

APLICACIÓN WEB PARA CONTROL DE LOS INVENTARIOS Y EL STOCK CON TECNOLOGÍA RFID

TRABAJO FINAL DE GRADO



UNIVERSIDAD POLITÉCNICA DE VALENCIA, CAMPUS ALCOY.
4º GRADO INGENIERÍA INFORMÁTICA (PROMOCIÓN 2011-2015)

ALCOY

JULIO 2016

tag ingenieros
tecnología RFID



APLICACIÓN WEB PARA CONTROL DE LOS INVENTARIOS Y EL STOCK CON TECNOLOGÍA RFID

TRABAJO FINAL DE GRADO PRESENTADO POR:

Alejandro Martínez Giménez

TUTOR DEL TRABAJO:

Javier Silvestre Blanes

UNIVERSIDAD POLITÉCNICA DE VALENCIA, CAMPUS ALCOY.

4º GRADO (PROMOCIÓN 2011-2015)

ALCOY

JULIO 2016

ÍNDICE DE CONTENIDO:

1. INTRODUCCIÓN	6
2. OBJETIVO.....	7
3. CONTEXTO	8
4. HERRAMIENTAS	9
4.1. ECLIPSE.....	9
4.2. OPENXAVA.....	9
4.3.POSTGRESQL	10
5. CONCEPTOS	11
5.1. SHOWROOM.....	11
5.2.TECNOLOGÍA RFID	11
5.2.1.TRANSPONDEDOR(TAG)	11
5.2.2. LECTOR	12
5.3. INVENTARIO	13
5.4. APLICACIÓN WEB.....	13
5.4.1. SERVICIO WEB.....	14
5.4.2 SERVIDOR WEB	14
5.5. SISTEMA GESTOR DE BASES DE DATOS	14
5.6.OPENXAVA.....	15
5.6.1 DESARROLLO DIRIGIDO POR EL MODELO LIGERO	15
5.6.2 COMPONENTES DE NEGOCIO	15
6. TECNOLOGIA RFID.....	17
6.1. FUNCIONAMIENTO DE UN SISTEMA RFID Y COMPONENTES	19
6.1.1. TRANSPONDEDOR	20
6.1.2. LECTORES.....	22
6.1.3. MIDDLEWARE.....	25
6.1.4. SISTEMA DE INFORMACIÓN	26
6.2. TIPOS DE SISTEMAS	27
6.2.1. SISTEMAS DE BAJA FRECUENCIA (LF -135 KHZ)	27
6.2.2. SISTEMAS DE ALTA FRECUENCIA (HF – 13’56 MHZ).....	28
6.2.3. SISTEMA DE ULTRA ALTA FRECUENCIA (UHF -928 KHZ)	29
6.2.4. SISTEMAS EN FRECUENCIA DE MICROONDAS (2’45 GHZ – 5’8 GHZ)	30
6.3. PROBLEMÁTICA DEL SISTEMA RFID	30
6.4. RFID VS. CÓDIGO DE BARRAS	34

6.5. SEGURIDAD, CONFIDENCIALIDAD Y PRIVACIDAD.....	36
6.5.1. ASPECTOS DE SEGURIDAD	37
6.5.2. ASPECTOS DE PRIVACIDAD Y CONFIDENCIALIDAD.....	38
7. APLICACIÓN WEB.....	40
7.1. ESTRUCTURA DE LA APLICACIÓN WEB	40
7.2. BASE DE DATOS	41
7.3. DIAGRAMA DE FLUJO.....	41
7.4. PROCESOS.....	44
7.5. ENTIDADES	45
7.5.1. CLASE ARTÍCULO	45
7.5.2. CLASE ALMACEN	50
7.5.3. CLASE INVENTARIO.....	52
7.5.4. CLASE CUADRE INVENTARIO	54
7.5.5. CLASE STOCK.....	59
7.5.6. CLASE TRAZABILIDAD	61
7.5.7. CLASE UBICACIÓN	65
7.5.8. CLASE VALIDACION UBICACIÓN	66
7.5.9. CLASE DE ÓRDENES DE IMPRESIÓN.....	69
8. CONCLUSIÓN	72
ANEXO.....	73
BIBLIOGRAFÍA.....	104

1. INTRODUCCIÓN

Éste trabajo se originó durante las prácticas universitarias realizadas en la empresa Tag Ingenieros S.L. Tras ver el trabajo diario de los compañeros de Tag Ingenieros con la tecnología de identificación por Radiofrecuencia (RFID), decidí investigar más sobre esta tecnología, ya que aporta grandes beneficios a las empresas de sectores logísticos, viendo reducido el tiempo de realización de inventarios, y gestionar de forma fácil y rápida la información sacada sobre los productos mostrados en una aplicación web.

La tecnología RFID es una de las tecnologías de comunicación que ha experimentado un crecimiento más acelerado y constante en los últimos tiempos.

Las posibilidades que RFID ofrece son: la lectura a distancia de la información contenida en una etiqueta sin necesidad de contacto físico, y la capacidad para realizar múltiples lecturas simultáneamente (y en algunos casos escrituras).

Esto abre la puerta a un amplio conjunto de aplicaciones, en una gran variedad de ámbitos desde, soluciones de trazabilidad y control de inventario en un almacén, como es el caso de este trabajo, hasta la localización, seguimiento e identificación de productos e incluso personas. Durante el siguiente trabajo, se va a detallar que es un sistema RFID, como funciona, que tipos hay, junto con la aplicación web que va a mostrar los datos del sistema, de una forma más extensa.

2. OBJETIVO

La finalidad de este trabajo final de carrera (TFG), es aprender que es la tecnología RFID, el funcionamiento de la misma, en que ámbitos se aplica esta tecnología y aplicarla para optimizar la realización de inventarios en un almacén de exposición (Showroom), y con la ayuda de una aplicación web poder visualizar los datos y realizar una gestión de los productos del Showroom.

Teniendo en cuenta que, en la empresa que se va a instalar el sistema RFID y la aplicación web, es del sector logístico, tiene la necesidad de realizar el seguimiento logístico a los productos, identificar los productos a lo largo de todo el proceso, desde la producción hasta que el artículo es dado de baja o sale de la empresa. Esta situación, es un elemento esencial para poder gestionar los recursos que tiene la empresa.

Es por eso que desde hace años se hace uso de los códigos de barras, para este propósito. Sin embargo, la tecnología de lectura de los códigos de barras presenta ciertas desventajas, como la escasa cantidad de datos que pueden almacenar y la imposibilidad de ser reprogramados. El más importante es el hecho de que es obligatoria una línea visual directa entre el lector y el código, cosa que impide distancias elevadas y la posibilidad de realizar múltiples lecturas simultáneas en un segundo.

Todo esto hace que, pese a haberse convertido en un sistema prácticamente imprescindible en las grandes empresas para el control de sus productos, el sistema de códigos de barras no sea lo mejor para según qué casos.

En este proyecto se pretende resolver esos problemas, donde para mantener el control de la trazabilidad de los productos y el stock que se tiene en el Showroom, la empresa emplea una cantidad de tiempo y de gente elevada que se dedica a la realización de los inventarios, dejando de trabajar en el departamento que le corresponde para poder contabilizar los productos manualmente.

Para ello utilizaremos la tecnología RFID, la cual, permite leer múltiples elementos a la vez, sin que eso aumente significativamente el tiempo de lectura y, además, eliminando la restricción de la línea visual entre lector y etiqueta(chip/tag en RFID), y todo ello quedara reflejado en la aplicación web ITL(Identificación, trazabilidad y logística) Showroom.

3. CONTEXTO

Este proyecto, se realiza como Trabajo Final de Grado (TFG) de la carrera de Grado en Ingeniería Informática de la Universidad Politécnica de Valencia (UPV), extensión de Alcoy. Este TFG ha sido realizado en la empresa Tag Ingenieros S.L.

Tag Ingenieros es una Empresa de Base Tecnológica, especializada en la identificación automática y la trazabilidad de productos y personas. Tag Ingenieros es experto en la integración de la tecnología RFID en Empresas, y dispone de un departamento de I+D donde se desarrollan los sistemas que responden a las necesidades de los clientes, permitiendo ofrecerles soluciones con un rápido retorno de la inversión.

El principal objetivo de la empresa es, aportar soluciones tecnológicas que permitan a las empresas alcanzar las más altas cotas de competitividad en el ámbito logístico, la trazabilidad y de la identificación de personas y productos.

Tag Ingenieros es una empresa joven que aúna la experiencia de más de 20 años en el sector con la aplicación de las últimas tecnologías logísticas y de identificación por radiofrecuencia, RFID.

La cuidada metodología de trabajo garantiza soluciones que aportan valor en los puntos críticos de los procesos del cliente y con un rápido retorno de la inversión.

El departamento de I+D trabaja en nuevas soluciones tecnológicas, adelantándose a las necesidades de los clientes. Esto permite adaptar las soluciones a sus necesidades específicas, ofreciéndole soluciones.

Una vez descrito la empresa donde se va a trabajar para realizar el TFG, indicar que el trabajo se origina como una solución solicitada a la empresa Tag Ingenieros S.L. por parte de un cliente que quiere tener un mayor control y rapidez a la hora de realizar inventarios en su almacén, para evitar la cantidad de horas que se dedican a realizar inventarios, y tener localizado el stock que tiene en cada momento de su almacén Showroom donde se encuentran los productos. Para ello, se implantará la tecnología RFID y una aplicación web donde se mostrarán los datos que se encuentran en el sistema.

4. HERRAMIENTAS

En el siguiente apartado se va a nombrar y realizar una descripción de las distintas herramientas utilizadas durante el TFG.

4.1.ECLIPSE

Eclipse es una potente y completa plataforma de desarrollo open source basada en Java. En sí mismo, Eclipse, es un entorno de desarrollo integrado (IDE) en el que contiene herramientas y funciones necesarias para el trabajo a realizar. Contiene un conjunto de servicios, que hacen a eclipse una interfaz atractiva que facilita mucho el trabajo de desarrollo.



En otras palabras, es un entorno de desarrollo integrado, de Código abierto y Multiplataforma.

4.2.OPENJAVA

OpenJava es un marco de trabajo para desarrollo rápido de aplicaciones de gestión con Java. Es rápido para desarrollar y al mismo tiempo es extensible y personalizable, además el código de la aplicación se estructura desde un punto de vista orientado a objetos puro. Por lo tanto, puedes enfrentarte a aplicaciones complejas con él.



La aproximación de OpenJava al desarrollo rápido no es por medio de usar entornos visuales (como Visual Basic o Delphi), o *scripting*, como PHP. Más bien, el enfoque de OpenJava es dirigido por el modelo (model-driven), donde el corazón de la aplicación son clases Java que describen tu problema.

4.3.POSTGRESQL



PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, con su código fuente disponible libremente.

PostgreSQL utiliza un modelo cliente/servidor y usa *multiprocesos* en vez de *multihilos* para garantizar la estabilidad del sistema. Porque en caso de que se produzca un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

5. CONCEPTOS

En este apartado se va a definir los conceptos básicos que posteriormente van a ser utilizados durante el trabajo.

5.1. SHOWROOM

Un Showroom es una sala de exposición, donde se tiene una serie de productos expuestos para que los clientes puedan ver muestras de ellos. En concreto en la sala de exposición que se ha empleado para el trabajo se encuentra una muestra de cada artículo que está disponible a la venta para los clientes, de ese modo se aprovecha mejor el espacio para tener una mayor cantidad de productos que ofrecer a los clientes.

5.2. TECNOLOGÍA RFID

La identificación por radiofrecuencia es una tecnología básicamente (aunque no sólo) de captura e identificación automática de información contenida en etiquetas (tags o transpondedores). Cuando estos transpondedores entran en el área de cobertura de un lector RFID, éste envía una señal para que la etiqueta le transmita la información almacenada en su memoria. Una de las claves de esta tecnología es que la recuperación de la información contenida en la etiqueta se realiza vía radiofrecuencia y sin necesidad de que exista contacto físico o visual (línea de vista) entre el dispositivo lector y las etiquetas, aunque en muchos casos se exige una cierta proximidad de esos elementos.

RFID puede proporcionar ventajas estratégicas en muy diversas áreas de negocio, ofreciendo seguimiento preciso en tiempo real de la cadena de suministro de bienes o materias primas, y en general, la posibilidad de monitorización en tiempo real de los activos de una empresa.

5.2.1. TRANSPONDEDOR(TAG)

El transpondedor es el dispositivo que va embebido en una etiqueta y contiene la información asociada al objeto al que acompaña, transmitiéndola cuando el lector la solicita.

Está compuesto principalmente por un microchip y una antena. Adicionalmente puede incorporar una batería para alimentar sus transmisiones o incluso algunas etiquetas más sofisticadas pueden incluir una circuitería extra con funciones adicionales

de entrada/salida, tales como registros de tiempo u otros estados físicos que pueden ser monitorizados mediante sensores apropiados (de temperatura, humedad, etc.).

El microchip incluye:

- Una circuitería analógica que se encarga de realizar la transferencia de datos y de proporcionar la alimentación.
- Una circuitería digital que incluye:
 - La lógica de control.
 - La lógica de seguridad.
 - La lógica interna o microprocesador.
- Una memoria para almacenar los datos. Esta memoria suele contener:
 - Una ROM (Read Only Memory) o memoria de sólo lectura, para alojar los datos de seguridad y las instrucciones de funcionamiento del sistema.
 - Una RAM (Random Access Memory) o memoria de acceso aleatorio, utilizada para facilitar el almacenamiento temporal de datos durante el proceso de interrogación y respuesta.
 - Una memoria de programación no volátil. Se utiliza para asegurar que los datos están almacenados aunque el dispositivo esté inactivo. Típicamente suele tratarse de una EEPROM (Electrically Erasable Programmable ROM).

Este tipo de memorias posee un consumo elevado, un tiempo de vida (número de ciclos de escritura) limitado (de entre 10.000 y 100.000) y un tiempo de escritura de entre 5 y 10 ms. Como alternativa aparece la FRAM (Ferromagnetic RAM) cuyo consumo es 100 veces menor que una EEPROM y su tiempo de escritura también es menor, de aproximadamente 0,1 μ s.

5.2.2. LECTOR

Un lector o interrogador es el dispositivo que activa las etiquetas proporcionando energía a las mismas, lee los datos que le llegan de vuelta y los envía al sistema de información. Asimismo, también gestiona la secuencia de comunicaciones con el lector.

Con el fin de cumplir tales funciones, está equipado con un módulo de radiofrecuencia (transmisor y receptor), una unidad de control y una antena. Además, el lector incorpora un interfaz a un PC, *host* o controlador, a través de un enlace local o

remoto: RS232, RS485, Ethernet, WLAN (RF, WiFi, Bluetooth, etc.), que permite enviar los datos del transpondedor al sistema de información.

El lector puede actuar de tres modos:

- Interrogando su zona de cobertura continuamente, si se espera la presencia de múltiples etiquetas pasando de forma continua.
- Interrogando periódicamente, para detectar nuevas presencias de etiquetas.
- Interrogando de forma puntual, por ejemplo cuando un sensor detecte la presencia de una nueva etiqueta.

5.3. INVENTARIO

El inventario es un registro documental de los bienes y demás objetos pertenecientes a una persona física, a una comunidad y que se encuentra realizado a partir de mucha precisión y prolijidad en la plasmación de los datos. También y como consecuencia de la situación recién mencionada, se llama inventario a la comprobación y recuento, tanto cualitativo como cuantitativo de las existencias físicas con las teóricas que fueron oportunamente documentadas.

