma que el usuario conozca los riesgos antes de instalar la aplicación.

Permisos que presenta nuestra aplicación Android:

```
android:name="android.permission.CAMERA"
```

Permite acceder a la cámara del dispositivo

```
android:name="android.permission.WRITE_EXTERNAL_STORAGE"
```

Permite el acceso al almacenamiento externo

```
android:name="android.permission.INTERNET"
```

Permite a la aplicación abrir sockets.

```
android:name="android.permission.ACCESS_NETWORK_STATE"
```

Permite a la aplicación el acceso al estado de la red wiffi

```
android:name="android.permission.CALL_PHONE"
```

Permite a la aplicación el acceso a las llamadas del teléfono

6.4. Spring e Hibernate en HCIS

Como se comentó en el capítulo 2 en el apartado Contexto Tecnológico, Se utilizaron Spring e Hibernate para el desarrollo de la parte Web del proyecto.

Spring sigue el modelo MVC como se ha mencionado en capítulos anteriores cuando hemos hablado de los patrones utilizados.

Lo primero que hemos hecho para utilizar este modelo, ha sido editar un servlet y se ha añadido como propiedad la una referencia con la ruta de que será nuestra nueva ventana con extensión .adt y el nombre del servicio que se encargará de lanzar la ventana, dentro de un been (contenedor), la vista se comunicará con este fichero cuando se invoque una nueva ventana.

El been hace referencia a una clase que actúa como modelo y contendrá el servicio que invocará la vista .jsp que hemos creado previamente.

En esta clase se encuentran varios servicios pertenecientes a otros beens, aquí hemos hecho referencia a nuestro servicio que es básicamente una interfaz que devuelve una lista de pacientes, Aquí es donde entra en juego Hibernate que comentaremos más adelante como funciona.

En la clase citada anteriormente se ha creado un método asociado a este servicio que recibe una petición y crea un objeto MVC (Model And View), carga una los pacientes recogidos mediante el servicio, y devuelve al servlet un objeto MVC con los atributos que se le han añadido.

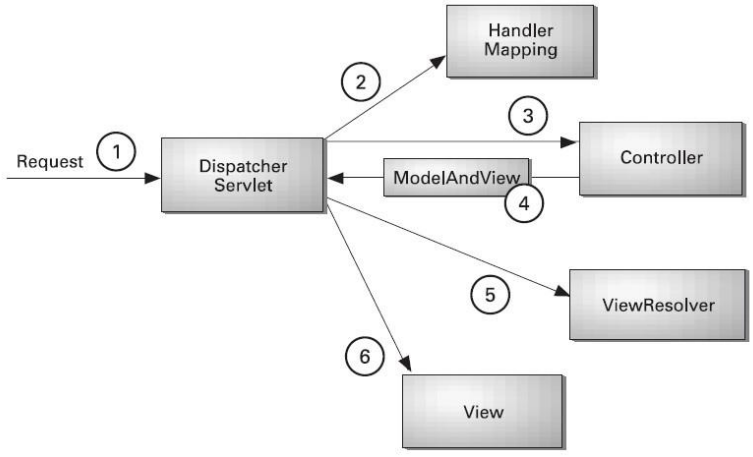


Ilustración 35 Secuencia en Spring

Añadimos en otro servlet que funciona como solucionador de vistas la ruta a nuestro .jsp con su ruta real dentro del proyecto y con su extensión .jsp. Ahora el primer servlet que funciona como despachador cargará la vista referenciada.

Para la parte de persistencia de la base de datos con Hibernate lo primero que se ha hecho ha sido crear un POJO del objeto (PacienteAmbulancia) que vamos a utilizar con todos sus atributos.

Una vez hecho, necesitamos una interfaz con métodos para el servicio junto a su implementación así como la interfaz DAO y su implentación.

Para poder ejecutar las sentencias y recuperar los datos se crea un fichero con la extensión .hbm.xml con el fin de mapear la tabla de la base de datos en los atributos del POJO que hemos creado.

La siguiente imagen resume de manera muy simple este sistema completo:

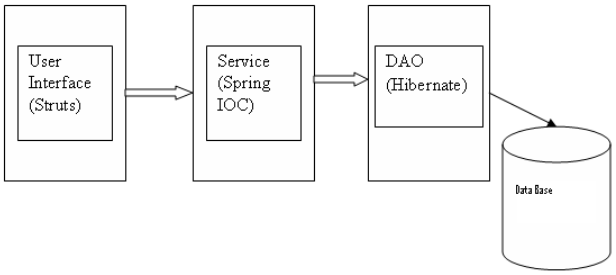


Ilustración 36 Comunicación usuario-Spring-Hibernate-BD

6.5. Red VirtualBox

Debido a un problema de incompatibilidad entre Oracle 11G y Windows 8, se tomó la decisión de crear una máquina virtual con Windows XP tal y como estaba establecido en la gestión de riesgos.

Lo que se hizo fue descargar una versión de Windows XP e instalar la base de datos.

Para poder acceder a la máquina desde el sistema anfitrión se configuró la red en las opciones de VirtualBox, de tal manera que tomase la red del sistema anfitrión.

