Universitat Autònoma de Barcelona

escola d'enginyeria

2127: APLICACIÓN PARA LA ADMINISTRACIÓN DE DEPARTAMENTO DE VENTA DE UN CENTRO COMERCIAL (iPhone)

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per
Alejandro Navarro Romero
i dirigit per
Diego Javier Mostaccio Mancini
Bellaterra, 17 de juny de 2010

Universitat Autònoma de Barcelona

Escola Tècnica Superior d'Enginyeria

El sotasignat, Diego Javier Mostaccio Mancini

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en

I per tal que consti firma la present.

Signat: .....

Bellaterra, 17 de Juny de 2010

A Chantal.

Por soportar todas las ausencias.

StockTouch

Tabla de Contenidos

Índice de Figuras

Índice de tablas

Capítulo 1

Introducción

En la actualidad, un gran número de centros comerciales importantes, siguen utilizando técnicas muy primarias para el control de las existencias de todos sus productos. De este modo, resulta imposible ofrecer un buen servicio a sus clientes.

En este capítulo, se presenta una descripción de la problemática , con el objetivo principal de encontrar una solución más acorde al siglo XXI, dónde la tecnología avanza a gran velocidad.

También se detalla la propuesta de proyecto, los objetivos que se esperan alcanzar y la planificación para realizarlos.

1.1 Experiencia personal

Observando cómo se controlan las existencias de todos los productos, surge la necesidad de mejorar estos procesos diarios.

Actualmente, trabajo en el departamento de ventas informáticas de uno de los centros comerciales más importantes de nuestro país, donde el "stock" se controla principalmente por una hoja creada con Excel. Ésta, se compone de una tabla con productos ordenados según una referencia, marca, modelo, precio, almacén y cantidad.

Todo tipo de gestiones, como las reparaciones, reservas, adecuaciones y devoluciones, son controladas por albaranes, que posteriormente son guardados en diversos archivadores.

Este sistema, aun ha de considerarse, como una ventaja que tenemos los vendedores de informática respecto a los demás, ya que he observado que se gestiona incluso de peor forma en toda la sección de electrónica

El funcionamiento de estos grandes almacenes es el siguiente: Los jefes de departamento, con su capacidad, son los encargados de la administración y gestión, y disponen de pocas herramientas para poder hacerlo de una manera eficaz.

Debido a esto hay departamentos en los cuales, se gestiona el "stock" con los productos anotados en una libreta, otros con una hoja creada por Excel y en otros directamente no se gestiona. Los vendedores se guían de una existencia teórica consultada en la base de datos central del centro comercial. Ésta, es un recurso que poseen todos los departamentos, pero sólo da una cifra orientativa de los

productos disponibles, ya que es una existencia teórica que no controla artículos reservados, reparaciones, productos a adecuar, o expuestos en tienda.

Como futuro ingeniero, y trabajador de la empresa, creo en la necesidad de un cambio radical que pueda adaptarse mejor a nuestros tiempos y en un sistema adecuado para una de las principales empresas de nuestro país que da servicio a miles de personas diariamente.

Por lo tanto, de la experiencia diaria en mi actual lugar de trabajo, nace *"StockTouch".* Una herramienta que controla todos los artículos en el departamento, llevando un control exhaustivo del número, lugar y estado en el que se encuentra cada producto.

1.2 Propuesta de proyecto

A raíz de los problemas que comporta para las diferentes *secciones* la gestión de su mercancía, y las causas que derivan, es cuando surge la idea inicial de proyecto.

StockTouch es una herramienta pensada para el control de las existencias "stock" en un departamento de venta de un centro comercial.

Esta herramienta se ha de poder implementar en cualquier sección de venta, ya sea, en la de productos informáticos, libros o cualquier otro artículo. El tipo de departamento y por lo tanto, del artículo, hará que la aplicación pueda utilizarse en mayor o menor medida.

Controlará de forma exhaustiva, todo producto que pertenezca a cada zona de venta, ya sea un artículo que se encuentra físicamente en el centro o en un proceso de transito. También administrará, si la mercancía lo requiere, las reservas de los clientes o el estado de las reparaciones. En general cualquier estado que pueda llegar a tener el producto.

El entorno es dinámico, y precisamente es esto lo que se quiere conseguir. Una aplicación capaz de adaptarse a las necesidades específicas de cada departamento.

"StockTouch" ha de poder aplicarse en todos los departamentos gracias a su dinamismo y esto se consigue mediante dos interfaces, una para el administrador y otra para el vendedor.

La interfaz del administrador ha de ser única y han de poder controlarse todos los aspectos necesarios, dependiendo de cada sección del centro comercial. La del vendedor ha de ser fácil de utilizar e intuitiva, ayudándose de herramientas gráficas de diseño, que garanticen la atención personalizada a cada cliente. No hay que olvidar, por las características del lugar de trabajo en este centro comercial, la movilidad del vendedor. Por lo tanto la de éste ha de ser portátil, por lo que tiene que ser implantada en un dispositivo móvil.

1.2.1 Objetivos

De la inexistencia de una herramienta eficaz de control y gestión, surge como principal objetivo la necesidad de crear una aplicación que permita manejar de forma exhaustiva todos los productos de cada sección del centro, tanto si están a la venta, como sino.

En primer lugar , el producto, ha de poder identificarse. Es decir, ha de tener un numero de serie que lo haga único, y además una referencia que indica a que familia pertenece.

Esta familia tendrá unas características, que han de poder generarse. Por ejemplo no posee las mismas características un portátil (tamaño, cpu, memoria, etc…) que una cámara de fotos. Por motivos obvios todos los productos tienen marca, modelo y precio y por lo tanto estas tres características las tendrán por defecto.

La primera vez que se introduzca un articulo en el "stock" ha de poder introducirse con todas sus características. De esta manera todos los artículos que lleguen al departamento ya estarán referenciados y no hará falta describirlos.

Una vez introducido el producto, ha de estar localizado en algún lugar del departamento. Por este motivo todo producto tendrá asignado un almacén, que será cualquier lugar donde se pueda almacenar mercancía, desde un mueble, un cajón, una habitación, un proveedor. Éstos almacenes serán internos o externos, así existe la posibilidad de tener almacenes fuera del centro, donde poder pedir un producto.

Una vez introducido el artículo, descrito con unas características, y localizado en un almacén se ha de poder saber en que estado se encuentra. Por lo tanto el siguiente objetivo es poder generar estados. Dichos estados han de poder determinar si un producto esta a la venta o no, y si no lo esta en que estado se encuentra. Esto es debido a que el departamento no sólo posee productos a la venta, sino también productos que se han de reparar, enviar al servicio técnico, devolver, adecuar a la venta, entregar, etc… Por lo tanto otro de los objetivos es el control por parte de la aplicación de la gestión de estos productos que no están a la venta.

La parte móvil del proyecto será de gran ayuda, ya que a través de ésta se podrá realizar todo tipo de gestiones sobre las reparaciones, devoluciones, reservas, etc.. que lleven los vendedores.

Tanto las gestiones, como el stock ha de poder ser controlado por el administrador y alterado si él lo considera oportuno.

Como el proyecto no pretende ser sólo una herramienta de control de existencias, también se dará la posibilidad al vendedor de realizar sus gestiones y al jefe de asignar tareas. Debido a esto último el administrador ha de poder crear usuarios y autorizarles a usar la aplicación móvil. También ha de poder asignarles tareas y saber si las han realizado.

Para finalizar, y como resumen de los objetivos marcados, el entorno ha de poder ser lo más dinámico posible, ya que se han de poder generar productos muy distintos, con características muy distintas entre sí. Se han de poder crear almacenes donde localizar los productos. También asignar estados distintos, para las necesidades de cada departamento en concreto, y la gestión de los tramites de productos que no estén a la venta. Finalmente se han de poder asignar tareas a los usuarios y el seguimiento de su realización.

1.2.2 División del proyecto

La estructura del proyecto ha de estar dividida según los siguientes puntos:

- Base de datos.
  Donde se guarda toda la información de los productos, clientes, vendedores, etc.
  Esta base de datos estará diseñada para contener el "stock", y los datos de las gestiones.

- Aplicación web.
  Será la herramienta mediante la cual el administrador del departamento en cuestión, podrá consultar todos los datos que cree necesarios sobre el "stock" del departamento, modificarlos y crearlos. Además podrá asignar tareas a los vendedores, y lo más importante, podrá generar todo el entorno, adaptándolo al departamento en el cual se vaya a implantar la aplicación. Esto implica que la aplicación deberá dar la posibilidad de poder crear productos, familias de productos con todas sus características, almacenes donde poder guardarlos, vendedores, reparaciones, adecuaciones, reservas, etc… También podrá hacer modificaciones según crea necesario.

- Aplicación móvil.
  Una aplicación la cual estará destinada a ser usada por todos los vendedores del departamento, ésta tendrá más de un usuario, y en ella el vendedor podrá hacer todas las gestiones que necesite. Al contrario de la aplicación del administrador, estará orientada a ser una utilidad para el vendedor, con la que podrá realizar las gestiones del departamento, pero no podrá generar el entorno, ya que está será una tarea exclusiva del administrador.

- Interfaz de comunicación (web services).
  Para unir el concepto de aplicación móvil con el de base de datos, se creará una interfaz entre ésta y la aplicación para poder suministrar todos los datos que la aplicación requiera. Además dicha interfaz podrá realizar modificaciones sobre la base de datos y atenderá peticiones de varios usuarios.

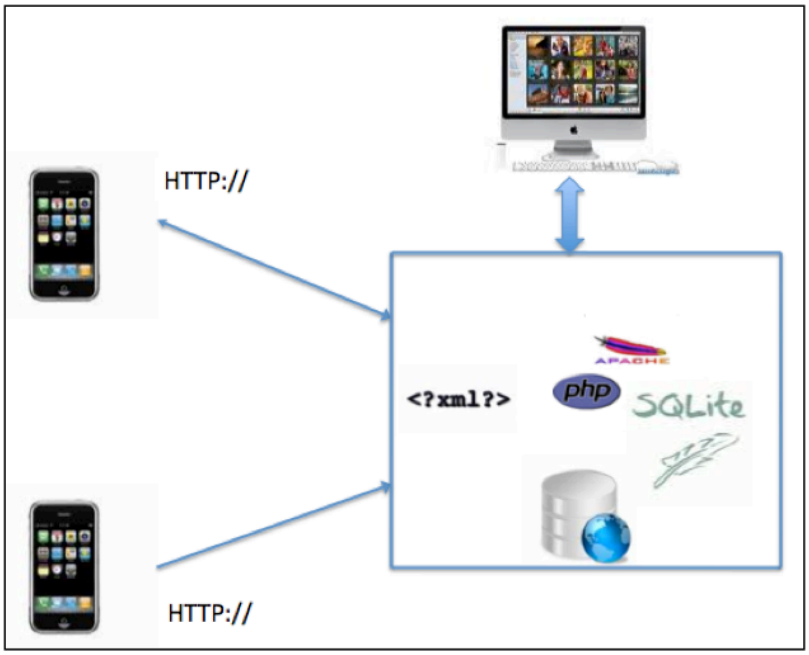


Figura 1 División del proyecto

1.2.3 Requisitos y limitaciones

A parte de los objetivos descritos con anterioridad el proyecto ha de cumplir una serie de requisitos para que el resultado y su implementación sean aceptables.

Se pretende implantar en cualquier departamento de un centro comercial, con lo cual a priori, no se dispone de toda la información del entorno bajo el que va a funcionar. Esto implica una serie de limitaciones, pero a su vez también es lo que hace que la aplicación sea tan adaptable.

Los siguientes puntos detallan los requisitos que ha de cumplir el entorno donde se pretenda implantar y las dificultades a las que deberá hacer frente.

1. **Entorno de administrador suficiente**. El proyecto esta ideado para poder aplicarse con la división descrita en el punto 1.2.2, pero si el entorno en el que se ha de implantar el proyecto no dispone de las herramientas móviles necesarias, la aplicación del administrador ha de ser suficiente para gestionar todo el stock del departamento.

2. **Portabilidad**. El proyecto ha de poder implantarse bajo cualquier distribución de software existente. Por lo tanto la aplicación del administrador y la base de datos han de ser lo más ligeras posibles, y compatibles con la mayoría de los entornos existentes en la actualidad. Por el contrario si se desea el óptimo funcionamiento de la misma, se recomienda un pc o mac con un servidor web apache o tomcat instalado.

El servidor web ha de tener el modulo php activo, ya que el lenguaje bajo el cual se implementeará la aplicación web será este mismo.

3. **Topología común.** Para que la instalación y el funcionamiento del proyecto sea el correcto, la topología bajo la cual ha de funcionar debe cumplir con la mayoría de topologías que pueden encontrarse en cualquier establecimiento o centro comercial. Por este motivo la distribución mínima que se recomienda es un pc que haría de servidor, con un Punto de Acceso inalámbrico, para la conexión con el dispositivo móvil. Como se puede observar la mayoría de establecimientos, o departamentos de un centro comercial poseen como mínimo un pc para consultas y un router inalámbrico para conectarlo a Internet. Con lo cual, sólo con esto, ya se tendría la topología mínima para el correcto funcionamiento.

4. **Radio de acción.** A causa de las características técnicas de las actuales distribuciones inalámbricas (AP Wifi) más comunes y del dispositivo móvil iPhone, que es bajo el cual estará realizada la parte móvil del proyecto, el radio de acción estará bastante limitado. Dependerá del radio de acción que es capaz de aportar el punto de acceso que tengamos instalado en el lugar de trabajo. Gracias a la conexión 3G del dispositivo móvil iPhone, éste puede conectarse a Internet fuera del radio de acción de un AP. La aplicación está totalmente preparada para poder conectarse desde fuera del área local. En lugares donde las limitaciones del wifi sean muy grandes es posible la conexión mediante este tipo de tecnologías, entonces será tarea del administrador decidir si permite este tipo de conexiones, permitiendo una mayor disponibilidad, pero por el contrario una reducción de la seguridad. El coste de la conexión de datos 3G viene incluido por el momento en la tarifa mensual al adquirir un iPhone, por el contrario un iTouch no dispone de este servicio.

5. **IPhone o iTouch.** La aplicación móvil se ha decidido implantar en este dispositivo por varios motivos. El primero es la capacidad de interacción con el usuario. El dispositivo posee una gran pantalla con tecnología táctil multitouch, esto permite que el usuario pueda visualizar todos los datos con gran claridad, a parte de poder interactuar con ellos , este tipo de pantallas permite tener el control del dispositivo con los dedos, esto hace que sea mas cómodo para el usuario, que no ha de manejarse con teclados con teclas pequeñas e incomodas o lápices para pantallas táctiles con la posibilidad de perderlos. También por la eliminación de sistemas como el texto predictivo o la selección bajo presión que suelen ser más lentos e incómodos de utilizar. Otro de los motivos es el sdk (Software Development Kit) [IV] disponible para esta plataforma, que permite la creación de aplicaciones bajo una programación orientada a objetos y posee una gran cantidad de apis que permiten la realización de unas aplicaciones muy interesantes con gran funcionalidad y usabilidad. El último motivo, y no menos importante, es el canal de distribución. Apple ofrece una utilidad, llamada AppStore [III] que esta implantada en todos los dispositivos móviles iPhone e iTouch y en el software iTunes. Esta utilidad es un canal de distribución mediante el cual, si se dispone de una licencia de desarrollador suministrada por apple con un precio de 99$ [IV] y bajo la supervisión de esta, se puede publicar el desarroyo al precio que se considere adecuado. De esta manera

cualquier persona que tenga un iPhone o iTouch con una cuenta de iTunes puede adquirir ese producto.

1.2.4 Entorno de ejecución

El entorno físico que hará posible que la aplicación funcione será el siguiente.

- Un servidor.
  El cual alojará la base de datos, la aplicación web y la interfaz de comunicación (web services).
  El servidor ha de alojar un servidor web, el cual de servicio por el puerto 80 a peticiones locales. Dicho servidor ha de poder tener el módulo de php activado, y ha de tener restringidas, si se desea, las peticiones externas a la red local. Este servicio local está recomendado por seguridad, pero la aplicación es capaz de funcionar a través de Internet dando un servicio remoto. La base de datos será un archivo generado con sqlite3, el cual ha de tener permisos para que el servidor web sea capaz de modificarlo.
  Las pruebas realizadas y que se comentarán con posterioridad se han realizado bajo un servidor apache, tanto en un sistema Macintosh como en un pc con Windows7 y Windows vista.

- Un AP wifi.
  Un punto de acceso wifi, conectado con el servidor para que la aplicación móvil sea capaz de conectarse con la base de datos a través de la interfaz.
  En el caso que se desee tener acceso remoto, el punto de acceso wifi, si la red local se gestiona con nat, debe tener redireccionada la ip privada del servidor para poder aceptar peticiones por el puerto 80.

  La omisión de este punto, no haría que dejase de funcionar la aplicación, ya que toda la administración del entorno es capaz de hacerse desde la aplicación web. En este caso, no podría utilizarse la aplicación móvil que da servicio a las gestiones del vendedor.

- Uno o varios iPhone's o iTouch.
  La aplicación móvil está diseñada para utilizarse nativamente en un iPhone o Ipod Touch con OS iPhone 3.1. en adelante.
  La instalación de la aplicación se ha de realizar a través del sistema de gestión de dispositivos Apple iTunes. El cual necesitará de un dispositivo al que se le ha aplicado jailbreak para poder instalarse.
  Si no se dispone de un dispositivo con estas características la aplicación móvil sólo podrá ejecutarse en el simulador que Apple ofrece en su sdk.

1.2.5 Planificación temporal inicial

El proyecto se planificó inicialmente en varias fases:

La fase inicial se planificó para el estudio del SDK del iPhone. Esto incluye el lenguaje Objective-C y sus frameworks [VII][VIII]. También a toda la documentación sobre la programación en Cocoa Touch y a la experimentación tanto con Interface Builder [X] (utilidad de creación de la interfaz gráfica de aplicaciones en iphone) y Xcode [V](entorno de programación Cocoa).

La segunda fase se planificó para la realización en primer lugar del diseño, tanto de la base de datos, con todas las tablas y relaciones entre ellas, como de la interfaz de comunicaciónn. En segundo lugar se planificó la implementación de la base de datos y la introducción de datos no reales a modo de prueba.

La tercera fase consistía en el diseño y desarrollo de la aplicación web. Realización de pruebas unitarias y de integración sobre ellas. En este punto la parte del proyecto realizada tenia que funcionar por si sola. Por último se ideo la realización de la interfaz de comunicación entre la base de datos y la aplicación.

La cuarta fase se ideo en primer lugar, para la realización del diseño del cuerpo de la aplicación móvil, de todas sus clases y las relaciones entre ellas. Esto significa como tratará los datos la aplicación, como los recibirá, los parseará, los enviará, también del diseño de clases, los objetos que tendrá, la gestión de memoria, etc. En segundo lugar el desarrollo de todo el diseño de la interfaz gráfica de la aplicación móvil, además de implementarla. En este punto es donde pasará a utilizarse las herramientas de creación de la interfaz gráfica.

La última fase será la dedicada a realizar todos los tests necesarios, y la prueba tanto de la aplicación del administrador como la móvil en un entorno real.

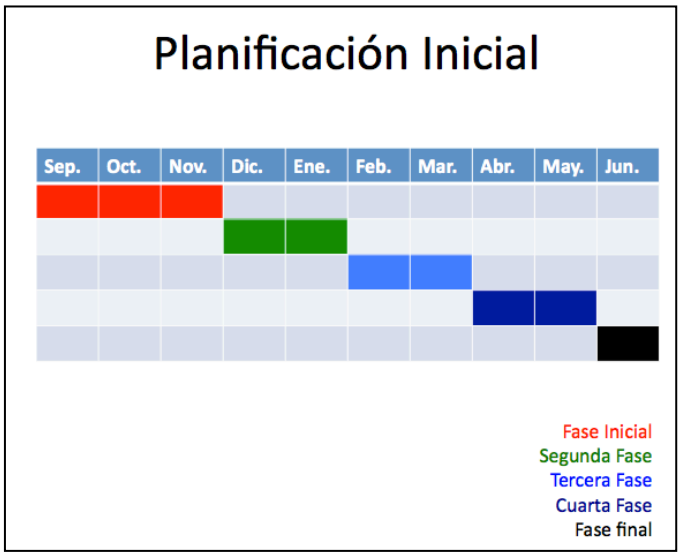


Figura 2 Planificación inicial

Capítulo 2

Estado del arte

En este capítulo se van a mostrar las herramientas, de uso comercial, que actualmente se utilizan para solucionar problemas parecidos al planteado. Además se explicarán las herramientas que existen en la actualidad para realizar programas como el propuesto.

2.1 IMS de IBM

Esta herramienta es la utilizada por el centro comercial en el que trabajo, como administración de la gestión de empresa. En concreto también se utiliza para el control del stock. A través de esta utilidad, un trabajador puede realizar varias acciones relacionadas con la mercancía.

Entre éstas acciones se encuentra la posibilidad de consultar el stock teórico de cualquier centro de la empresa, el surtido que tiene asignado, las ventas que se han realizado, etc.… Pero además ésta herramienta no sólo permite la consulta, sino también la realización de pedidos a almacén, traspasos de mercancía desde otro centro, envíos de almacén a un cliente, etc.…

Digamos que esta utilidad permite al trabajador controlar todos los aspectos relacionados con la mercancía, ya que controla todos los datos que de está dependen.

Pero además también permite la realización de todo tipo de gestiones, éstas pueden ser por ejemplo la realización de una financiación para una venta, la consulta de los datos de un cliente, la modificación de estos datos y todo un conjunto muy variado de operaciones. Éstas abordan el conjunto de todos los procedimientos que se puede llegar a realizar por parte de la empresa.

IBM Information Management System (IMS) es un gestor de datos jerárquicas y un gestor transaccional con alta capacidad de proceso. IBM diseñó el IMS con Rockwell y Caterpillar en 1966 debido al Programa Apollo. El desafío de IBM era inventariar la extensísima lista de materiales del cohete lunar Saturno V y de la nave Apollo. IMS soporta aplicaciones desarrolladas en Java, JDVC, XML y servicios Web. [XIII]

IMS, el principal sistema de gestión de bases de datos jerárquico y de transacción, es el mejor producto para aplicaciones y datos operativos importantes en línea, en las que los factores clave son que admitan una disponibilidad, un rendimiento, una capacidad y una integridad altos, manteniendo el bajo coste. [IX]

2.2 Herramientas Comerciales del control de Stock

Existen diferentes herramientas comerciales para el control y la gestión del stock. Éstas pueden ser muy variadas, desde utilidades creadas exclusivamente para funcionar en un determinado entorno, hasta otras que se comercializan al gran público y que tras una fase de adaptación pueden encajar muy bien en cualquier pequeña o median empresa.

2.2.1 Herramientas Sage

Conjunto de herramientas SAGE [XI] comercializadas al gran público, que son un producto destinado a PYMES y permite el control y la gestión de varios aspectos de su propio funcionamiento.

Entre éstas destacan:

- TPV plus. Entre el conjunto de utilidades que la componen, destaca la posibilidad de funcionar en un terminal con pantalla táctil, para la optimización de los ciclos comerciales. Este producto da la posibilidad del control total de artículos con gestión de productos compuestos (kit) y fabricados, sistema de trazabilidad, gestión por propiedades, tallas y colores, y campos extra definibles por el usuario (marca, editorial…).
  Completa gestión del ciclo de ventas y compras, pedidos, albaranes, presupuestos, abonos…

- Sage Comercio. Destinado a Tiendas de ropa, ferreterías, tiendas de bebidas y alimentación, perfumerías y droguerías, librerías, tiendas de electrodomésticos, tiendas de muebles, panaderías y pastelerías. Donde entre varias características destacan el control de stock por propiedades y almacén y el control de inventario.

- Gestión de Barra. Destinado a bares, pubs, discotecas y cafeterías donde también destaca el control de stock y almacenes.

2.2.2 Herramientas ARS

Conjunto de herramientas [VI] comercializadas al gran público. Se puede decir que desarrollan el mismo tipo de productos que Sage y por lo tanto ambos conjuntos de herramientas son la competencia el uno del otro.

Los productos a destacar por Ars son:

- Pack solución TPV Profesional. Aplicación que permite la gestión de la empresa desde el punto de venta. Entre las diferentes características destacan la venta Táctil con galería de imágenes, el control de almacenes y stock de productos, el cálculo del coste promedio según la NIC2, y la gestión

de artículos: tallas y colores, periodos de oferta, subcódigo de barra y lectura de código de barras EAN13.

- PackSolución Gestión Comercial Profesional. Entre las principales características destacan el control de almacenes y stock de productos. Artículo con descripciones más amplias. Gestión de artículos con fotografía y lectura de código de barras EAN13.

2.3 Aplicaciones Iphone/Itouch

En la actualidad hay varias formas de desarrollar aplicaciones para la plataforma iphone OS.

Hay tres caminos posibles a escoger a la hora de desarrollar una aplicación para esta plataforma, cada uno de ellos con sus ventajas e inconvenientes.

2.3.1 Aplicaciones nativas

Este tipo de aplicaciones se desarrollan mediante el sdk [IV] proporcionado por apple. Para poder descargarte este sdk tienes que estar registrado como desarrollador en Apple Developer. Este sdk permite la utilización de las numerosas apis de cocoa touch desarrolladas por apple y de todo su entorno de programación.

El lenguaje utilizado para programar es Objective-C, paradigma orientado a objetos compatible como C++ con la sintaxis tradicional de C, pero evolucionado con su propia sintaxis y modo de funcionamiento. A causa de la retrocompativilidad del lenguaje Objective-C, las aplicaciones programadas en C y C++ que utilicen las apis de Cocoa Touch también funcionarán. [VII]

El sdk de apple aporta un simulador donde poder ejecutar el código compilado. Pero si el desarrollo se quiere instalar en un dispositivo real, apple obliga al pago de 99$ [IV] por una licencia de desarrollador.

Una de las ventajas que tiene el pago de esta licencia es el canal de distribución. Una vez adquirida la licencia, no sólo puedes instalar tus aplicaciones en un dispositivo, sino que también apple proporciona un canal donde poder venderlas.

AppStore es un portal donde se pueden adquirir aplicaciones realizadas para el iPhone o iTouch. A este portal se puede acceder mediante una aplicación con el mismo nombre, instalada en todos los dispositivos de Apple. También se puede acceder mediante iTunes, software de Apple para pc o Mac, que es necesario para la gestión de cualquier iPhone o iPodTouch. Para realizar una compra en dicho portal es necesaria la obtención de una cuenta de iTunes, registrándote y aportando tus datos personales y un numero de tarjeta de crédito o debito.

Este tipo de aplicaciones, al estar desarrolladas nativamente para el iphone, suelen ser más potentes, ya que utilizan al máximo todas las características que el dispositivo ofrece. Además se puede encontrar mucha variedad de ellas, [III] desde juegos que utilizan openGL, pasando por aplicaciones que retocan fotografias, aplicaciones matemáticas, utilidades financieras, de acceso a redes sociales, etc…

2.3.2 Aplicaciones externas

Existe otro tipo de aplicaciones, no desarrolladas con el sdk de apple. Éste tipo de aplicaciones se desarrollan con otro lenguaje las cuales se enlazan directamente con APIs documentadas a través de un sistema de traducción, capa de compatibilidad o herramienta. Esto permite que puedan utilizarse en un Iphone o Ipod Touch.

Este tipo de aplicaciones pueden realizarse en la actualidad mediante herramientas como MonoTouch, Unity3D, y PhoneGap.

A finales de junio del 2010 se pública iPhone OS 4 que sustituirá al actual SO. Este SO bloqueará este tipo de aplicaciones, lo cual impedirá que puedan utilizarse. [II]

2.3.3 Aplicaciones WebKit

Este es el tipo de aplicaciones que podían ejecutarse en este dispositivo desde el principio.

Al ponerse a la venta el primer modelo de iPhone con Iphone OS 1, las aplicaciones nativas realizadas por terceros estaban prohibidas. No fue hasta Iphone OS 2.0 que se permitió la creación de éstas mediante el sdk de apple.

Las únicas aplicaciones que han podido realizarse desde entonces eran aplicaciones web, desarrolladas con el webkit para el iPhone y con el permiso de javascript como único lenguaje de soporte.