Un inventario es el proceso que se utiliza por las empresas para controlar el stock que se tiene en su tienda o almacén.

5.4. APLICACIÓN WEB

Se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un Servidor Web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación (Software) que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador.

Las aplicaciones web son populares debido a lo práctico del navegador web como Cliente ligero, a la independencia del Sistema operativo, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales.

Es importante mencionar que una página web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el

usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo.

5.4.1. SERVICIO WEB

Servicios Web son el conjunto de aplicaciones o tecnologías con capacidad para interoperar en la Web. Estas tecnologías intercambian datos entre ellas con el fin de ofrecer unos servicios.

La World Wide Web no es sólo un espacio de información, también es un espacio de interacción. Utilizando la Web como plataforma, los usuarios, de forma remota, pueden solicitar un servicio que algún proveedor ofrezca en la red. Pero para que esta interacción funcione, deben existir unos mecanismos de comunicación estándares entre diferentes aplicaciones. Estos mecanismos deben poder interactuar entre sí para presentar la información de forma dinámica al usuario. Se precisa, pues, una arquitectura de referencia estándar que haga posible la interoperabilidad y extensibilidad entre las distintas aplicaciones y que permita su combinación para realizar operaciones complejas.

5.4.2 SERVIDOR WEB

Para la realización del proyecto se ha optado, por su sencillez y extenso uso, a Apache Tomcat como servidor Web. Funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP) de Sun Microsystems. Tomcat está escrito en Java y por lo tanto puede funcionar en cualquier Sistema Operativo que disponga de una máquina virtual de Java. Se ha utilizado la última versión de Tomcat, la 7x. Dicho servidor es mantenido en buena parte por la Apache Software Foundation y es de uso gratuito.

5.5. SISTEMA GESTOR DE BASES DE DATOS

Los Sistemas Gestores de Bases de Datos (SGBD, por sus siglas en inglés), también conocidos como sistemas manejadores de bases de datos o DBMS (DataBase Management System), son un conjunto de programas que manejan todo acceso a la base

de datos, con el objetivo de servir de interfaz entre ésta, el usuario y las aplicaciones utilizadas.

Gracias a este sistema de software específico el usuario puede gestionar la base de datos (almacenar, modificar y acceder a la información contenida en ésta) mediante el uso de distintas herramientas para su análisis, con las que puede realizar consultas y generar informes.

5.6.OPENXAVA

Aunque OpenXava tiene una visión muy pragmática del desarrollo, está basado en un refinamiento de conceptos preexistentes, algunos populares y otros no tanto. El más popular es el Desarrollo Dirigido por el Modelo (*Model-Driven Development*, MDD), que OpenXava usa de una manera ligera. El otro concepto, el Componente de Negocio, es raíz y principio básico de OpenXava, además de ser la alternativa opuesta a MVC.

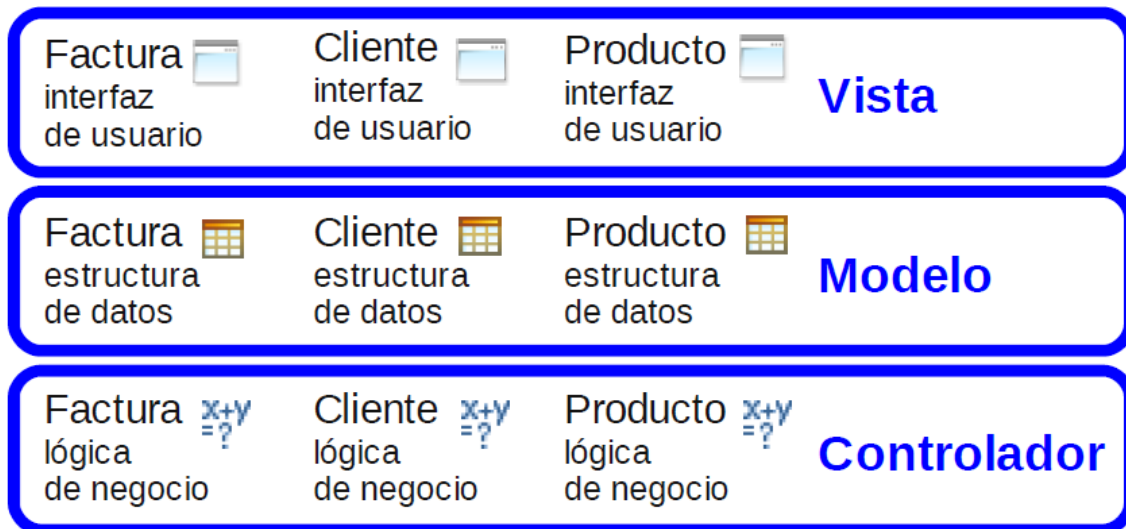
5.6.1 DESARROLLO DIRIGIDO POR EL MODELO LIGERO

Básicamente, MDD establece que únicamente se ha de desarrollar la parte del modelo de una aplicación, y el resto se generará a partir de este modelo.

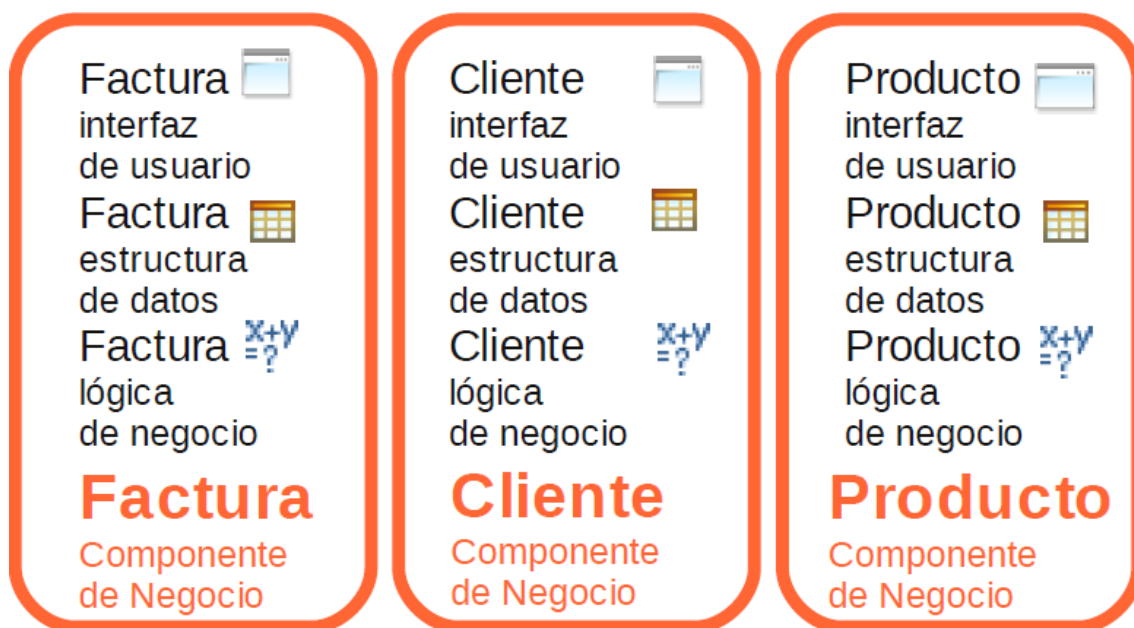
5.6.2 COMPONENTES DE NEGOCIO

Los componentes de negocio son tan solo una forma de organizar el software. La otra forma de organizar software es MVC (Model-View Controller), donde clasificas el código por datos (modelo), interfaz de usuario (vista) y lógica (controlador).

La figura que se adjunta a continuación, muestra como se organizan los artefactos de software en una aplicación MVC. Todos los artefactos para la interfaz de usuario de la aplicación, tales como páginas JSP, JSF, Swing, código JavaFX, etc. están en el mismo lugar, la capa de la vista. Lo mismo ocurre para el modelo y el controlador.



Esto contrasta con una arquitectura basada en componentes de negocio donde los artefactos de software se organizan alrededor de los conceptos de negocio, como se puede ver en la siguiente figura. Aquí, todos los artefactos de software acerca del concepto de Factura, como la interfaz de usuario, Acceso a base de datos, lógica de negocio, etc. están en un mismo lugar.



6. TECNOLOGIA RFID

RFID (Identificación por Radiofrecuencia) es un método de almacenamiento y recuperación remota de datos, basado en el empleo de etiquetas o “tags” en las que reside la información. RFID se basa en un concepto similar al del sistema de código de barras; la principal diferencia entre ambos reside en que el segundo utiliza señales ópticas para transmitir los datos entre la etiqueta y el lector, y RFID, en cambio, emplea señales de radiofrecuencia (en diferentes bandas dependiendo del tipo de sistema).

Todo sistema RFID se compone principalmente de cuatro elementos:

- Una **etiqueta RFID**, también llamada tag o transpondedor (transmisor y receptor). La etiqueta se inserta o adhiere en un objeto, animal o persona, portando información sobre el mismo. En este contexto, la palabra “objeto” se utiliza en su más amplio sentido: puede ser un vehículo, una tarjeta, una llave, un paquete, un producto, una planta, etc.

Consta de un microchip que almacena los datos y una pequeña antena que habilita la comunicación por radiofrecuencia con el lector.

- Un **lector** o interrogador, encargado de transmitir la energía suficiente a la etiqueta y de leer los datos que ésta le envíe. Consta de un módulo de radiofrecuencia (transmisor y receptor), una unidad de control y una antena para interrogar los tags vía radiofrecuencia.

Los lectores están equipados con interfaces estándar de comunicación que permiten enviar los datos recibidos de la etiqueta a un subsistema de procesamiento de datos, como puede ser un ordenador personal o una base de datos.

Algunos lectores llevan integrado un programador que añade a su capacidad de lectura, la habilidad para escribir información en las etiquetas.

A lo largo del presente estudio, cuando hablemos de lector, se considerará que es un dispositivo capaz de leer la etiqueta, independientemente de si puede sólo leer, o leer y escribir.

- Un **ordenador, host o controlador**, que desarrolla la aplicación RFID. Recibe la información de uno o varios lectores y se la comunica al sistema de información. También es capaz de transmitir órdenes al lector.
- Adicionalmente, un **middleware** y en *backend* (*es la parte encargada de procesar los datos recibidos del usuario y devolver una respuesta que el usuario pueda entender*) un **sistema ERP de gestión de sistemas IT** son necesarios para recoger, filtrar y manejar los datos.

A continuación muestro esquemáticamente una clasificación de los distintos sistemas RFID según sus características técnicas de cada uno de los componentes:

- Según su capacidad de programación:
 - *De sólo lectura*: las etiquetas se programan durante su fabricación y no pueden ser reprogramadas.
 - *De una escritura y múltiples lecturas*: las etiquetas permiten una única reprogramación.
 - *De lectura/escritura*: las etiquetas permiten múltiples reprogramaciones.
- Según el modo de alimentación:
 - *Activos*: si las etiquetas requieren de una batería para transmitir la información.
 - *Pasivos*: si las etiquetas no necesitan batería.
- Según el rango de frecuencia de trabajo:
 - *Baja Frecuencia (BF)*: se refiere a rangos de frecuencia inferiores a 135 KHz.
 - *Alta Frecuencia (AF)*: cuando la frecuencia de funcionamiento es de 13,56 MHz.
 - *Ultra Alta Frecuencia (UHF)*: comprende las frecuencias de funcionamiento en las bandas de 433 MHz, 860 MHz, 928 MHz.
 - *Frecuencia de Microondas*: comprende las frecuencias de funcionamiento en las bandas de 2,45 GHz y 5,8 GHz.

- Según el protocolo de comunicación:
 - *Dúplex*: el transpondedor transmite su información en cuanto recibe la señal del lector y mientras dure ésta. A su vez pueden ser:
 - *Half dúplex*, cuando transpondedor y lector transmiten en turnos alternativos.
 - *Full dúplex*, cuando la comunicación es simultánea. En estos casos la transmisión del transpondedor se realiza a una frecuencia distinta que la del lector.
 - *Secuencial*: el campo del lector se apaga a intervalos regulares, momento que aprovecha el transpondedor para enviar su información. Se utiliza con etiquetas activas, ya que el tag no puede aprovechar toda la potencia que le envía el lector y requiere una batería adicional para transmitir, lo cual incrementaría el coste.

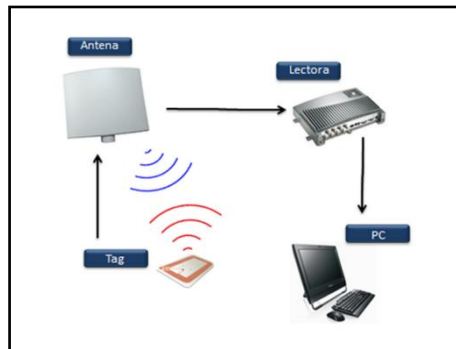
- Según el principio de propagación:
 - *Inductivos*: utilizan el campo magnético creado por la antena del lector para alimentar el tag. Opera en el campo cercano y a frecuencias bajas (BF y AF).
 - *Propagación de ondas electromagnéticas*: utilizan la propagación de la onda electromagnética para alimentar la etiqueta. Opera en el campo lejano y a muy altas frecuencias (UHF y microondas).

6.1. FUNCIONAMIENTO DE UN SISTEMA RFID Y COMPONENTES

Como se ha visto, existe una gran variedad de sistemas RFID con los que se puede abarcar un amplio abanico de aplicaciones. Sin embargo, aunque los aspectos tecnológicos varíen, todos se basan en el mismo principio de funcionamiento:

1. Todos los artículos u objetos que se quieran identificar y controlar su trazabilidad en el sistema, tienen que etiquetarse con la etiqueta RFID.
2. La antena del lector emite un campo de radiofrecuencia que activa las etiquetas.
3. En nuestro caso utilizamos etiquetas pasivas con lo que, las etiquetas que han sido activadas por dicho campo utilizan la energía recibida para ponerse a transmitir los datos que tiene almacenados en su memoria.

4. En el caso de que las etiquetas sean activas no requieren de la energía que emite el lector directamente cuando el lector emite ya están obteniendo la respuesta de las etiquetas activas.
5. El lector va recibiendo los datos de las etiquetas y los va enviando al ordenador de control (servidor) para su procesamiento.



Como se puede ver en la imagen anterior, intervienen dos interfaces de comunicación:

- Lector - Sistema de información. La conexión se realiza a través de un enlace de conexiones estándar que puede ser local de forma cableada o inalámbrica como el RS 232, RS485, USB, Ethernet, WLAN, GPRS, UMTS, etc.
- Lector – etiqueta. Se trata de un enlace de radio con sus propias características de frecuencia y protocolos de comunicación.

A continuación, se van a describir los distintos componentes de un sistema RFID, ya nombrados en el apartado anterior.

6.1.1. TRANSPONDEDOR

Es el dispositivo que va integrado en una etiqueta o tag y contiene la información asociada al objeto al que acompaña, transmitiéndola cuando el lector la activa.

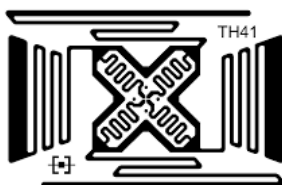
Está compuesto por un microchip y una antena. Además puede incorporar una batería para alimentar sus transmisiones.