Ilustración 37 Configuración VirtualBox

7. PRUEBAS

7.1. Introducción

Especificaremos las pruebas realizadas al sistema con el fin de asegurar que cumple todas las especificaciones marcadas en la fase de análisis

7.2. Casos de prueba

A continuación se exponen los casos de prueba para la aplicación móvil y HCIS

7.2.1. Casos de prueba Aplicación móvil

Cada caso de prueba ha sido probado en dos dispositivos:

Smartphone Samsung Galaxy Grand Neo Android 4.4 (KitKat) Procesador Quad Core 1.2 GHz 1 GB RAM
Sony Xperia E3 Android 4.4 (KitKat) Procesador Qualcomm MSM8926-2 Snapdragon 400 quad-core 1.2GHz 1 GB RAM

A continuación se muestran las distintas pruebas realizadas durante la batería de pruebas en la aplicación móvil:

Identificador	Cp1
Objetivo	Iniciar aplicación
Descripcion	Usuario inicia la aplicación
Entrada	Icono aplicación
Salida	El sistema muestra la pantalla principal de login.
Resultado	Esperado

Identificador	Cp2
Objetivo	Iniciar aplicación sin conexión en el dispositivo
Descripción	El usuario inicia la aplicación
Entrada	Icono de la aplicación
Salida	Dialogo de alerta sin conexión
Resultado	Esperado

Identificador	Cp3
Objetivo	Iniciar sesión login correcto
Descripción	Iniciar sesión en la aplicación móvil
Entrada	Nombre de usuario y contraseña
Salida	Ventana giro de pantalla
Resultado	Esperado

Identificador	Cp4
Objetivo	Iniciar sesión con nombre de usuario erróneo
Descripción	El usuario inicia sesión en la aplicación
Entrada	Nombre de usuario equivocado y contraseña
Salida	Toast fallo en el login
Resultado	Esperado

Identificador	Cp5
Objetivo	Iniciar sesión con nombre de contraseña errónea
Descripción	El usuario inicia sesión en la aplicación
Entrada	Nombre de usuario y contraseña erronea
Salida	Toast fallo en el login
Resultado	Esperado

Identificador	Cp6
Objetivo	Iniciar sesión campos vacíos
Descripción	El usuario inicia sesión en la aplicación
Entrada	Campos vacíos
Salida	Toast campos vacíos
Resultado	Esperado

Identificador	Cp7
Objetivo	Iniciar sesión chequeando recordarme y cerrar sesión y volver a iniciar.
Descripción	El usuario inicia sesión en la aplicación
Entrada	Nombre de usuario, contraseña y check recordarme
Salida	Ventana principal login con campos rellenos
Resultado	Esperado

Identificador	Cp8
Objetivo	Iniciar sesión chequeando recordarme y mantener sesión iniciada y salir y volver a entrar
Descripción	El usuario inicia sesión en la aplicación
Entrada	Nombre de usuario, contraseña, check recordarme y mantener sesión
Salida	Ventana giro.
Resultado	Esperado

Identificador	Cp9
Objetivo	Girar móvil en ventana de giro
Descripción	Lanzar al cámara en ventana de giro
Entrada	Giro pantalla
Salida	Cámara del móvil escaneando
Resultado	Esperado

Identificador	Cp10
Objetivo	Pulsar botón en introducir nss en ventana de giro
Descripción	Abrir ventana para intoducir nss
Entrada	Pulsación de botón
Salida	Ventana introducir nss
Resultado	Esperado

Identificador	Cp11
Objetivo	Introducir nss longitud diferente a 16 dígitos
Descripción	El usuario introduce un nss incorrecto
Entrada	Nss incorrecto
Salida	Toast aviso de longitud
Resultado	Esperado

Identificador	Cp12
Objetivo	Introducir nss vacío
Descripción	El usuario introduce nss vacío
Entrada	Nss vacío
Salida	Toast aviso de nss vacío
Resultado	Esperado

Identificador	Cp13
Objetivo	Introducir datos paciente en la ventana introducir datos paciente
Descripción	El usuario introduce datos del paciente
Entrada	Motivo, tratamiento, observaciones y prioridad
Salida	Toast aviso sincronización.
Resultado	Toast si se ha conseguido sincronizar o si no se ha podido sincronizar

Identificador	Cp14
Objetivo	Introducir datos paciente vacíos en la ventana introducir datos paciente
Descripción	El usuario introduce datos del paciente
Entrada	Datos de paciente en blanco
Salida	Toast aviso datos vacíos
Resultado	Esperado

Identificador	Cp15
Objetivo	Introducir algún dato paciente vacíos en la ventana introducir datos paciente
Descripción	El usuario introduce datos del paciente
Entrada	Algún dato de paciente en blanco
Salida	Toast aviso datos vacíos
Resultado	Esperado

Identificador	Cp16
Objetivo	Cerrar sesión desde cualquier ventana
Descripción	El usuario cierra sesión
Entrada	Cierre de sesión
Salida	Ventana de login
Resultado	Esperado

Identificador	Cp17
Objetivo	Llamar al paciente al sincronizar.
Descripción	El usuario pulsa el botón de llamar
Entrada	Botón de llamar contacto paciente
Salida	Llamada al contacto del paciente
Resultado	Esperado

7.2.2. Casos de prueba HCIS

A continuación se muestran las distintas pruebas realizadas durante la batería de pruebas en la aplicación web HCIS:

Identificador	Cp18
Objetivo	Acceder al área de urgencias y visualizar el botón creado
Descripción	Acceder al área de urgencias
Entrada	Vínculo urgencias
Salida	Vista pacientes en urgencias
Resultado	Esperado