Estas aplicaciones no se ejecutan nativamente en el dispositivo, sino que lo están haciendo mediante un interprete, en este caso un navegador web (safari). [I]

Utilizando este tipo de desarrollo, se conseguirá por parte del programador, una sensación parecida a la que puede causar una aplicación nativa, ya que el webkit proporciona unas herramientas muy eficientes, pero nunca se llegará al grado de efectividad de una aplicación desarrollada en Objective-C.

Capítulo 3

Análisis y diseño

Tras ver en la introducción los objetivos del proyecto, la división de este mismo y los requisitos que ha de cumplir, se pasa a detallar el diseño y el análisis de cada componente de la aplicación.

3.1 Casos de uso

Se pasa a detallar mediante un diagrama de casos de uso, la funcionalidad que se pretende y la interacción con los diferentes usuarios. En las siguientes figuras se observa el esquema descrito.

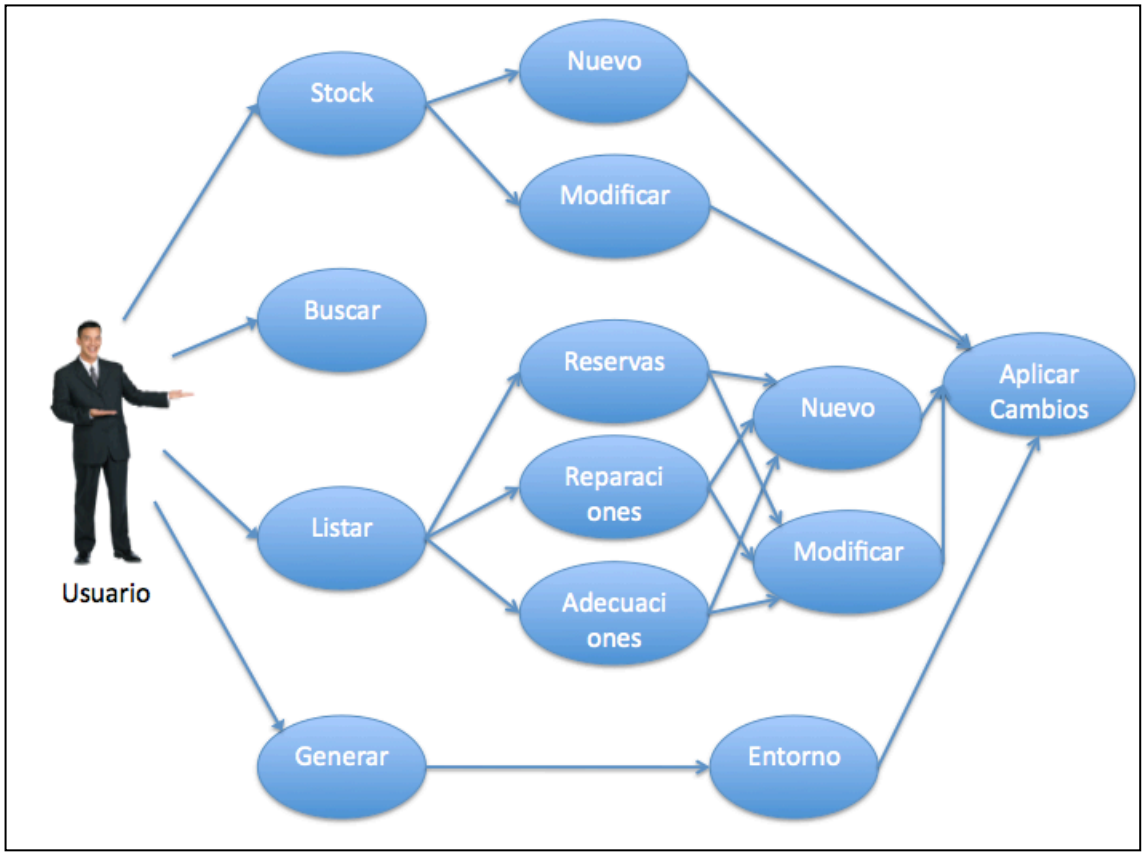


Figura 3 Casos de uso usuario administrador

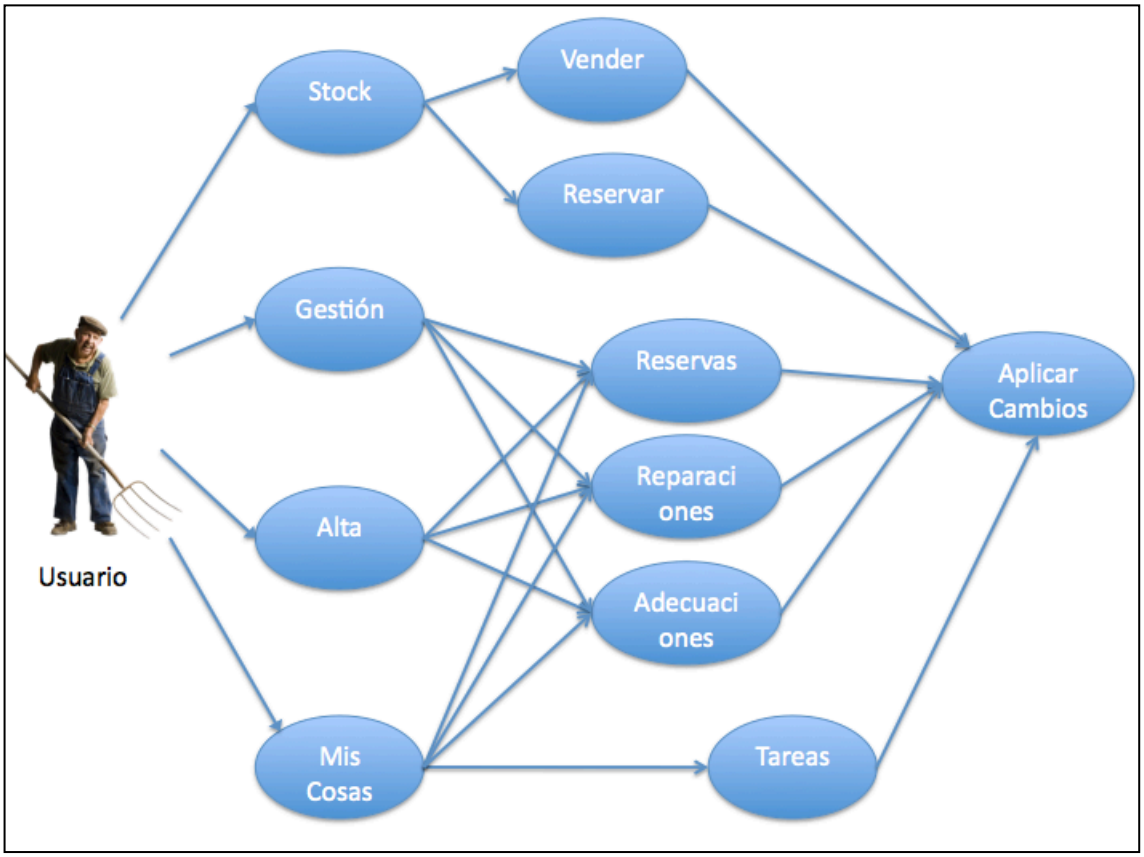


Figura 4 Casos de uso usuario vendedor

Caso de uso	Descripción
Stock	Proceso mediante el cual el usuario entra en el apartado de introducción de stock.
Buscar	Buscar producto, ya sea por características, almacén, vendedor, etc…
Listar	Listar productos y características según su estado.
Generar	Es el proceso que permite al usuario administrador generar todo el entorno.
Nuevo	Proceso que crea una nueva entrada.
Modificar	Proceso que modifica una entrada existente.
Reservas	Proceso que rescata el conjunto de productos en estado pendiente o venta en proceso.
Reparaciones	Conjunto de productos con estado reparación.
Adecuaciones	Proceso que accederá a los productos a adecuar a la venta.

Entorno	Proceso que accede al entorno, para generarlo o crearlo. Éste esta compuesto por vendedores, almacenes, familia de productos, tareas, etc…
Aplicar Cambios	Proceso final que guarda los cambios realizados por el administrador.

Tabla 1 . Descripción de casos de uso del administrador

Casos de Uso	Descripción
Stock	Proceso mediante el cual el usuario entra en el apartado de stock.
Gestión	Proceso que gestiona productos con estados diferente a venta.
Alta	Mediante Alta el usuario puede generar productos con estados diferentes a venta.
Mis Cosas	Proceso que controla productos asignados al usuario.
Vender	Se inicia proceso de venta de un producto.
Reservar	Se inicia proceso de reserva de un producto.
Reservas	Proceso que rescata el conjunto de productos en estado pendiente o venta en proceso.
Reparaciones	Conjunto de productos con estado reparación.
Adecuaciones	Proceso que accederá a los productos a adecuar a la venta.
Tareas	Proceso que accederá a las tareas asignadas al usuario.
Aplicar Cambios	Proceso final que guarda los cambios realizados por el usuario.

Tabla 2 Casos de uso del usuario vendedor

Mediante estas tablas se puede observar como la aplicación del administrador puede controlar todo producto en el departamento y a su vez generar todo su entorno.

En cambio el usuario vendedor puede acceder a los productos, cambiar su estado e incluso generar nuevos productos ya existentes, pero no puede generar el entorno.

Con esto se observa que el destino de la aplicación del administrador es controlar y generar tanto artículos como el propio entorno, en cambio la del vendedor es poder gestionar artículos.

3.2 Entorno

El diseño del entorno es la pieza clave del proyecto. Éste es un conjunto de elementos que describen todo lo que se quiere gestionar mediante la aplicación. Por lo tanto, el entorno debe describir minuciosamente todos los elementos, que como piezas de un rompecabezas encajarán para formar parte de todo el conjunto de componentes que lo formarán.

Todo el entorno se basa en una pieza clave, el producto, que pasará a ser el eje central del proyecto. Éste posee una serie de características que se describen a continuación.

- **Numero de serie**. Este sn es un numero que lo identifica como único, por lo tanto todos los productos tendrán un sn distinto, el cual será único e irrepetible.

- **Referencia.** Número que lo identifica entre los diferentes tipos de modelos. La referencia es una marca que se compone de tres campos.

    - **Campo departamento**. Un conjunto de tres dígitos que identifica el departamento al que pertenece el producto, por ejemplo un producto de informática pertenecerá al departamento "152" y un producto de televisiones pertenecerá al departamento "148".
    - **Campo familia** . Un conjunto de tres dígitos que identifica a la familia de productos a los que pertenece, es decir por ejemplo, un portátil pertenece a la familia "155" y una impresora a la familia "260".
    - **Campo barra**. Un conjunto de cinco cifras que identifica al modelo dentro de la familia de productos, un ejemplo seria el modelo el Toshiba-a500-20c con barra 10002 y otro el hp-dv6-2015 con barra 20302.

  Todos los productos pertenecientes a una familia en concreto tendrán el mismo tipo de características. Dentro de cada familia habrán distintos modelos, identificados por la barra, donde cada barra se distinguirá de otra por ser diferente modelo, como ya se ha comentado, y además tener las mismas características, pero diferentes configuraciones. Para poner un ejemplo, todos los productos pertenecientes a la familia "155" portátiles, tendrán la característica en común "cpu". Lo que distinguirá una barra de la otra es que serán diferentes modelos y por eso la cpu en un caso puede que sea un Intel core2duo, en cambio en el otro sea un amd turion x2. Esto llevará a la creación de un conjunto de características por cada familia.

- **Almacén**, el cual determina el lugar donde se encuentra el producto. Los almacenes se clasificaran en dos tipos.

    - **Internos**, todo tipo de almacén, ya sea cajón, mueble, habitación, muelle, etc.. que esté en el centro y que por lo tanto pueda ser accesible desde el mismo.
    - **Externo**, todo tipo de almacén que no sea accesible desde el centro, puede ser tanto un almacén de productos a la venta, como un servicio técnico, un proveedor, etc.. cualquier tipo de almacén que el administrador considere oportuno.

  En este caso los almacenes serán generados por el administrador, pudiéndose alterar si él lo considera necesario.

  Existirá un almacén por defecto, "libre", dicho almacén será un recurso para el administrador. Será donde irán a parar los productos que no tengan un almacén asignado, es decir, aquellos productos que por cualquier motivo han llegado al centro y aún no han estado colocados, pero si introducidos en la aplicación. También se utilizará, cuando haya una diferencia en "stock" y difiera del numero de productos en ese almacén, por ejemplo la aplicación dice que hay 5 productos en el almacén3 pero al pasar revisión se observa que hay 1, esos 2 de diferencia serán asignados al almacén "libre". Posteriormente será decisión del administrador eliminarlos o no. Por otro lado también será un recurso para la misma aplicación, ya que a la hora de alterar los almacenes, por ejemplo eliminando uno, todos los productos que se encuentren en dicho almacén pasarán a "libre".

- **Estado**, el cual determina el estado en que se encuentra el producto.
  Hay dos clases de estados.

    - De **grupo**. Este es un estado que se comparte por todos los productos pertenecientes a una barra y es sólo un estado para productos que están a la venta.
      Se distinguen dos tipos:

        - **Tipo venta**: 10 estados posibles a generar por el administrador (0-9) para productos que están a la venta. El primer estado "0" está asignado por defecto y será "vendido".

        - **Tipo destacados**: 10 estados posibles a generar por el administrador (10-19) para productos que están a la venta y que por cualquier motivo tienen algo en especial. Por ejemplo, productos en rebajas o en promociones. El primer estado "10" esta reservado como estado "sin-clasificar"

- **Únicos**. Es un estado que se asigna únicamente a **un producto** que **no está a la venta**. Se distinguen 3 tipos:

    - **Venta en proceso**: 10 estados posibles a generar por el administrador (20-29) que determina como está un producto que ha entrado en un proceso de venta pero aun no ha finalizado. Es decir por ejemplo el caso de un producto reservado para un cliente, pedido a un proveedor, pendiente de la aprobación de una financiación, etc… *El estado "20" está reservado* a estado *cumplimentado,* que es el estado en el que se encuentran los productos que finalizan el proceso de venta y por lo tanto se pueden contar como vendidos.

    - **Reparaciones**: 10 estados posibles a generar por el administrador (30-39) que determina en que estado se encuentra un producto que esta en reparación. Por ejemplo el producto que acaba de ser entregado por parte del cliente y permanece en el centro tendrá un estado diferente al producto que ya se ha enviado al servicio técnico o el producto que se ha recibido del servicio técnico y por lo tanto ya se puede llamar al cliente para que venga a recogerlo. *El estado "30" está reservado* a estado *entregado,* que es el estado en el que se encuentra un producto cuando finalmente ya se ha reparado y se ha entregado al cliente.

    - **Adecuaciones**: 10 estados posibles a generar por el administrador (40-49) que determina en que estado se encuentra un producto a adecuar. Un producto a adecuar es un producto en poder del departamento, es decir que no esta vendido, pero que por cualquier motivo no se puede poner a la venta, este producto ha de pasar por un proceso de adecuación para poner a la venta y en el caso de que este proceso no sea posible, su gestión para la devolución a proveedor. *El estado "40" esta reservado* para el estado *Devuelto* que es el estado en que queda un producto que ha finalizado su proceso de adecuación, pero no se ha podido poner a la venta y ha sido devuelto.

Estados	Nombre	Tipo	Reservado
0-9	Venta	Grupo	0-Vendido
10-19	Destacados	Grupo	10-Sin-Clasificar
20-29	Venta en proceso	Único	20-Cumplimentado
30-39	Reparaciones	Único	30-Reparado
40-49	Adecuaciones	Único	40-Devuelto

Tabla 3 Estados

Resumiendo, hay dos grupos posibles de estados, de grupo o individual. Los estados de grupo se asignan a toda una barra de productos a la venta y los únicos a un sn.

Un producto, si esta a la venta tiene asignado un estado de grupo, en el caso que tenga asignado un estado único significará que ese producto no está a la venta.

Cada grupo de estados tiene reservado el primero, como finalización del estado en el que se encontraba, es decir cada producto dentro de un grupo de estados tiene como destino llegar al primer estado. Esta norma se rompe únicamente en la familia de venta en proceso y adecuaciones que pueden pasar a la venta, y por lo tanto cambiar de grupo de estados.

Por ejemplo un producto del grupo venta tiene como destino llegar al estado 0 que seria vendido o un producto dentro del grupo destacados tiene el mismo destino, llegar al 0 porque ambos grupos son productos a la venta.

Un producto en venta en proceso tiene como destino ser cumplimentado, estado "20", que es el hecho de acabar la venta y entregar el producto al cliente.

En cambio un producto con un estado del grupo reparar tiene como destino llegar al primer estado de esa familia, es decir a reparado "30", no puede cambiar de grupo de estados ya que si un producto, por ejemplo, está a reparar pertenece a un cliente, no puede pasar a productos a la venta.

Como ya he comentado antes, esta norma se rompe en el grupo de estados venta en proceso y adecuaciones. Ya que un producto en venta en proceso siempre puede acabar rompiendo este proceso y ser devuelto a la familia de venta, por ejemplo si no se acepta una financiación o un producto reservado al final no es comprado. Otro caso seria el de las adecuaciones, ya que finalizada una adecuación si no se devuelve, el producto ha de volver al estado venta.

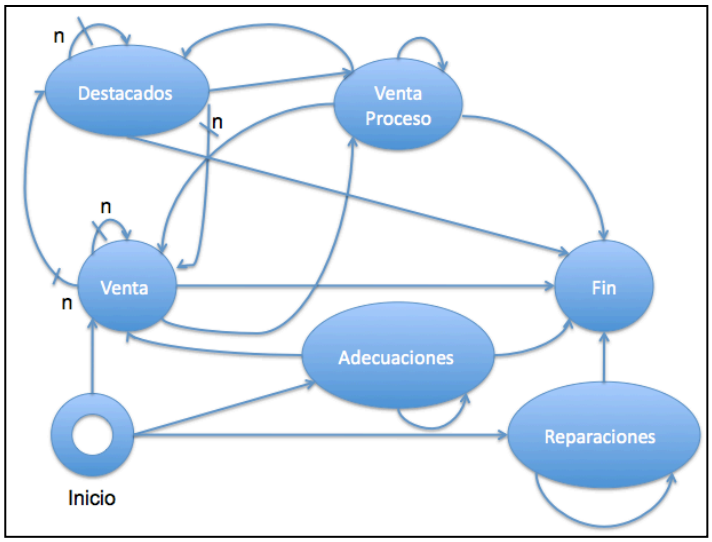


Figura 5 Máquina de estados

Comentar en estos dos casos que finalizar un estado en estado venta significa asignarle al producto un estado de grupo, con lo cual, ha de comprobarse en que estado se encuentra la barra y asignarle el estado de grupo al que corresponde. Por ejemplo si un producto en rebajas se reserva a un cliente pasa al estado único venta en proceso. Posteriormente si el cliente desea no finalizar la compra el producto retornara al estado rebajas que es al que pertenece a su barra.

El entorno además se compone de tres piezas más. Estas piezas son 3 apartados que ayudan a realizar las gestiones de productos que no se encuentran a la venta.

Los 3 apartados que se distinguen son los siguientes:

- **Pendientes**: Apartado el cual describe la venta en proceso y se compone de varios subapartados.

    - El **producto**. En este caso el producto que cursa esta venta en proceso.
    - **Vendedor**. El vendedor que ha iniciado esta venta en proceso
    - **Cliente**. Al cual pertenece la venta en proceso.
    - **Fecha**. La fecha en que se ha iniciado la venta en proceso.
    - **Fecha de compromiso**. La fecha prevista de finalización de la venta.

- **Reparaciones**: Apartado el cual describe los aspectos de una reparación y se distinguen varios subapartados.

    - El **producto**. El producto a reparar.
    - **Vendedor**. El vendedor que está gestionando la reparación.
    - **Cliente**. El cliente al que pertenecen el producto.
    - **Servicio técnico**.
    - **RMA**. Si se tiene, es el numero de incidencia que aporta el servicio técnico.
    - **Fecha**. La fecha en que se inicio el proceso de reparación.
    - **Fecha de Actualización**. Es la ultima fecha en la que se realizó algún cambio en la reparación.

- **Adecuaciones**: Apartado el cual describe los aspectos de una adecuación a venta o devolución y se compone de varios subapartados:

    - El **producto**. El producto a adecuar
    - **Vendedor**. El vendedor que está realizando la adecuación
    - **Proveedor**. El proveedor del producto en caso de devolución.
    - **Fecha**. La fecha en que se inicio el proceso de reparación.
    - **Fecha de Actualización**. Es la ultima fecha en la que se realizó algún cambio en la reparación.

3.3 Base de datos

La base de datos esta implementada en sqlite3. [XII] Dicho lenguaje "sql" permite gestionar una base de datos, realizando tanto consultas sobre ésta como cambios y modificaciones en ella.

Sql permite explotar la flexibilidad y la potencia de los sistemas relacionales, lo cual ayudará en gran manera a implementar la base de datos en cuestión, ya que los campos de las tablas en la base de datos están relacionados entre si.

Sqlite3 en concreto, por la sencillez de la base de datos resultante, ya que ésta es simplemente un archivo. Además del manejo, y la configuración de la misma, que son extremadamente sencillos, al no tener parámetros de configuración. El cliente Sqlite3, como ya se ha comentado trabaja sobre un sólo archivo, sin necesidad de ninguna configuración previa. Esto es muy útil para la portabilidad de la aplicación a cualquier sistema, ya que simplemente transportando o generando dicho archivo la aplicación es capaz de funcionar.

En un entorno, en el cual, la variedad de sistemas que se pueden encontrar es amplia, es clave poder disponer de la mayor sencillez posible. Esto se ajusta a la implementación que se pretende por parte del entorno generado, ya que, en concreto con el ejemplo de un centro comercial, la aplicación se ha de poder implantar en departamentos distintos, los cuales puede ser que tengan sistemas distintos. Por ejemplo el departamento de informática, dispone de un portátil con windows7, el cual es el que haría de servidor. En cambio en el departamento de telefonía, disponen de un imac, sistema diferente al de otros departamentos.

Esta sencillez en sqlite3 a su vez tiene puntos en contra, ya que no permite la configuración de acceso a varios usuarios. Dicha base de datos es accesible por cualquiera, sin el control de quien ha realizado los cambios.

Para poder solucionar esto, cosa que en un entorno con varios usuarios es clave, se opta por dos procedimientos.

El primero, el cual garantiza que nadie ajeno al entorno, puede alterar la base de datos, es modificar los permisos que los usuarios tienen sobre el archivo donde se encuentra la base de datos. Estos permisos han de ser de acceso total por parte del usuario servidor web y restringidos a todos los demás. Para garantizar que además sólo el administrador puede modificar la base de datos en el servidor, la aplicación web estará protegida con un nombre de usuario y contraseña. De esta manera solo el administrador puede modificar la base de datos desde la aplicación web y ningún usuario que tenga acceso al servidor podrá hacerlo de otra manera.

El segundo es la modificación de la base de datos por parte de la aplicación móvil. Como la base de datos tiene permisos de acceso total por parte del usuario servidor web, la aplicación móvil tiene total acceso, de esta manera cualquier usuario dado de alta como usuario vendedor, a través de la aplicación móvil puede modificar la base de datos. Por lo tanto todo el entorno y la aplicación móvil están diseñados para que los cambios efectuados por dichos usuarios queden registrados en la base de datos. De esta manera cualquier cambio, a parte de ser

reversible, es controlado por el administrador, pudiendo alterarlo según sus intereses.

Otro punto que se tiene en cuenta es la relación entre las tablas. Sqlite3 soporta relaciones entre tablas, claves primarias y foráneas. Pero a pesar de eso, se ha tomado la decisión de controlar este sistema de relaciones desde la propia aplicación, de esta manera se tiene una mayor robustez con la coherencia de los datos almacenados, al contemplarse todos los casos posibles desde la misma aplicación.

El diseño tal como se puede ver en la figura 6 parte del concepto del entorno descrito con anterioridad, y por lo tanto el eje central sobre el que gira toda la base de datos es también el producto.

La tabla "**producto**" es la base de todas las relaciones y el campo sn es el que une a las otras tablas.

La tabla "**altas**" es una extensión de la tabla producto que permite unir las características de familias de productos con estos mismos, teniendo la gran adaptabilidad que ha de tener todo el entorno, ya que se ha de poder aplicar en departamentos distintos, por lo tanto las características serán muy distintas.

A causa del gran dinamismo que ha de aportar la aplicación, no se podía optar por una tabla estática que guardará las características de los productos, ya que estos a priori, no se conocen. Por lo tanto se ha optado por la generación de una tabla por cada familia de productos que se introduzca en la base de datos. De esta manera cada vez que se añada una familia nueva al departamento se creará a su vez (por parte de la aplicación del administrador) una tabla, con las características de esta familia que el administrador crea que la describen con exactitud.

De esta manera la tabla posee los campos departamento y familia, que sirven de clave primaria y el campo tabla_c que sirve de puntero a la tabla que se ha añadido con la descripción de los productos. Dicho campo es el nombre de la tabla **FDepartamentoFamilia**. Por lo tanto será un enlace entre el departamento y la familia del producto y la tabla que contiene las características de esa familia.

La tabla "**Estados**" describe los estados que el administrador ha generado para cada producto.

"**Reparaciones**", "**pendientes**" , "**adecuaciones**" son tablas que contendrán todos los datos sobre la gestión de dichos procedimientos.

La tabla "**Almacenes**" contiene información sobre los almacenes generados por el administrador, y forma parte de la creación del entorno dinámico.

"**Cliente**" tal como su nombre indica contiene información sobre todos los clientes que han cursado alguna gestión en el departamento.

"**Vendedores**" es la tabla que contiene la información sobre que usuarios pueden utilizar la aplicación.

"**Tareas**" es donde se añaden las tareas a realizar por los vendedores.

"**Historial**" contiene información sobre los productos ya finalizados o en gestión y el vendedor que ha realizado la operación.

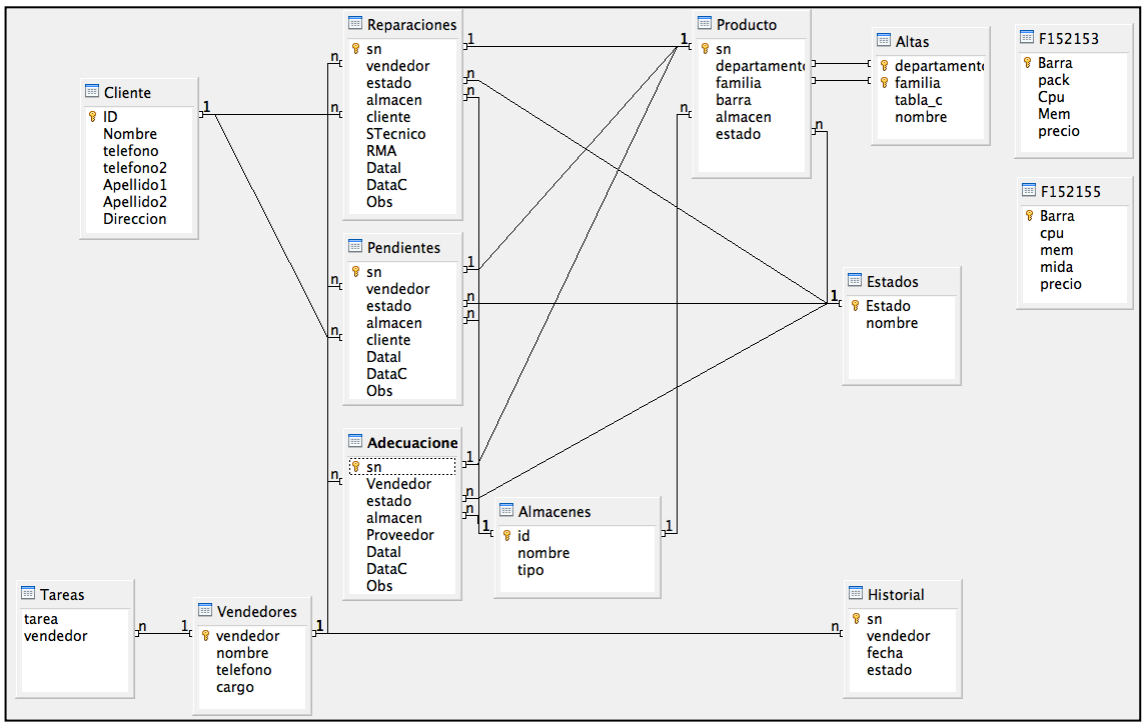


Figura 6 Diseño de la Base de Datos

3.3.1 Particularidades

Para poder interactuar con la aplicación, la base de datos esta dotada de unas particularidades que hacen que funcione adecuadamente. Sin éstas, la coherencia de la base de datos es correcta, y el funcionamiento de la misma también, pero a la hora de utilizar la aplicación, tanto la parte del administrador como la parte móvil, no podrá ejecutarse de manera correcta, ya que su implementación requiere de estas particularidades en concreto.