El microchip de la etiqueta incluye:

- Una circuitería analógica que realiza la transferencia de datos y proporciona la alimentación.
- Una circuitería digital que incluye la lógica de control, la lógica de seguridad y un microprocesador.
- Una memoria para almacenar los datos, que puede contener una ROM (para datos de seguridad e instrucciones de funcionamiento del sistema), una RAM (para el almacenamiento temporal de datos), una EEPROM (para asegurarse el almacenamiento de los datos aún con el dispositivo inactivo) y registros de datos (buffer).

La antena de las etiquetas puede ser de dos tipos:

- Un elemento inductivo (bobina).
- Un dipolo. Es la utilizada en el trabajo.



Las etiquetas RFID se caracterizan por una serie de parámetros. A continuación se van a enumerar sus características en función de esos parámetros:

- Modo de alimentación
 - Etiquetas activas (se alimentan de una batería) o pasivas (obtienen la potencia necesaria del campo generado por el lector).
- Capacidad y tipo de datos almacenados
 - Permiten almacenar desde un único bit hasta centenares de kilobits.
 - Las etiquetas se usan con el fin de transportar: un identificador (para representar una identidad o una clave de acceso) o ficheros de datos (PDF, para transmitir la información o iniciar acciones).
- Velocidad de lectura de los datos
 - Depende de la frecuencia portadora (cuanta mayor frecuencia, más velocidad)

- La lectura de las etiquetas que posean una alta capacidad de almacenamiento de datos llevara más tiempo.
- En función del número de etiquetas a leer, el tiempo de lectura se multiplica por el número de etiquetas.
- Opciones de programación
 - Dependiendo del tipo de memoria del transpondedor, los datos transportados pueden ser: de solo lectura, de una escritura y múltiples lecturas o de lectura y escritura.
- Costes
 - El tipo y la cantidad de etiquetas que se adquieran son los factores que influyen en el coste.
 - Según el tipo, se consideran los siguientes factores: la complejidad de la lógica del circuito, la forma de la etiqueta, la frecuencia de trabajo de la etiqueta (de baja frecuencia más económicos) y el tipo de etiqueta (lectura/escritura, pasivas o activas).
 - Actualmente, el precio de una etiqueta no sofisticada es de unos 5 céntimos.

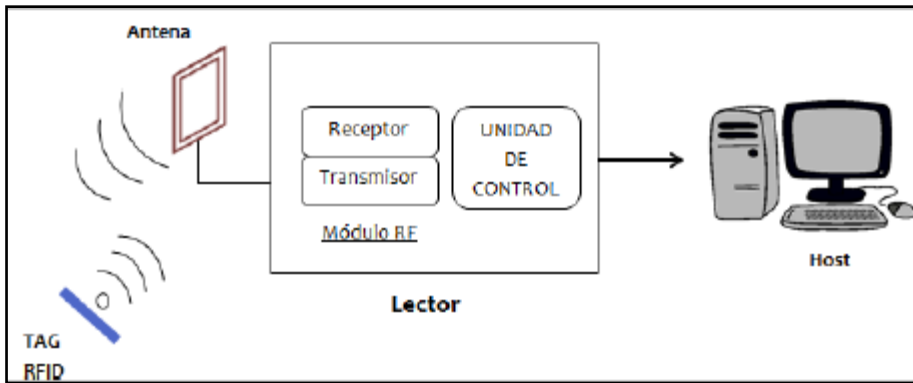
6.1.2. LECTORES

Es el dispositivo que proporciona energía a las etiquetas, lee los datos que le llegan y los envía al sistema de información.

El lector está equipado con un modulo de radiofrecuencia (transmisor y receptor), una unidad de control y una antena.

El lector puede actuar de tres modos:

- Interrogador su zona de cobertura continuamente, si se espera la presencia de muchas etiquetas pasando de forma continúa.
- Interrogando periódicamente, para detectar nuevas presencias de etiquetas.
- Interrogando de forma puntual, al detectar la presencia de una nueva etiqueta.



El **módulo de radiofrecuencia** consta de un transmisor que genera la señal y de un receptor que recibe los datos enviados por las etiquetas. Sus funciones son:

- Generar la señal de radiofrecuencia para activar el transpondedor y proporcionarle energía.
- Modular la transmisión de la señal para enviar los datos al transpondedor.
- Recibir y demodular las señales enviadas por el transpondedor.

La **unidad de control** está constituida por un microprocesador. Sus funciones son:

- Codificar y decodificar los datos procedentes de los transpondedores.
 - Verificar la integridad de los datos y almacenarlos.
 - Gestionar el acceso al medio: activar las etiquetas, inicializar la sesión, autenticar y autorizar la transmisión, detectar y corregir errores, gestionar el proceso de multilectura, cifrar y descifrar los datos, etc.
- Para evitar problemas durante la transmisión, se utilizan procedimientos de comprobación (checksum).
- Comunicarse con el sistema de información transmitiéndole la información obtenida de las etiquetas.

La antena del lector es el elemento que posibilita la comunicación entre el lector y el transpondedor .

El elemento más característico de la antena del lector es la frecuencia de operación a la que trabaja el sistema. Además existen otra serie de parámetros físicos que es necesario considerar (pero no se van a detallar) ya que crean el campo de radiofrecuencia: impedancia, máxima potencia permitida, ganancia, patrón de polarización (X-Y o circular). Éstos a su vez están influenciados por otros parámetros, como la *eficiencia* de la antena o el tipo de *acoplamiento* con la antena de la etiqueta.

En general, las posibilidades que brinda el tipo de antena, su conexión al lector y su ubicación son innumerables.

Cabe destacar que algunos lectores (los que trabajan en campo cercano, como los de mano), incorporan la antena integrada al lector, lo que reduce enormemente estas posibilidades.

La mayor parte de las antenas se engloban en una de las siguientes características:

- Antenas de puerta (uso ortogonal)
- Antenas polarizadas circularmente
- Antenas polarizadas linealmente
- Antenas de varilla
- Antenas omnidireccionales
- Dipolos o multipolos
- Antenas adaptativas o de arrays

El principal aspecto a considerar a la hora de elegir una antena es el área de cobertura que requiere la aplicación, de modo que sea lo suficientemente grande para detectar las etiquetas y lo suficientemente pequeño para evitar lecturas erróneas que confundan al sistema.

La orientación de la antena del lector respecto a la etiqueta también es importante, ya que influye sobre la cantidad de potencia transferida al tag y puede afectar a la lectura. Por ello resulta conveniente buscar el acoplamiento óptimo entre la antena del lector y la antena de la etiqueta.

Una vez que el lector ha leído una etiqueta, la descarta para proceder a interrogar a la siguiente. Existen algoritmos “Protocolo Orden-Respuesta”, en el que el lector ordena a un transpondedor que cese su transmisión cuando ya ha recibido la información. Otro método alternativo (más lento y costoso) se denomina “Sondeo Selectivo”, donde el lector busca las etiquetas que tienen una determinada identificación y las interroga por turnos.

Dependiendo del tipo de transpondedor que tengan que alimentar y las funciones a desarrollar, los lectores se dividen en:

- Lectores fijos. Se posicionan en puntos estratégicos como puertas de acceso, lugares de paso o puntos críticos dentro de una cadena de ensamblaje, para monitorizar las etiquetas de la aplicación.



- Lectores móviles. Dispositivos de mano. Incorporan una pantalla LCD, un teclado y una antena integrada dentro de una unidad portátil (su radio de cobertura es menor).



6.1.3. MIDDLEWARE

Es el software encargado de la comunicación entre el hardware RFID y los sistemas de información existentes en la aplicación.

Se ocupa del encaminamiento de los datos entre los lectores, las etiquetas y los sistemas de información.

Las cuatro funciones principales del middleware RFID son:

1. **Adquisición de datos.** Responsable de la extracción, agrupación y filtrado de los datos procedentes de los lectores RFID en un sistema (sin el middleware los sistemas de información de las empresas se colapsarían).
2. **Encaminamiento de los datos.** Dirige los datos al sistema apropiado dentro de la aplicación.
3. **Gestión de procesos.** Se puede utilizar para programar eventos en función de las reglas de la empresa (envíos no autorizados, bajada de stock, pedido enviado, etc.)
4. **Gestión de dispositivos.** Monitoriza y coordina los lectores RFID.

6.1.4. SISTEMA DE INFORMACIÓN

Las etiquetas RFID son un modo automatizado de proporcionar datos de entrada al sistema cliente. También son capaces de proporcionar una salida del sistema hacia la etiqueta, permitiendo la actualización dinámica de los datos que ésta porta.

El sistema de información se comunica con el lector según el principio maestro-esclavo. Cuando el lector recibe una orden de la aplicación, establece una comunicación con los transponders, en la que el lector ejerce de maestro y éstos de esclavos.

El principal objetivo de la aplicación es gestionar y tratar los datos recibidos por el lector. El sistema de información debe ser lo suficiente robusto para poder manejar las múltiples lecturas, coordinar tiempo y flujos de información, gestionar eventos, soportar las realimentaciones de los usuarios y actualizar e integrar el sistema con otros sistemas de información de la empresa.

Algunos de los sistemas de información de la empresa con los que se puede integrar un sistema RFID son:

- Sistema de planificación de recursos (ERP)
- Sistema de gestión de almacenes (WMS)
- Sistema de albaranes y comprobantes de entrega (POD)
- Sistema de comprobantes de recogida (POC)

6.2. TIPOS DE SISTEMAS

El elemento más determinante a la hora de desplegar un sistema RFID es la frecuencia de utilización. En este apartado se van a analizar las características de los distintos tipos de sistemas según la banda de frecuencia a la que trabajen.

Las características a considerar son:

- Capacidad de almacenamientos de datos
- Cobertura
- Zona de lectura
- Costes
- Áreas de aplicación más adecuadas

6.2.1. SISTEMAS DE BAJA FRECUENCIA (LF -135 KHZ)

Suelen emplear etiquetas RFID pasivas y utilizan acoplamiento inductivo.

Capacidad de almacenamiento de datos:

Baja, alrededor de 64 bits. Con etiquetas activas, hasta 2 kbits.

Cobertura:

El campo magnético decrece muy rápidamente con la distancia y las dimensiones de la antena.

Las antenas que utilizan son pequeñas y complejas; las etiquetas pasivas poseen una cobertura pequeña, que alcanza como mucho 0,5 metros. Las activas pueden superar los 2 metros, aunque depende de más parámetros del sistema.

Zona de lectura:

No funcionan bien con materiales conductores (este problema se incrementa con la frecuencia). Son muy susceptibles a interferencias electromagnéticas industriales de baja frecuencia.

Costes:

Dependen de la forma y necesidades del sistema. Tanto las etiquetas activas como las pasivas son caras (en relación a las que se utilizan en sistemas de frecuencias

superiores). Sin embargo, la construcción del chip y el encapsulado resultan más baratos, así como los lectores e impresoras.

Áreas de aplicación más adecuadas:

Aplicaciones que requieran leer poca cantidad de datos y en pequeñas distancias. Por ejemplo: control de accesos, identificación de animales, gestión de bienes, identificación de vehículos y contenedores, etc

6.2.2. SISTEMAS DE ALTA FRECUENCIA (HF - 13'56 MHZ)

La mayoría utilizan etiquetas RFID pasivas y acoplamiento inductivo, al igual que en baja frecuencia.

Capacidad de almacenamiento de datos:

Desde 512 bits hasta 8kbits, divididos en sectores o bloques que permiten direccionar los datos (etiquetas pasivas).

Cobertura:

Alrededor de un metro.

Zona de lectura:

Buena penetración en materiales y líquidos no conductores. No funciona bien cuando hay materiales metálicos en la zona de lectura, ya que producen reflexiones en la señal.

Mejor inmunidad al ruido por interferencias electromagnéticas que en los sistemas LF. Requiere una correcta orientación de la etiqueta respecto de la antena lectora.

Costes:

Dependen de la forma de la etiqueta y su aplicación. El diseño de la antena del tag es sencillo (menor coste que a LF).

Áreas de aplicación más adecuadas:

Igual que en LF, son aptos para aplicaciones que requieran leer poca cantidad de datos y a pequeñas distancias. Por ejemplo: gestión de maletas en aeropuertos, bibliotecas y servicios de alquiler, seguimiento de paquetes y logística.

6.2.3. SISTEMA DE ULTRA ALTA FRECUENCIA (UHF -928 KHZ)

Estos sistemas basan su funcionamiento en la propagación por ondas electromagnéticas para comunicar los datos y alimentar la etiqueta en el caso de que ésta sea pasiva.

Capacidad de almacenamiento de datos:

Etiquetas activas y pasivas con capacidades desde los 32 bits hasta los 4 kbits, divididos en páginas de 128 bits para permitir direccionar los datos

Cobertura:

Las etiquetas pasivas pueden alcanzar 3 o 4 metros. Trabajando con etiquetas activas y a la menor frecuencia (433 MHz), la cobertura puede alcanzar los 10 metros.

La cobertura está influenciada por las regulaciones de los distintos países correspondientes a la cantidad de potencia permitida, que es menor en Europa que en Estados Unidos, por ejemplo.

Zona de lectura:

Buena penetración en materiales conductores y no conductores, pero presenta dificultades ante la presencia de líquidos.

Mejor inmunidad al ruido por interferencias electromagnéticas que en los sistemas LF. Requiere una correcta orientación de la etiqueta respecto de la antena del lector.

Costes:

Dependen principalmente de la forma. Las tarjetas inteligentes presentan un coste razonable, siendo la opción más barata dentro de la categoría UHF.

Áreas de aplicación más adecuadas:

Aptos para aplicaciones que requieran distancias de transmisión superiores a las bandas anteriores. Por ejemplo: trazabilidad y seguimiento de bienes y artículos y logística de la cadena de suministros.

6.2.4. SISTEMAS EN FRECUENCIA DE MICROONDAS (2'45 GHZ – 5'8 GHZ)

Estos sistemas son los más habituales para las etiquetas activas.

Capacidad de almacenamiento de datos:

Etiquetas activas y pasivas con capacidades desde los 128 bits hasta los 512 kbits, que pueden dividirse en sectores para permitir direccionar los datos.

Cobertura:

Abarca regiones de entre 1 y 2 metros para dispositivos pasivos y hasta 15 metros para los activos.

Zona de lectura:

Buena penetración en materiales no conductores, pero no en líquidos (agua) donde el coeficiente de absorción es importante.

Costes:

Dependen principalmente de la forma y el modo de alimentación (activo o pasivo).

Áreas de aplicación más adecuadas:

Aptos para aplicaciones que requieran alta cobertura y velocidades de transmisión elevadas. Por ejemplo: automatización en la fabricación, peaje de carretera, control de acceso, logística militar, etc...

6.3. PROBLEMÁTICA DEL SISTEMA RFID

A pesar de la fiabilidad de la tecnología RFid, existen diversos factores que influyen de manera negativa en su funcionamiento. Esta influencia puede ir, desde una disminución de las prestaciones óptimas de un sistema RFid, hasta incluso su completa inutilización.

La gran mayoría de estos factores están identificados, y sus consecuencias son previsibles de una manera teórica. Sin embargo, no existen fórmulas que permitan cuantificar los valores exactos de variación de funcionamiento que se producirían ante esos factores.

Por ello, es necesario afrontar una fase inicial de pruebas a la hora de implantar cualquier sistema de RFID para cada aplicación concreta.