Identificador	Cp19
Objetivo	Comprobar que la ventana creada funciona al pulsar el botón
Descripción	Pulsar botón consulta ambulancia
Entrada	Botón consulta ambulancia
Salida	Ventana con lista de pacientes en ambulancia
Resultado	Esperado

Identificador	Cp20
Objetivo	Ver información de un paciente seleccionado
Descripción	Ver información de un paciente
Entrada	Paciente de lista
Salida	Ventana con la información del paciente
Resultado	Esperado

Identificador	Cp21
Objetivo	Ver información de un paciente sin seleccionarlo
Descripción	Ver información de un paciente
Entrada	Botón ver información paciente
Salida	Alerta selección
Resultado	Esperado

Identificador	Cp22
Objetivo	Ver información de un paciente seleccionando varios pacientes
Descripción	Ver información de un paciente
Entrada	Selección de varios pacientes
Salida	Alerta selección
Resultado	Esperado

Identificador	Cp23
Objetivo	Preparar llegada de un paciente seleccionado
Descripción	Preparar llegada de un paciente
Entrada	Paciente de lista
Salida	Ventana para preparar la llegada del paciente
Resultado	Esperado

Identificador	Cp24
Objetivo	Preparar llegada de un paciente sin seleccionarlo
Descripción	Preparar llegada de un paciente
Entrada	Paciente de lista
Salida	Alerta selección
Resultado	Esperado

Identificador	Cp25
Objetivo	Preparar llegada de un paciente seleccionando varios pacientes
Descripción	Preparar llegada de un paciente
Entrada	Paciente de lista
Salida	Alerta selección
Resultado	Esperado

8. CONCLUSIONES Y TRABAJOS FUTUROS

8.1. Conclusiones

En este aparto se detallan las conclusiones y logros obtenidos durante el desarrollo de este proyecto.

Los requisitos han sido cumplidos, pueden ser mejorables pero las soluciones tomadas se adaptan a ellos.

Se ha logrado una mejora en la gestión de proyectos, ha mejorado la capacidad de planificar los tiempos para la elaboración de proyectos, algo que seguro vendrá bien en el futuro.

Conocimientos obtenidos en el lenguaje Android del que partíamos sin apenas conocimientos previos. Se ha visto mejorado notablemente durante la implementación de la aplicación.

Aprendizaje de nuevas tecnologías y frameworks utilizados como Spring e Hibernate, usadas con bastante frecuencia en proyectos empresariales.

Aprender cómo trabajan las empresas y haber establecido una toma de contacto antes de finalizar mis estudios académicos.

Y por último satisfacción personal con el trabajo hecho.

8.2. Trabajos futuros

Analizando la aplicación después de haberla terminado surgen nuevas ideas que podrían implantarse en la aplicación, entre ellas podemos destacar:

- -Geolocalización para obtener la ubicación de la ambulancia.
- -Mejoras en el rendimiento de OCR
- -Utilización de Ajax en la parte web que avise al enfermero de turno de una emergencia.
- Más plataformas móviles

ANEXO

A: Manual de Usuario Aplicación Móvil

En este apartado se explica cómo utilizar la aplicación.

Tras haber instalado la aplicación en nuestro dispositivo, ésta se iniciará pulsando sobre el icono que se habrá creado en el menú del dispositivo:

Al iniciar la aplicación se comprobará la conexión a internet, si no existe se mostrará un aviso al usuario, la ventana principal es la siguiente:

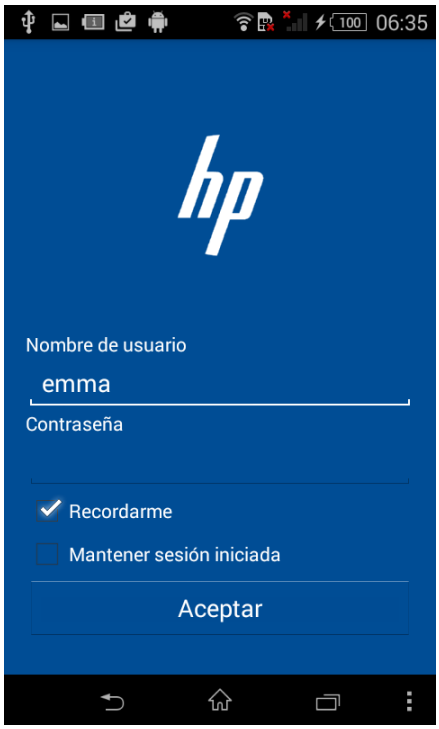


Ilustración 38 Ventana Login

En esta ventana el usuario deberá loguearse y estar registrado en el sistema para poder acceder a las funcionalidades de la aplicación.

El usuario tiene la opción de mantener la sesión y también de recordar su nombre de usuario si marca la opción de “Recordarme”.

Si se ha marcado la opción “Mantener la sesión iniciada” y el usuario sale de la aplicación y después vuelve a iniciar la aplicación no será necesario autenticarse, esto será así hasta que el usuario cierre sesión y desmarque el checkbox correspondiente.

Una vez autenticado en el sistema aparecerá la siguiente ventana:

Ilustración 39 Ventana de Principal

Aquí tiene dos opciones introducir el pulsando el botón introducir NSS o Girar el móvil para empezar a escanear la tarjeta.