- Estados
  Los estados deben de estar asignados siguiendo un orden.
  El estado 0 esta reservado a estado "vendido", el 1 a "venta", y del 2 al 9 son posibles estados venta.
  El estado 10 esta reservado a estado "sin clasificar", el 11 a "destacado" y del 12 al 19 son posibles estados destacados.
  El estado 20 esta reservado a estado "Cumplimentado", el 21 a "reservado" y del 22 al 29 son posibles estados pendientes o reservados.
  El estado 30 esta reservado a estado "Reparado", el 31 "A-Reparar" y del 32 al 39 son posibles estados reparación.

El estado 40 esta reservado a estado “Devuelto”, el 41 “A-Adecuar” y del 42 al 49 son posibles estados adecuación.
La aplicación del administrador asigna por defecto los estados reservados y sólo deja modificar estados siguiendo la coherencia descrita anteriormente.

- Barra=’00000’
  Gracias a este pequeño recurso en la tabla FDepartamentoFamilia (donde se establecen las características de la familia), se podrá capturar si se desea los campos de las características de la familia en cuestión, haciendo una consulta sobre un producto de una familia determinada con campo barra=’00000’.

- Marca, modelo, precio
  Todos los campos de la descripción de una familia no están asignados previamente, han de escogerse libremente por el administrador, excepto tres campos que se asignan por defecto a toda familia generada. Marca, modelo y precio que son tres campos que por razones obvias cualquier familia posee.

3.4 Aplicación Web

La aplicación del administrador es una aplicación Web, implementada en lenguaje php que se ejecuta en el servidor e interactúa con la base de datos para proporcionar una Web en HTML que podrá ser ejecutada bajo cualquier cliente HTML. Para la corrección de posibles errores en la introducción de datos por parte del cliente Web, se ha utilizado JavaScript, por lo tanto el navegador también ha de soportar este tipo de lenguaje.

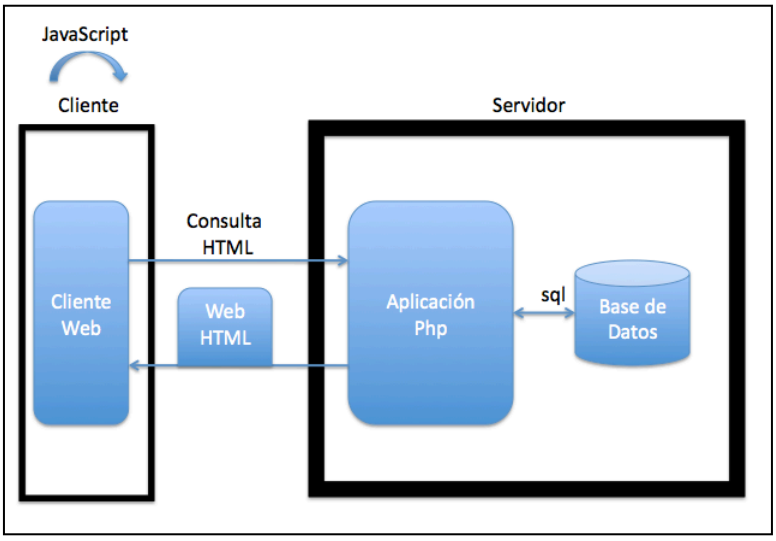


Figura 7 Diseño estructura aplicación Web

La aplicación esta diseñada para estar dividida en 4 apartados que interactúan con el entorno descrito en el punto 3.2, más el apartado de configuración de la propia aplicación.

El apartado stock está destinado a la creación y modificación de "stock" en el entorno. Este apartado es la herramientas mediante la cual el administrador puede modificar el stock de productos a la venta, modificando la cantidad de productos, donde se encuentran, en que estado (estados venta) se encuentra y las características (por barra) que tienen. A su vez este apartado también nos permite la introducción de nuevo "stock" con nuevas características.

El apartado Buscar nos permitirá buscar y modificar cualquier producto ya sea por referencia, características, almacén o vendedor .

El apartado listar nos listará cualquier producto en un estado de venta en proceso (pendientes), reparación, adecuación o ya finalizado, a su vez podremos modificar las características de los mismos. También nos mostrará un historial de cambios.

El apartado generación es el apartado que debe utilizar el administrador para generar todo el entorno. En este apartado encontraremos las herramientas para generar departamentos y familias nuevas, poder crear y modificar almacenes, generar los estados de los productos, dar de alta vendedores y asignarles tareas, introducir clientes y generar productos en venta en proceso, reparaciones y adecuaciones.

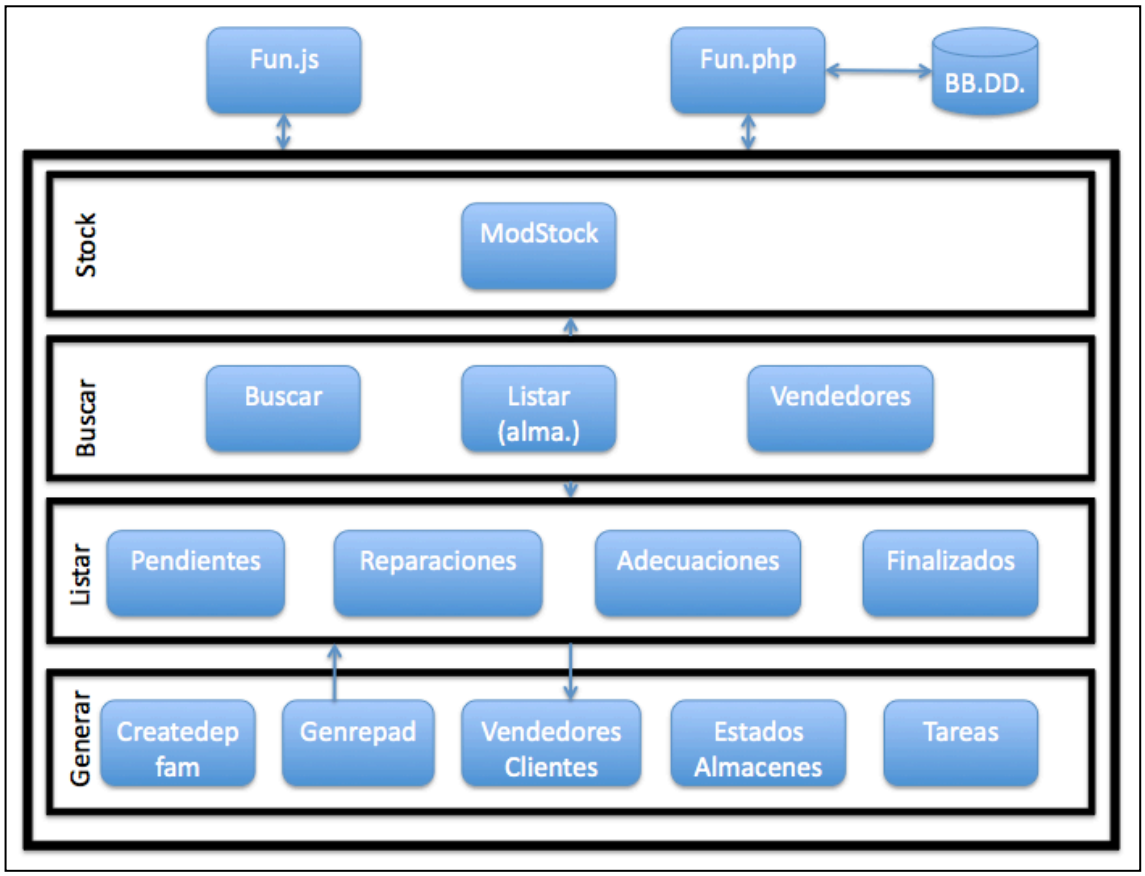


Figura 8 Diagrama de módulos Aplicación Web

Todos los módulos están conectados con fun.js que es el módulo que almacena todas las funciones JavaScript y con fun.php que es el módulo encargado de hacer la conexión y consulta con la base de datos.

El diseño se divide en 4 conjuntos de módulos como anteriormente se ha comentado. El primero, stock, contiene el módulo modstock que se encarga de introducir y modificar todos los aspectos relacionados con el stock. Almacén, cantidad, estado y características son las modificaciones que el modulo modstock puede realizar sobre el producto.

El conjunto de módulos buscar incluye los módulos destinados a realizar búsquedas sobre los productos. Éstas son por características o barra de las cuales se encarga el modulo Buscar, por almacenes con el módulo listar o por vendedores de la que se encarga el módulo vendedores. Este conjunto permite después de realizar una búsqueda hacer modificaciones sobre lo encontrado, por este motivo esta conectado con el conjunto stock y listar ya que estos dos conjuntos permiten hacer modificaciones sobre el stock.

El conjunto Listar es el encargado de listar los productos que se encuentran en un estado de grupo, tales como venta en proceso (pendientes), reparaciones, adecuaciones y productos que ya hayan finalizado su proceso y que por lo tanto se encuentren en un estado finalizado. Una vez listado el producto estos módulos también permiten hacer modificaciones sobre lo listado. A consecuencia de esto, están conectados con el modulo cliente del apartado generar, que es el que permite hacer modificaciones sobre estos mismos. Para los demás campos, tales como estado, vendedores, etc… es el mismo módulo el que accede a la información y por lo tanto no necesitan conexión ninguna con otro módulo.

El conjunto Generar constituye todos los módulos encargados de la generación del entorno, ya sea creando departamentos y familias con el módulo createdepfam, vendedores, clientes, estados, almacenes y tareas. El módulo Genrepad es el encargado de generar nuevas reparaciones, adecuaciones o reservas por lo tanto tiene una conexión con el conjunto de módulos listar.

3.5 Interfaz de comunicación

Para la conexión entre la base de datos y el dispositivo móvil se ha de diseñar una interfaz de comunicación que reciba como entrada consultas remotas y devuelva el resultado de la misma.

El diseño de ésta, ha de estar basado en web services, que implementada en código php sea capaz de recibir una consulta sql como parámetro, lanzar dicha consulta a la base de datos y devolver un xml bien formado como respuesta, este xml ha de tener los campos de la consulta como tags del xml y el valor de estos tags como el resultado de la consulta realizada.

Para describir el proceso de una consulta a través de la interfaz se muestra un ejemplo práctico.

Esta es una entrada posible que pueda recibir:

/interficie/consulta.php?consulta=select++from+vendedores*

Debemos obtener una respuesta de este tipo.

```
<?xml version="1.0" encoding="UTF-8"?>
<Tabla>
<tupla id="0"><vendedor>78193596</vendedor><nombre>Joan
Valls</nombre><telefono>639892043</telefono><cargo>Administrador</carg
o></tupla>
<tupla id="1"><vendedor>65460271</vendedor><nombre>Alejandro
Navarro</nombre><telefono>679143520</telefono><cargo>Vendedor</cargo><
/tupla>
<tupla id="2"><vendedor>65221210</vendedor><nombre>Oscar
Esclusa</nombre><telefono>648302123</telefono><cargo>Vendedor</cargo><
/tupla>
</Tabla>
```

Como se observa, el modulo consulta.php recibe como parámetro una consulta sql, (select * from vendedores) unidos por el símbolo "+". Obteniendo como salida un xml, estructurado con el resultado de la consulta. Éste está anidado dentro del tag general <Tabla>, donde cada tupla está anidado dentro del tag <tupla id=...>, donde cada id es la posición de la una tupla en la respuesta sql.

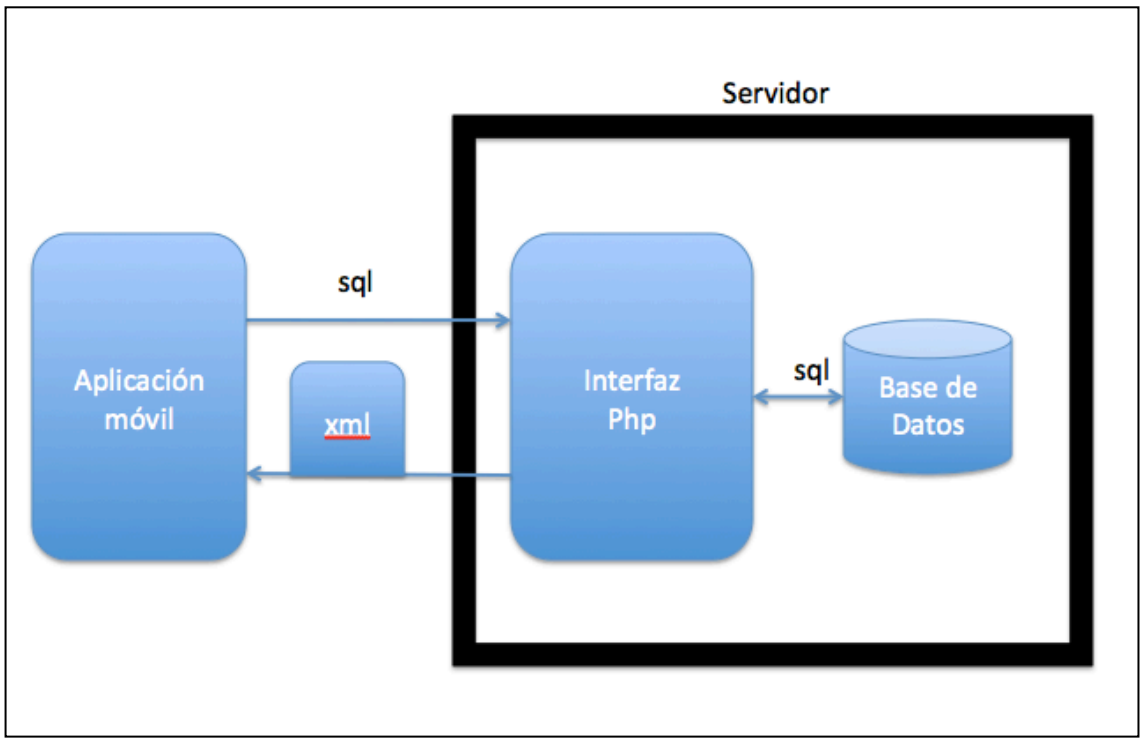


Figura 9 Diagrama módulos Interfaz de comunicación

3.6 Aplicación móvil

Como se ha comentado anteriormente ésta aplicación "StockTouch", estará implementada para funcionar nativamente en un dispositivo iPhone o iTouch. Este tipo de dispositivos ejecutan aplicaciones implementadas bajo xcode en ObjectiveC con las apis CocoaTouch. [V] El paradigma de programación que se plantea es orientada a objetos, por lo tanto se ha de crear un diseño de clases para la aplicación, que responda a los requerimientos de la misma.

3.6.1 Parser

La primera clase que se ha de diseñar será la clase "parser", esta clase deriva de NSobject (como todas las clases CocaTouch), y al ser iniciada ha de conectar con el servidor. Se ha de poder crear en cualquier otra clase de la aplicación. Ha de recibir una consulta como parámetro de entrada, abrir una conexión bloqueante para lanzar dicha consulta a la interfaz, recibir el xml proporcionado por ésta, parsear dicho xml y devolver una tabla con el resultado de la consulta.

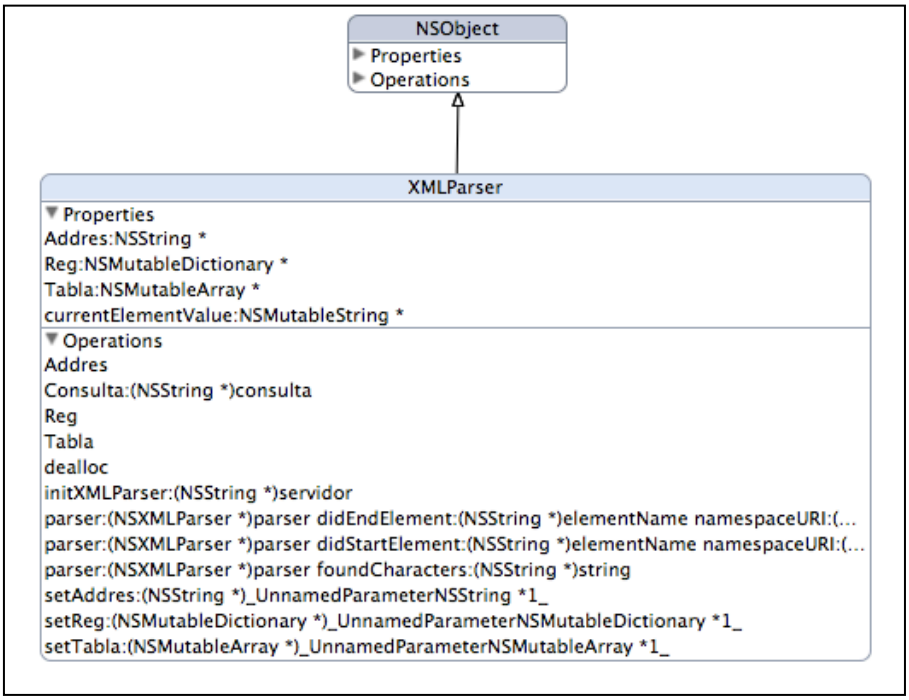


Figura 10 Clase XMLParser

Por lo tanto el método initXMLParser que se observa en el diseño de la clase será el encargado de iniciar el objeto e iniciar los atributos de la clase, en este caso la dirección del servidor. El método Consulta, que recibe una consulta como entrada, será el encargado de realizar la conexión, parsear y devolver la tabla con el resultado.

3.6.2 Cuerpo

El cuerpo de la aplicación esta diseñado para almacenar todos los datos provenientes del producto. La aplicación está ideada para poder realizar modificaciones sobre el producto, tanto si está a la venta, como si esta en otro estado. Para efectuar estas modificaciones la aplicación primero recogerá en algún punto los datos actuales del producto sobre el que esté actuando. Para éstos, se ha diseñado un conjunto de clases a modo de estructura de datos, que permitirán interactuar con ellos, de tal forma que cuando se haya acabado la modificación, éstos estarán en la estructura y permitirán la salida hacia la base de datos para aplicar los cambios.

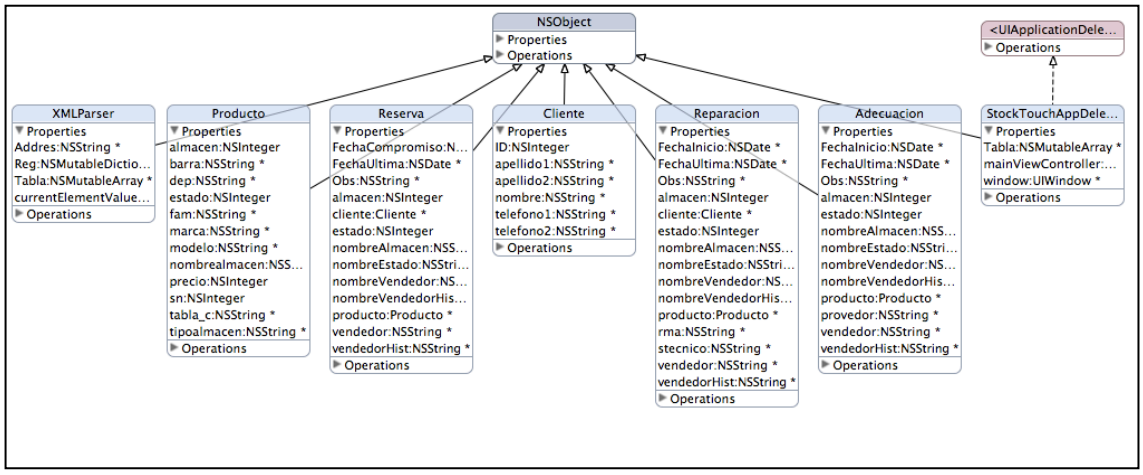


Figura 11 Clases del Cuerpo

3.6.2.1 Diseño de clases (cuerpo)

El diseño se basa en un objeto básico, el producto, con todas las características que puede llegar a tener. Éstas pueden ser (tal como se ve en la figura 12) sn, referencia, almacén, etc.. a partir de producto se crean reservas, reparaciones y adecuaciones que a parte de tener sus atributos específicos, tienen el atributo "producto" que será un objeto de la clase con el mismo nombre. De esta manera un objeto de la clase, por ejemplo reservas, tiene sus atributos específicos, como vendedor o cliente y a parte el producto en cuestión con su sn, referencia, estado, marca, modelo, precio, etc…

Si se observa (fig.12), la clase producto y las que se crean a partir de él, se advierte que hay atributos por duplicado, es decir almacén, por ejemplo se encuentra tanto en la clase producto como en la clase reserva. Esto se ha diseñado así, porqué una operación puede jugar con varios campos idénticos a la vez. Por poner un ejemplo, en una reserva, el producto esta alojado en un almacén y cuando se reserva puede ir a otro almacén, de este modo es necesario la duplicación de estos atributos.

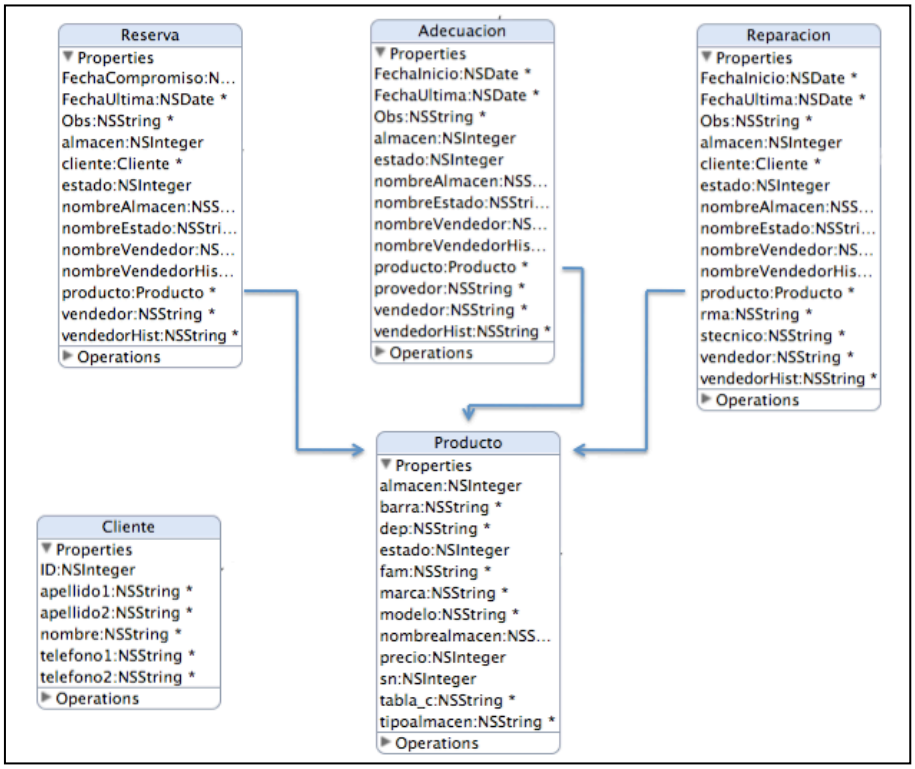


Figura 12 Diseño clases (Cuerpo)

Otro caso es la clase cliente, que actuará como estructura para almacenar los datos del cliente en el cual la aplicación este realizando en ese momento una operación.

3.6.3 View

Existen varios tipos de diseño de vistas aplicables a un proyecto de software para el iPhone. Se ha escogido uno en el cual la aplicación esta compuesta de 4 grandes apartados. Éstos derivan de UIViewController *(clase de control de vista) [V]*, donde cada apartado será un UITabBarController *[punto 3.6.3.1]* dividido en varios UInavigationController *[punto 3.6.3.2].* A su vez este UInavigationController estará compuesto por varios UIView *(vista)* donde en este caso serán un UITableView *(vista de tabla).*

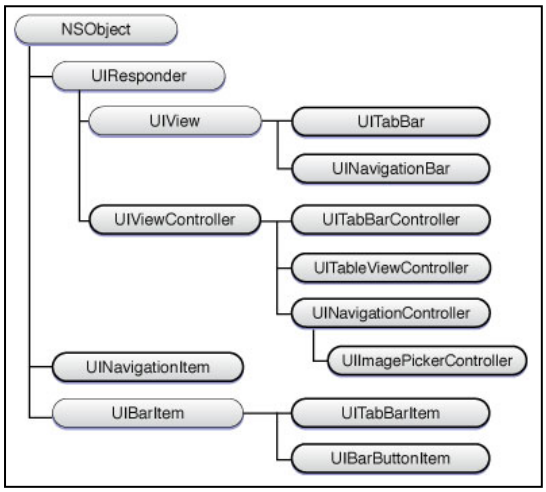


Figura 13 Herencia clases View

3.6.3.1 UITabBarController

UItabBarController es un tipo de vista el cual puede almacenar otros tipos de vistas controlables desde la barra inferior, pudiendo escoger la vista deseada desde esta barra.

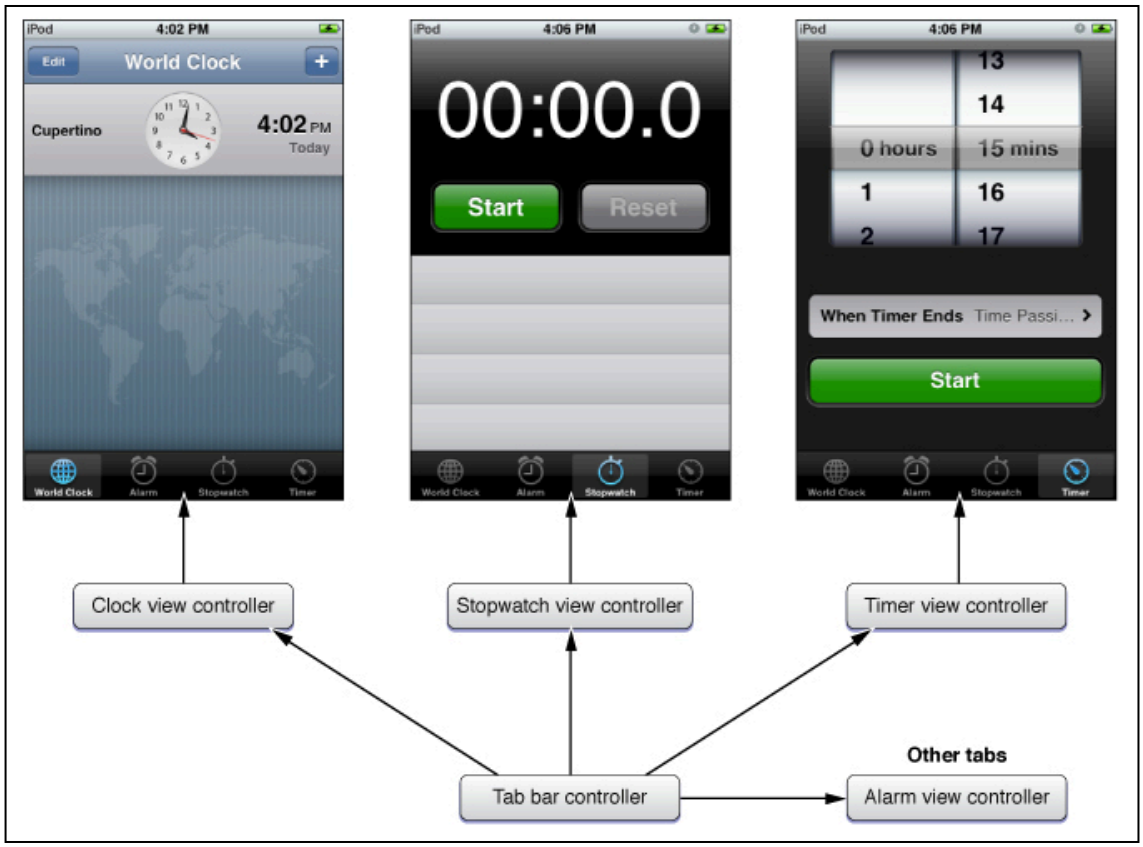


Figura 14 Vistas en UItabBarController

Para llevar a cabo este tipo de vistas se inicializa una array con todos las vistas que se desea incorporar en UITabBarController. Estas vistas pueden ser simples, compuestas, o como en el caso que nos ocupa, cada vista almacenada en esta array será un UINavigationController. *[punto 3.6.3.2]*

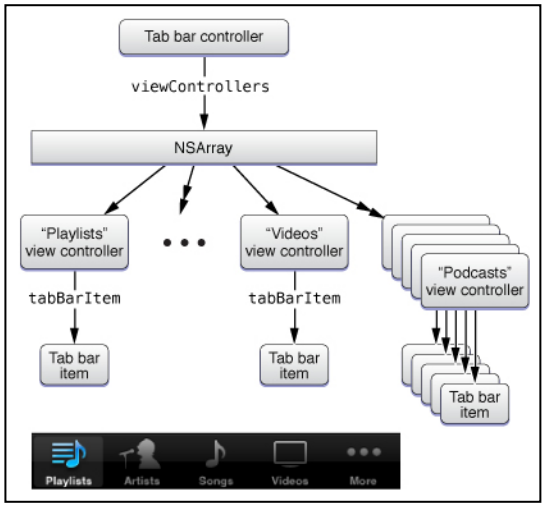


Figura 15 Array UItabBarController

3.6.3.2 UINavigationController

UInavigationController es un tipo de vista compuesto de muchas vistas, a modo de navegación estas vistas van apareciendo a medida que se va avanzando en la aplicación. Por ejemplo, para seleccionar un producto primero tendríamos que seleccionar su familia, después la marca y por ultimo el modelo. Para conseguir esto se navega desde una tabla con todas las familias posibles, se selecciona una y aparece otra tabla con todas las marcas posibles de esa familia, si por el contrario se desea seleccionar otra familia existe una barra superior la cual incluye unos botones para poder navegar hacia atrás, todo esta forma de navegar entre vistas se consigue mediante un UINavigationController.