Los principales factores que influyen en el funcionamiento de un sistema RFid son los siguientes:

- Potencia de transmisión.
- Tamaño de los elementos radiantes (antenas).
- Patrones de radiación de las antenas.
- Tamaño, tipo y estado de los tags.
- Frecuencia de trabajo.
- Entorno de trabajo.
- Materiales sobre los que se trabaja (Posibles jaulas de Faraday).
- Posición del tag en el momento de la lectura/escritura.
- Proximidad entre distintos campos de acción de distintos lectores/escritores.

Todos estos factores influyen de manera crítica en aspectos como la distancia de lectura, el ángulo de lectura y el número de tags que puede leer simultáneamente (velocidad de lectura).

Así, cuanto mayor sea la potencia de transmisión, normalmente conseguiremos mayor distancia y ángulo de lectura.

El tamaño de las antenas es un factor decisivo en la distancia de lectura. Cuanto mayor sea el elemento radiante (antena), generalmente mayor será la distancia que podremos alcanzar, y, en el caso de la antena del lector, mayor será el campo magnético que se genere. El tamaño de las antenas en los tags también será determinante en la distancia. Tratándose de tags pasivos, la antena podrá inducir una mayor corriente eléctrica cuanto mayor sea su tamaño, por lo que tendrá mayor potencia y podrá contestar a mayor distancia.

El tamaño y tipo de los tags también influyen de una manera crítica en el funcionamiento óptimo de un sistema RFID. Así, normalmente, un tag grande tendrá una antena grande, y por lo tanto alcanzará mayor distancia de lectura. En

cuanto al tipo de tag, aparte del tamaño, los tags difieren unos de otros por el diseño de su antena. Existen muchos diseños de antena, especialmente en los tags de UHF que utilizan una antena de tipo dipolo, con comportamientos distintos.

La frecuencia de trabajo es quizá el factor más determinante en un sistema RFID.

De hecho, la elección de la frecuencia de trabajo a utilizar definirá las características del sistema. Así, el uso de frecuencias bajas limita la distancia y la simultaneidad de lectura, pero aumenta las prestaciones en entornos húmedos. Por el contrario, el empleo de frecuencias altas mejora la distancia de lectura y la velocidad, aunque el sistema resulta más afectado por los entornos húmedos. El ángulo de lectura también se ve afectado por la frecuencia, pues dependiendo de la frecuencia de trabajo se utilizará un tipo de antena concreto, con un determinado patrón de radiación.

El entorno de trabajo también influye en el comportamiento del sistema: entornos húmedos, metálicos o con interferencias electromagnéticas influyen de manera negativa en el comportamiento de un sistema RFID.

Los materiales sobre los que se trabaja son igualmente determinantes del tipo de sistema RFID a utilizar (frecuencia, tipo de tag, etc.). Así, los materiales líquidos y los metálicos influyen de manera negativa en el funcionamiento del sistema, llegando incluso a hacer inviable la implantación del sistema de identificación por radiofrecuencia.

La posición del tag influye tanto en la distancia como en la simultaneidad de lectura, debido a las propiedades físicas de las ondas. Por ejemplo, dependiendo de la posición del tag, el campo magnético podrá inducir mayor o menor cantidad de corriente eléctrica en el tag.

La proximidad entre dos o más lectores tendrá también una clara influencia en el comportamiento de los sistemas RFID, ya que varios lectores podrían interrogar a un mismo tag. En esas circunstancias entrarían en funcionamiento protocolos anticolidión que influirían en la velocidad de lectura. Además, la perturbación electromagnética que podría generar un lector en el entorno de otro influiría negativamente en la distancia de lectura.

Además de aquellos factores puramente materiales que resultan críticos en el comportamiento de un sistema RFID, existen otros que influyen de manera negativa en el funcionamiento de un sistema RFID y que no dependen de los componentes del propio sistema (lectores, antenas y tag), sino que guardan relación con el entorno de trabajo.

Así, entornos húmedos, entornos metálicos o entornos con interferencias electromagnéticas de frecuencias próximas a la frecuencia de trabajo del sistema RFid, influyen de una manera muy negativa en la identificación por radiofrecuencia.

Esta influencia negativa encuentra su explicación en las propiedades físicas del electromagnetismo. Los entornos húmedos pueden provocar un efecto en la onda llegando a imposibilitar la comunicación.

Los entornos metálicos son mucho más críticos que los húmedos, ya que se pueden generar jaulas de Faraday que aíslen completamente el tag del campo generado por el lector. Además, cualquier elemento metálico tiene una polaridad que perturbará el campo electromagnético o podrá producir un apantallamiento del campo hasta el punto de anular la comunicación entre el tag y el lector de un sistema RFID.

La velocidad a la que pasa el tag por el campo de acción del lector es otro de los factores que pueden resultar decisivos a la hora de que un sistema funcione óptimamente o no. Así, como norma general, cuanto mayor sea esa velocidad, menor será la eficiencia del sistema. Por ello, los tag RFID activos tendrán mayor velocidad de respuesta, ya que no tienen que “perder tiempo” en generar la energía necesaria para enviar la respuesta. A pesar de que estas contingencias estarán siempre presentes en cualquier tecnología basada en radiofrecuencia y en electromagnetismo, existen algunas soluciones que permiten atenuar sus efectos negativos. Por ejemplo, ya hemos comentado que las frecuencias más bajas se ven menos perjudicadas por la humedad. En el caso de la velocidad, el uso de tags activos elimina el problema. Y el efecto de los entornos metálicos se minimiza encapsulando los tags con elementos aislantes y situando las antenas de los lectores en lugares donde se eviten los posibles apantallamientos.

Sobre los problemas que ocasiona un sistema RFID, podemos indicar que tienen algunas limitaciones que afectan de una forma clara al rendimiento de esta tecnología. Para empezar el tamaño y chip de la etiqueta puede llegar a afectar a un sistema de tal forma que por no tener una etiqueta adecuada para el sistema que se tiene puede producir una serie de dudas de que el sistema este funcionando de una forma correcta.

Indicar también que, factores como el tiempo, o los materiales donde se peguen las etiquetas afectan de una manera muy importante, tanto es así que puede no leer una etiqueta que tiene delante el lector.

6.4. RFID VS. CÓDIGO DE BARRAS

Durante los últimos años el código de barras ha sido el principal medio de identificación automática de productos en la cadena de abastecimiento. Los códigos de barra han probado ser muy efectivos, no obstante, también tienen limitaciones.

En este apartado se van a analizar una serie de características para comparar la tecnología RFID y el código de barras.

Una migración hacia RFID involucra un conjunto de consideraciones, siendo una de las principales si el código de barras debe ser complementario o si será reemplazado definitivamente.

Método de lectura:

Los lectores ópticos de código de barras necesitan una verificación visual directa. El lector indica cuando obtiene una lectura, asociándola a una etiqueta y un ítem específicos. Este tipo de lecturas son “uno a uno”.

La lectura por RFID no requiere línea de visión para obtener la información de la etiqueta, ya que la señal de radiofrecuencia es capaz de viajar a través de la mayoría de materiales. Un lector RFID es capaz de distinguir e interactuar con una etiqueta individual a pesar de que haya muchas más en el rango de lectura. A pesar de esto, la discriminación de etiquetas no provee la ubicación física absoluta de un ítem, que si ofrece el código de barras. Los tags que no responden por alguna razón, requieren de búsqueda manual.

Velocidad de lectura:

Las etiquetas RFID pueden ser leídas más rápidamente que las etiquetas de código de barras. Esto tiene gran valor en las aplicaciones de recepción y despacho de mercaderías en grandes volúmenes, donde se necesitan contabilizar un gran número de ítems con rapidez.

Por ejemplo cuando se recibe un pale de cajas etiquetadas en un almacén, un lector RFID puede identificar todas las cajas sin tener que desembalar el pale y escanear cada una individualmente.

Durabilidad:

Para mayor protección, las etiquetas RFID se pueden insertar en protectores de plástico u otros materiales. A pesar de que son significativamente más duraderas que las etiquetas de papel de código de barras, ambas dependen del adhesivo que las mantiene intactas y pegadas a un ítem.

El punto débil de una etiqueta RFID es la unión de la antena con el chip. Un corte que le dañase dejaría la etiqueta inservible, mientras que el código de barras solo se degradaría levemente.

Almacenamiento de datos:

El código de barras UPC identifica la clasificación de un ítem genérico; en cambio, el código EPC permite identificar un ítem de forma unívoca a través de una clave asignada.

Los códigos de barras genéricos pueden almacenar hasta 30 caracteres.

Los tags RFID contendrán varios bits de memoria (dependiendo de sus características). Esto permite que un gran número de productos puedan ser rastreados, con datos como la fecha de realización, el tiempo transcurrido, su ubicación en el almacén o la fecha de vencimiento del ítem.

Flexibilidad de información:

Las etiquetas RFID son capaces de realizar operaciones de lectura y escritura, permitiendo la actualización de información en tiempo real de un ítem que se va moviendo a lo largo de la cadena de producción, por ejemplo.

Redundancia de información:

Las etiquetas RFID contienen información que ofrecen únicamente cuando un lector las activa. La integridad del sistema no es lineal (se puede rechazar lo que el lector reciba). Los códigos de barra, por otro lado, tienen un formato de legibilidad de caracteres humanos, lo que permite una recuperación directa en caso de que se falle al leer.

Seguridad:

Algunas etiquetas RFID pueden ir cifradas, para que sólo puedan ser leídas por lectores que tengan acceso. En el caso del código de barras no se usa cifrado y el estándar es bien conocido.

Costo:

RFID requiere inversiones en capital. Los principales costos vienen del equipamiento (impresores, lectores, antenas y etiquetas) y por los servicios profesionales (ingeniería de proyectos, instalación y puesta en marcha, capacitación de los usuarios). Por su parte, el código de barras requiere costes muy bajos.

-tabla . Diferencias entre el código de barras y RFID-

Características	Código de Barras	RFID
Capacidad	Espacio limitado	Mayor cantidad de información
Identificación	Estandarizado	Unívoca por producto
Actualización	Sólo lectura	Lectura/escritura
Lectura	Una cada vez	Simultanea
Tipo de lectura	Solo en superficie	A través de diversos materiales
Flexibilidad	Requiere línea de visión para lectura	No requiere línea de visión para lectura
Precisión	Requiere intervención humana	100% Automático
Durabilidad	Puede estropearse fácilmente	Soporta ambientes agresivos

6.5. SEGURIDAD, CONFIDENCIALIDAD Y PRIVACIDAD

A pesar de los potenciales beneficios que conlleva la implantación de sistemas RFID, existe una creciente corriente en contra de esta tecnología, debido a que cualquier persona con un lector apropiado, puede leer la información que llevan las etiquetas.

Por ello todo sistema RFID debe protegerse, en mayor o menor medida de:

- Lecturas/escrituras indeseadas, con objeto de obtener información o modificar datos de forma fraudulenta.
- La existencia de etiquetas falsas dentro de una zona restringida, que tratan de burlar la seguridad del sistema accediendo a lugares no autorizados o recibiendo determinados servicios sin previo pago.
- Escuchas ilegales con objeto de copiar los datos y falsificar etiquetas.

6.5.1. ASPECTOS DE SEGURIDAD

La forma más simple de ataque a un sistema RFID es, evitar la comunicación entre el lector y la etiqueta. Esto se puede hacer apantallado con metales. Existen otras formas de ataque más sofisticadas, cuyo objetivo son las comunicaciones por radiofrecuencia. Los ataques más importantes se resumen en:

- Spoofing Se suministra información falsa que parece ser válida y el sistema la acepta. Por ejemplo, enviando un código de producto (EPC) falso, cuando el sistema espera uno correcto.
- Inserción Se insertan comandos del sistema donde se esperan datos. Por ejemplo, inserción SQL en una base de datos.
- Replay Se intercepta una señal RFID y se graban los datos. Después se retransmiten al sistema que los acepta como validos.
- Denegación de servicio (DoS). Se colapsa al sistema transmitiéndole más datos de los que puede manejar.
- Man in the Middle (MiM) Se aprovecha de la confianza mutua en el proceso de comunicación, suplantando una de las entidades.
- Modificación de chips Se lee la información de una etiqueta y se reescribe con la información deseada.
- Inutilización de etiquetas Se somete a la etiqueta a un fuerte campo electromagnético para inutilizarla.

Medidas de seguridad

- Para etiquetas:
 - Utilizar etiquetas de solo lectura.
 - No escribir los datos directamente en la etiqueta, sino un único código que enlace la información a una base de datos en el sistema backend.

- Utilizar métodos de autenticación previos para evitar borrados y desactivaciones.
- Cifrar la información.
- Para el lector:
 - Utilizar métodos de autenticación para validar la comunicación entre el lector y la etiqueta, y evitar falsificaciones de identificadores de lector.
 - Uso de cifrado y protocolos seguros a nivel de middleware.
 - Uso de buffers para evitar ataques DoS.
 - Realizar análisis de patrones de eventos para evitar eventos espurios.
 - Uso de extensiones de seguridad para el DNS.

6.5.2. ASPECTOS DE PRIVACIDAD Y CONFIDENCIALIDAD

RFID hace posible la captura de información personal de forma silenciosa y a veces transparente para el usuario. Esto hace que se vea a la tecnología como causante de un mayor impacto en la privacidad.

Aunque es cierto que la tecnología RFID puede atentar contra la privacidad y confidencialidad de las personas, existen soluciones técnicas para controlar las utilidades indeseadas de los sistemas RFID, como son los procedimientos de cifrado y autenticación. El cifrado se utiliza para asegurar que la información solo pueda ser entendida por los usuarios de la aplicación y evitar de ese modo lecturas indeseadas. La autenticación se utiliza para que únicamente personal autorizado pueda acceder a dicha información, tanto para leer como para escribir.

Las principales amenazas a la privacidad y confidencialidad en los sistemas RFID provienen de:

- Lecturas no autorizadas de las etiquetas. Las etiquetas pueden contener información personal, como nombres, fecha de nacimiento, direcciones, etc. Pueden contener también datos en forma de una clave de acceso a una base de datos.
- Seguimiento de las personas, preferencias, gustos. Cuando una persona porta una etiqueta con sus datos y la emplea para pagos de compras, transportes, etc, sus movimientos y gustos pueden ser seguidos y almacenados, extrayendo por ejemplo preferencias y gustos personales.

- Uso de datos para extracción de información personal. A partir del conjunto de datos de una persona extraídos del uso de RFID se pueden emplear, por ejemplo, técnicas de minería de datos para encontrar patrones, correlaciones de comportamiento o prioridades de una persona e incluso de su relación con las demás.
- Uso de datos para propósitos diferentes de su empleo original.
- Uso de datos para monitorización de comportamientos específicos. Esta monitorización se podría realizar en tiempo real, pero también mediante almacenamiento de datos y estudio posterior de los mismos. Por ejemplo, un comerciante podría estudiar los patrones de comportamiento de los usuarios en sus compras para establecer las políticas de precios que le resultaran más ventajosas.

Como respuesta al planteamiento de estos problemas, la empresa EPCglobal formo una comisión encargada de buscar el equilibrio entre los aspectos de privacidad y los posibles beneficios de la implantación de la tecnología RFID. Uno de los resultados de esta comisión fueron unas directrices para la protección de la privacidad de los consumidores. Estas directrices son:

- Información al consumidor. Los consumidores deben ser advertidos claramente de la presencia de códigos electrónicos en los productos o envases.
- Elección del consumidor. Los consumidores deben ser informados de la elección de un producto de este tipo, por si desean descartarlo o quitar las etiquetas RFID.
- Educación al consumidor. Los consumidores deben tener la posibilidad de informarse correctamente sobre el uso de las etiquetas electrónicas y sus aplicaciones.
- Grabación de usos, retención y seguridad. De la misma forma que con el código de barras, las empresas deben almacenar registros de uso, mantenimiento y protección de la información obtenida con esta tecnología, y deben publicar en sus sitios web sus políticas al respecto.