Si el usuario decide girar el móvil la cámara se activará. Para escanear la tarjeta lo único que se necesita es un fondo opaco, como puede ser una cartulina negra y enfocar hacia la tarjeta.

Si esperamos unos segundos a que el software identifique los vértices de la tarjeta que aparecerá recuadrada con una figura de color rojo, si la figura apareje roja el sistema nos estará indicando que aún no encuentra el número en la base de datos, podemos verlo en la siguiente imagen:

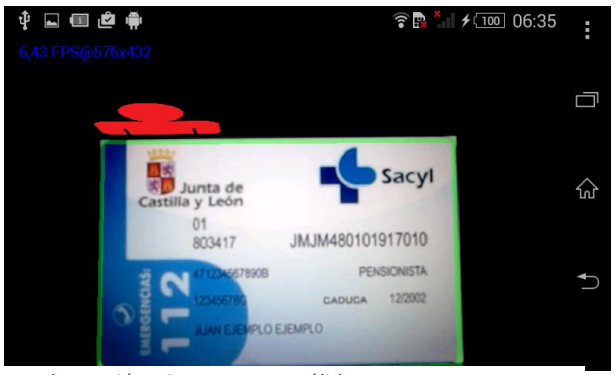


Ilustración 40 Escaneo no válido

Pasados unos segundos si existen coincidencias con el número de tarjeta de la seguridad social reconocido por la aplicación, la figura pasará a ser de color verde y nos indicará que existen coincidencias en base de datos, ahora solo tenemos que girar el móvil para recuperar los datos y poder visualizarlos.

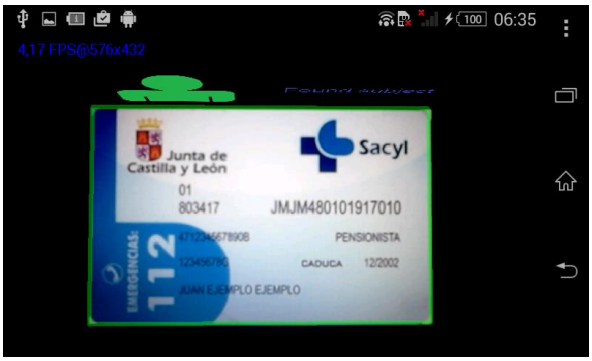
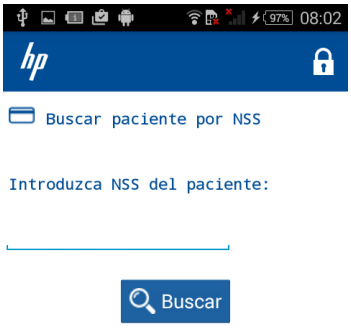


Ilustración 41 escaneo válido

Si abortamos el escaneo, por ejemplo porque la tarjeta está desgastada y los números no son reconocidos por el OCR, podemos volver a la pantalla anterior y pulsar sobre el botón introducir NSS.



La ventana que nos aparecerá será la de la izquierda, en la que el usuario puede introducir el número de la seguridad del paciente y pulsar sobre el botón buscar.

Si existen no existen coincidencias el sistema nos avisará y si existen la aplicación nos mostrara la información relativa al paciente.

En la ilustración 37 podemos ver la información que devuelve el sistema del número asociado a esa tarjeta.

Ilustración 42 Ventana introducir nss

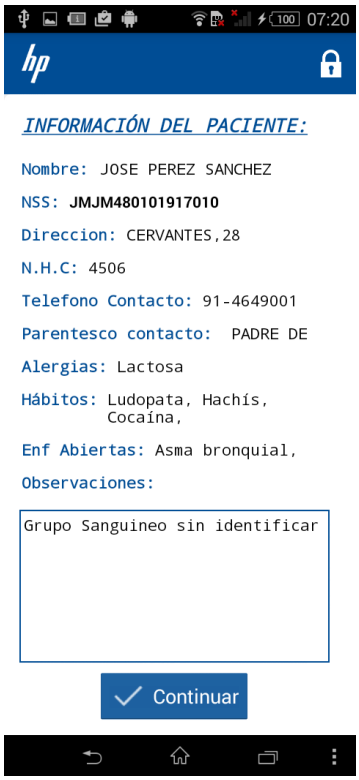


Ilustración 43 ventana información

Podemos ver el nombre del paciente, su número de seguridad social, dirección del paciente, su número de historia clínica, el teléfono de un contacto, el parentesco del contacto, las alergias que puede tener el paciente, hábito del paciente, Enfermedades que tiene el paciente abiertas algunas observaciones que puedan resultar importantes para atender al paciente.

A continuación si pulsamos sobre el botón continuar nos aparecerá una ventana en la que podremos introducir diferente información importante para que el hospital puede ir preparando la llegada del paciente a urgencias y tengan todo listo para su llegada.

En la siguiente imagen podemos ver la información que se puede introducir, es el motivo, el tratamiento que está recibiendo el paciente en la ambulancia, alguna observación que el usuario que está utilizando la aplicación considere oportuna y por último la prioridad del paciente que está asociado al grado de gravedad del paciente.

Una vez introducidos los datos sin dejar ninguno en blanco, el siguiente paso será enviar los datos. Tan solo hay que pulsar en el botón enviar.