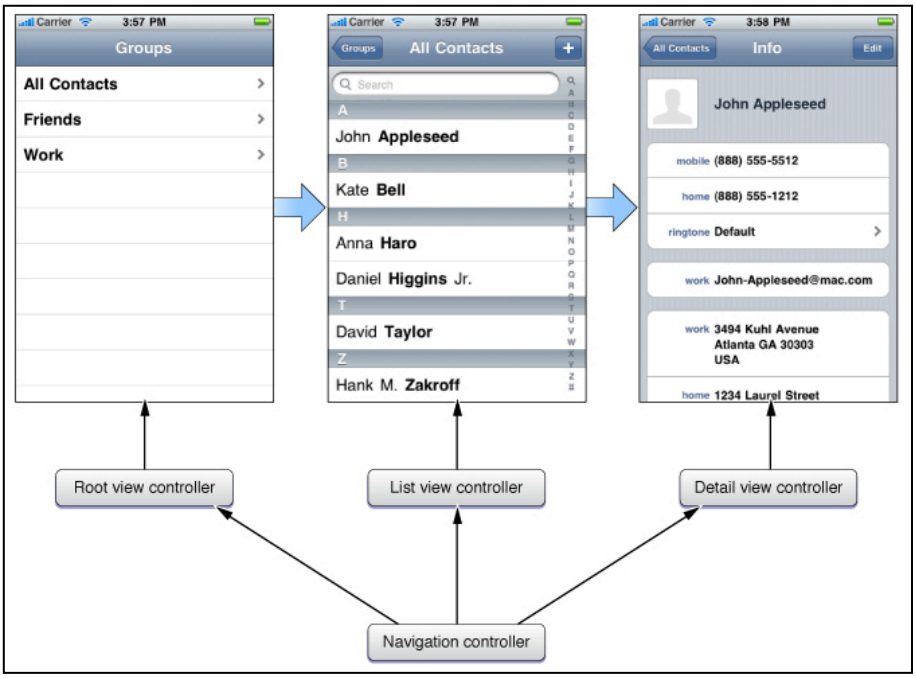


Figura 16 Vistas en UINavigationController

UINavigationController se inicializa con la primera vista, y a medida que se avanza en la aplicación se añaden vistas, de forma que se acumulan en una pila de navegación.

De este modo cuando se navega hacia atrás se extrae la ultima vista de la pila de navegación y cuando se navega hace adelante se añade una vista a la pila de navegación.

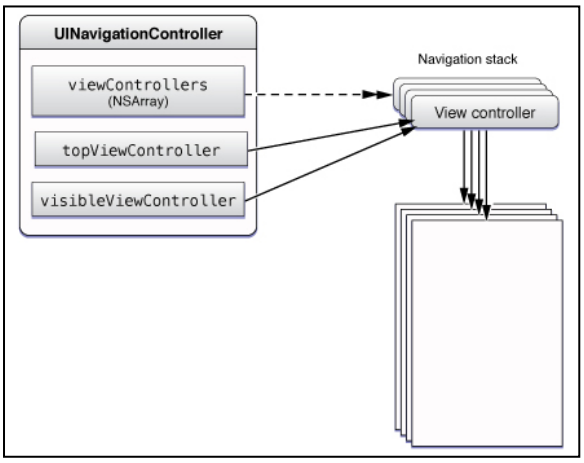


Figura 17 Pila de navegación

3.6.4 Diseño de Clases

Como ya se ha comentado con anterioridad, la aplicación del dispositivo móvil esta dividida en 4 grandes apartados:

- Venta
- Gestión
- Alta
- Mis Cosas

Al iniciar la aplicación, ésta pasa a un estado de carga controlado por el objeto StockTouchAppDelegate que es el objeto delegado de controlar la aplicación. Una vez finalizada la carga del programa, este objeto llama al objeto MainViewController que se encarga de comprobar si hay conexión, y si la hay se encarga de inicializar las vistas de los 4 apartados mencionados anteriormente, mostrar la pantalla principal, los botones para acceder a ellas y las vistas de ayuda.

VentaRootViewController, GestionRootViewController, AltaRootViewController y MisCosasRootViewController son objetos UINavigationControllers embebidos dentro de un UITabBarController, que mediante el atributo "caso" de cada clase se inicializan en una vista u otra.

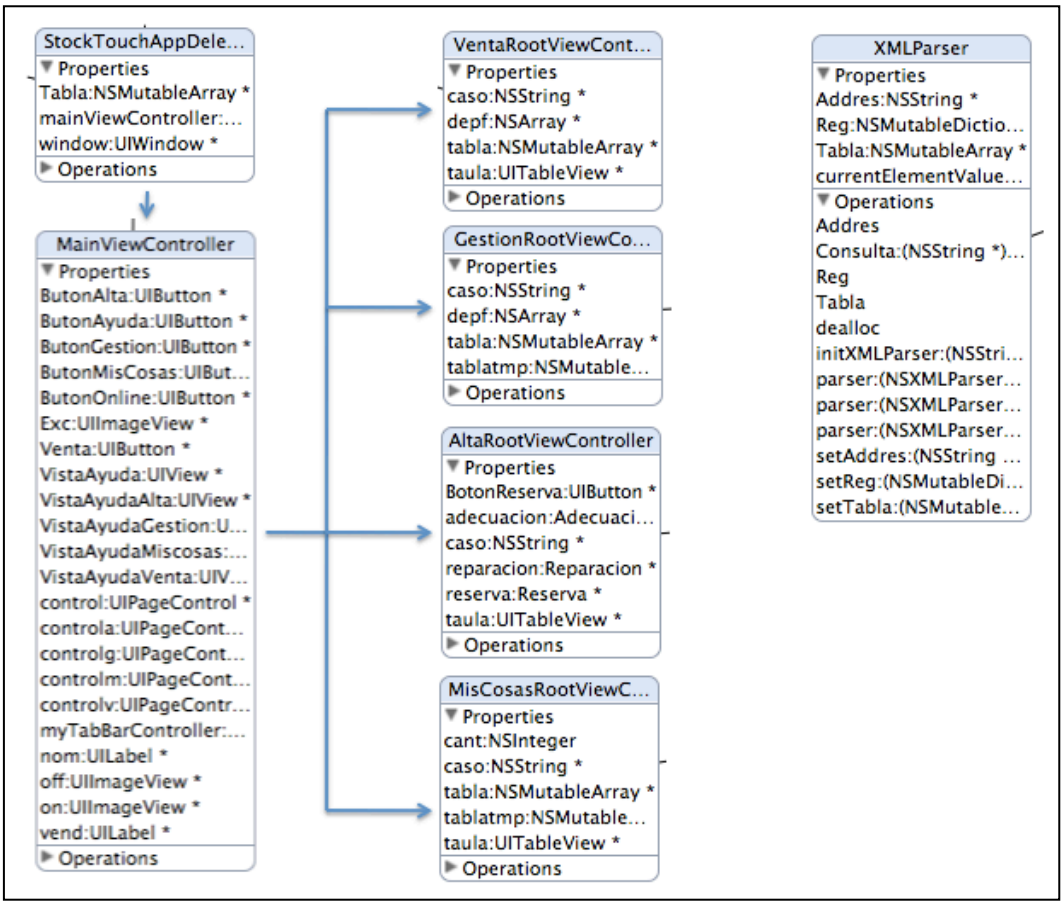


Figura 18 Llamada a clases MainViewController

Es decir MainViewController se encarga de iniciar 4 UItabBarControllers y dentro de cada uno de ellos iran los UINavigationControllers correspondientes.

Un ejemplo seria el apartado venta. MainViewController al seleccionar el botón "Venta" se encarga de crear un UItabBarController, crea una array, crea 5 objetos nuevos de la clase UINavigationController, añade a la pila de navegación de cada UINavigationController uno de estos 5 objetos nuevos de la clase VentaRootViewController, con un caso distinto para cada uno de ellos, y coloca estos UINavigationControllers dentro de la array de UItabBarController.

Por último carga la vista del UItabBarController, añadiendo a la barra de navegación un botón de menú desde el cual se descarga dicha vista para volver a MainViewController.

3.6.4.1 Venta

El apartado Venta está dividido en 2 diseños. El primero es la selección o búsqueda de un producto para su posterior venta, y por lo tanto un diseño de clases para mostrar vistas que ayuden al usuario a buscar el producto deseado. El segundo es un formulario de reserva, donde la clase ReservaViewController estará respaldada por varias clases que permitirán completar los campos del formulario.

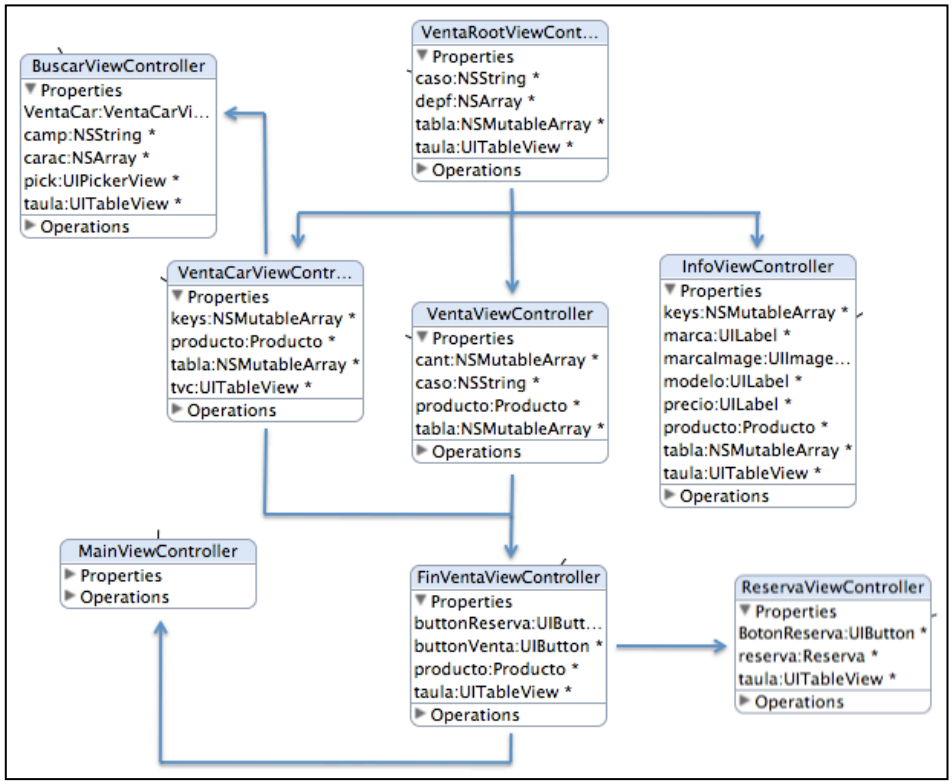


Figura 19 Llamadas a clases Venta

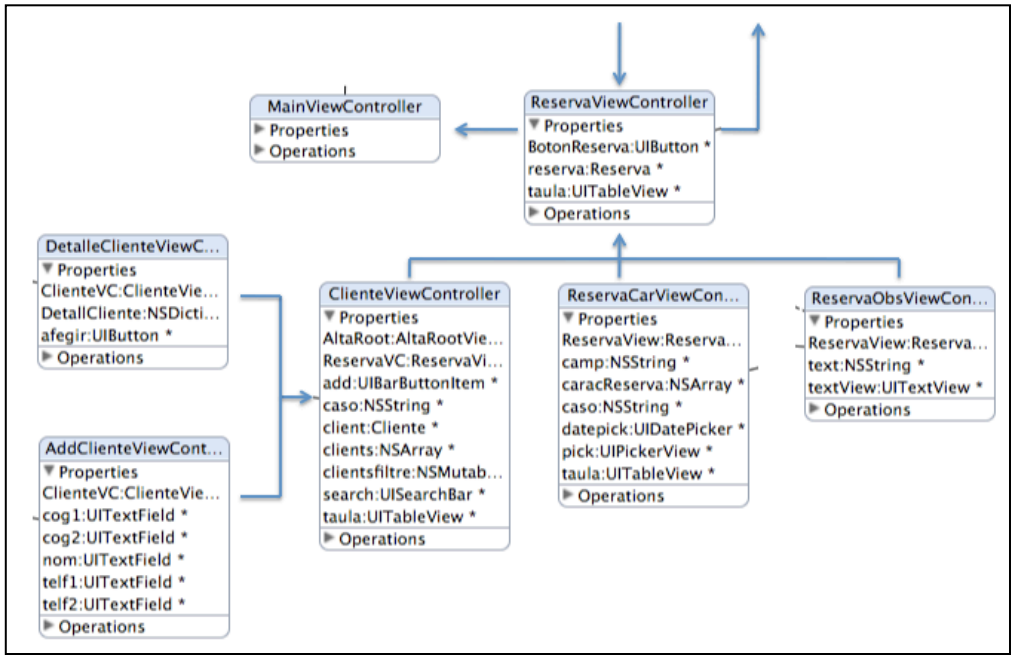


Figura 20 Diseño Clases Reserva

ReservaViewController es la clase encargada de mostrar el formulario para rellenar la reserva, ReservaObsViewController es una clase que deriva de TextViewController, que servirá para mostrar vistas en las que se pueda rellenar un campo de texto, ReservaCarViewController es un vista que se encargara de mostrar un UIpicker, herramienta de selección proporcionada por la api del iPhone al estilo de un menú contextual y clienteViewController será la clase encargada de elegir un cliente para la reserva o crearlo. Una vez completos los campos del formulario el objeto de la clase ReservaViewController se encarga de dar de alta la reserva y devolver el control al objeto de la clase MainViewController.

3.6.4.2 Gestión

El apartado gestión permite gestionar todos los productos del departamento que no se encuentren a la venta. Sigue la misma estructura descrita en el apartado Venta, excepto que en este apartado a diferencia del de venta, hay dos clases raiz,. Una clase GestionRootViewController es llamada por MainViewController y según el "caso" inicializa una vista u otra, y GestionClienteView que es la encargada de mostrar una lista con todos los cilentes que tienen algún producto en tramitación, para su selección y gestión. GestionViewController se encarga de generar las vistas del cuerpo del apartado Gestión, y FinGestiónViewController muestra el formulario final con los diferentes campos del producto en cuestión, se ayuda de la misma manera que en el apartado venta, de las clases GestionCarViewController y GestionObsViewController, que mostraran una vista de texto o un UIPicker.

Al finalizar la gestión se debe devolver el control a la clase MainViewController.

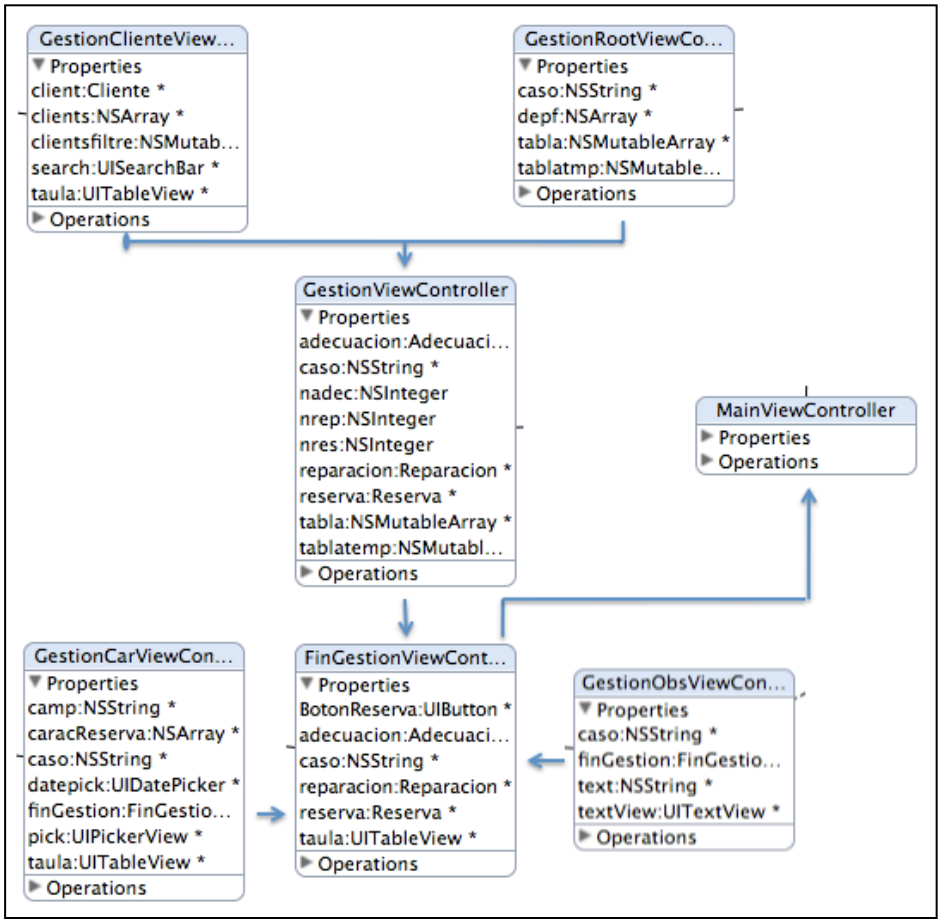


Figura 21 Diseño Clases Gestión

3.6.4.3 Alta

El apartado Alta, muestra un formulario para dar de alta un producto que no esté a la venta. Por lo tanto estará, en los diferentes grupos de estados que se pueden generar en la aplicación. Resumiendo, este apartado servirá para dar de alta una reparación, adecuación, reserva o cambio (adecuación + eliminación de un producto a la venta).

AltaRootViewController será la raíz de este apartado y también la clase encargada de generar el formulario para realizar el alta de un producto en reparación, reserva, adecuación o cambio. Se ayuda de las clases AltaCarViewController, AltaObsViewController y ClienteViewController para rellenar los campos del formulario. Estas clases del mismo modo que en los apartados anteriores, muestran una vista formada por un texto o un UIPicker. ClienteViewController, mostrará una tabla o formulario con los diferentes clientes, para poder añadirlos.

Además la clase AltaViewController, será la encargada de buscar el producto, que se dará de alta, ya sea por marca-modelo, o por referencia, para rellenar el campo producto del formulario de alta.

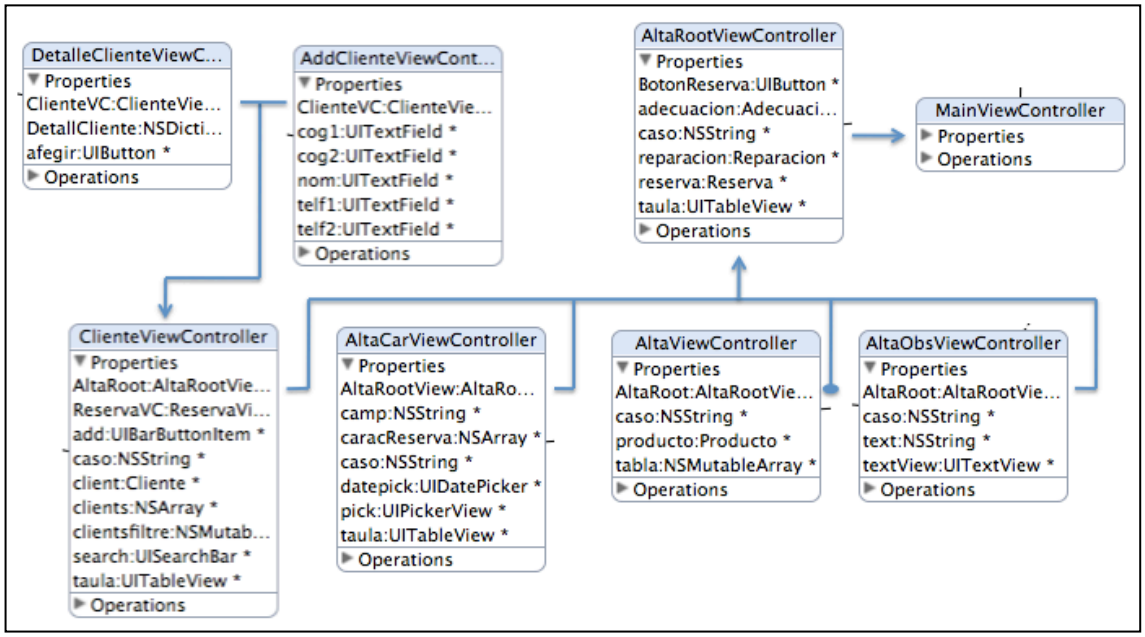


Figura 22 Diseño de Clases Alta

3.6.4.4 Mis Cosas

Mis Cosas es un apartado especial, parecido en parte al apartado gestión, pero con algunas diferencias y con unas funcionalidades extras exclusivas de este apartado.

En primer lugar el apartado "Mis Cosas" gestiona todos los productos que se encuentran en un estado fuera de venta, pero en este caso, en vez de controlarlos todos, sólo controla los del usuario de la aplicación. De esta manera funciona como una agenda personal, con la gestión de todos los productos que controla el vendedor.

En segundo lugar, el apartado Mis Cosas muestra un historial con los productos vendidos por el vendedor, y los cambios en los estados de los productos que ha generado este usuario.

En tercer lugar, el apartado Mis Cosas muestra una lista con las tareas asignadas por parte del administrador, y da la posibilidad de finalizar dichas tareas.

Como último punto, si la tarea Inventario esta asignada por parte del administrador al vendedor en cuestión, se activa un subapartado más, "Inventario", mediante el cual, éste puede hacer el inventario de la cantidad productos que hay en cada almacén y que están a la venta en el departamento.

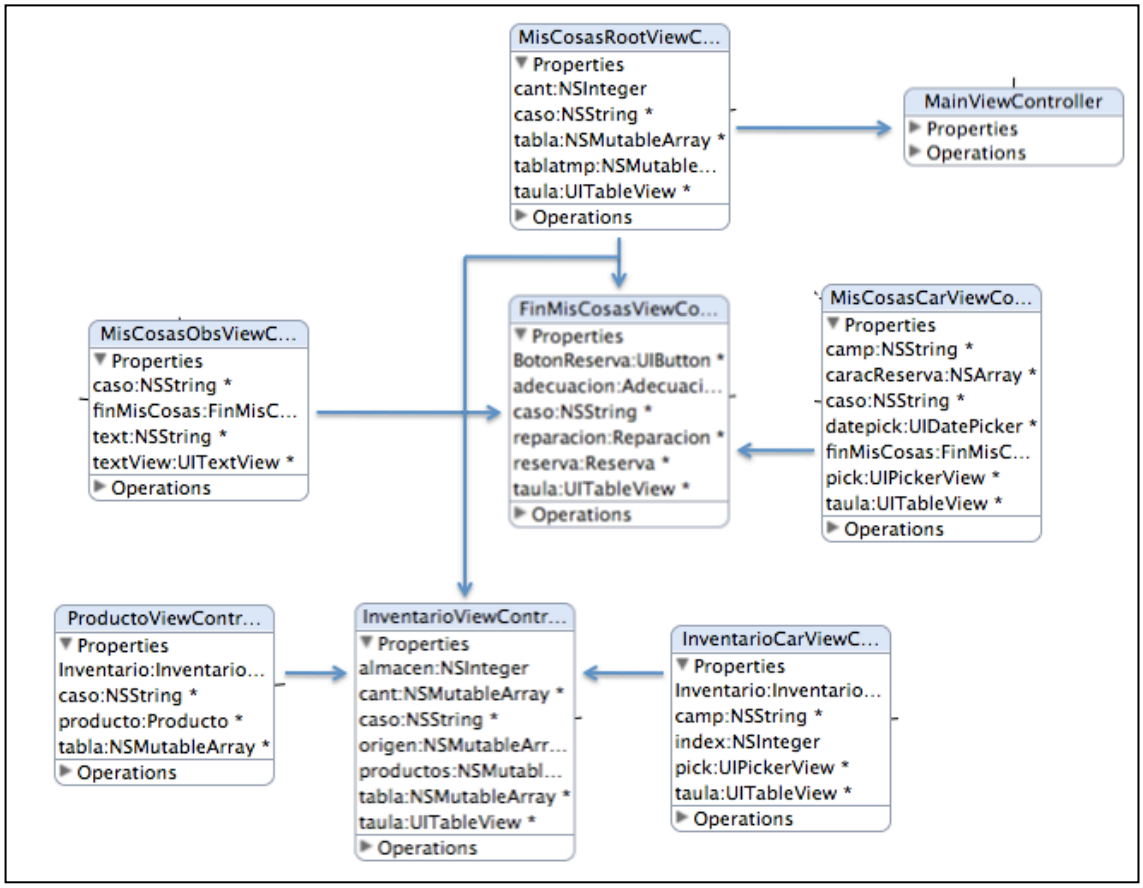


Figura 23 Diseño Clases Mis Cosas

Como en los diseños anteriores la clase raíz MisCosasRootViewController es la llamada por MianViewController, ésta se encarga de mostrar las vistas necesarias para poder seleccionar el producto que se desea gestionar, cuando finaliza la búsqueda pasa el control a FinMisCosasViewController que genera un formulario ayudada por las clases, MiscosasCarViewController y MisCosasObsViewController mediante la cual, puede rellenarlo y aplicar los cambios.

La clase raiz, en este caso, también es la encargada de mostrar la lista con las tareas asignadas al vendedor y ofrece la posibilidad de eliminarlas. Por otra parte también es la encargada de mostrar el historial del vendedor.

Si la tarea Inventario está asignada entra en juego la clase InventarioViewController que ayudada por las clases InventarioCarViewController y ProductoViewController será la encargada de modificar el stock en los diferentes almacenes con los productos que el usuario crea conveniente.

Capítulo 4

Implementación y funcionamiento

En este capítulo se presentan los detalles concretos de la implementación del proyecto y del funcionamiento del mismo. Éste, en gran parte, es una experiencia interactiva, por lo tanto se explicarán la mayoría de sus procesos con ejemplos visuales y argumentándolos con diagramas que ayuden a entender el flujo de trabajo que puede percibir el usuario.