7. APLICACIÓN WEB

En la ingeniería de software se denomina aplicación web a aquellas herramientas que los usuarios pueden utilizar accediendo a un servidor web a través de internet o de una intranet mediante un navegador.

Las aplicaciones web son populares debido a lo práctico del navegador como cliente ligero, a la independencia del sistema operativo, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales. También tiene la posibilidad de ser ejecutadas en múltiples plataformas.

Es importante mencionar que una página web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, o acceder a gestores de base de datos de todo tipo.

7.1. ESTRUCTURA DE LA APLICACIÓN WEB

Aunque existen muchas variaciones posibles, una aplicación web está normalmente estructurada como una aplicación de tres-capas. El navegador web ofrece la primera capa, un motor capaz de usar alguna tecnología web dinámica, por ejemplo: PHP, Java Servlets o ASP, ASP.NET, embPerl o Python, constituye la capa intermedia. Por último, una base de datos constituye la tercera y última capa.

El proceso que se realiza es desde un navegador web manda peticiones a la capa intermedia que ofrece servicios valiéndose de consultas y actualizaciones a la base de datos y a su vez proporciona una interfaz de usuario.

Después de una breve descripción de aplicación web y la estructura de la aplicación web, se configuraran todas las herramientas que se necesitan para desarrollar con OpenXava, las cuales se mencionan en el punto 4 de este trabajo.

Los pasos realizados para desarrollar la aplicación web son:

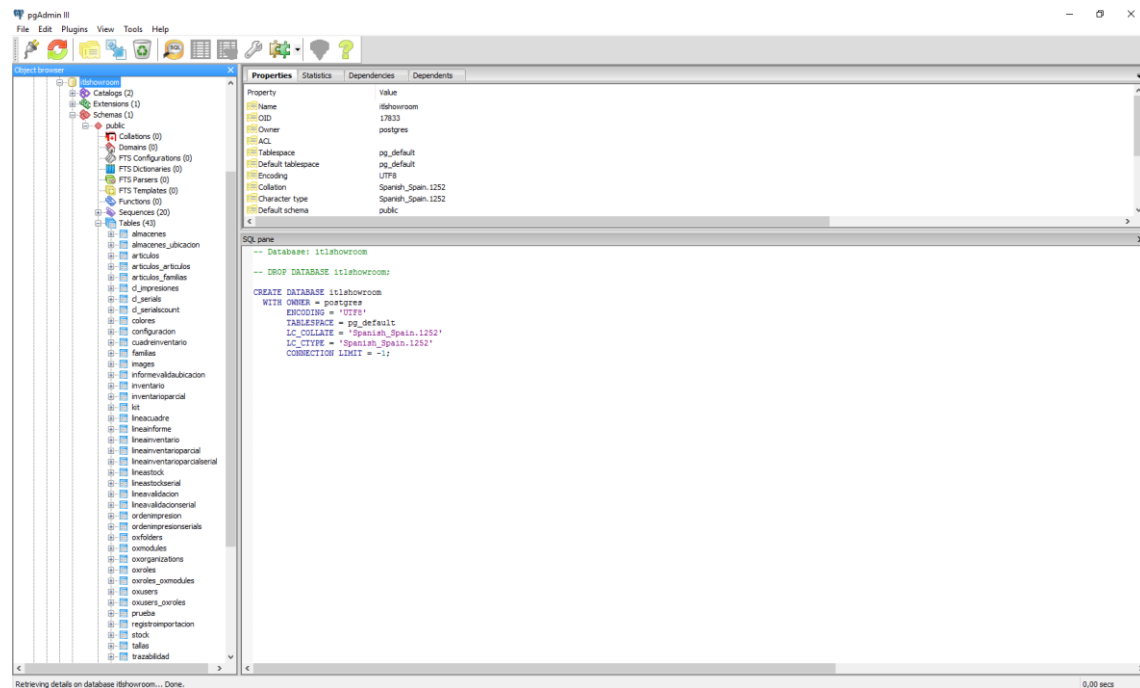
- Instalar PostgreSQL
- Instalar Eclipse;

- Configurar el Tomcat dentro del Eclipse
- Crear el proyecto para la aplicación

Una vez se han realizado estos pasos ya se tiene preparado el entorno de desarrollo.

7.2. BASE DE DATOS

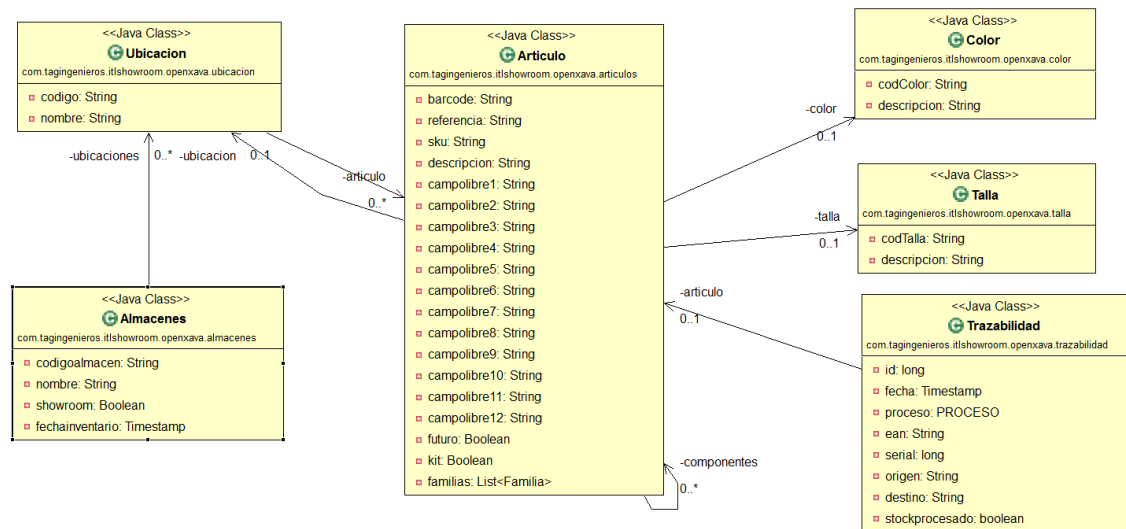
El resultado de la estructura de la base de datos utilizada es el que se muestra a continuación:



7.3. DIAGRAMA DE FLUJO

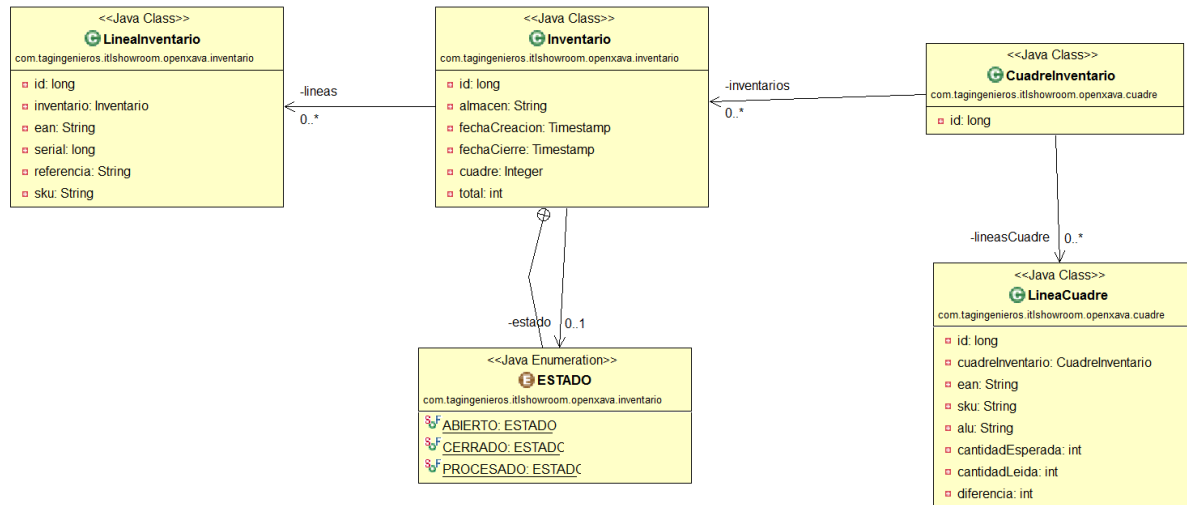
Esquema UML de la entidad de Artículo y las entidades relacionadas. Se muestra el esquema de la entidad Artículo por el motivo de que en la aplicación, el artículo es el componente sobre el que se trabaja, tanto para ver los propios datos, como saber donde está ubicado, como también tener localizado los artículos con la trazabilidad. Ya que el objetivo de la aplicación que se pidió a la empresa fue la de identificar y controlar los artículos de su almacén. Donde se ve también en el diagrama que un almacén tiene ubicaciones.

DIAGRAMA DE ARTICULOS



En el diagrama siguiente se ve que un artículo puede estar en una única ubicación y que cada ubicación puede contener distintos artículos, también como un almacén puede tener varias ubicaciones, por lo que simplificando un almacén puede tener muchas ubicaciones y en cada una de ellas puede contener distintos artículos, donde cada artículo puede tener una talla y un color. A su vez, cada registro de trazabilidad contiene un artículo, donde se tiene la información de que se ha realizado con el artículo y donde se encuentra ahora. Una vez explicado brevemente como esta relacionado el artículo con el resto de entidades, se pasara a realizar la descripción de los procesos que se realizan sobre él.

DIAGRAMA DE INVENTARIO



Se ha adjuntado el siguiente diagrama con la finalidad de mostrar la relación entre el inventario y el cuadro de inventario, donde se visualiza que un inventario puede tener muchas líneas de inventario y un cuadro inventario solo se realiza de un inventario, donde el cuadro puede tener muchas líneas de cuadro. Un inventario puede tener tres estados: Abierto, Cerrado y Procesado.

Una vez procesado el inventario realizado y comprobado en el cuadro inventario, los artículos leídos con la PDA, pasaran a ser el stock del almacén en el que se ha realizado el inventario.

Este proceso queda más detallado en el anexo, donde se adjunta el manual de usuario, que realice para que el cliente que solicitó la aplicación web, sepa cómo funciona la aplicación web.

7.4. PROCESOS

Hay procesos que se han realizado para tratar los artículos en la aplicación web, a continuación se enumeraran y se realizara una breve descripción de su funcionalidad. En este apartado no se mostrará el código realizado para su funcionalidad por motivos de confidencialidad con la empresa.

-Importación de Artículos: La importación de artículos se realiza desde el módulo Artículos, como se puede visualizar en la vista de la clase Artículos. Hay un botón “Load Artículos” que cuando se pulsa, abre una ventana para introducir el fichero de artículos que se quiere introducir a la aplicación, en el Anexo esta explicado y detallado como tiene que ser este fichero, una vez introducido se importará y pasará a reflejarse los datos en el modulo de Artículos.

-Inventario: Este proceso se inicia desde el dispositivo de lectura móvil RFID, una vez inicializado se van haciendo lecturas desde el mismo dispositivo y se van mostrando en el modulo de Inventario, en el se podrá cerrar y realizar el cuadro de inventario, para posteriormente procesarlo y que se almacenen los artículos leídos por el dispositivo en el modulo stock donde se controla la cantidad leída por cada almacén.

-Cuadre Inventario: Este proceso se realiza una vez cerrado el Inventario, en el modulo de Cuadre de Inventario se calculan y comparan los datos que hay en la aplicación con los del inventario. Básicamente, se compara las lecturas realizadas en el inventario con el stock que hay en la aplicación, sacado de un inventario anterior o de un inventario teórico, que es un archivo con una estructura donde nos indican la cantidad de artículos que hay en el almacén según los datos del cliente. En el Anexo esta detallado todas las funcionalidades con más detalle del Cuadre de Inventario.

-Validación Ubicación: El proceso de validación de ubicación es el mismo proceso que un inventario pero se realiza solo las lecturas y comprobaciones de una ubicación.

Para más información, en el anexo se detalla más ampliamente cada proceso de los mencionados, su funcionalidad y los requisitos que requiere para el correcto funcionamiento de la aplicación web.

7.5. ENTIDADES

A continuación, se muestra el código de las clases del modelo (las entidades) que se han realizado para la aplicación web. Se van a mostrar las principales entidades que se van a utilizar, donde más adelante se verá el resultado visual de estas entidades, con sus vistas gráficas.

7.5.1. CLASE ARTÍCULO

```
package com.tagingenieros.itlshowroom.openxava.articulos;

import java.util.*;
import javax.persistence.*;
import org.openxava.annotations.*;
import com.tagingenieros.itlshowroom.openxava.color.*;
import com.tagingenieros.itlshowroom.openxava.familia.*;
import com.tagingenieros.itlshowroom.openxava.talla.*;
import com.tagingenieros.itlshowroom.openxava.ubicacion.*;

@Entity
@Table(name="articulos")
@Tabs({
    @Tab(name="simple", properties="barcode, referencia, sku, descripcion")
})

public class Articulo {

    @Id
    @Required
    @Column(length=20, name="barcode")
    private String barcode;

    @Required
    @Column(length=50, name="referencia")
    private String referencia;

    @Required
    @Column(length=50, name="sku")
    private String sku;

    @Column(length=100, name="descripcion")
    private String descripcion;

    @Column(length=100, name="campolibre1")
    private String campolibre1;

    @Column(length=100, name="campolibre2")
    private String campolibre2;

    @Column(length=100, name="campolibre3")
    private String campolibre3;

    @Column(length=100, name="campolibre4")
    private String campolibre4;
}
```

```

@Column(length=100, name="campolibre5")
private String campolibre5;

@Column(length=100, name="campolibre6")
private String campolibre6;

@Column(length=100, name="campolibre7")
private String campolibre7;

@Column(length=100, name="campolibre8")
private String campolibre8;

@Column(length=100, name="campolibre9")
private String campolibre9;

@Column(length=100, name="campolibre10")
private String campolibre10;

@Column(length=100, name="campolibre11")
private String campolibre11;

@Column(length=100, name="campolibre12")
private String campolibre12;

private Boolean futuro;

private Boolean kit;

@ReferenceView(value="simple")
@ManyToOne
private Ubicacion ubicacion;

@ManyToOne
private Talla talla;

@ManyToOne
private Color color;

@ManyToMany
private List<Articulo> componentes;

//GETTERS AND SETTERS
public String getBarcode() {
    return barcode;
}

public void setBarcode(String barcode) {
    this.barcode = barcode;
}

public String getReferencia() {
    return referencia;
}

public void setReferencia(String referencia) {
    this.referencia = referencia;
}

public String getSku() {