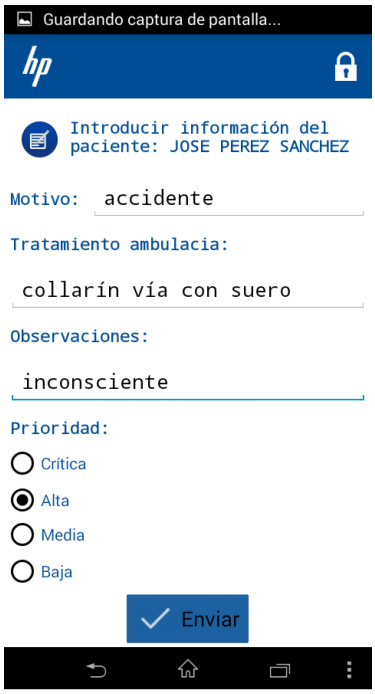
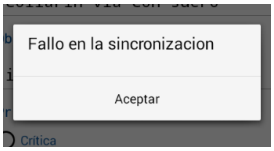
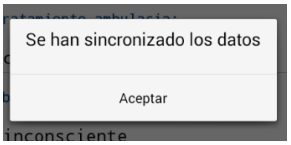


Ilustración 44 Ventana introducir datos

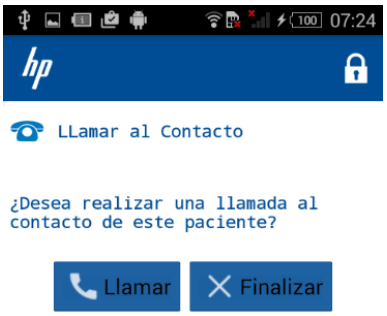
Si se produce algún error en la sincronización de los datos el sistema nos avisará, si los datos se sincronizan correctamente también será avisado.





Por último una vez sincronizados los datos al usuario se le permite realizar la llamada al contacto del paciente, como podemos ver en la siguiente imagen:

Si el usuario pulsa en el botón llamar, se iniciara la llamada con el número de teléfono del contacto que nos devolvió la aplicación asociado a la tarjeta.



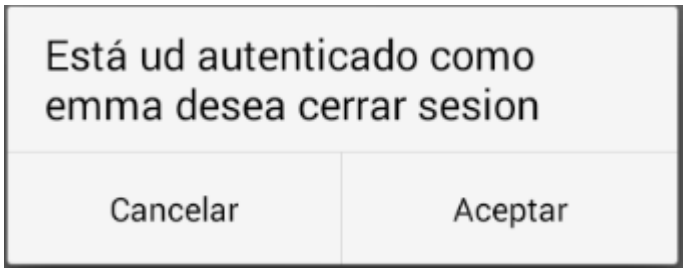
Si pulsamos en finalizar volveremos a la ventana en la que aparece la información del paciente, por si nos interesara mirar algo más.

Para salir de la aplicación podemos hacerlo en cualquier momento, en todas las ventanas aparece un icono en la parte superior derecha que si pulsamos, podemos cerrar sesión:

Ilustración 45 Ventana llamada

Al pulsar sobre este icono aparecerá el siguiente mensaje:

Si aceptamos cerraremos sesión en la aplicación y volveremos a la ventana de login. Si por el contrario cancelamos permaneceremos en la ventana actual.



B: Manual de Usuario HCIS

El usuario accederá a la aplicación web por la url correspondiente, lo primero que le aparece es una ventana de login.

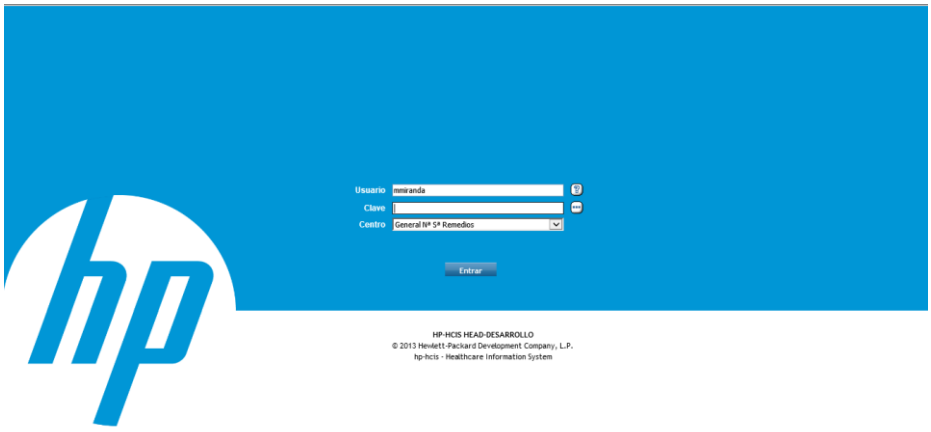
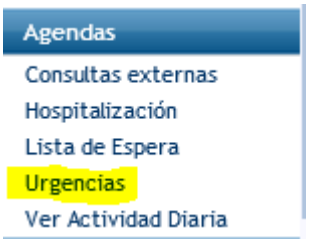


Ilustración 46 Login HCIS

El usuario deberá autenticarse con sus credenciales y elegir un centro, para poder acceder a las funcionalidades de la aplicación.

En la parte izquierda de la página principal podemos ver un menú de tipo acordeón con diferentes apartados.

Para acceder a la parte que nos interesa debemos pulsar en el apartado de agenda para poder acceder a la parte de urgencias.