4.1 Aplicación Web

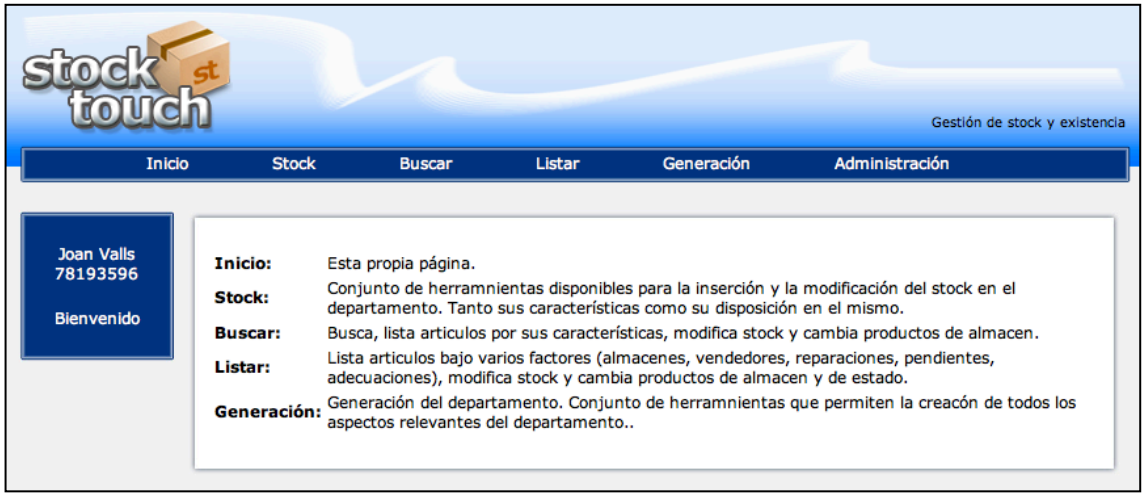


Figura 24 Aplicación Web

A continuación se describirá paso por paso de que forma un usuario puede interactuar con la aplicación. Se mostrarán capturas con la visión que el usuario tiene de los diferentes apartados en cuestión.

Los diagramas explicarán el flujo que el usuario puede seguir en cada apartado.

Como se puede observar en la figura 24 la aplicación esta dividida en 4 apartados que interactúan con el entorno descrito en el tercer capítulo, más el apartado de administración de la propia aplicación.

4.1.1 Stock

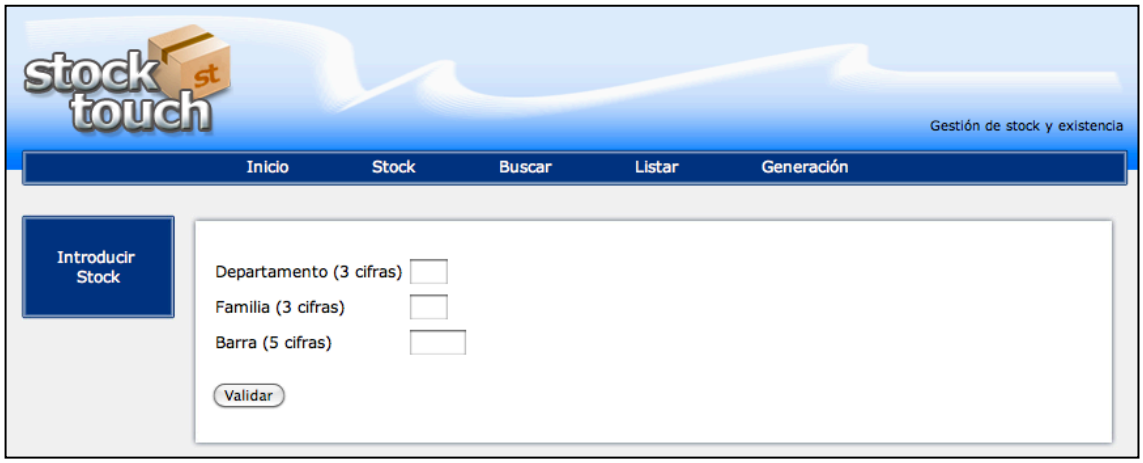


Figura 25 Aplicación Web Stock

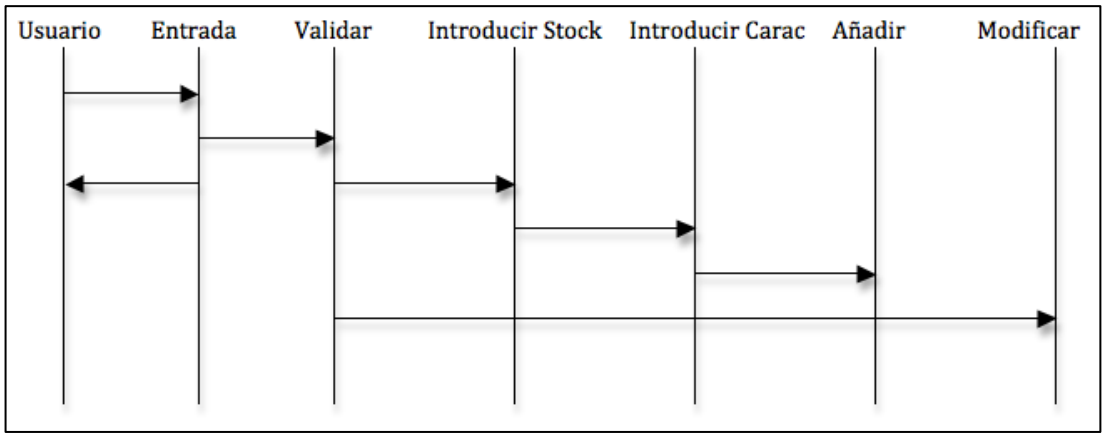


Figura 26 Flujo Stock

El usuario entra el código de barras correspondiente, si el departamento y familia no están creados devuelve un mensaje de error. Si es correcto y la barra no existe, aparecen todos los campos de las características de la familia en blanco para rellenar por el usuario, si ya existe la barra, aparecen los campos con las características correspondientes para poderse modificar.

En ambos casos puede escogerse el almacén, estado y la cantidad de productos a almacenar. En caso de que la barra exista, por defecto te escoge el estado de grupo venta o destacado que le corresponde y el almacén donde más productos de su misma barra hay.

Al modificar cantidad de productos la aplicación utiliza como recurso el almacén por defecto libre. Al añadir productos a un almacén, primero los restará del almacén libre, si hay, y a continuación añadirá nuevos. En cambio al eliminar productos de algún almacén la aplicación los pasa a almacén libre, si se desea eliminarlos definitivamente, hay que eliminarlos del almacén libre.

4.1.2 Buscar

Figura 27 Aplicación Web Buscar

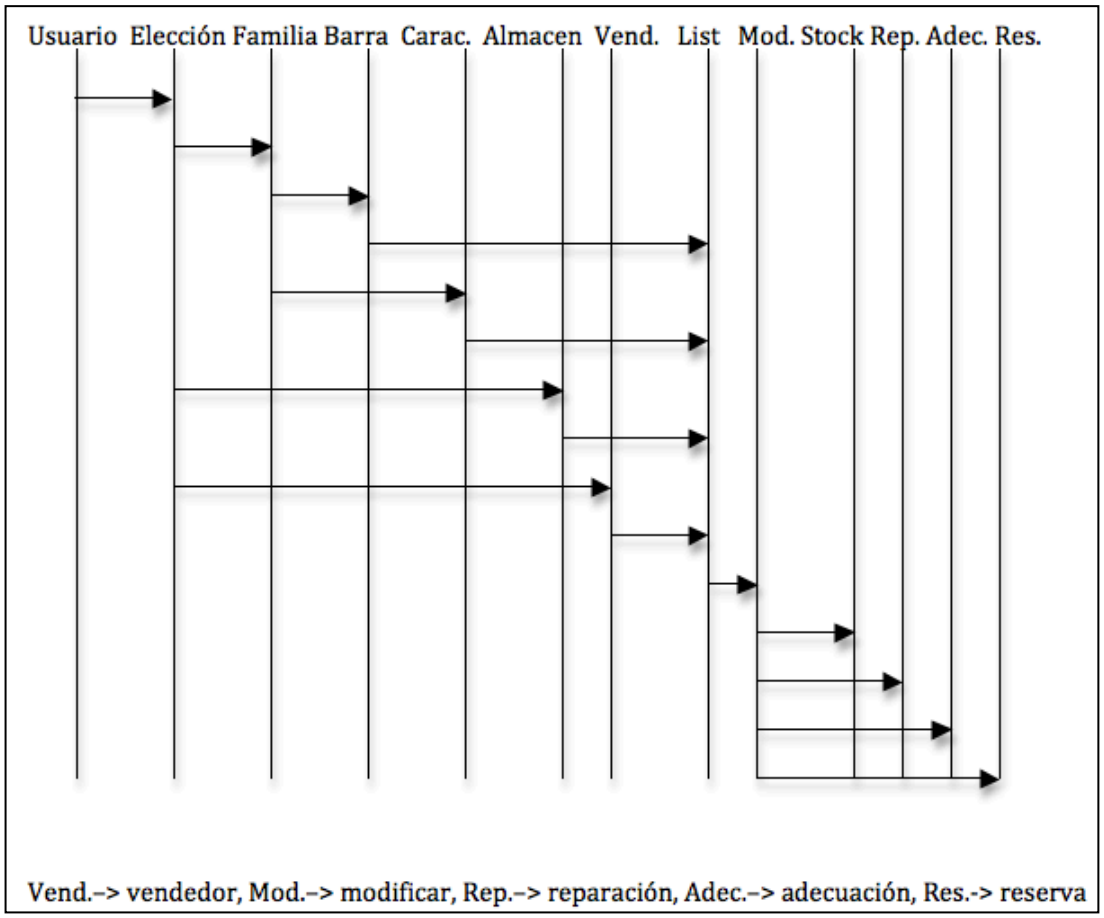


Figura 28 Flujo Buscar

El usuario puede escoger búsqueda por referencia, característica de familia, almacén o vendedor. Una vez realizada la selección el programa mostrará un listado con los productos filtrados, y dará la posibilidad de modificarlos según sea su estado, a la venta (stock), reparación, adecuación o venta en proceso.

4.1.3 Listar

Figura 29 Aplicación Web Listar

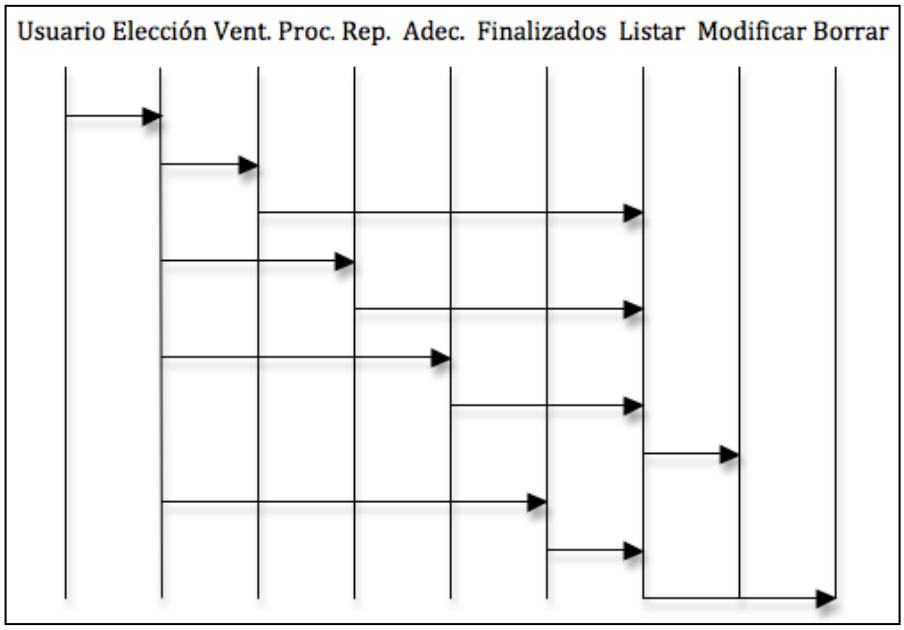


Figura 30 Flujo Listar

El usuario puede escoger que tipo de productos desea listar, productos en estado reparación, venta en proceso o adecuación. La aplicación lista estos productos y se pueden modificar según el estado en el que estén.

A parte el usuario puede listar el historial de cambios realizado y el conjunto de productos en un estado finalizado es decir en los estados 0,10,20,30 o 40 y eliminarlos de la base de datos.

4.1.4 Generación

El apartado Generación incluye varios apartados que permiten al administrador construir todo el entorno, a parte de generar reservas, reparaciones o adecuaciones.

4.1.4.1 Generación reservas, reparaciones o adecuaciones

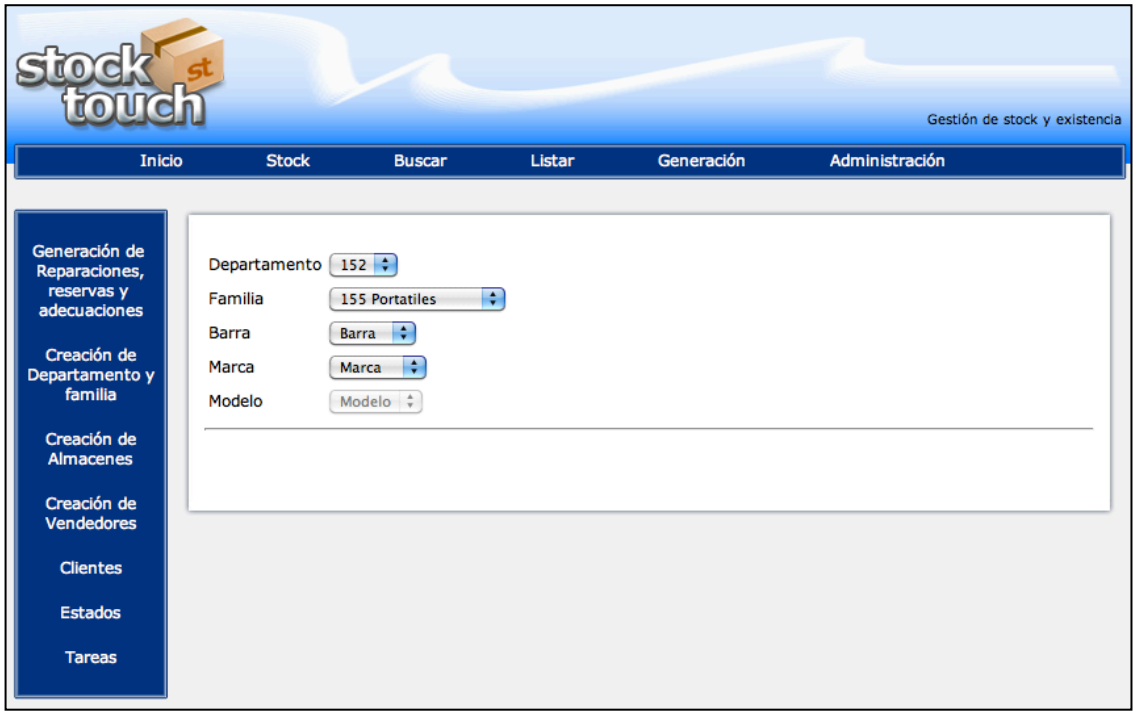


Figura 31 Aplicación Web generar reparación, reservas o adecuaciones

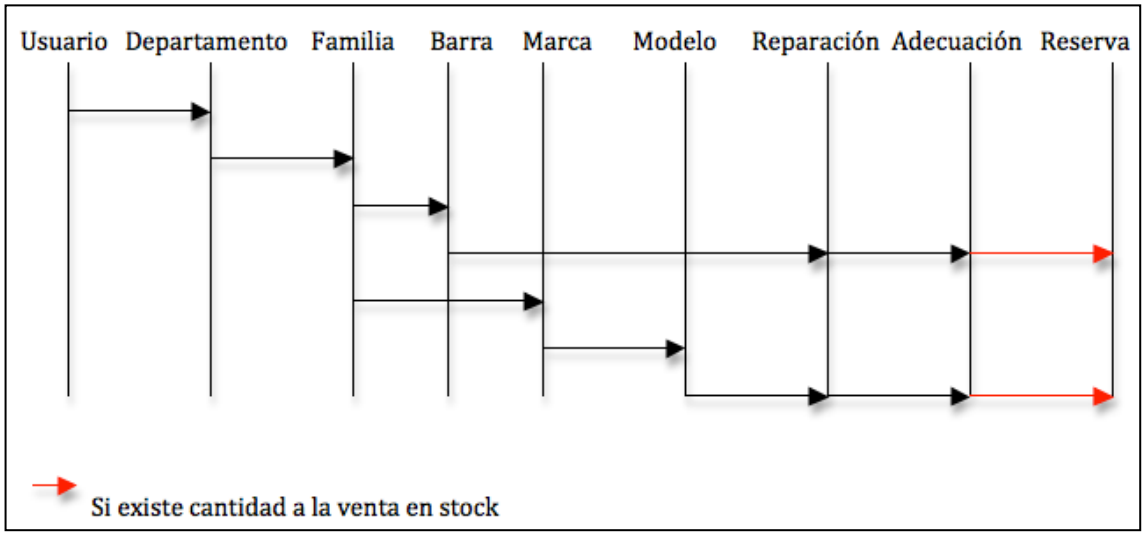


Figura 32 Flujo Generar reparaciones, reservas o adecuaciones

El usuario escoge el producto para generar la reparación, adecuación o reserva. Una vez escogido el producto aparece un formulario de alta mediante el cual puede dar de alta el producto a reparar, adecuar o reservar.

La selección de este producto se realiza mediante marca-modelo de la familia seleccionada, o realizando una búsqueda por referencia, es decir seleccionando departamento, familia y barra.

Éste se puede escoger, haya existencias en el departamento o no, ya que se asume que una reparación o adecuación no provienen de productos a la venta. En cambio, si no hay ninguna cantidad de ese producto a la venta, el formulario reserva no aparece, ya que no se puede reservar algo de lo que no hay existencias.

Como la generación de una reparación, adecuación o reserva son distintas entre si, se pasa a detallar el funcionamiento de cada apartado en concreto.

4.1.4.1.1 Reparación

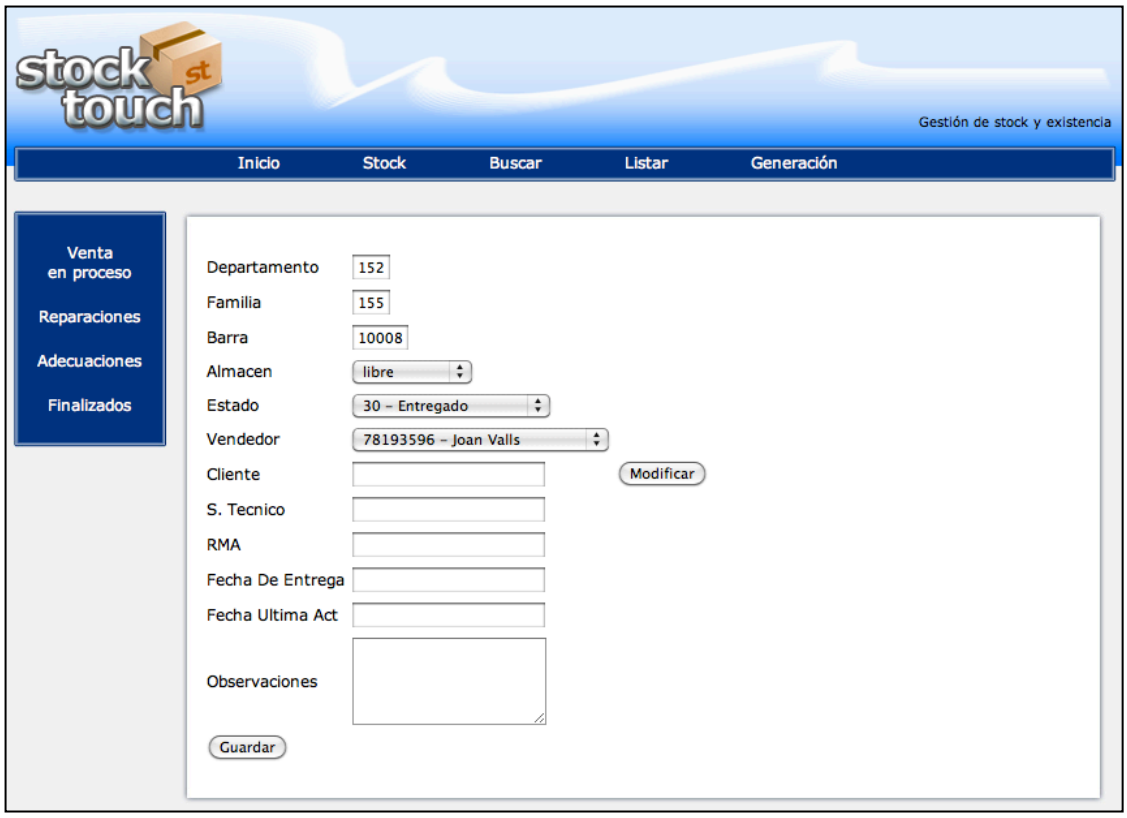


Figura 33 . Aplicación Web generar reparación

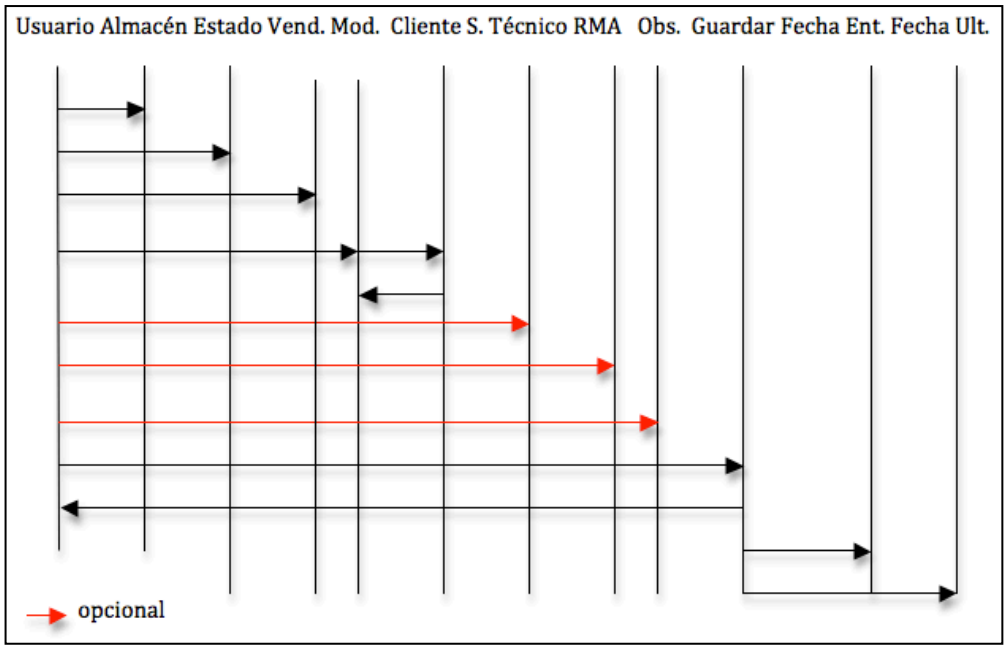


Figura 34 Flujo generar reparación

El usuario Introduce los datos de la reparación en el formulario. El campo cliente es un formulario que pertenece a la sección cliente. Los campos servicio técnico, rma y observaciones son opcionales. Si los campos no opcionales están vacíos la aplicación devuelve un error.

4.1.4.1.2 Adecuación

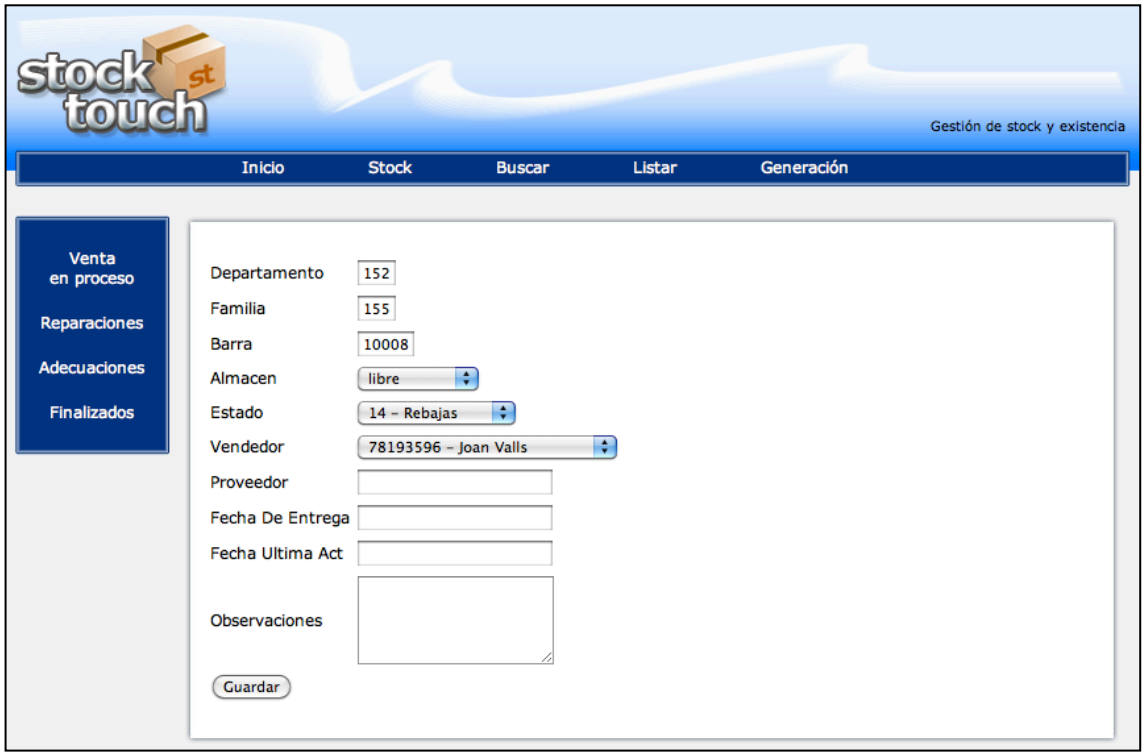


Figura 35 Aplicación Web generar adecuación

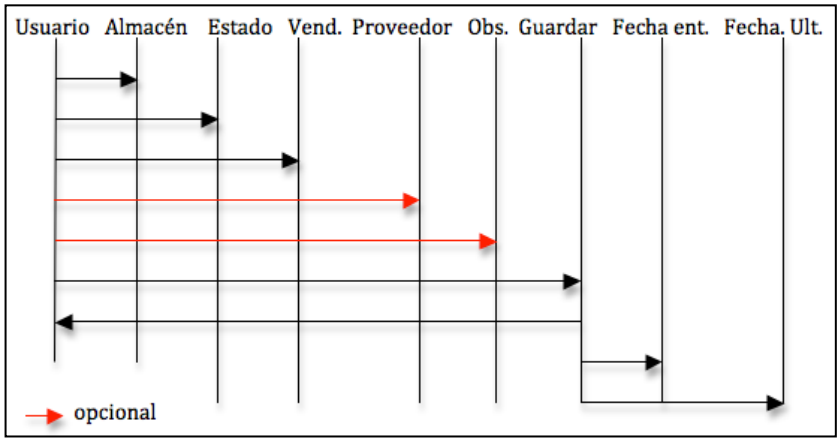


Figura 36 Flujo generar adecuación

El usuario introduce los datos de la adecuación en el formulario. El proveedor y las observaciones son opcionales. Si los campos no opcionales están vacíos la aplicación devuelve un error. Igual que en el punto anterior y en el siguiente, al guardar, la aplicación se encarga de actualizar los campos de fecha.