```

```

        return sku;
    }

    public void setSku(String sku) {
        this.sku = sku;
    }

    public String getDescripcion() {
        return descripcion;
    }

    public void setDescripcion(String descripcion) {
        this.descripcion = descripcion;
    }

    public String getCampolibre1() {
        return campolibre1;
    }

    public void setCampolibre1(String campolibre1) {
        this.campolibre1 = campolibre1;
    }

    public String getCampolibre2() {
        return campolibre2;
    }

    public void setCampolibre2(String campolibre2) {
        this.campolibre2 = campolibre2;
    }

    public String getCampolibre3() {
        return campolibre3;
    }

    public void setCampolibre3(String campolibre3) {
        this.campolibre3 = campolibre3;
    }

    public String getCampolibre4() {
        return campolibre4;
    }

    public void setCampolibre4(String campolibre4) {
        this.campolibre4 = campolibre4;
    }

    public String getCampolibre5() {
        return campolibre5;
    }

    public void setCampolibre5(String campolibre5) {
        this.campolibre5 = campolibre5;
    }

    public String getCampolibre6() {
        return campolibre6;
    }
}

```

```

public void setCampolibre6(String campolibre6) {
    this.campolibre6 = campolibre6;
}

public String getCampolibre7() {
    return campolibre7;
}

public void setCampolibre7(String campolibre7) {
    this.campolibre7 = campolibre7;
}

public String getCampolibre8() {
    return campolibre8;
}

public void setCampolibre8(String campolibre8) {
    this.campolibre8 = campolibre8;
}

public String getCampolibre9() {
    return campolibre9;
}

public void setCampolibre9(String campolibre9) {
    this.campolibre9 = campolibre9;
}

public String getCampolibre10() {
    return campolibre10;
}

public void setCampolibre10(String campolibre10) {
    this.campolibre10 = campolibre10;
}

public String getCampolibre11() {
    return campolibre11;
}

public void setCampolibre11(String campolibre11) {
    this.campolibre11 = campolibre11;
}

public String getCampolibre12() {
    return campolibre12;
}

public void setCampolibre12(String campolibre12) {
    this.campolibre12 = campolibre12;
}

public Boolean getFuturo() {
    return futuro;
}

public void setFuturo(Boolean futuro) {
    this.futuro = futuro;
}

```



```
public Talla getTalla() {
    return talla;
}

public void setTalla(Talla talla) {
    this.talla = talla;
}

public Color getColor() {
    return color;
}

public void setColor(Color color) {
    this.color = color;
}

public Boolean getKit() {
    return kit;
}

public void setKit(Boolean kit) {
    this.kit = kit;
}

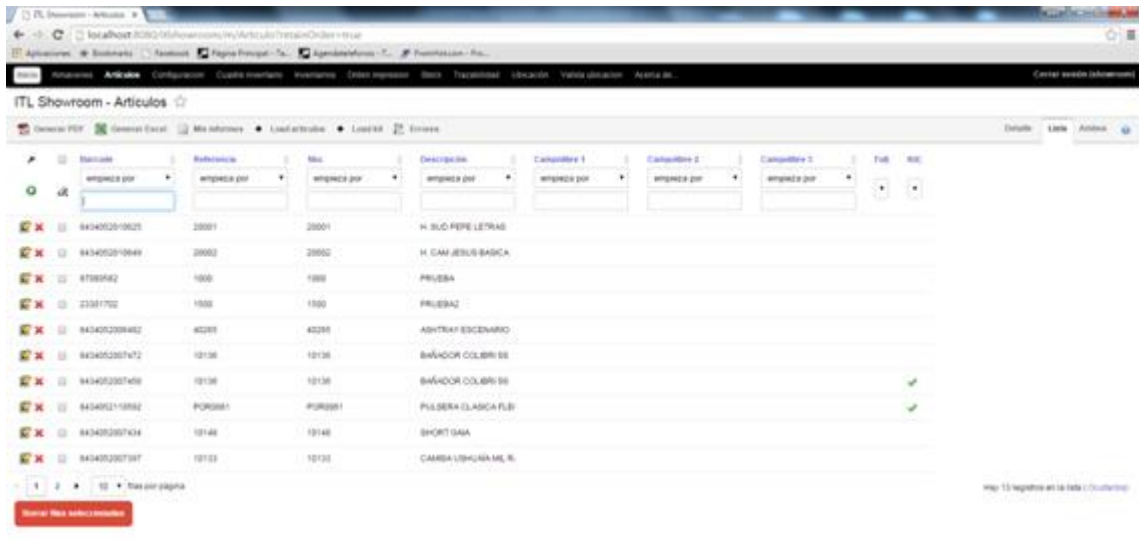
public List<Articulo> getComponentes() {
    return componentes;
}

public void setComponentes(List<Articulo> componentes) {
    this.componentes = componentes;
}

public Ubicacion getUbicacion() {
    return ubicacion;
}

public void setUbicacion(Ubicacion ubicacion) {
    this.ubicacion = ubicacion;
}
}
```

VISTA



7.5.2. CLASE ALMACEN

```
package com.tagingenieros.itlshowroom.openxava.almacenes;
```

```
import java.sql.*;
import java.util.*;
import javax.persistence.*;
import org.openxava.annotations.*;
import com.tagingenieros.itlshowroom.openxava.ubicacion.*;
```

```
@Entity
@Table(name = "almacenes")
@Views({ @View(members = "codigoalmacen;nombre;showroom;ubicaciones"),
         @View(name = "almacen_ubicacion", members =
            "codigoalmacen;nombre;showroom"),
        })
@Tab(properties = "codigoalmacen,nombre,showroom,fechainventario")
```

```
public class Almacenes {

    @Id
    @Column(length = 20, name = "codigoalmacen")
    private String codigoalmacen;

    @Required
    @Column(length = 50, name = "nombre")
    private String nombre;

    @ManyToMany
    @NewAction(value = "FiltrarUbicacionShowroom.add")
    private Set<Ubicacion> ubicaciones;

    private Boolean showroom;

    @ReadOnly
    private Timestamp fechainventario;

    // GETTERS AND SETTERS
```

```

public String getCodigoalmacen() {
    return codigoalmacen;
}

public void setCodigoalmacen(String codigoalmacen) {
    this.codigoalmacen = codigoalmacen;
}

public String getNombre() {
    return nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public Boolean getShowroom() {
    return showroom;
}

public void setShowroom(Boolean showroom) {
    this.showroom = showroom;
}

public Set<Ubicacion> getUbicaciones() {
    return ubicaciones;
}

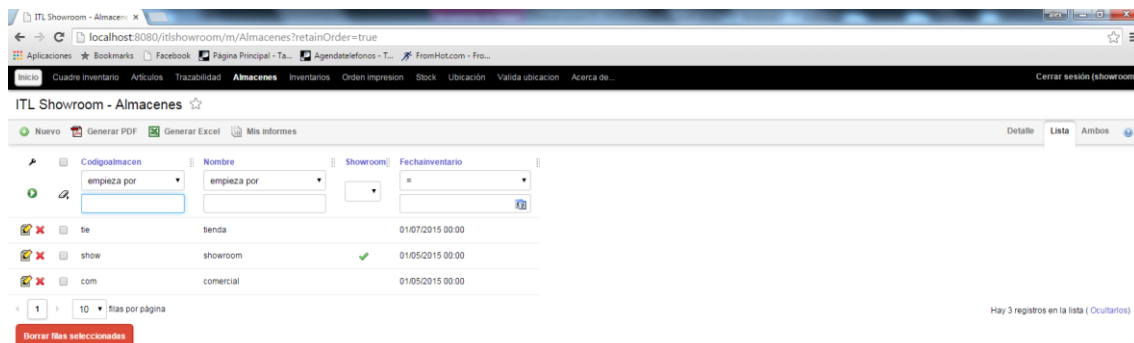
public void setUbicaciones(Set<Ubicacion> ubicaciones) {
    this.ubicaciones = ubicaciones;
}

public Timestamp getFechainventario() {
    return fechainventario;
}

public void setFechainventario(Timestamp fecha) {
    this.fechainventario = fecha;
}
}

```

VISTA



7.5.3. CLASE INVENTARIO

```
package com.tagingenieros.itlshowroom.openxava.inventario;

import java.math.*;
import java.sql.*;
import java.util.*;
import javax.persistence.*;
import org.openxava.annotations.*;
import com.tagingenieros.itlshowroom.openxava.inventario.*;

@Entity
@Tab(properties="id, estado, fechaCreacion, fechaCierre, almacen, total,
cuadre")
@View(members="inventario [#id;estado;fechaCreacion, fechaCierre, almacen,
total, cuadre;];lineas")
public class Inventario {

    public enum ESTADO {
        ABIERTO, // Se pueden añadir lecturas al inventario
        CERRADO, // Se puede realizar el cuadro de inventario
        PROCESADO // Se ha realizado ya el cuadro de inventario
    }

    @Id
    @ReadOnly
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private long id;

    @Required
    @ReadOnly
    @Enumerated(EnumType.STRING)
    @DefaultValueCalculator(EstadoDefaultValue.class)
    private ESTADO estado;

    @Required
    @Column(length=20)
    @ReadOnly
    private String almacen;//almacen inventario

    @Required
    @ReadOnly
    @Stereotype("DATETIME")
    @Column(length=14)
    @DefaultValueCalculator(FechaDefaultValue.class)
    private Timestamp fechaCreacion;//fecha creacion inventario

    @ReadOnly
    @Stereotype("DATETIME")
    @Column(length=14)
    private Timestamp fechaCierre;//fecha cierre inventario

    @OneToMany(targetEntity=LineaInventario.class, mappedBy="inventario",
    cascade = {CascadeType.REMOVE, CascadeType.DETACH,
        CascadeType.MERGE, CascadeType.PERSIST})
    @ListProperties("ean, sku, referencia, serial")
    @NewAction("Inventariolinea.add") //añadir una lineainventario al
inventario
```

```

    @RemoveSelectedAction("InventarioLinea.removeSelected")//acción delete
de una línea de inventario en vista detalle de Inventario
    @RemoveAction("InventarioDialogoLinea.remove") //acción que se ejecuta
dentro del dialogo de un elemento de la colección
    @SaveAction("InventarioDialogoLinea.save") //acción Save dentro del
dialogo de líneaInventario
    private List<LineaInventario> lineas;

    @org.hibernate.annotations.Formula("(SELECT inventarios_id FROM
inventario i WHERE i.id=id)")
    private Integer cuadre;
    public Integer getCuadre(){
        return cuadre;
    }

    //GETTERS AND SETTERS
    public long getId() {
        return id;
    }

    public void setId(long id) {
        this.id = id;
    }

    public ESTADO getEstado() {
        return estado;
    }

    public void setEstado(ESTADO estado) {
        this.estado = estado;
    }

    public Timestamp getFechaCreacion() {
        return fechaCreacion;
    }

    public void setFechaCreacion(Timestamp fechaCreacion) {
        this.fechaCreacion = fechaCreacion;
    }

    public List<LineaInventario> getLineas() {
        return lineas;
    }

    public void setLineas(List<LineaInventario> lineas) {
        this.lineas = lineas;
    }
    //almacen de la ubicacion de los articulos
    public String getAlmacen() {
        return almacen;
    }

    public void setAlmacen(String almacen) {
        this.almacen = almacen;
    }

    @org.hibernate.annotations.Formula("(SELECT count(1)
FROM lineainventario li WHERE li.inventario_id = id)")
    private int total;

```

```

public int getTotal(){
    return total;
}

public Timestamp getFechaCierre() {
    return fechaCierre;
}

public void setFechaCierre(Timestamp fechaCierre) {
    this.fechaCierre = fechaCierre;
}
}

```

VISTA

Id	Estado	Fecha	Fecha cierre	Almacén	Total	Cuadre
1	PROCESADO	05/04/2015 00:00	08/04/2015 00:00	showroom	3	1
3	CERRADO	10/05/2015 00:00	11/06/2015 11:12	tie	0	2
4	PROCESADO	09/05/2015 00:00	11/06/2015 13:02	show	2	3
5	PROCESADO	11/06/2015 13:22	17/06/2015 12:44	show	1	4
2	CERRADO	11/06/2015 10:04	23/06/2015 12:58	com	3	5
6	CERRADO	25/06/2015 00:00	03/07/2015 13:17	com	9	

7.5.4. CLASE CUADRE INVENTARIO

```
package com.tagingenieros.itlshowroom.openxava.cuadre;
```

```

import java.util.*;
import javax.persistence.*;
import org.openxava.annotations.*;
import com.tagingenieros.itlshowroom.openxava.inventario.*;

```

```

@Entity
@Tab(properties = "id, idsInventarios, totalEsperada, totalLeida,
totalDiferenciaPostivas, totalDiferenciaNegativas")
@View(members = "cuadreinventario [#id,idsInventarios;" +
"totalEsperada,totalLeida;"
+ "totalDiferenciaPostivas,totalDiferenciaNegativas;"
+ "];lineasCuadre")

```

```

public class CuadreInventario {

    @Id
    @ReadOnly
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;

    @OneToMany(targetEntity = Inventario.class, cascade = CascadeType.ALL)
    @JoinColumn(referencedColumnName = "id")
    private List<Inventario> inventarios;
}

```

```

// para relacionar las líneas con su Cuadre
@OneToMany(targetEntity = LineaCuadre.class, mappedBy =
"cuadreInventario", cascade = { CascadeType.REMOVE,
CascadeType.DETACH, CascadeType.MERGE, CascadeType.PERSIST })
@ListProperties("ean, sku, alu, cantidadEsperada, cantidadLeida,
diferencia")
@NewAction("LineaCuadre.add")
// añadir una lineaCuadre al CuadreInventario
private List<LineaCuadre> lineasCuadre;

// GETTERS AND SETTERS

public List<Inventario> getInventarios() {
    return inventarios;
}

public void setInventarios(List<Inventario> inventarios) {
    this.inventarios = inventarios;
}

public List<LineaCuadre> getLineasCuadre() {
    return lineasCuadre;
}

public void setLineasCuadre(List<LineaCuadre> lineasCuadre) {
    this.lineasCuadre = lineasCuadre;
}

@Transient
public String getIdsInventarios() {
    StringBuilder sb = new StringBuilder();

    if (this.inventarios != null) {
        if (this.inventarios.isEmpty() == false) {
            for (Inventario inventario : this.inventarios) {
                sb.append(inventario.getId()).append(',');
            }

            sb.deleteCharAt(sb.length() - 1);
        }
    }

    return sb.toString();
}

@Transient
public int getTotalLeida() {
    int suma = 0;

    if (lineasCuadre != null) {
        for (LineaCuadre linea : this.lineasCuadre) {
            suma += linea.getCantidadLeida();
        }
    }

    return suma;
}

```

```

@Transient
public int getTotalEsperada() {
    int suma = 0;

    if (lineasCuadre != null) {
        for (LineaCuadre linea : this.lineasCuadre) {
            int cantidad = linea.getCantidadEsperada();

            if (cantidad > 0) {
                suma += cantidad;
            }
        }
    }

    return suma;
}

@Transient
public int getTotalNegativos() {
    int suma = 0;

    if (lineasCuadre != null) {
        for (LineaCuadre linea : this.lineasCuadre) {
            int cantidad = linea.getCantidadEsperada();

            if (cantidad < 0) {
                suma += cantidad;
            }
        }
    }

    return suma;
}

@Transient
public int getTotalDiferenciaPostivas() {
    int suma = 0;

    if (lineasCuadre != null) {
        for (LineaCuadre linea : this.lineasCuadre) {

            int cantidad = linea.getDiferencia();

            if (cantidad > 0) {
                suma += cantidad;
            }
        }
    }

    return suma;
}

@Transient
public int getTotalDiferenciaNegativas() {
    int suma = 0;

    if (lineasCuadre != null) {
        for (LineaCuadre linea : this.lineasCuadre) {