Una vez hemos accedido a la parte de urgencias veremos la siguiente pantalla:

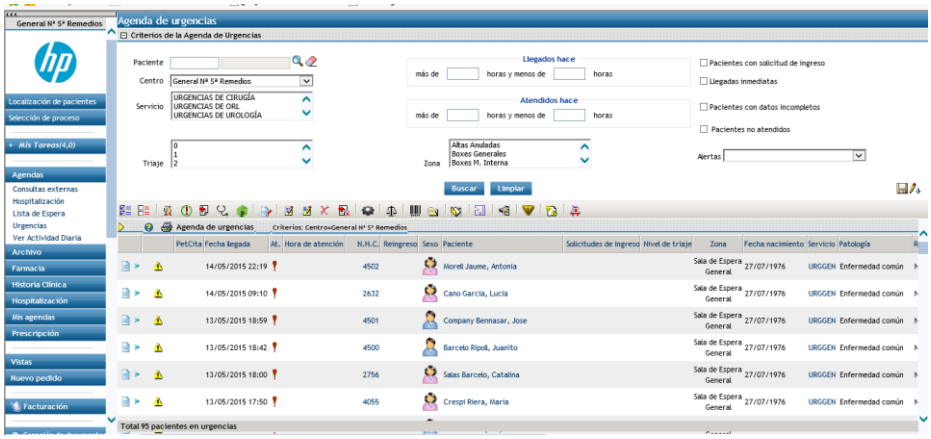


Ilustración 47 Ventana Urgencias

En ella podemos ver un listado de pacientes en urgencias en la parte inferior de la imagen, y una botonera en la parte central.

Para poder visualizar las alertas de pacientes en ambulancias debemos pulsar sobre el siguiente botón de la botonera.

A continuación se nos abrirá una nueva ventana como la siguiente, en la que podemos ver una lista con los pacientes que están siendo atendidos por los sanitarios de las ambulancias.

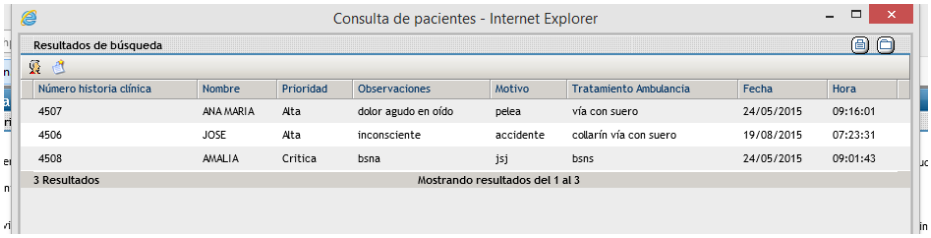


Ilustración 48 Ventana Consulta de pacientes

Puede seleccionar un paciente de la lista y pulsar sobre uno de los botones de la parte superior izquierda.

El botón  mostrará en una nueva ventana la información del paciente de la siguiente manera:

Ilustración 49 Ventana Información paciente

El botón  sirve para preparar la llegada del paciente a urgencias, si lo pulsamos se nos abrirá una nueva ventana con un formulario el que se debe cumplimentar:

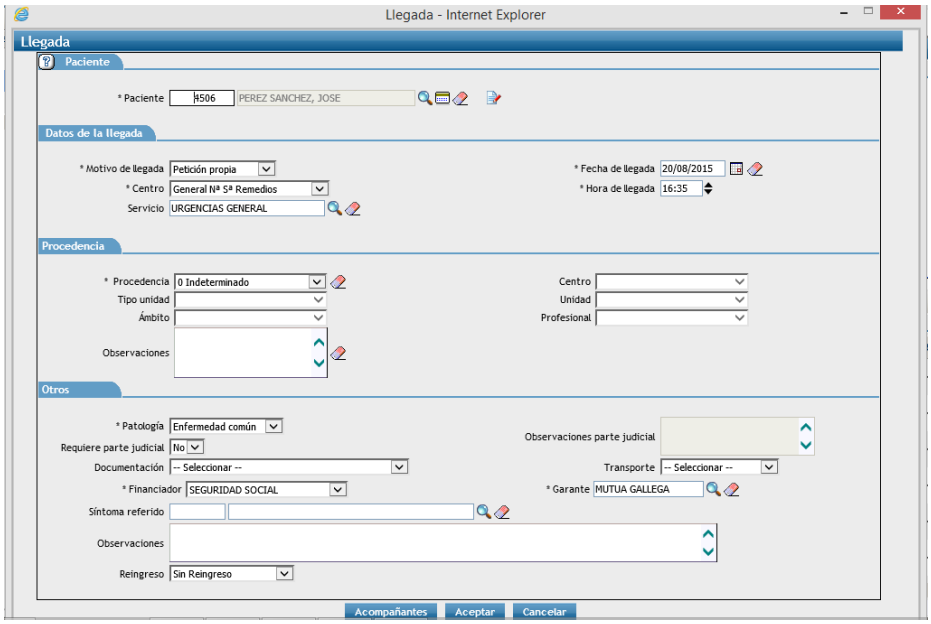


Ilustración 50 Ventana preparar llegada

Una vez rellenado y pulsado el botón de aceptar se cerrará la ventana y posteriormente podemos ver al paciente en la ventana de pacientes en urgencias:

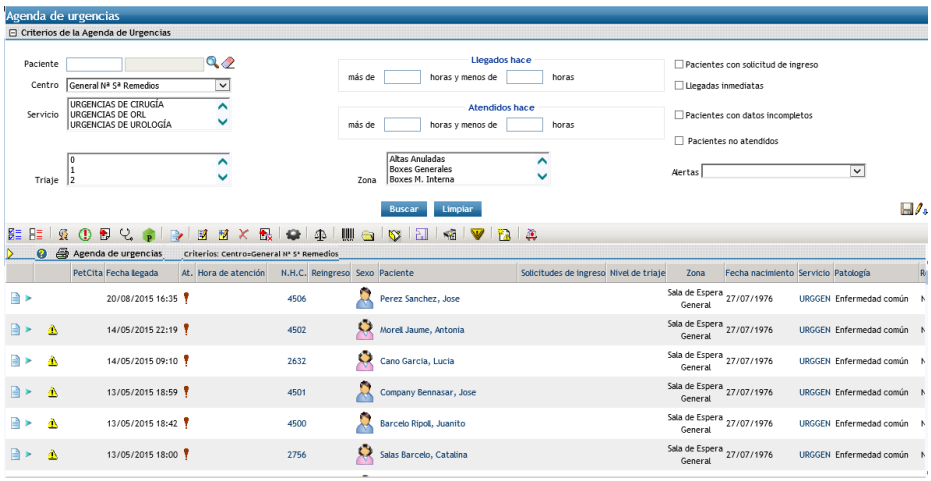


Ilustración 51 ventana Urgencias Actualizada

C: Manual de Instalación Aplicación Móvil

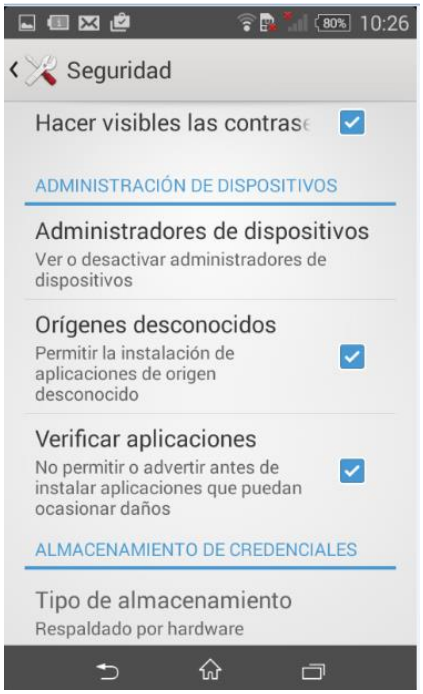
Para la instalación de la aplicación en un dispositivo Android, se precisa de algún medio para transmitirle datos desde el ordenador. Se aconseja el uso de un cable USB por su rapidez.

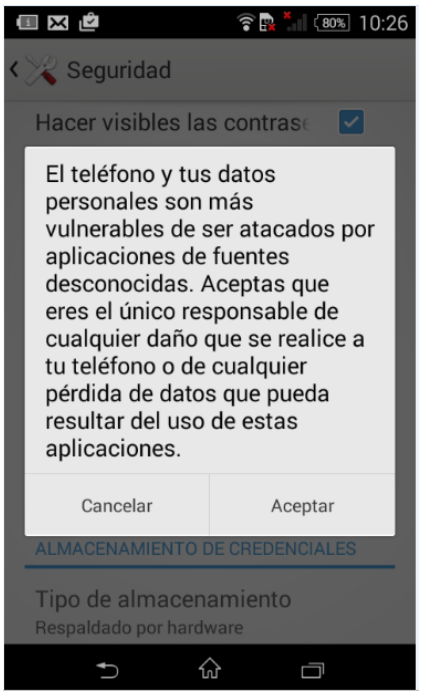
Igualmente será necesario disponer de un navegador de archivos, que viene por defecto instalado en la mayoría de los dispositivos Android del mercado. Mediante el navegador de ficheros, buscaremos nuestra aplicación APK, previamente almacenada en el dispositivo.

La aplicación tendrá el nombre: HCISscan.apk. Cuando el archivo se ha localizado mediante el navegador de archivos, pulsamos sobre él.

El sistema nos mostrará un diálogo con varias opciones, donde seleccionamos la opción de instalar. Una vez que se acepte, ya quedará instalada en el dispositivo.

Debe asegurarse primero de que en el dispositivo se permite la instalación de aplicaciones externas. Esto se comprueba en el menú de ajustes, dentro de aplicaciones, verificando si aparece marcada la opción “Orígenes desconocidos”.





D: Glosario de Términos

API: *Application Programming Interface*. La Interfaz de Programación de Aplicaciones, es un grupo de rutinas que definen cómo invocar desde un programa, un servicio que éstos prestan. En otras palabras, una API representa una interfaz de comunicación entre componentes software.

APK: *Application PacKage File.* Se usa para distribuir e instalar componentes empaquetados para la plataforma Android para smartphones y tablets.

CP: Caso de Prueba

CU: Caso de Uso

DAO: Data Access Object (DAO, Objeto de Acceso a Datos) es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo

DSS: Diagrama de secuencia del sistema.

FRAMEWORK: Estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado.

HTTP: *Hypertext Transfer Protocol,* es un protocolo cliente-servidor utilizado en la World Wide Web para el intercambio de información.