4.1.4.1.3 Reservas

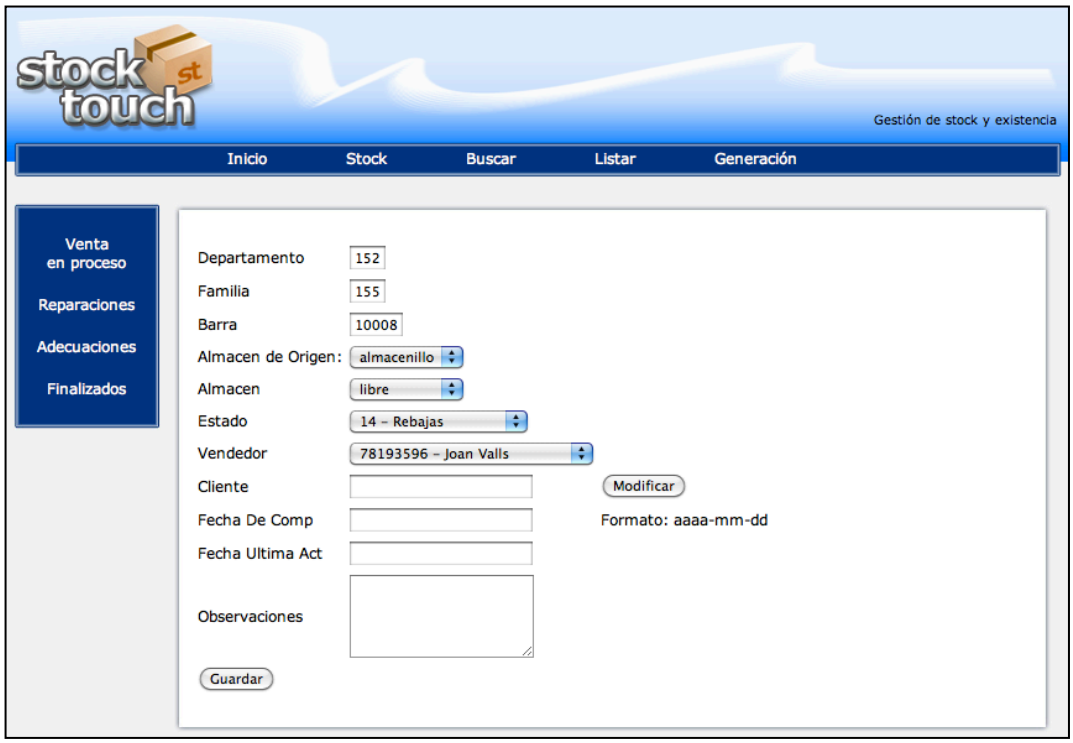


Figura 37 Aplicación Web generar reservas

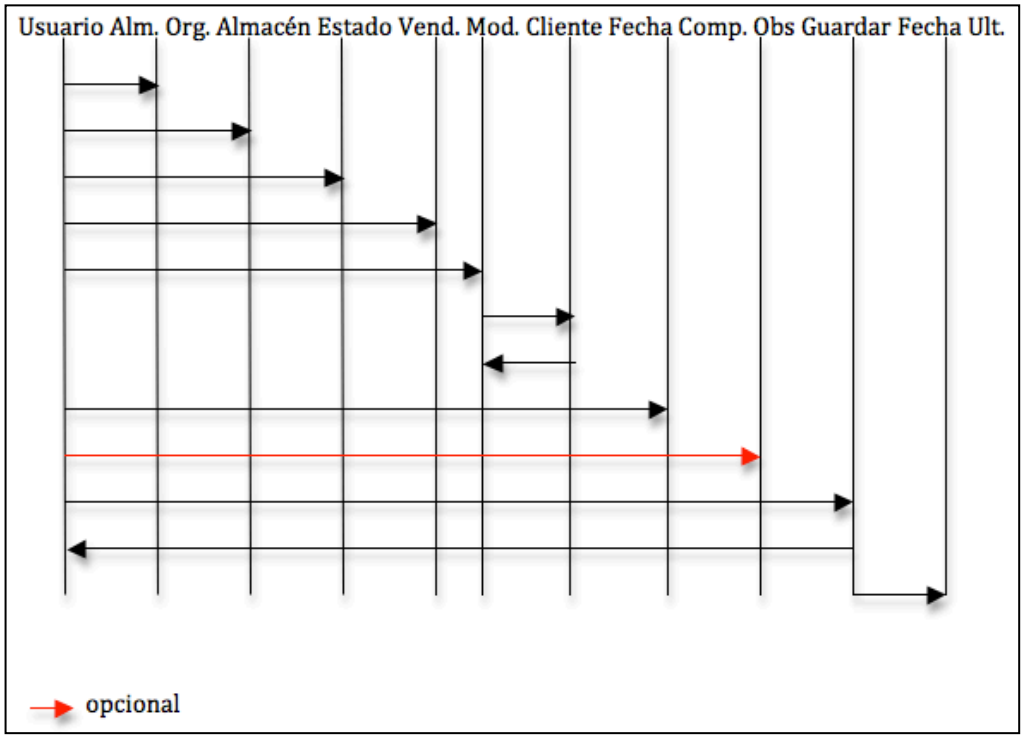


Figura 38 Flujo Generación Reserva

El usuario Introduce los datos de la reserva en el formulario. El campo cliente es un formulario que pertenece a la sección cliente. Las observaciones son opcionales. En este caso, al contrario que los puntos anteriores, es el usuario es el que introduce el campo fecha de compromiso.

4.1.4.2 Creación de departamento familia

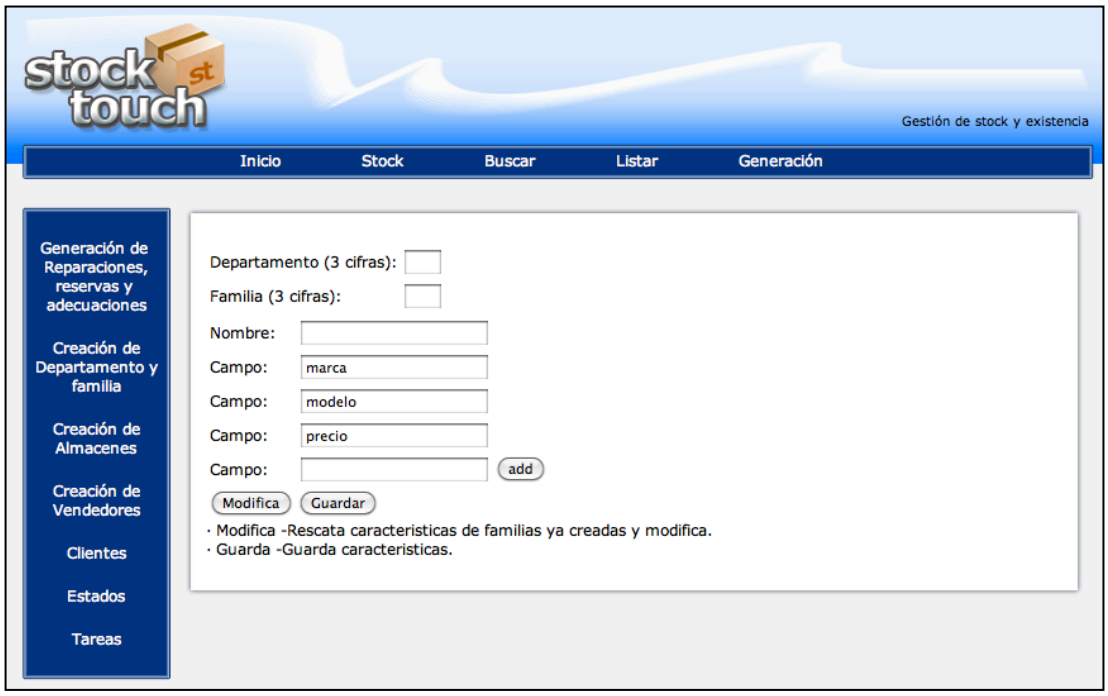


Figura 39 Aplicación Web creación departamento familia

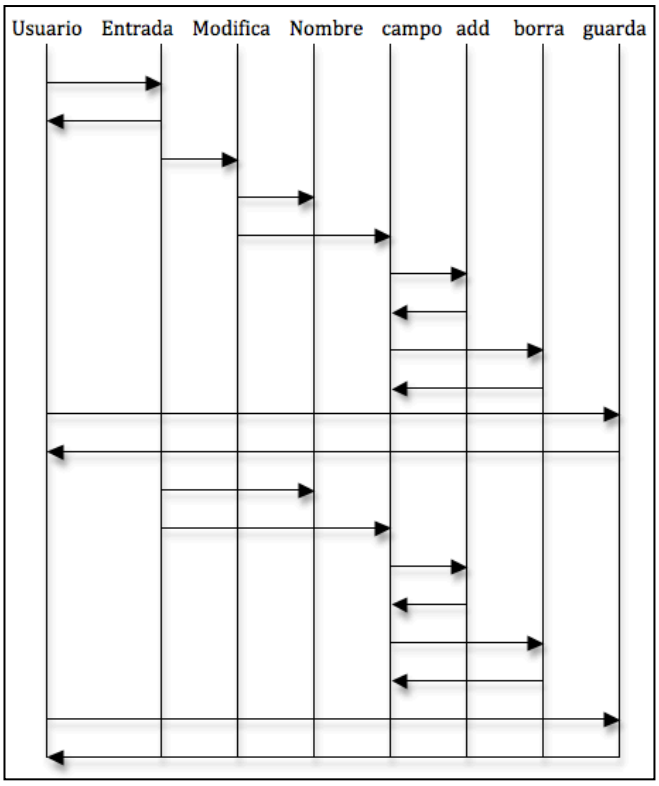


Figura 40 Flujo Creación departamento familia

El usuario puede crear una familia nueva con las características que cree oportunas o modificarla si ya esta creada.

4.1.4.3 Creación de almacenes

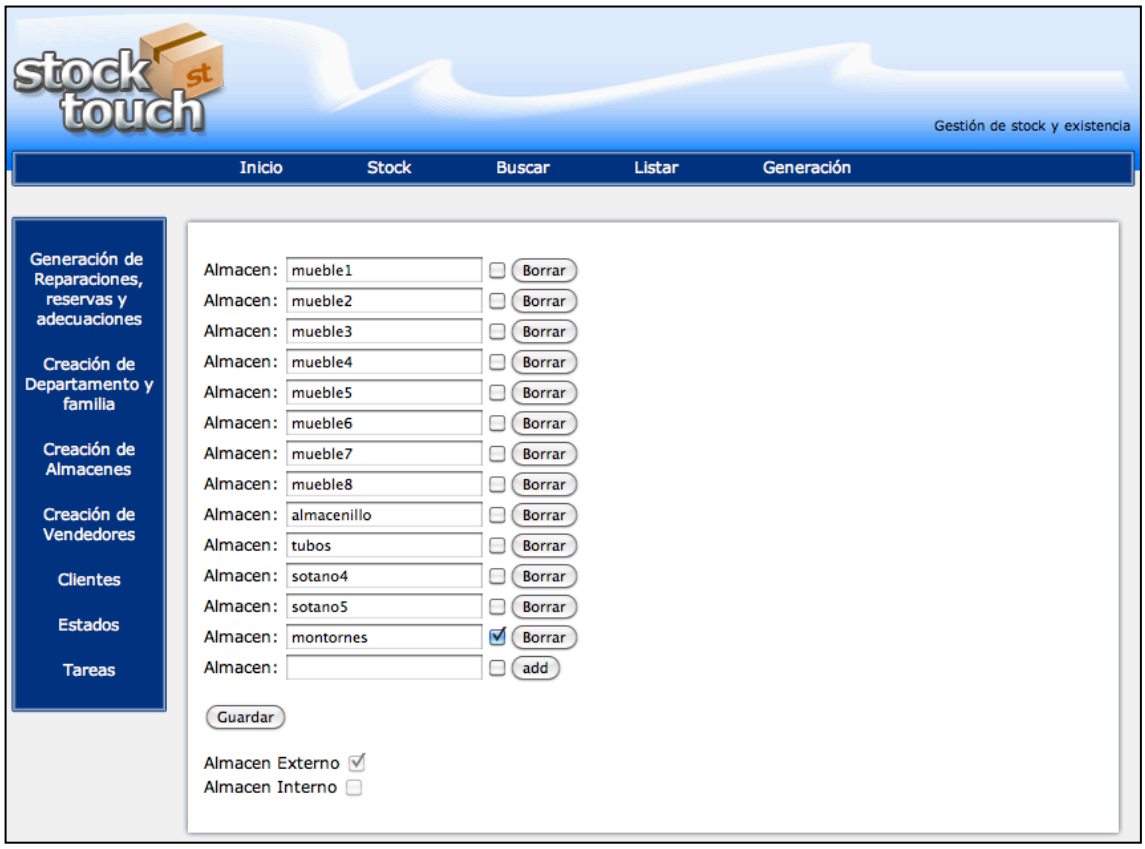


Figura 41 Aplicación Web Creación de almacenes

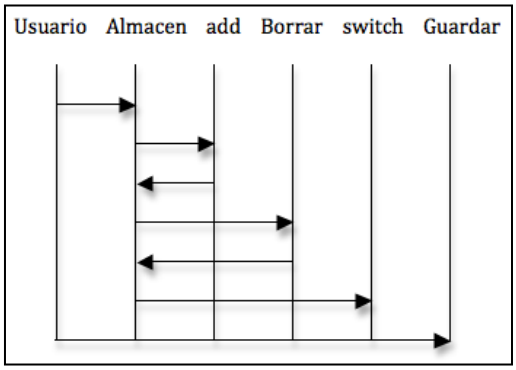


Figura 42 Flujo Creación de almacenes

El usuario puede modificar los almacenes, añadir nuevos y borrarlos. Si un almacén se borra todos sus artículos pasan a almacén libre.

También puede, a través de un switch, determinar si el almacén es interno o externo. Si el almacén es externo los productos que hay en él no están en el centro. De esta manera se determina que los productos de un almacén externo eliminado no pasarán a libre, y aquellos que estén en dicho almacén no podrán venderse, sólo reservarse.

4.1.4.4 Clientes

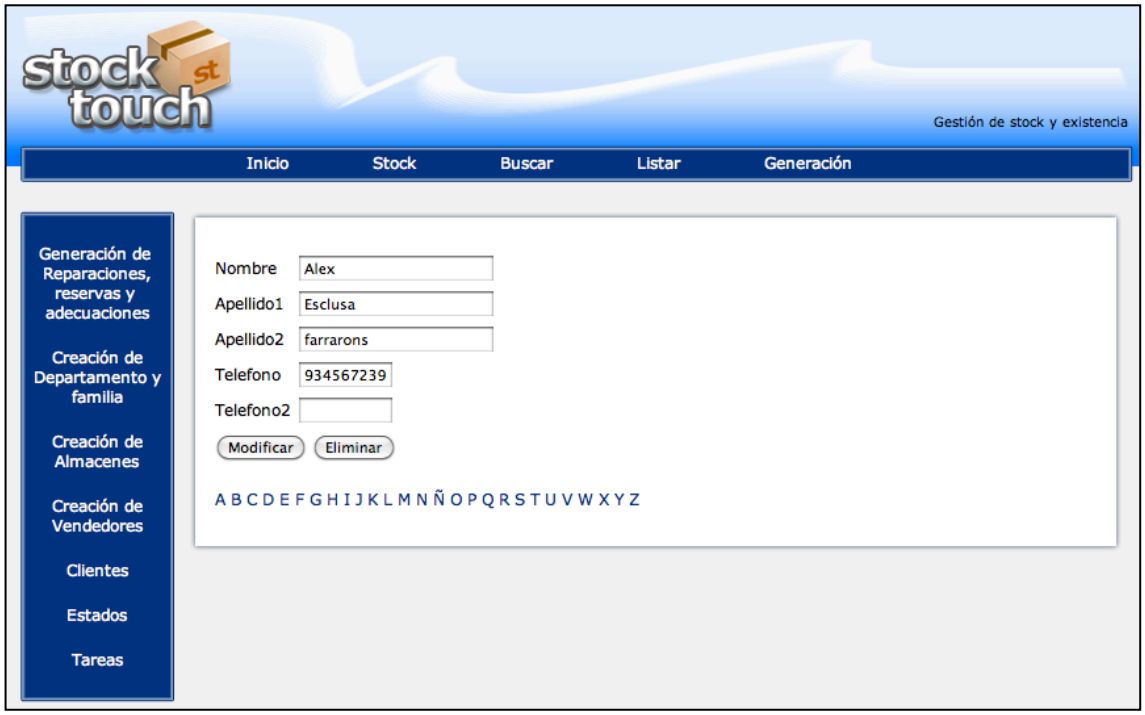


Figura 43 Aplicación Web Clientes

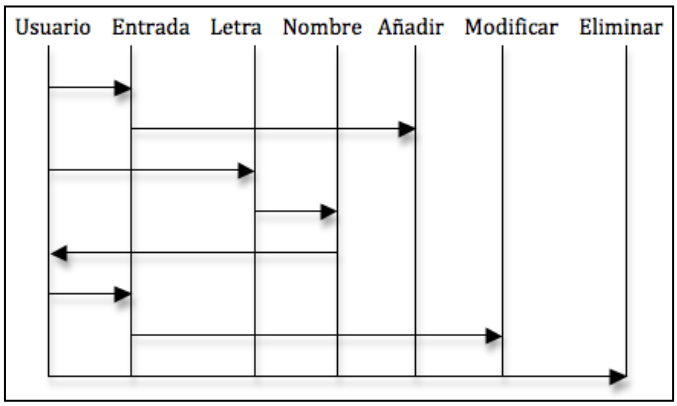


Figura 44 Flujo Clientes

El usuario puede Crear nuevos clientes, así como modificar y eliminar los que ya existen en la base de datos. Si el cliente tiene algún producto asignado, la aplicación no deja eliminarlo.

4.1.4.5 Estados

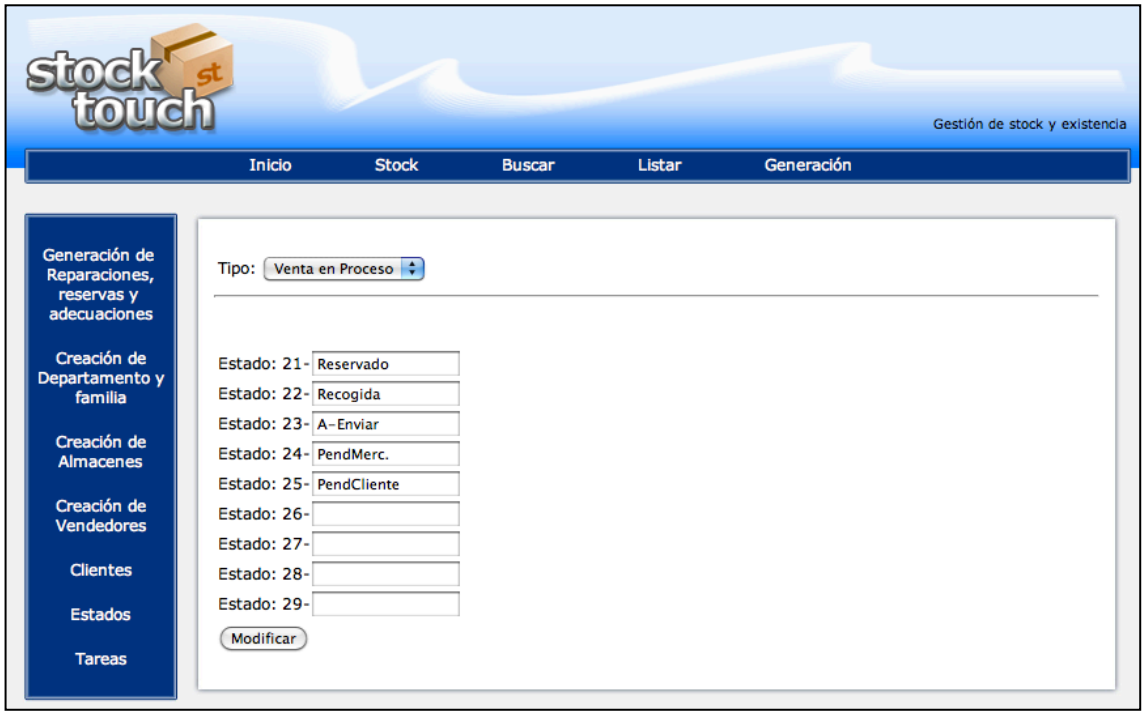


Figura 45 Aplicación Web. Estados

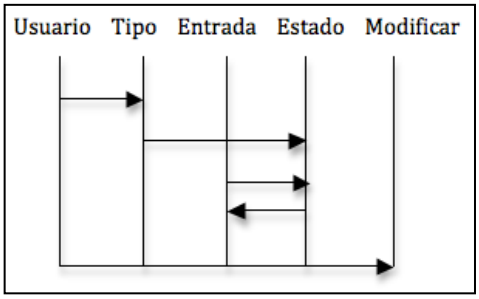


Figura 46 Flujo estados

El usuario puede tanto modificar los estados, como añadir nuevos y eliminarlos. Siempre teniendo en cuenta el rango que se especifica por defecto, es decir del 1 al 9 estados venta. Del 11 al 19 estados destacados. Del 21 al 29 estados venta en proceso. Del 31 al 39 estados en reparación y del 41 al 49 estados en adecuación o devolución. Los estados 0,10,20,30,40 están reservados y 1,11,21,31,41 ya están asignados.

4.1.4.6 Tareas

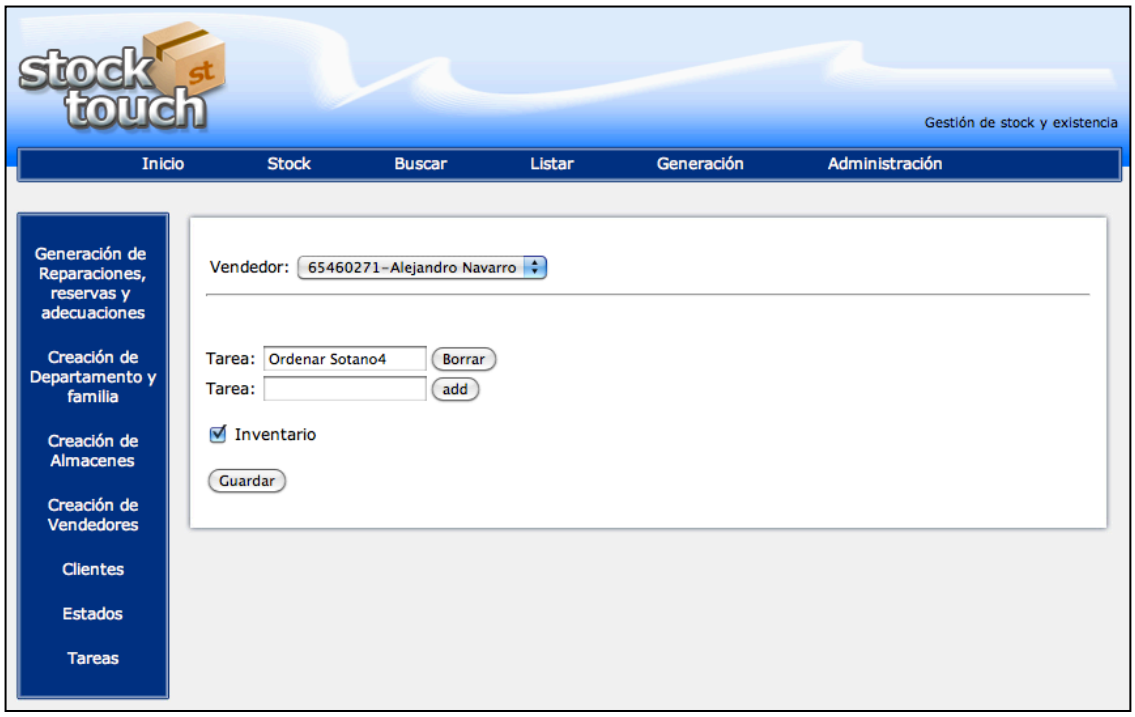


Figura 47 Aplicación Web Tareas

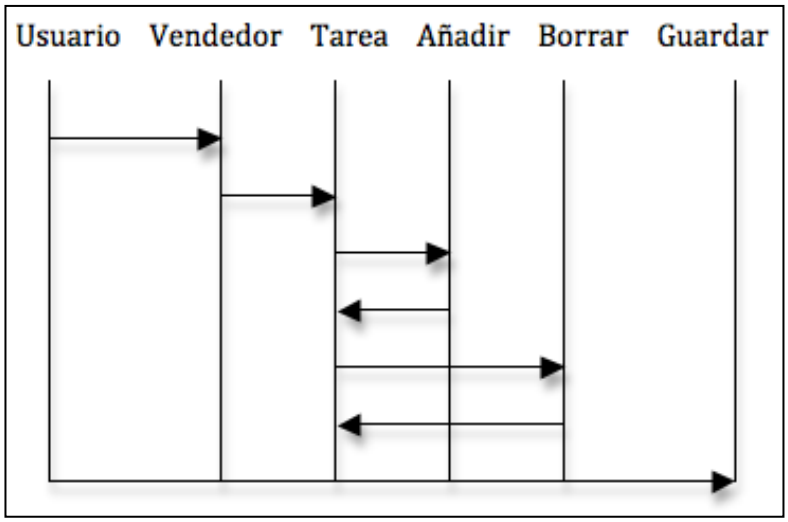


Figura 48 Flujo estados

El usuario puede seleccionar al vendedor, y de esta manera asignarle tareas. Mención especial a la tarea Inventario, que activará un modulo especial (modulo inventario) en la aplicación móvil.

4.1.5 Administración

El apartado administración es el dedicado a generar, cargar o descargar la base de datos. Como administración también tiene el apartado dedicado a cambiar la contraseña del administrador.

Figura 49 Aplicación Web. Nueva base de datos.

La aplicación es capaz de crear una base de datos desde cero, formateándola con las características correspondientes para el buen funcionamiento de la misma. Además permite la carga de una base de datos desde el cliente, tras comprobar el correcto formato de la misma y descargarla en formato de archivo "base.db".

Por otra parte, el apartado administración también permite cambiar el nombre de usuario y contraseña. Éstos serán necesarios, si se utilizan, en la configuración de la aplicación móvil, ya que la conexión desde al servidor estará, desde el mismo momento en que se de alta este usuario y contraseña, bajo la comprobación de estos.

4.2 Interfaz de comunicación

El código de la interfaz es el siguiente:

```
<?php
      include("fun.php");

$patron = "^[[:digit:]]+$";
      echo "<?xml version=\"1.0\" encoding=\"UTF-8\"?>";

if (isset($_REQUEST[consulta]))
{

      $_REQUEST[consulta] = str_replace ("\\", "", "$_REQUEST[consulta]");
      $cons="";
      $i=0;
      $p=select($_REQUEST[consulta]);
      foreach ($p as $val)
      {
        $cons ="$cons<tupla id=\"$i\">";
        if($val>2){
              foreach ($val as $key => $value){
                    $key=preg_replace("/\(\S*\)/", "", "$key");
                    if (!(eregi($patron, $key)))
                          $cons="$cons<$key>$value</$key>";
              }
        }
        $cons="$cons</tupla>";
        $i++;

      }
echo "<Tabla>";
echo $cons;
echo "</Tabla>";
}
?>
```

“fun.php” Incluye los métodos para conectarse la base de datos y lanzar la consulta “select()”.

Se puede observar en el código que primero se adapta la consulta, eliminando el string “\\” ya que no permitía lanzar una consulta adecuadamente, y posteriormente adaptando la respuesta para que genere un xml bien formado.

Esta interfaz como ya he comentado anteriormente devuelve un xml bien formado, el siguiente paso, es implementar en la aplicación móvil el parseador xml que pueda extraer toda la información obtenida a través de la interfaz.

Este parseador es capaz de realizar una conexión con el servidor, generar una consulta bien formada, que la interfaz la puede procesar, parsear el xml resultante e introducir los datos en una estructura para poder utilizarla. En caso de no poder realizar la conexión devuelve una estructura vacía que posteriormente puede ser tratada.

4.3 Parser XML

En el capítulo 3 se explicó en que debería consistir el diseño del parseador xml. Seguidamente se expone en que consiste su implementación. Ya que es un punto clave del proyecto, será la única clase que se explicará en que consisten cada uno de sus métodos al detalle.

La clase xmlparser es una clase delegada del objeto nsxmlparser que deriva de nsObject en la api CocoaTouch [V]. Por lo tanto la clase xmlparser es la encargada de implementar los métodos de este objeto.

Estos métodos son llamados cada vez que el parseador xml se encuentra con un elemento que reconoce, y por lo tanto tienen que ser implementados para que devuelvan el resultado tal y como lo queremos tratar.