```



```

        int cantidad = linea.getDiferencia();

        if (cantidad < 0) {
            suma += cantidad;
        }
    }
}

return suma;
}

// la suma de las cantidades leídas cuya diferencia es negativa
@Transient
public int getTotalLeidasConDiferenciaNegativa() {
    int suma = 0;

    if (lineasCuadre != null) {
        for (LineaCuadre linea : this.lineasCuadre) {

            int cantidad = linea.getDiferencia();

            if (cantidad < 0) {
                suma += linea.getCantidadLeida();
            }
        }
    }

    return suma;
}

// la suma de las cantidades leídas cuya diferencia es positiva
@Transient
public int getTotalLeidasConDiferenciaPositiva() {
    int suma = 0;

    if (lineasCuadre != null) {
        for (LineaCuadre linea : this.lineasCuadre) {

            int cantidad = linea.getDiferencia();

            if (cantidad > 0) {
                suma += linea.getCantidadLeida();
            }
        }
    }

    return suma;
}

@Transient
public int getTotalLineasPositivas() {
    int suma = 0;

    if (lineasCuadre != null) {
        for (LineaCuadre linea : this.lineasCuadre) {

            int cantidad = linea.getDiferencia();

```

```

                if (cantidad > 0) {
                    suma += 1;
                }
            }
        }

        return suma;
    }

    @Transient
    public int getTotalLineasDiferenciasNegativas() {
        int suma = 0;

        if (lineasCuadre != null) {
            for (LineaCuadre linea : this.lineasCuadre) {

                int cantidad = linea.getDiferencia();

                if (cantidad < 0) {
                    suma += 1;
                }
            }
        }

        return suma;
    }

    @Transient
    public int getTotalLineasNoDiferencias() {
        int suma = 0;

        if (lineasCuadre != null) {
            for (LineaCuadre linea : this.lineasCuadre) {

                int cantidad = linea.getDiferencia();

                if (cantidad == 0) {
                    suma += 1;
                }
            }
        }

        return suma;
    }

    @Transient
    public int getTotalNoDiferencias() {
        int suma = 0;

        if (lineasCuadre != null) {
            for (LineaCuadre linea : this.lineasCuadre) {

                int cantidad = linea.getDiferencia();

                if (cantidad == 0) {
                    suma += linea.getCantidadLeida();
                }
            }
        }
    }

```

```

    }

    return suma;
}

public long getId() {
    return id;
}

public void setId(long id) {
    this.id = id;
}
}

```

VISTA

The screenshot shows a web browser displaying a page titled 'ITL Showroom - Cuadre inventario'. The page features a table with the following columns: 'Almacén', 'Cantidad', 'Total Stock', 'Línea de', and 'Total No de'. The table contains several rows of data, including a row with '1' in the 'Almacén' column and '1' in the 'Cantidad' column. The table is displayed in a grid format with a search bar and a 'Detalle' button above it.

7.5.5. CLASE STOCK

```

package com.tagingenieros.itlshowroom.openjava.stock;

import java.math.*;
import java.util.*;
import javax.persistence.*;
import org.openjava.annotations.*;
import org.openjava.jpa.*;

import com.tagingenieros.itlshowroom.openjava.almacenes.*;

@Entity
@Tab(properties="almacen,nombrealmacen,total")
@View(members="stock [#almacen,nombrealmacen,total];lineasStock")

public class Stock {

    @Id
    @Required
    @Column(length=20)
    @ReadOnly
    private String almacen;//almacén de stock
}

```

```

@Required
@Column(length=50)
@ReadOnly
private String nombrealmacen;//almacen de stock

//para relacionar las lineas de Stock
@OneToMany(targetEntity=LineaStock.class, mappedBy="registroStock",
cascade = {CascadeType.REMOVE, CascadeType.DETACH,
CascadeType.MERGE, CascadeType.PERSIST})
@ListProperties("ean, sku, referencia, stock")
@NewAction("LineaStock.add") //anadir una lineaStock al Stock
@RemoveSelectedAction("LineaStock.removeSelected")//acción delete de
una lineaStock en vista detalle de Stock
@RemoveAction("StockDialogoLineaStock.remove") //acción que se ejecuta
dentro del dialogo de un elemento de la coleccion
@SaveAction("StockDialogoLineaStock.save") //acción Save dentro del
dialogo de lineaStock
private List<LineaStock> lineasStock;

//GETTERS AND SETTERS

public List<LineaStock> getLineasStock() {
return lineasStock;
}

public String getAlmacen() {
return almacen;
}

public void setAlmacen(String almacen) {
this.almacen = almacen;
}

public String getNombrealmacen() {
return nombrealmacen;
}

public void setNombrealmacen(String nombrealmacen) {
this.nombrealmacen = nombrealmacen;
}

public void setLineasStock(List<LineaStock> lineasStock) {
this.lineasStock = lineasStock;
}

@Transient
public int getTotal() {
EntityManager em = XPersistence.getManager();
Query query = em.createNativeQuery(String.format(
"SELECT count(1) FROM lineastock ls
INNER JOIN lineastockserial lx ON ls.id= lx.stocklinea_id
WHERE ls.registrostock_almacen = '%s'",this.almacen));
List result = query.getResultList();
if (result == null || result.isEmpty()) {
return 0;
} else {
BigInteger quantity = (BigInteger) result.get(0);
if (quantity == null) {
return 0;
}
}
}

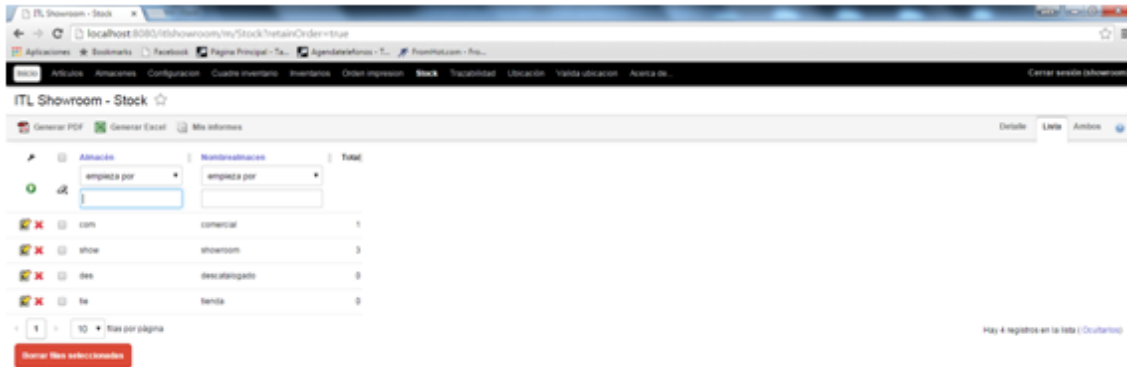
```

```

    }
    int q = ((BigInteger) quantity).intValue();
    return q;
}
}
}

```

VISTA



7.5.6. CLASE TRAZABILIDAD

```
package com.tagingenieros.itlshowroom.openxava.trazabilidad;
```

```
import java.sql.*;
import javax.persistence.*;
import org.openxava.annotations.*;
import org.openxava.jpa.*;
import com.tagingenieros.itlshowroom.openxava.articulos.*;
```

```
@Entity
```

```
@Tab(properties="fecha,proceso,ean,sku,referencia,origen,destino")
```

```
public class Trazabilidad {
```

```
    public enum PROCESO {
        ALTA, // No afecta a stock
        IMPRESION, // No afecta a stock si ya esta impreso
        INVENTARIO, // No se procesa
        MOV_ALMACEN, //Añadimos para movimientos entre almacenes.
        MOV_UBICACION, // No afecta a stock
        BAJA
    }
```

```
@Id
```

```
@ReadOnly
```

```
@GeneratedValue(strategy=GenerationType.IDENTITY)
```

```
private long id;
```

```
@Required
```

```
@ReadOnly
```

```
@Stereotype("DATETIME")
```

```
@Column(length=14)
```

```
private Timestamp fecha;
```

```

@Required
@Column(length=15)
@ReadOnly
private PROCESO proceso; // origen

@Required
@ReadOnly
@Column(length=20)
private String ean;

@Required
@ReadOnly
private long serial;

//no requerida para que permita valores nulos en los procesos de
trazabilidad que no tienen origen
@Column(length=20)
@ReadOnly
private String origen;

//no requerida para que permita valores nulos en los procesos de
trazabilidad que no tienen destino
@Column(length=20)
@ReadOnly
private String destino;

@Column
@ReadOnly
@DefaultValueCalculator(StockProcesadoDefaultValue.class)
private boolean stockprocesado=false;

//GETTERS AND SETTERS
public boolean isStockprocesado() {
    return stockprocesado;
}

public void setStockprocesado(boolean stockprocesado) {
    this.stockprocesado = stockprocesado;
}

public long getId() {
    return id;
}

public void setId(long id) {
    this.id = id;
}

public Timestamp getFecha() {
    return fecha;
}

public void setFecha(Timestamp fecha) {
    this.fecha = fecha;
}

public PROCESO getProceso() {

```

```

        return proceso;
    }

    public void setProceso(PROCESO proceso) {
        this.proceso = proceso;
    }

    public String getEan() {
        return ean;
    }

    public void setEan(String ean) {
        this.ean = ean;
    }

    public long getSerial() {
        return serial;
    }

    public void setSerial(long serial) {
        this.serial = serial;
    }

    @Transient
    private Articulo articulo = null;

    private void getArticulo(){
        if(this.articulo == null){
            EntityManager em = XPersistence.createManager();
            this.articulo = em.find(Articulo.class, this.ean);
            em.close();
        }
    }

    @Transient
    public String getReferencia(){
        getArticulo();
        return (this.articulo == null) ? "--" : articulo.getReferencia();
    }

    @Transient
    public String getSku(){
        getArticulo();
        return (this.articulo == null) ? "--" : articulo.getSku();
    }

    public String getOrigen() {
        return origen;
    }

    public void setOrigen(String origen) {
        this.origen = origen;
    }

    public String getDestino() {
        return destino;
    }

    public void setDestino(String destino) {

```

```

        this.destino = destino;
    }

    @Override
    public String toString() {
        StringBuilder builder = new StringBuilder();
        builder.append("Trazabilidad [id=");
        builder.append(id);
        builder.append(", fecha=");
        builder.append(fecha);
        builder.append(", proceso=");
        builder.append(proceso);
        builder.append(", ean=");
        builder.append(ean);
        builder.append(", serial=");
        builder.append(serial);
        builder.append(", origen=");
        builder.append(origen);
        builder.append(", destino=");
        builder.append(destino);
        builder.append(", stockprocesado=");
        builder.append(stockprocesado);
        builder.append(", articulo=");
        builder.append(articulo);
        builder.append("]");
        return builder.toString();
    }
}
}

```

VISTA

ITL Showroom - Trazabilidad

Generar PDF Generar Excel

Detalle Lista Ambos

Fecha	Proceso	Ean	Sku	Referencia	Origen	Destino
12/08/2015 10:51	MOV ALMACEN	8434052010632	--	--	com	show
25/02/2016 14:04	BAJA	23301702	493	06129F-6DZ		
03/08/2015 00:00	ALTA	7450045465617	24309020006	24309020006		show
03/08/2015 00:00	ALTA	7450045476231	BOH-02-CAS-CE290MM	BOH-02-CAS-CE290MM		show
03/08/2015 00:00	ALTA	87080582	--	--		show
04/08/2015 00:00	ALTA	23301696	491	0540F-6DZ		show
04/08/2015 00:00	ALTA	23301702	493	06129F-6DZ		show
25/02/2016 14:09	BAJA	23301702	493	06129F-6DZ		
04/08/2015 09:04	INVENTARIO	87080582	--	--	show	show
04/08/2015 09:04	INVENTARIO	87080582	--	--	show	show

1 2 3 4 5 6 10 filas por página

Hay 377 registros en la lista (Ocultarlos)

7.5.7. CLASE UBICACIÓN

```
package com.tagingenieros.itlshowroom.openxava.ubicacion;

import java.util.*;
import javax.persistence.*;
import org.openxava.annotations.*;
import com.tagingenieros.itlshowroom.openxava.articulos.*;

@Entity
@Table(name="ubicacion")
@View(name="simple", members="codigo, nombre;")

public class Ubicacion {

    @Id
    @Required
    @Column(length=15, name="codigo")
    private String codigo;

    @Column(length=150)
    @Required
    private String nombre;

    //relacion con los articulos
    @NewAction(value="FiltrarArticuloUbicacion.add")
    @RemoveSelectedAction("FiltrarArticuloUbicacion.removeSelected")
    @OneToMany
    @PrimaryKeyJoinColumn(name="articulos_barcode",
        referencedColumnName="barcode")
    @JoinColumn(name="ubicacion_codigo", referencedColumnName="codigo")
    private Set<Articulo> articulo;

    //GETTERS AND SETTERS

    public String getCodigo() {
        return codigo;
    }

    public void setCodigo(String codigo) {
        this.codigo = codigo;
    }

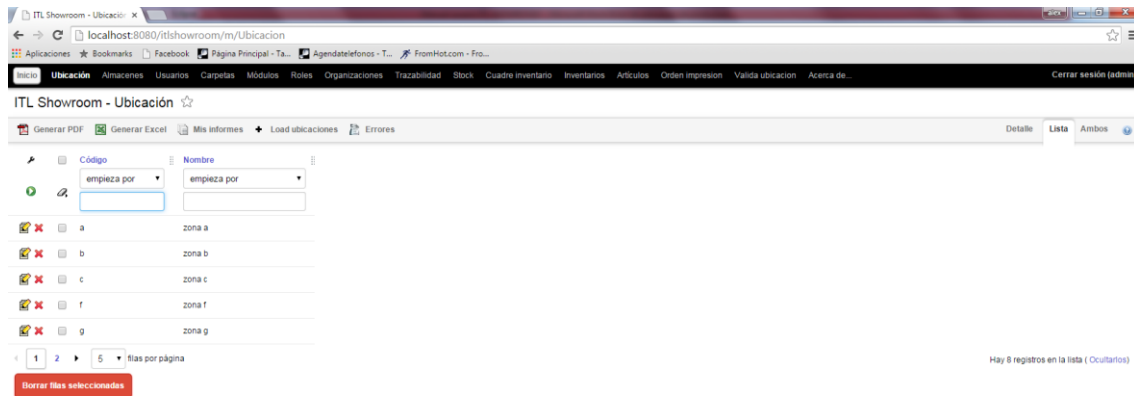
    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public Set<Articulo> getArticulo() {
        return articulo;
    }

    public void setArticulo(Set<Articulo> articulo) {
        this.articulo = articulo;
    }
}
```