IDE: *Integrated Development Environment.* Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

JDBC: *Java Database Connectivity,* es una API que permite realizar operaciones sobre una base de datos, utilizando el lenguaje de programación Java.

Layout: Esquema de distribución de los elementos en un diseño de una interfaz.

MVC: El *Modelo Vista Controlador* es un patrón de arquitectura del software que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos.

SDK: Software Development Kit. Un kit de desarrollo de software es un conjunto de herramientas de desarrollo que permite a un programador crear aplicaciones para un sistema concreto.

URL: UniformResourceLocator. El localizador uniforme de recursos es una secuencia de caracteres, de acuerdo a un formato modélico y estándar, que se usa para nombrar recursos en Internet para su localización o identificación.

WebService: es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet.

Been: objeto creado y manejado por el contenedor Spring.

ModelAndView: es una composición del modelo lógico de datos y vista o interfaz de usuario

E: Apéndice contenidos del soporte digital

En el soporte digital, podemos encontrar tres carpetas, Memoria, Fuentes, Ejecutable.
Memoria contiene un pdf con la documentación del proyecto.
Fuentes contiene los códigos fuentes comprimidos de la aplicación móvil y HCIS.
Ejecutable contiene la apk de a aplicación móvil.

Bibliografía

[1] Sébastien Pérochon, *Android. Guía de desarrollo de aplicaciones para Smartphones y Tabletas* (Ed. eni). 2013. ISBN: 9782746092297

[2] LARMAN, Craig. *Applying UML and patterns* (Ed. Prentice Hall). 2001. ISBN: 0-13-092569-1

[3] GIRONÉS, Jesús Tomás. *El gran libro de Android, 3ª Edición*. (ed. Marcombo). 2013. ISBN: 978-84-267-1976-8

[4] SOMMERVILLE, Ian. Ingeniería del Software. (Ed. Pearson Educación) 2005. ISBN: 84-7829-074-5

Referencias Web

[1] Wikipedia, [Web en línea]; Spring Framework

https://es.wikipedia.org/wiki/Spring_Framework [Consulta: 18/4/14]

[2] Wikipedia, [Web en línea]; Servicio Web

https://es.wikipedia.org/wiki/Servicio_web [Consulta: 3/3/2015]

[3] Wikipedia, [Web en línea]; SOAP

https://es.wikipedia.org/wiki/Simple_Object_Access_Protocol [Consulta: 4/5/2015]

[4] Universidad de Alicante. Dpto de ciencia de la computación, [Web en línea];

http://www.jtech.ua.es/j2ee/publico/servc-web-2012-13/sesion02-apuntes.html [Consulta: 8/3/2015]

[5] OpenCv, [Web en línea];

http://wwwhttp://opencv.org/ [Consulta: 15/3/2015]

[6] Wikipedia, [Web en línea]; Tesseract

https://es.wikipedia.org/wiki/Tesseract_OCR> [Consulta: 17/2/2015]

[7] QODE, [Web en línea];

http://qode.pro/blog/web-services-rest-vs-soap [Consulta: 13/12/2015]

[8] PMBOK, [Web en línea];

http://es.slideshare.net/andrewcastro7169/gestion-de-proyectos-pmbok [Consulta: 2/3/2015]

[9] Developer android, [Web en línea];

http://developer.android.com/guide/index.html [Consulta: 15/4/2015]

[10] Software de comunicaciones, [Web en línea];

https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android [Consulta: 12/4/2015]

[11] Wikipedia, [Web en línea]; Oracle11G

https://es.wikipedia.org/wiki/Oracle_Database [Consulta: 23/3/2015]

[12] Telecomhall, [Web en línea];

http://www.telecomhall.com/es/usando-toad-como-la-mejor-herramienta-libre-de-gestion-de-base-de-datos.aspx [Consulta: 3/3/2015]

[13] Documentación Jboss, [Web en línea];

https://docs.jboss.org/hibernate/orm/3.5/reference/es-ES/html/mapping.html [Consulta: 3/3/2015]

[14] Tech Resources, [Web en línea];

http://www.ots.ac.cr/tech/node/318 [Consulta: 2/2/2015]

[15] Plastic SCM, [Web en línea];

http://codicesoftware-es.blogspot.com.es/2010/03/plastic-scm-29-ya-esta-aqui.html [Consulta: 20/11/2014]

[16] Diagrama de clases UML, [Web en línea];

http://ocw.unizar.es/ciencias-experimentales/modelos-matematicos-en-bases-de-datos/uml/03UML_DiagramaClases.pdf [Consulta: 1/6/2015]

[17] Java Oracle, [Web en línea];

http://docs.oracle.com/javase/tutorial/java/javaOO/accesscontrol.html [Consulta: 22/5/2015]

[18] Hibernate, [Web en línea];

http://cursohibernate.es/doku.php >. [Consulta: 21/4/2015]

[19] Developer android, [Web en línea];

http://developer.android.com/reference/android/content/SharedPreferences.html [Consulta: 21/4/2015]

[20] Virtual Box, [Web en línea];

https://www.virtualbox.org/manual/ch06.html [Consulta: 22/11/2014]

[21] Wikipedia, [Web en línea]; OCR

https://es.wikipedia.org/wiki/Reconocimiento_%C3%B3ptico_de_caracteres [Consulta: 13/03/2015]