Al crear un objeto de la clase nsxmlparser y asignarle la clase delegada xmlparser, se le ha de dar como entrada un xml a parsear, este xml puede ser proveniente de una dirección local (un archivo) o remota (una dirección web).

Una vez realizada la inicialización del objeto de la clase nsxmlparser, éste pasa a parsear el archivo en cuestión.

Este parseo se realiza tras la llmada a estos metodos, cada vez que encuentra un elemento que forma parte de un xml bien formado.

- (void)parser:(NSXMLParser *)parser didStartElement:
  Este método es llamado en primera instancia cada vez que el parseador se encuentra con un elemento. Si el elemento es <tupla> se crea un elemento diccionario. Éste es una lista indexada con el formato clave-valor.

- (void)parser:(NSXMLParser *)parser foundCharacters
  Este método es llamado cada vez que encuentra un carácter dentro de corchetes > … <. Este string se asignará a una propiedad string de la clase.

- (void)parser:(NSXMLParser *)parser didEndElement
  Este método es llamado por ultima instancia cada vez que el parseador se encuentra con un elemento que finaliza un valor xml </>.
  En este caso si el parser se encuentra con un elemento lo añade al diccionario junto con el campo clave que se encuentre entre <>. Si el elemento encontrado es </tupla> el parseador añadirá el diccionario a una tabla propiedad de la clase.
  Si el elemento es </Tabla> el parser finaliza.

De esta manera y siguiendo los métodos en cuestión, el parser devuelve una tabla donde cada elemento es un diccionario. Este diccionario estará compuesto por clave-valor, donde la clave es el campo de la tabla sobre la que se ha hecho la consulta y valor es la respuesta correspondiente a la consulta. Por lo tanto cada elemento de la tabla será una tupla de la consulta realizada.

4.4 Aplicación móvil

La aplicación móvil es una aplicación orientada a servir de herramienta para el usuario vendedor, está ha de poder realizar consultas a la base de datos, y modificaciones en la misma, además ha de poder servir como herramienta para realizar las gestiones sobre los productos.

Para este fin, la aplicación ha de poder consultar todo el "stock", mediante varias herramientas, a través por ejemplo de búsquedas por producto, por referencia o por características. Poder vender productos del "stock", o cambiar su estado, reservándolos, adecuándolos, reparándolos, o haciendo un cambio.

A causa de esto, la aplicación ha de poder tener una parte de gestión de todos los productos que están en el departamento y otra parte en la que poder gestionar los productos que están asignados exclusivamente al usuario en ese momento.

Por este motivo la aplicación no será sólo una herramienta de consulta de "stock", sino que además podrá realizar venta o reserva de productos, gestiones como reparaciones y adecuaciones, llevar un control sobre las tareas a realizar por el vendedor e incluso pasar inventario de los productos que se encuentran en el centro.

4.4.1 Estructura

La aplicación esta dividida en 4 apartados, donde cada uno de ellos esta seccionado en otros tantos. Esto sigue la estructura que a continuación se detalla:

Venta	Gestión	Alta	Mis Cosas
Producto	Reservas	Reserva	Reservas
Referencia	Reparaciones	Reparación	Reparaciones
Búsqueda	Adecuaciones	Adecuación	Adecuaciones
Destacado	Vendedores	Cambio	Tareas
Info	Clientes		Historial
			Inventario

Tabla 4 Estructura Aplicación móvil

Sección "Venta" mediante la cual el vendedor puede realizar una búsqueda del producto deseado y venderlo o reservarlo.

La búsqueda del producto se puede hacer siguiendo varios criterios. Si elige "producto" la búsqueda se hará por marca y modelo, en cambio si el criterio es "referencia" el producto se buscara siguiendo este mismo parámetro. En el subapartado "búsqueda" el usuario puede filtrar productos por características y en "destacado" aparecerán los productos que en ese momento tengan ése estado. Al finalizar la búsqueda el usuario puede escoger vender o reservar. Por el contrario el apartado "info" realiza un búsqueda por marca y modelo, y muestra las características del mismo. Es una vista informativa, y ésta no permite ni vender ni reservar.

Sección "Gestión" mediante la cual el vendedor puede realizar las gestiones de los artículos que no están a la venta.

Esto quiere decir que hay una sección para cada tipo de estado, reservas (venta en proceso), reparaciones y adecuaciones, además el usuario puede hacer una búsqueda de productos en gestión por vendedor o por cliente.

Sección "Alta" donde puede generar un artículo en un estado de venta en proceso, reparación o adecuación. Además existe la posibilidad de hacer un cambio, que no es mas que entregar una mercancía a la venta y generar una adecuación, todo en la misma operación.

Mis cosas, donde el usuario puede realizar sus gestiones, consultar las tareas transmitidas por el administrador y consultar el historial de ventas y gestiones. Además como elemento extra, el usuario puede realizar inventario, sólo si el administrador se lo ha asignado como tarea.

4.4.2 Funcionamiento

Descrito en que consiste cada apartado de la aplicación, paso a detallar cual es su funcionamiento, de la misma manera en que se describió el funcionamiento de la aplicación Web.

Al tener como principal característica el efecto visual, se ha escogido una descripción apoyada por imágenes de lo que puede ver el usuario en cada momento, acompañadas de diagramas donde se pretende explicar el flujo de la aplicación. Estos diagramas detallan que acción se puede realizar en cada momento y en que sentido.

Como elemento principal en cada sección primero se mostrará el flujo inicial, es decir cuando el usuario entra en un apartado (venta, gestión…), seguido de cada acción si escoge un subapartado u otro.

4.4.2.1 Venta

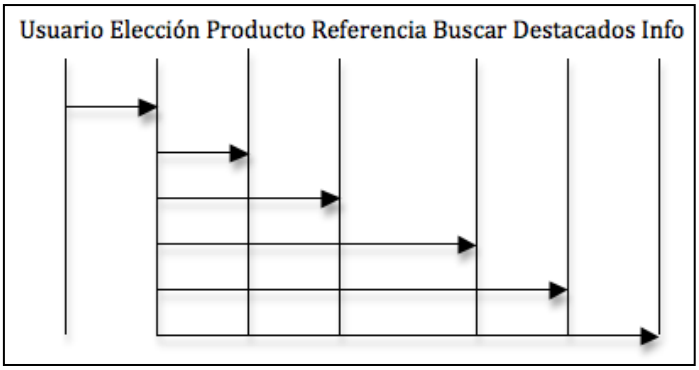


Figura 50 Flujo Venta

Este diagrama indica que una vez dentro del apartado gestión, el usuario puede escoger (elección) diferentes subapartados, producto, referencia, buscar, destacados o info.

En todos los subapartados de la sección productos, (como se detallará en los siguientes diagramas) la primera elección que tiene el usuario para escoger una mercancía, es la familia.

4.4.2.1.1 Venta por producto

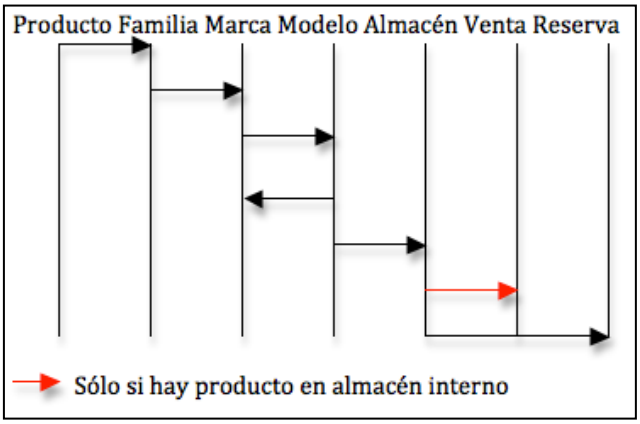
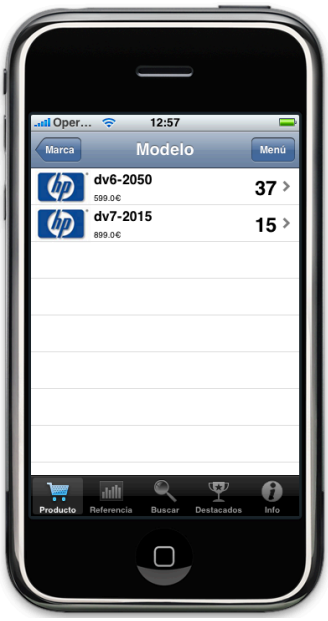


Figura 51 Flujo Venta Producto



En primer lugar el usuario selecciona la familia, la aplicación muestra todas las marcas que hay sobre esa familia, a continuación el usuario escoge una de ellas, si hay mercancía de esta marca la aplicación muestra todos los modelos, por el contrario la aplicación vuelve a mostrar todas las marcas. Tras seleccionar el modelo se muestra en que almacenes se encuentra y la cantidad de cada uno de ellos. Se selecciona uno y seguido se ofrece la posibilidad de vender o reservar.

4.4.2.1.2 Venta por referencia

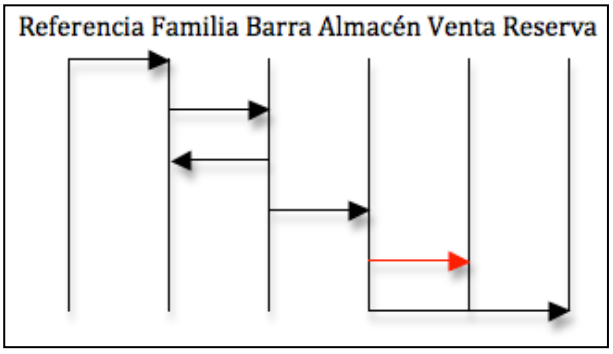


Figura 52 Flujo Venta por Referencia

El apartado referencia funciona de la misma manera que el apartado producto, la diferencia es que en este caso la búsqueda del producto se realiza por referencia, primero seleccionando la familia y posteriormente la barra.

4.4.2.1.3 Venta Buscar

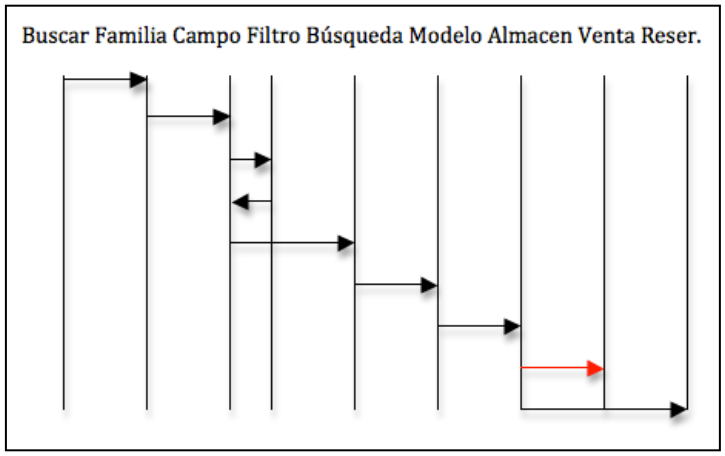


Figura 53 Flujo Búsqueda

El usuario primero selecciona la familia tras la cual desea realizar la búsqueda. Acto seguido se muestra un formulario donde cada campo corresponde a una de las características de esa familia. El usuario puede detallar cada campo con la característica que esta buscando, además también puede realizar una selección por rango de precio, con la posibilidad de seleccionar un precio máximo y uno mínimo.

Tras esto se realiza la búsqueda y aparecen todos los producto de esa familia con las características seleccionadas. Seguidamente se continua como en los apartados anteriores.

4.4.2.1.4 Venta Destacados

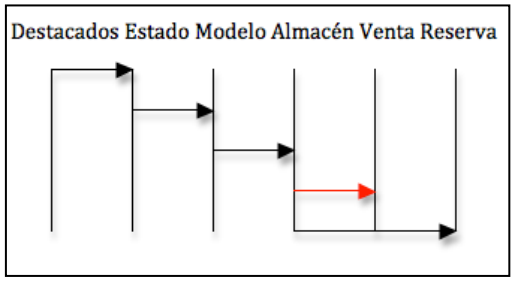


Figura 54 Flujo Venta destacados

Este apartado es como el apartado venta o barra, la selección de la mercancía se hace en este caso por el estado destacado, que el administrador haya asignado a los productos que considere.

4.4.2.1.5 Venta Info

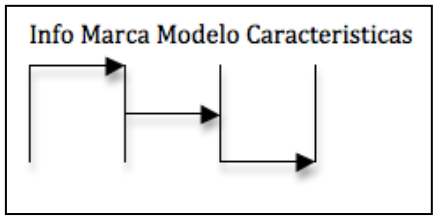


Figura 55 Venta Info

Este apartado es el único de la sección venta que no da la posibilidad de vender o reservar el producto, ya que el destino de éste es simplemente informar al vendedor de las características del producto en cuestión.

4.4.2.1.6 Venta Reserva

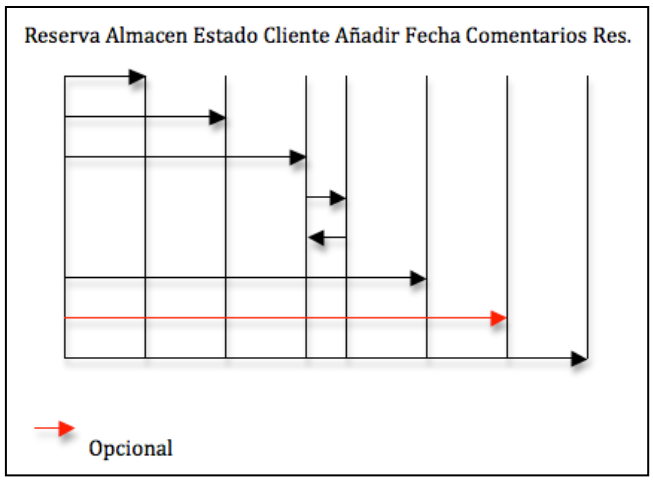
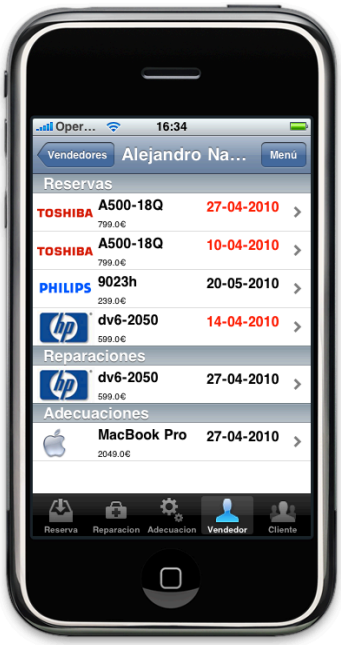


Figura 56 Venta Reserva

Al finalizar la búsqueda del articulo a vender, puede surgir la necesidad de reservarlo, este es el destino de este apartado que se incluye dentro de la sección venta.

Se nos muestra un formulario, que tras rellenar los campos de los que está compuesto, podemos reservar un producto.

4.4.2.2 Gestión



El apartado gestión esta implementado de tal forma que el usuario puede escoger que conjunto de productos que se hayan en algún proceso de reserva, reparación o adecuación desea seleccionar para gestionarlos. Al mismo tiempo el apartado vendedor muestra todos los productos gestionados por un vendedor y el apartado cliente muestra todos los productos asignados a un cliente.

Una vez seleccionado el producto en cuestión la aplicación abre un formulario para seguir gestionando el proceso.

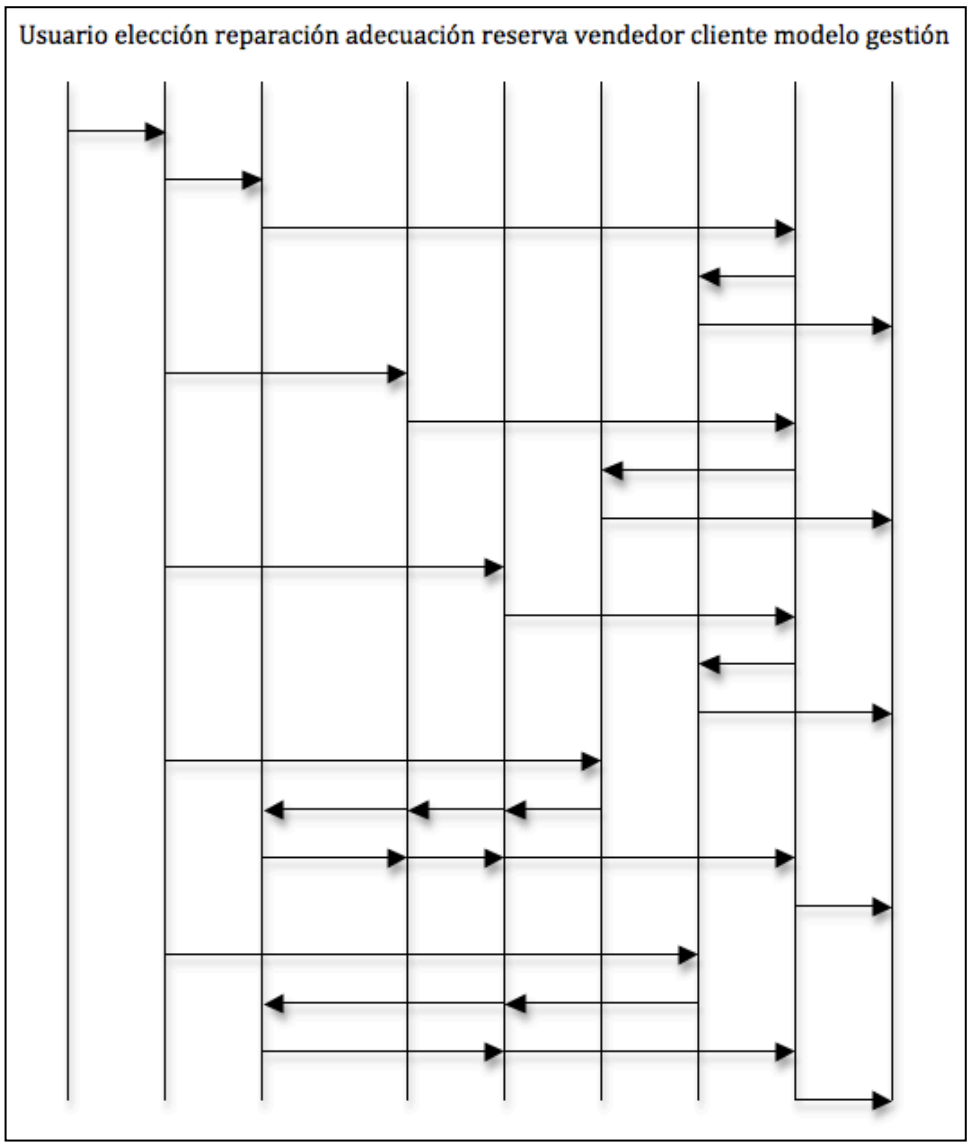


Figura 57 Gestión

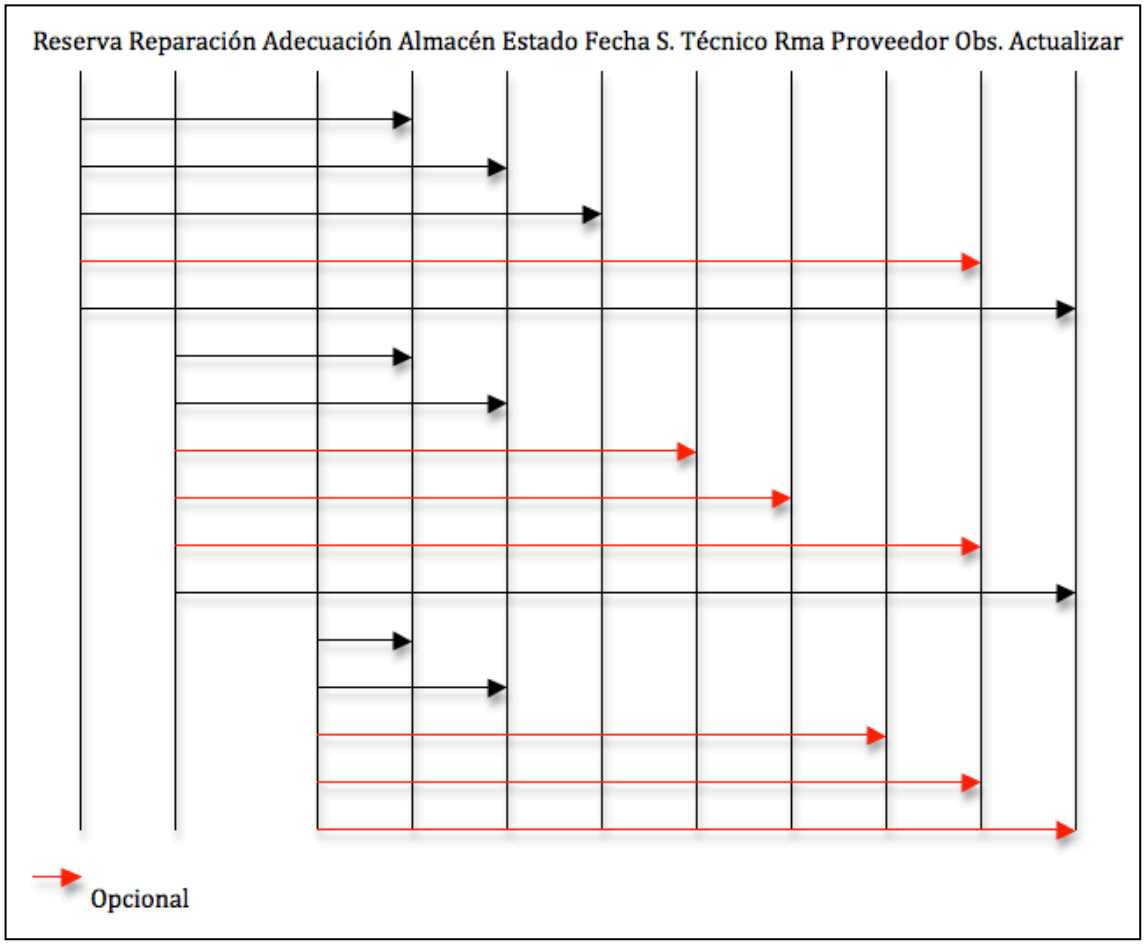


Figura 58 Selección de gestión

Al finalizar la gestión, la aplicación nos permite realizar los cambio en el proceso de cada producto. Hay campos del formulario que son fijos, el campo producto es uno de ellos, la fecha de entrega o el vendedor que lleva la gestión también lo son, en cambio el resto de características son totalmente actualizables, de hecho se entiende por gestión a la actualización de estos mismos. Si el vendedor que realiza algún cambio sobre la gestión del producto es distinto al vendedor propietario de dicha operación, éste queda registrado apareciendo como un campo extra en el formulario del producto. Toda operación puede finalizar cambiando el estado. En el caso de una reserva cumplimentando, de una reparación cambiando a estado reparado y una adecuación a estado devuelto. Los casos reserva y adecuaciones pueden volver a "stock" cambiando a estado venta.

4.4.2.3 Alta

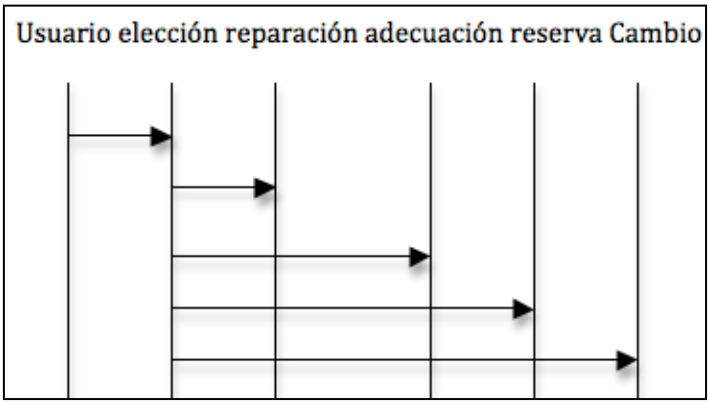


Figura 59 Alta

El apartado Alta nos permite rellenar un formulario para asignar un estado de reserva, reparación o adecuación a un producto. Este formulario tendrá todos los campo que dicha gestión requiera.

4.4.2.3.1 Alta Producto

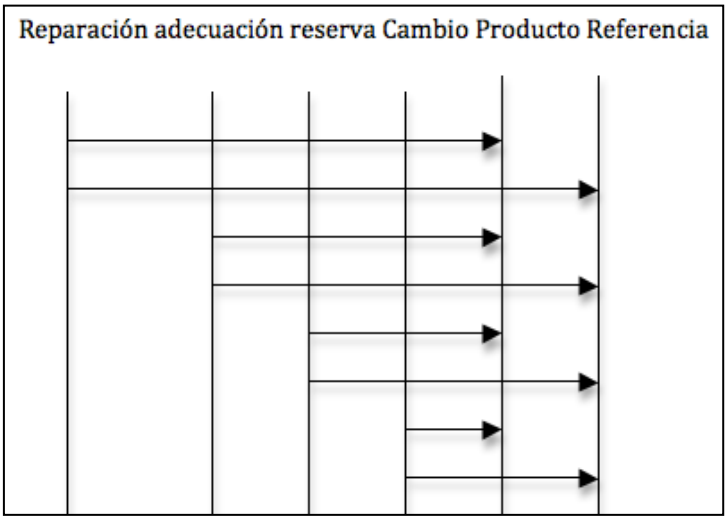


Figura 60 Flujo Alta

El primer campo del formulario de alta que ha de rellenarse es “producto” este campo permite hacer una búsqueda por marca y modelo o por referencia del producto del cual se quiere hacer el alta en cuestión.

4.4.2.3.2 Cliente

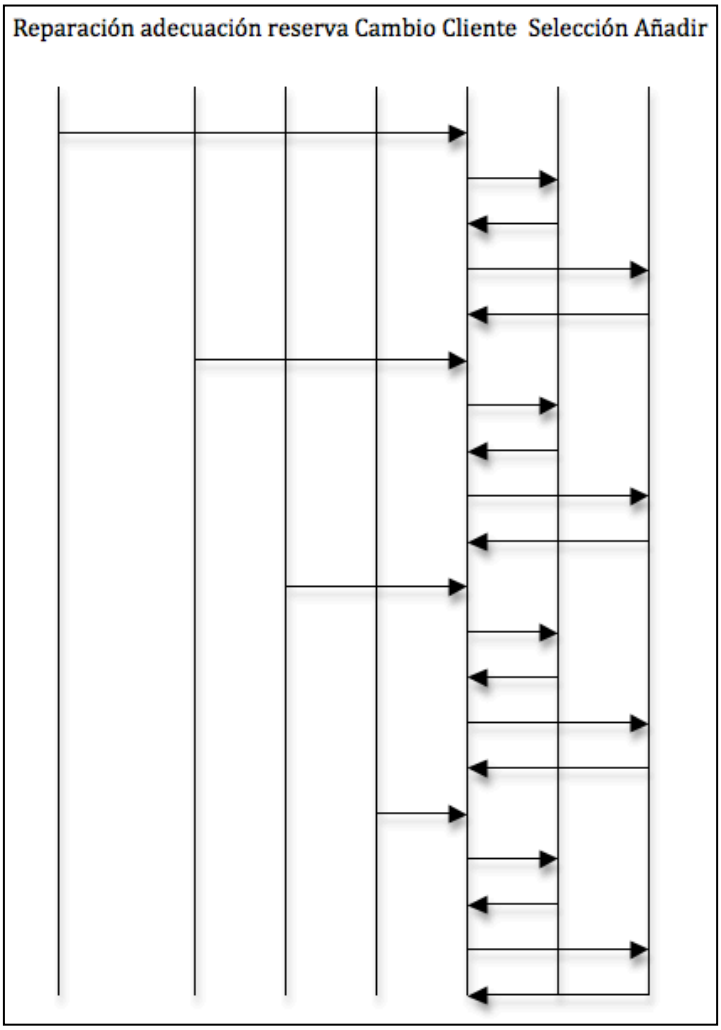
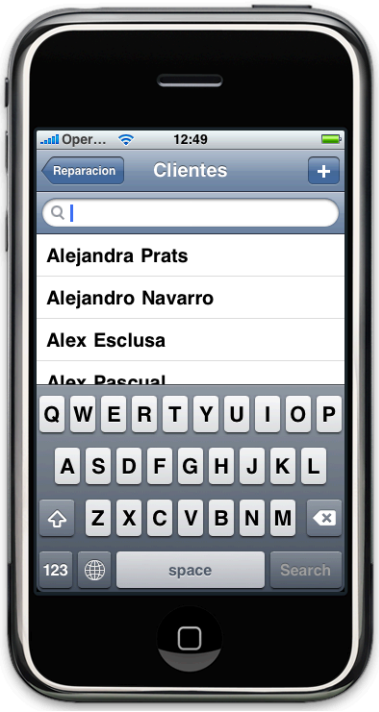
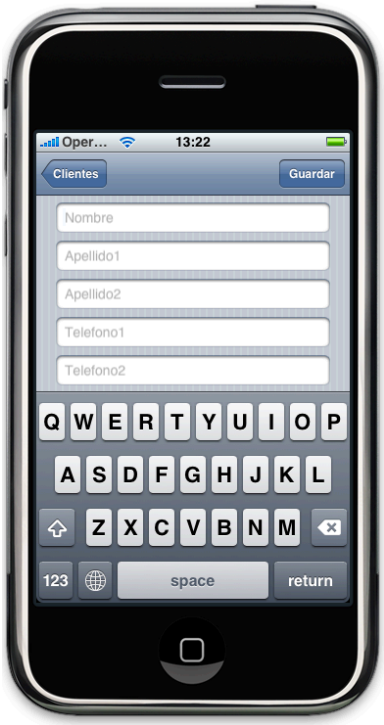


Figura 61 Alta Cliente



Rellenar el campo cliente implica la selección de un cliente que ya esté introducido en la base de datos, y por lo tanto no haga falta introducir sus datos. La aplicación nos permitirá realizar una búsqueda por nombre o apellido.

Por el contrario, si éste no se encuentra, se ofrece la posibilidad de añadirlo. En este caso, la aplicación mostrará un formulario con los campos que un cliente puede tener. El campo nombre, apellido1 y telefono1 serán obligatorios, mientras que los campos apellido2 y telefono2 serán totalmente opcionales.



4.4.2.3.3 Formulario Alta

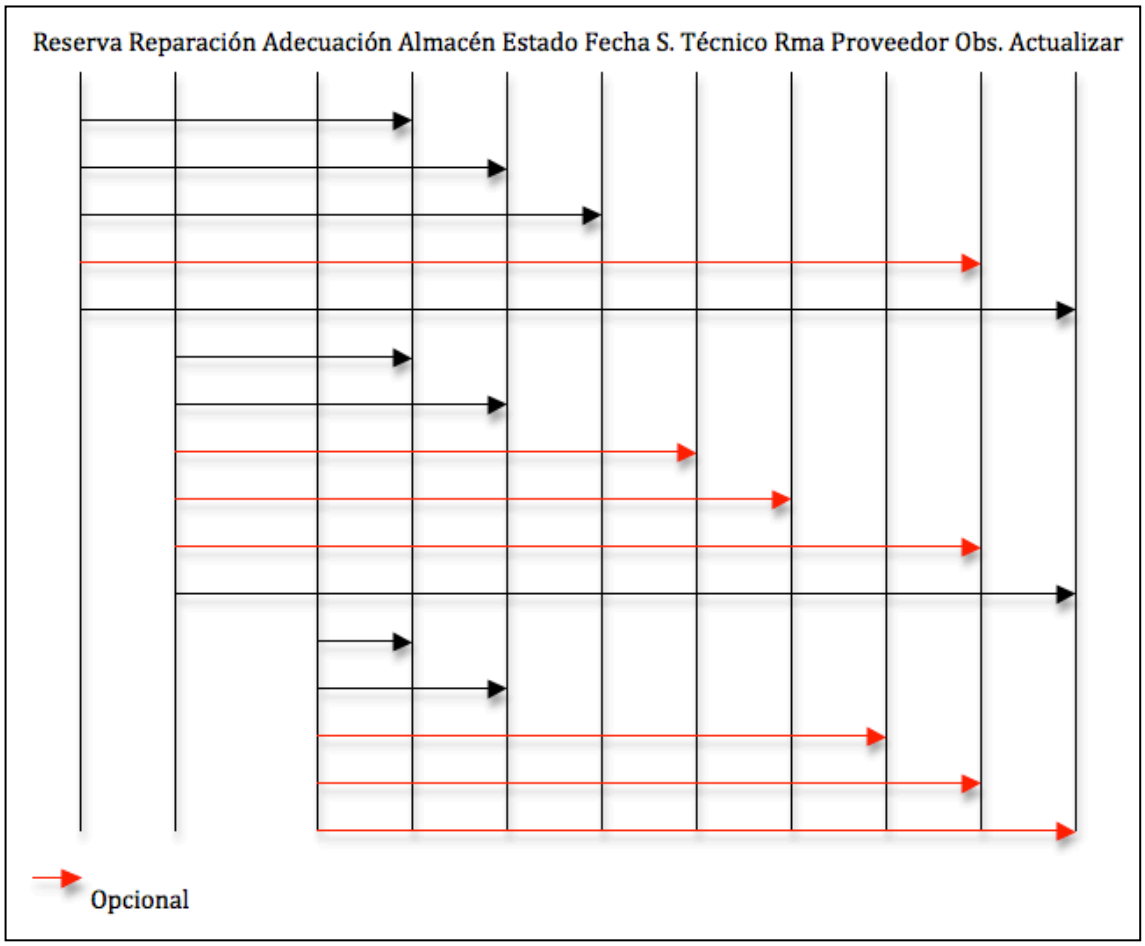


Figura 62 Alta Formulario

Al finalizar la selección de producto y cliente se acaba rellenando los campos que faltan para abrir una alta, ya si es una reparación, una adecuación o una reserva con sus campos correspondientes. El cambio es el mismo caso que el alta de una adecuación, pero al seleccionar el producto la aplicación se encarga de retirar uno de éstos del stock a la venta, para ello también solicita el almacén donde se encuentra esta mercancía.

4.4.2.4 Mis Cosas

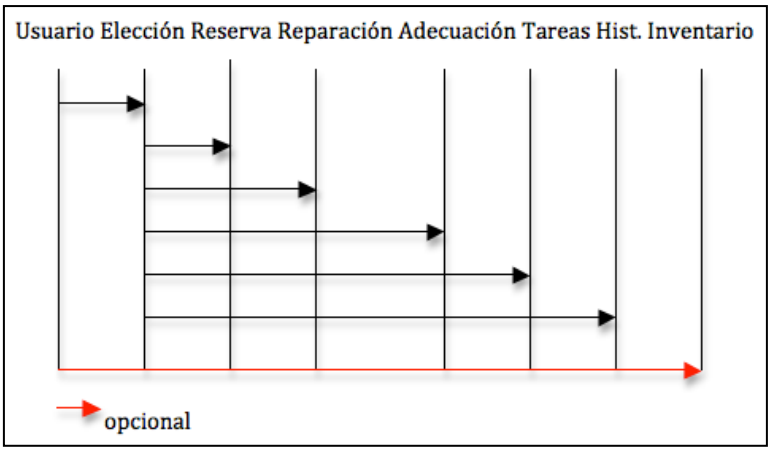
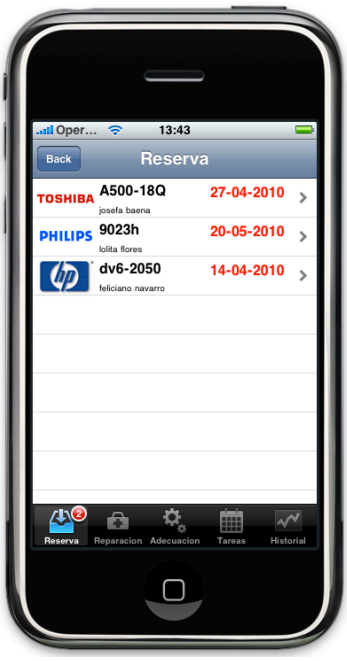


Figura 63 Mis Cosas



Esta sección nos permite llevar una gestión de los productos. El apartado reserva, reparación y adecuación es exactamente igual que en el apartado gestión, la diferencia es que en este caso solo aparecen los productos gestionados por el usuario de la aplicación.

4.4.2.4.1 Tareas

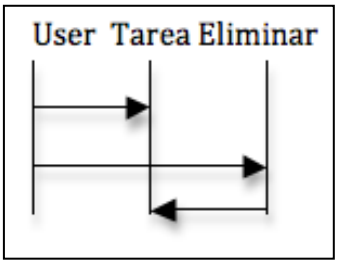
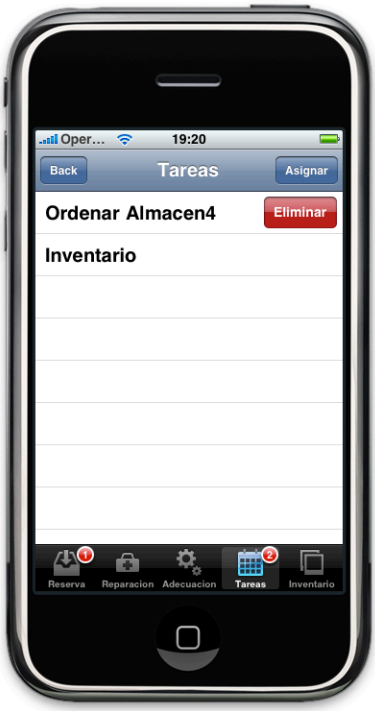


Figura 64 Tareas

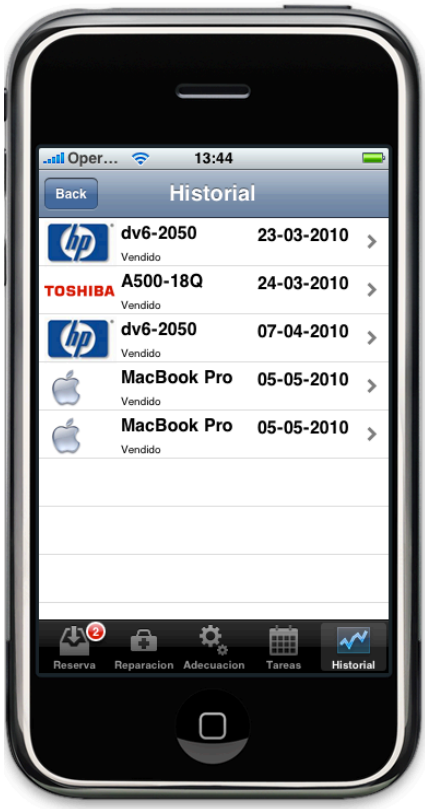


La aplicación lista las tareas asignadas al vendedor.

Una vez que el vendedor cree que ha finalizado la tarea, puede eliminarla desplazando el dedo horizontalmente sobre la tarea escogida. Si lo desea puede asignarle una tarea al administrador, a través de asignar.

Si la tarea Inventario esta asignada, desaparece el apartado historial para aparecer el apartado Inventario. Si se elimina esta tarea de la lista el apartado Inventario desaparece.

4.4.2.4.2 Historial



El apartado historial muestra una lista con todo el producto que se ha gestionado por parte del vendedor ordenado por fecha.

Esto no quiere decir necesariamente que ese vendedor sea quien lleve la gestión. Se puede dar el caso que una tramitación la este gestionando un vendedor, pero que por cualquier motivo, otro vendedor realice una gestión sobre ella. Este tipo de gestiones, las que realiza el propio vendedor, son las que aparecen en historial.

En el ejemplo se puede observar como aparecen los productos vendidos ordenados por fecha.

4.4.2.4.3 Inventario

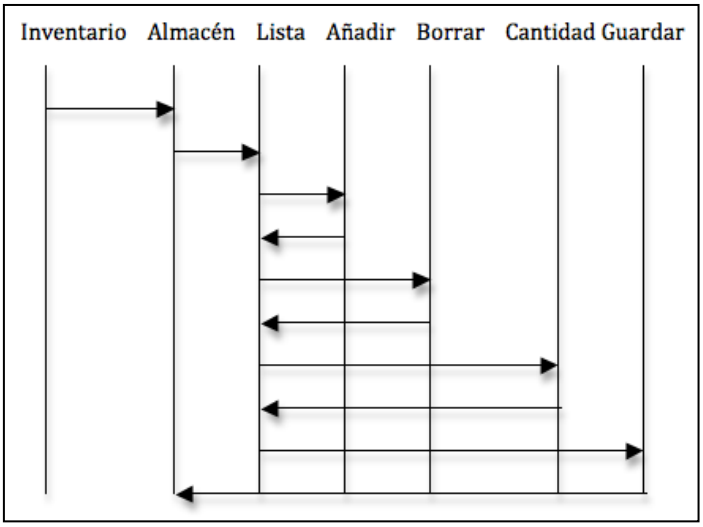
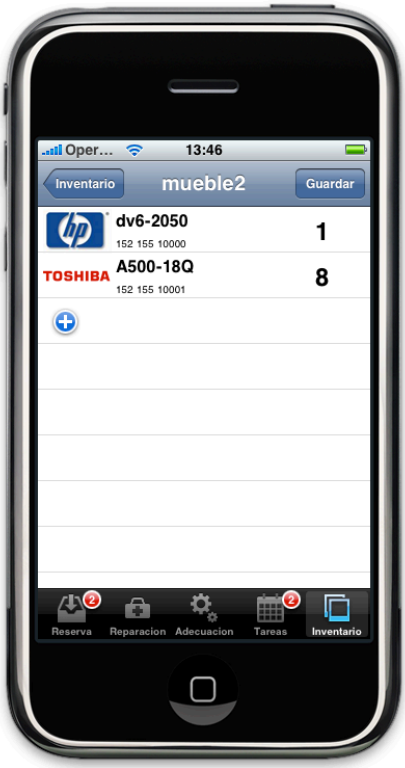


Figura 65 Inventario



Inventario es una tarea que se asigna por parte del administrador. Esto es así, ya que se supone que es una tarea delicada, ya que altera las cantidades de productos que se encuentran en el stock y por lo tanto tiene que realizarse bajo la autorización de éste.

El usuario selecciona un almacén, donde aparecen las cantidades de los productos que hay a la venta, él puede alterar estas cantidades, eliminar productos o añadirlos. Al finalizar guarda los cambios que se almacenan en la base de datos.

Capítulo 5

Pruebas

En este apartado se describen las diferentes pruebas que se han aplicado para asegurar la robustez del proyecto, la integridad de la base de datos, y su coherencia. Por lo tanto existen dos grupos de pruebas. Las que se han ido realizando a medida que iba avanzando el proyecto, y los tests que se han realizado al final, para comprobar como funcionaba en un entorno real.

5.1 Pruebas durante el desarrollo

A medida que ha ido avanzando el desarrollo, se han ido realizando pruebas sobre el mismo. Estas pruebas eran esenciales, ya que corrigian posibles errores y demuestraban que el desarrollo funcionaba tanto por si solo, como actuando con lo ya implementado y probado. Esto tipo de pruebas se pueden agrupar en dos conjuntos, las pruebas unitarias y las de integración.

5.1.1 Pruebas unitarias

Las pruebas realizadas a cada modulo de la aplicación web han permitido comprobar la ejecución correcta de cada uno de ellos. Para esto se ha realizado una batería de pruebas donde se contemplaba, en la medida de lo posible, a que tipo de inputs se podía enfrentar. Se ha intentado probar los casos más extremos, y de esta forma ajustar el código para que el módulo responda correctamente a ellos.

En el caso de la aplicación móvil se ha recurrido a una batería de pruebas por iteración. Esto se aplica buscando todos los casos posibles a los que se puede enfrentar una clase y probándolos uno a uno hasta ajustar el código para que responda correctamente a todos ellos.

La herramienta que se ha utilizado en el caso de la aplicación móvil, es el depurador de Xcode 3.2. Éste muestra que variables se utilizan en cada momento, el valor que se les asigna, y posibles fugas de memoria. Por lo tanto cada vez que se implementaba una clase, se comprobaba mediante el depurador, que la ejecución de la misma era correcta. Es decir que sus argumentos, en ejecución, tenían coherencia con lo implementado, y que sus métodos funcionaban correctamente.

5.1.2 Pruebas de integración

Una vez probados uno a uno todos los módulos, se han realizado las pruebas de integración, éstas consisten en probar la conexión de cada uno de los módulos con los demás.

En el caso de la aplicación Web se ha probado que la interconexión entre los diferentes módulos funcionase. Por lo tanto tenia que haber una coherencia cada vez que se pasaba de un módulo a otro. Por ejemplo, para que la conexión entre el módulo cliente y reservas fuera correcta al realizar una reserva y asignársela a un cliente, éste tenia que aparecer en el módulo correspondiente, si esto no sucedía la integración no había sido correcta.

En la aplicación móvil se ha probado la coherencia en la conexión de todas las clases, como la mayoría de ellas son clases que derivan de UIView, la prueba fehaciente de que no funcionaba era un error en la vista, o simplemente que la aplicación dejaba de funcionar.

Se han realizado pruebas de coherencia, éstas se realizaban haciendo gestiones desde el dispositivo y posteriormente comprobando si efectivamente estos datos aparecían reflejados coherentemente en la base de datos.

5.2 Tests

Los tests se han realizado una vez el proyecto iba cumpliendo hitos y consistían en un conjunto de pruebas alfa y otro de pruebas beta.

5.2.1 Pruebas alfa

Se han realizado básicamente tres pruebas alfa:

1. La primera prueba alfa que se realizó fue una vez cumplido el hito de la implementación de la aplicación Web. Ésta es capaz de funcionar por si misma, por lo tanto se le sometió a un batería de pruebas, como si ya estuviera en pleno funcionamiento.
2. El segundo hito fue la implementación de la interfaz. Una vez realizada se ideo un sencillo programa en iPhone que lanzaba consultas y las imprimía por pantalla. De esta manera se probo el correcto funcionamiento.

3. El tercer punto en realidad fue un conjunto de pruebas, ya que se considero un hito cada vez que se implementaba un apartado de la aplicación. Por lo tanto se realizaron pruebas en 4 ocasiones.

5.2.2 Pruebas beta

Al finalizar la fase de implementación, se ha pasado a realizar las pruebas tipo beta. Éstas consisten en probar la aplicación tal como ha de funcionar en su entorno real.

De hecho, esto fue así en realidad, ya que tuve la posibilidad, gracias a mi trabajo, de poder disponer de un entorno real. El proyecto se probó en el departamento de informática. La aplicación Web y la base de datos se instalaron en un portátil con Windows7 conectado a un router inalámbrico vía wifi. Se utilizaron un iPhone 2G y un Ipod Touch de segunda generación para instalar la aplicación. El proyecto estuvo funcionando durante media jornada laboral.

Se trabajó sobre la base inicial de 5 productos por marca, con 3 marcas por familia, y 4 familias en el departamento. A partir de ahí se fueron introduciendo datos y realizando gestiones con el mismo.

En un futuro esto se ha de poder probar con todos los elementos del departamento, y observar tanto la coherencia, como la estabilidad del programa, y la velocidad en que la aplicación móvil es capaz de acceder a los datos.

Capítulo 6

Resultados y conclusiones

Finalmente se analiza los resultados obtenidos y que conclusiones se extraen de ellos, también se propone un conjunto de mejoras para el futuro.

6.1 Rendimiento

En las pruebas realizadas en la fase de pruebas beta, la aplicación Web accedía a los datos de una manera casi instantánea.

En cambio la aplicación móvil oscilaba desde un tiempo de acceso casi inapreciable a un rango de entre 0 a 6 segundos. Este rango de tiempo se comprobó que era a causa del volumen de datos y la distancia con el punto de acceso. Cuando la aplicación tenia que acceder a listas grandes de mas de 15 elementos, oscilaba entre este rango.

Al desplazarse por el departamento, y llegar al limite del radio de acción del punto de acceso, la aplicación oscila entre los ya comentados 0 a 6 segundos unos más que si lo hacia cerca del centro de conexión.

Aun así las pruebas fueron satisfactorias, ya que la aplicación funcionó correctamente, y hubo en todo momento coherencia con los datos introducidos.

Por la difícil medición de la latencia en los tiempos de acceso, no se han realizado pruebas sobre estos. Simplemente se ha estimado, consultándolo con los futuros usuarios de la aplicación, si el tiempo de acceso era aceptable o no.

En todo momento, y bajo los tests efectuados, se ha llegado a la conclusión que el tiempo de acceso es inapreciable, ya que de 0 a 6 segundos para listas grandes, no es nada comparado con el tiempo que actualmente tardan en buscar cualquier dato.

6.2 Conclusiones

Se puede afirmar que el proyecto ha cumplido con los objetivos inicialmente propuestos. Es una herramienta actualmente muy valorada por el departamento y con altas expectativas.

De todas formas, cambiar los hábitos de trabajo de las personas es una tarea muy complicada, esto es un punto a tener en cuenta, ya que aunque haya grandes expectativas por la implantación de la herramienta en mi ámbito actual de trabajo, también se es consciente que es difícil que se llegue a implantar con total efectividad.

Este tipo de aplicaciones han de utilizarse en su totalidad, es decir que si algún vendedor no se adapta, no cambia su método de trabajo, y continua con el antiguo, es posible que la aplicación no pueda llegar a implantarse. Ésta controla todo el stock, pero también todas las operaciones han de pasar por ella. Si esto no es así existirá incoherencia con los datos. Puede ser, por ejemplo, que hayan productos a la venta que ya se hayan vendido. Para evitar esto, lo ideal seria que el proceso de venta pasase exclusivamente por la aplicación, y esto, en estos momentos es imposible.

Otra vía seria la comunicación con la base de datos del propio centro. Esto haría que si algún vendedor no utilizase la herramienta, ésta se actualizase correctamente de todas formas. Intentar esto sería lo idóneo, pero entonces estaríamos hablando de otro tipo de proyecto, mucho mayor del que tenemos entre manos.

6.3 Trabajo futuro

Existe en estos momentos 2 tipos de trabajo futuro.

Uno más general, y comentado en las conclusiones, que podría ser la conexión con la base de datos del centro comercial en cuestión. Esto haría mucho más fiable la aplicación. Ya que la mayoría de incoherencia de los datos almacenados en ésta suele deberse a errores humanos o simplemente a la no utilización de la misma.

Otro tipo de trabajos futuros, son la ampliación de la aplicación siguiendo la mima línea de trabajo actual, es decir conservando la aplicación tal como está pero añadiéndole funcionalidades nuevas.

Un ejemplo seria la posibilidad de subir las imágenes tanto del producto como de sus marcas desde la aplicación web. Poder visualizar estas imágenes desde el dispositivo.

Otro ejemplo seria la búsqueda de características a traves de Internet, de este modo el administrador no tendría que rellenar los campos de un producto cada vez que lo diera de alta.

Seguro existen muchas más funcionalidades que se puede añadir al proyecto sin cambiar la línea del mismo.

6.4 Historial de cambios

Al inicio de esta memoria se explicó la planificación del proyecto y la propuesta del mismo. Esto obviamente no corresponde en su totalidad con lo realizado al final.

El tiempo de diseño estimado ha sido mucho mayor del que se pretendía, por este motivo se ha tenido que acortar sensiblemente el tiempo de implementación.

A causa de todo esto, al final, se ha intercalado el tiempo que había de tests con el de realización de la memoria. Por este motivo se ha tenido que rediseñar la planificación y trabajar en paralelo en dos campos distintos, el de la realización de tests y el de la memoria a la vez.

Figura 66 Planificación Final

Referencias

[I] Allen, C., & Appelcline, S. (2009). *iPhone in Action Introduction to Web and SDK Development.* Greenwich: Mmanning.

[II] Apple. (2010). Licencia SDK. En *Seccion 3.3.2.* Anexo1.

[III] Apple. (15 de 06 de 2010). *Apple - iTunes - Qué hay en iTunes*. Recuperado el 15 de 06 de 2010, de apple.com/es: http://www.apple.com/es/itunes/whats-on/

[IV] Apple. (2010). *Iphone Developer Program* . Recuperado el 15 de 05 de 2010, de Apple Developer: http://developer.apple.com/programs/iphone/

[V] Apple. (07 de 07 de 2008). *iPhone OS Reference Library*. Recuperado el 15 de 05 de 2010, de developer.apple.com: http://developer.apple.com/iphone/library/navigation/index.html?section=Resource+Types&topic=Getting+Started

[VI] Ars. (s.f.). *Ars Software*. Recuperado el 15 de 05 de 2010, de http://www.ars-software.com/web/web/index.asp

[VII] Hernández, F. L. (2008). *El lenguaje Objective-C para programadores C++ y Java.* Madrid: MacProgramadores.

[VIII] Hernández, F. L. (2009). *Programación Cocoa con Fundation Framework.* Madrid: MacProgramadores.

[IX] IBM. (2010). *IBM*. Recuperado el 15 de 05 de 2010, de IBM -IMS -España: http://www-142.ibm.com/software/products/es/es/imsfamily

[X] Mark, D., & LaMarche, J. (2009). *Beginning iPhone 3 Development Exploring the iPhone SDK.* Berkeley: Apress.

[XI] Sage. (s.f.). *Sage*. Recuperado el 15 de 05 de 2010, de http://www.sage.es/sage/default.aspx

[XII] Sqlite. (s.f.). *Sqlite Home Page*. Recuperado el 15 de 05 de 2010, de http://www.sqlite.org/

[XIII] Wikipedia, c. d. (02 de 02 de 2010). *IMS (IBM)*. Recuperado el 15 de 05 de 2010, de Wikipedia, La enciclopedia libre.: http://es.wikipedia.org/w/index.php?title=IMS_(IBM)&oldid=33576455

Anexos

1. Fragmento sección 3 licencia SDK

3.3 Program Requirements for Applications
Any Application developed using this SDK must comply with these criteria and requirements, as they may be modified by Apple from time to time:
APIs and Functionality:
3.3.1 Applications may only use Documented APIs in the manner prescribed by Apple and must not use or call any private APIs.
3.3.2 An Application may not itself install or launch other executable code by any means, including without limitation through the use of a plug-in architecture, calling other frameworks, other APIs or otherwise. No interpreted code may be downloaded or used in an Application except for code that is interpreted and run by Apple's Documented APIs and built-in interpreter(s).
3.3.3 Without Apple's prior written approval, an Application may not provide, unlock or enable additional features or functionality through distribution mechanisms other than the App Store.
3.3.4 An Application may only read data from or write data to an Application's designated container area on the device, except as otherwise specified by Apple.
User Interface and Data:
3.3.5 Applications must comply with the Human Interface Guidelines and other Documentation provided by Apple.
3.3.6 Any form of user or device data collection, or image, picture or voice capture or recording performed by the Application (collectively "Recordings"), and any form of user data, content or information processing, maintenance, uploading, syncing, or transmission performed by the Application (collectively "Transmissions") must comply with all applicable privacy laws and regulations as well as any Apple program
requirements related to such aspects, including but not limited to any notice or consent requirements. In particular, a reasonably conspicuous audio, visual or other indicator must be displayed to the user as part of the Application to indicate that a Recording is taking place.

Alejandro Navarro Romero

Resumen

Proyecto orientado a ser una herramienta, para poder controlar y gestionar el stock de un departamento de venta de un centro comercial. Basado en dos aplicaciones. La primera, destinada a ser la herramienta mediante la cual el administrador del departamento, puede crear el entorno bajo la cual va a funcionar, controlar el stock y todas las operaciones que se realizan. La segunda, ejecutada en una plataforma móvil (iPhone), será una herramienta de gran utilidad para los vendedores, que les permitirá controlar el stock y realizar gestiones.

Projecte orientat a ser una eina, per a poder controlar i gestionar l'estoc d'un departament de venda d'un centre comercial. Basat en dues aplicacions. La primera, destinada a ser l'eina mitjançant la qual l'administrador del departament, pot crear l'entorn sota la qual va a funcionar, controlar l'estoc i totes les operacions que es realitzen. La segona, executada en una plataforma mòbil (iPhone), serà una eina de gran utilitat per als venedors, que els permetrà controlar l'estoc i realitzar gestions.

Project aimed at being a tool to control and manage the stock of a sales department of a mall. Based on two applications. The first, intended to be the tool by which the department administrator can create the environment under which it will work, stock control and all operations are performed. The second, executed in a mobile (iPhone) will be a useful tool for sellers of the mall, allowing them to control the stock and make arrangements.