VISTA



7.5.8. CLASE VALIDACION UBICACIÓN

```
package com.tagingenieros.itlshowroom.openxava.validaubicacion;
```

```
import java.sql.*;
import java.util.*;
import javax.persistence.*;
import org.openxava.annotations.*;
import com.tagingenieros.itlshowroom.openxava.inventario.*;
```

```
@Tab(properties="id,almacen,estado, fechaCreacion, fechaCierre,
totalUbicaciones")
@View(members="Cabecera [#estado, almacen; fechaCreacion, fechaCierre;
totalUbicaciones];lineasValidacion")
```

```
@Entity
```

```
public class ValidaUbicacion {
```

```
    public enum ESTADOVALIDACION {
        ABIERTO, // Se pueden añadir lecturas al inventario
        CERRADO, // Se puede realizar el cuadro de inventario
        PROCESADO // Se ha realizado ya el cuadro de inventario
    }
```

```
    @Id
    @ReadOnly
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;
```

```
    @Required
    @Column(length = 20)
    @ReadOnly
    private String almacen;// almacén de validación
```

```
    @Required
    @ReadOnly
    @Enumerated(EnumType.STRING)
    @DefaultValueCalculator(EstadoDefaultValue.class)
    private ESTADOVALIDACION estado;
```

```
    @Required
```

```

@ReadOnly
@Stereotype("DATETIME")
@Column(length = 14)
@DefaultValueCalculator(FechaDefaultValue.class)
private Timestamp fechaCreacion;// fecha creación

@ReadOnly
@Stereotype("DATETIME")
@Column(length = 14)
private Timestamp fechaCierre;// fecha cierre

// para relacionar las líneas de validación
@OneToMany(targetEntity = LineaValidacion.class, mappedBy =
"registroValidacion", cascade = { CascadeType.REMOVE, CascadeType.DETACH,
CascadeType.MERGE,
        CascadeType.PERSIST })
@ListProperties("ubicacion.codigo, ubicacion.nombre,
totalarticulos");//, serialOK
@NewAction("LineaValidacion.add")
// añadir una linea inventario al inventario
@RemoveSelectedAction("LineaValidacion.removeSelected")
//acción delete de una línea de inventario en vista detalle de
Inventario
@RemoveAction("ValidaDialogoLinea.remove")
//acción que se ejecuta dentro del dialogo de un elemento de la
colección
@SaveAction("ValidaDialogoLinea.save")
// acción Save dentro del dialogo de linea inventario
private List<LineaValidacion> lineasValidacion;

@org.hibernate.annotations.Formula("(SELECT validaubicaciones_id FROM
validaubicacion v WHERE v.id=id)")
private Integer informe;

public Integer getInforme() {
    return informe;
}

// GETTERS AND SETTERS

public long getId() {
    return id;
}

public void setId(long id) {
    this.id = id;
}

public ESTADOVALIDACION getEstado() {
    return estado;
}

public void setEstado(ESTADOVALIDACION estado) {
    this.estado = estado;
}

public Timestamp getFechaCreacion() {
    return fechaCreacion;
}

```

```

public void setFechaCreacion(Timestamp fechaCreacion) {
    this.fechaCreacion = fechaCreacion;
}

// almacén de la ubicación de los artículos
public String getAlmacen() {
    return almacen;
}

public void setAlmacen(String almacen) {
    this.almacen = almacen;
}

public Timestamp getFechaCierre() {
    return fechaCierre;
}

public void setFechaCierre(Timestamp fechaCierre) {
    this.fechaCierre = fechaCierre;
}

public List<LineaValidacion> getLineasValidacion() {
    return lineasValidacion;
}

public void setLineasValidacion(List<LineaValidacion>
lineasValidacion)
{
    this.lineasValidacion = lineasValidacion;
}

@org.hibernate.annotations.Formula("(SELECT count(1) FROM
lineavalidacion lv WHERE lv.registrovalidacion_id = id)")
private int totalUbicaciones;

public int getTotalUbicaciones() {
    return totalUbicaciones;
}
}

```

VISTA

id	Almacén	Estado	Fecha	Fecha cierre	Total ubicaciones
1	show	PROCESADO	23/06/2015 08:00	23/06/2015 13:42	3
2	show	CERRADO	25/06/2015 08:00	26/06/2015 13:21	1
7	show	ABIERTO	21/07/2015 16:03		3
12	show	CERRADO	01/07/2015 16:01	08/07/2015 11:55	3
5	show	CERRADO	26/06/2015 08:00	08/07/2015 13:55	3

7.5.9. CLASE DE ÓRDENES DE IMPRESIÓN

```
package com.tagingenieros.itlshowroom.openxava.ordenes;

import java.sql.*;
import java.util.*;
import javax.persistence.*;
import org.openxava.annotations.*;

@Tab(properties="id,fichero,fechaimportacion,total,imprimidas,ptesImprimir")
@View(members="Orden
[#id,fichero,fechaimportacion;total,imprimidas,ptesImprimir;];lineas")
@Entity
public class OrdenImpresion {

    @Id
    @Column(length = 50)
    private String id;

    @Required
    @ReadOnly
    @Stereotype("DATETIME")
    @Column(length = 14)
    private Timestamp fechaimportacion;

    @Column(length = 50)
    @ReadOnly
    private String fichero;

    @OneToMany(targetEntity = OrdenImpresionSerials.class,
mappedBy = "orden", cascade = CascadeType.ALL)
    @ListProperties("clave, ean, serial, ordendeimpresion,
numvecesimpreso, sku, referencia, descripcion,"
+"campolibre1,campolibre2,campolibre3,campolibre4,campolibre5,
campolibre6,campolibre7,campolibre8, campolibre9, campolibre10,
campolibre11, campolibre12")
    @NewAction(value = "OrdenImpresionLinea.add")
    // acción para añadir una nueva línea a la orden
    @RemoveSelectedAction("OrdenImpresionLinea.removeSelected")
    // acción delete de una línea (OrdenImpresionSerials) en vista detalle
de Orden de impresión
    @RemoveAction("OrdenDialogoLinea.remove")
    // acción que se ejecuta dentro del dialogo de un elemento de la
colección
    @SaveAction("OrdenDialogoLinea.save")
    // acción Save dentro del dialogo de una línea de una orden, es decir,
OrdenImpresionSerials
    private List<OrdenImpresionSerials> lineas;

    @org.hibernate.annotations.Formula("(SELECT count(*) FROM
ordenimpresionserials li WHERE li.orden_id = id)")
    private int total;

    public int getTotal() {
        return total;
    }

    @org.hibernate.annotations.Formula("(SELECT count(*) FROM
ordenimpresionserials li " + "INNER JOIN cl_impresiones cli ")
```

```

        + "ON cli.clave = li.clave AND li.orden_id=id")
private int imprimidas;

public int getImprimidas() {
    return imprimidas;
}

@org.hibernate.annotations.Formula("(" + "SELECT " + "(SELECT
count(*) FROM ordenimpresionserials li WHERE li.orden_id = ID) "
+ "- "
+ "(SELECT COUNT(*) FROM ordenimpresionserials li "
+ "INNER JOIN cl_impresiones cli "
+ "ON cli.clave = li.clave AND li.orden_id = ID) "
+ ")")
private int ptesImprimir;

public int getPtesImprimir() {
    return ptesImprimir;
}

// GETTERS AND SETTERS

public String getId() {
    return id;
}

public void setId(String id) {
    this.id = id;
}

public Timestamp getFechaimportacion() {
    return fechaimportacion;
}

public void setFechaimportacion(Timestamp fechaimportacion) {
    this.fechaimportacion = fechaimportacion;
}

public String getFichero() {
    return fichero;
}

public void setFichero(String fichero) {
    this.fichero = fichero;
}

public List<OrdenImpresionSerials> getLineas() {
    return lineas;
}

public void setLineas(List<OrdenImpresionSerials> lineas) {
    this.lineas = lineas;
}
}

```

VISTA

The screenshot shows a web browser window displaying the 'Orden Impresion' application. The browser address bar shows 'localhost:8080/itlshowroom/m/OrdenImpresion'. The application header includes navigation links like 'Inicio', 'Orden impresion', 'Vista ubicacion', 'Ubicacion', 'Almacenes', 'Articulos', 'Trazabilidad', 'Stock', 'Cuadro Inventario', 'Inventarios', and 'Ayuda de...'. Below the header, there are tabs for 'Detalle', 'Lista', and 'Ambos'. The main content area features a table with columns: 'Id', 'Archivo', 'Fechaimportacion', 'Total', 'Impresiones', and 'Pasa imprimir'. The table contains three rows of data. Below the table, there is a pagination control showing '1' of '10' items per page and a status message 'Hay 3 registros en la lista (Ocultar)'. A modal dialog titled 'Cargar orden impresion' is open in the center, with a text input field containing 'Seleccionar archivo' and the message 'Ningún archivo seleccionado'. The dialog has two buttons: 'Iniciar importacion orden impresion' and 'Cancelar'.

Id	Archivo	Fechaimportacion	Total	Impresiones	Pasa imprimir
1	impresion.txt	03/08/2015 12:54	5	5	0
2	impresion2.txt	11/08/2015 17:22	1	4	-3
3	impresion3.txt	12/08/2015 11:06	3	0	3

8. CONCLUSIÓN

Después de realizar este trabajo se ha llegado a la conclusión que, la tecnología RFID mejora en diversos aspectos al código de barras, y con ella, se puede mejorar el rendimiento de una empresa logística, en la que se pueda aplicar este tipo de tecnología.

Del mismo modo, se extrae la conclusión que, realizando una aplicación web como la que se ha mostrado, se puede tener el control de los datos de una forma fácil y práctica para el usuario final. Haciendo más ágil y rápido, el proceso de inventario para el almacén Showroom que se ha trabajado.

Para finalizar, cabe indicar que este Trabajo Final de Grado ha sido implantado en el almacén de una empresa de Panamá, en la cual he podido comprobar que el sistema ha sido efectivo, y se ha reducido el tiempo y la cantidad de gente para realizar inventarios, con lo que se ha llegado a conseguir los objetivos que se pretenden al implantar el sistema RFID.

ANEXO

1. ACCESO AL SISTEMA

1.1. ACCESO AL SISTEMA

El sistema de gestión puede ser accedido a través del navegador habitual del usuario (Google Chrome, Firefox, Internet Explorer, etc), la dirección URL que hay que acceder para poder utilizar el sistema de gestión está compuesta por la dirección IP del servidor, el puerto 8080 y el contexto itlshowroom, quedando de la siguiente manera:

http://direccion_ip:8080/itlshowroom

Ejemplo de dirección URL:

<http://10.10.40.150:8080/itlshowroom>

Esta dirección se puede guardar en la opción de “Favoritos” o “Marcadores” del navegador del usuario. Se puede guardar también en forma de acceso directo en el escritorio.

Una vez se accede a esta dirección la primera pantalla solicita el usuario y la contraseña:



Una vez validados en la aplicación, la vista que observa el usuario está formada por un menú vertical en la parte izquierda de la ventana. Dado que este menú habitualmente se oculta para facilitar la visualización de datos, el usuario debe clicar el botón “Inicio” para ver el menú.

En la parte superior de la ventana se muestra un menú que acumula los módulos a los que ha accedido el usuario recientemente y la opción de “Cerrar sesión”, recomendable en caso de que el ordenador de acceso sea compartido.

1.2. VISUALIZACION DE DATOS

Hay tres formas de visualizar los datos de cada módulo: lista, detalle o ambos.

En la vista modo lista es el primero que se carga al acceder a un módulo y permite visualizar todos los datos registrados con distribución por columnas y su paginación correspondiente.

Si se clica un registro se visualiza en modo detalle, sirve para ver el registro en forma tabular para editarlo. En esta vista es posible ver si el objeto está relacionado con otros tipos de objeto, mostrando al detalle esta relación. La vista detalle es la usada también cuando se crea un nuevo registro.

La vista “ambos” muestra una combinación de las dos anteriores. Una primera sección permite ver todos los registros existentes en formato de lista y una segunda sección muestra en vista detalle la opción de crear un nuevo registro o visualizar uno existente si se selecciona en la parte superior.

Vista modo lista

ITL Showroom - Artículos

Generar PDF Generar Excel Mis informes Load artículos Load kit Errores

Barcode	Referencia	Sku	Descripción	Campolbre 1	Campolbre 2	Campolbre 3	Futl	Kit
8434052010625	20001	20001	H. SUD PEPE LETRAS					
8434052010649	20002	20002	H. CAM JESUS BASICA					
87080582	1000	1000	PRUEBA					
23301702	1500	1500	PRUEBA2					
8434052006482	40265	40265	ASHTRAY ESCENARIO					
8434052007472	10136	10136	BAÑADOR COLIBRIS					
8434052007458	10136	10136	BAÑADOR COLIBRIS					✓
8434052110592	POR0061	POR0061	PULSERA CLASICA FLE					✓
8434052007434	10148	10148	SHORT GAIA					
8434052007397	10133	10133	CAMISA USHUJÁ MIL R					

1 2 10 filas por página

Borrar filas seleccionadas

Hay 13 registros en la lista (Ocultarlos)

Vista modo detalle

ITL Showroom - Artículos

Buscar Refrescar

Detalle Lista Ambos

Barcode: 8434052010625

Referencia: 20001

Sku: 20001

Descripción: H. SUD PEPE LETRAS

Campolbre 1:

Campolbre 2:

Campolbre 3:

Futuro:

Kit:

Ubicación

Código: a Nombre: zona a

Talla

Cod talla: S Descripción: S

Color

Cod color: AZULULIMA Descripción: AZULULIMA

Componentes

Añadir Quitar seleccionados

Barcode	Referencia	Sku	Descripción	Campolbre 1	Campolbre 2	Campolbre 3	Futl	Kit
---------	------------	-----	-------------	-------------	-------------	-------------	------	-----

Vista modo ambos

Barcode	Referencia	Sku	Descripción	Campolbre 1	Campolbre 2	Campolbre 3	FuB	KID
8434052010625	20001	20001	H. SUD PEPE LETRAS					
8434052010649	20002	20002	H. CAM JESUS BASICA					
87080582	1000	1000	PRUEBA					
23301702	1500	1500	PRUEBA2					
8434052006482	40265	40265	ASHTRAY ESCENARIO					
8434052007472	10136	10136	BAÑADOR COLIBRI SS					
8434052007456	10136	10136	BAÑADOR COLIBRI SS					✓
8434052110592	POR0061	POR0061	PULSERA CLASICA FLEI					✓
8434052007434	10148	10148	SHORT GAJA					
8434052007397	10133	10133	CAMISA USHUJAJA MIL R.					

1.3. ACCIONES CON LOS REGISTROS

Cada vista (lista/detalle) tiene su correspondiente barra horizontal para interactuar con los elementos.

En el caso de la lista las opciones habituales son la de crear nuevo registro “nuevo”, generar un PDF o un Excel con los datos del listado que se tiene en pantalla, o crear un informe personalizado con un filtro.

Además, en las cabeceras de las columnas existen opciones de filtrado según datos. Las columnas se pueden cambiar de tamaño clicando en los bordes verticales en forma de puntos, también se pueden ordenar por cada columna para ello se clica sobre el nombre de la columna y se ordena.

El sistema guarda la configuración de tamaño de las columnas para posteriores visitas a los módulos.

Inicio Artículos Almacenes Configuración Cuadre inventario Inventarios Orden impresión Stock Trazabilidad **Ubicación**

ITL Showroom - Ubicación ☆

Nuevo Generar PDF Generar Excel Mis informes Load ubicaciones Errores

	Código	Nombre
<input type="checkbox"/>	termina en	empieza por
<input type="checkbox"/>	empieza por	
<input type="checkbox"/>	termina en	
<input type="checkbox"/>	contiene	zona a
<input type="checkbox"/>	no contiene	
<input type="checkbox"/>	=	zona b
<input type="checkbox"/>	<>	
<input type="checkbox"/>	>=	zona c
<input type="checkbox"/>	<=	
<input type="checkbox"/>	>	zona f
<input type="checkbox"/>	<	
<input type="checkbox"/>	en grupo	zona g
<input type="checkbox"/>	no en grupo	
<input type="checkbox"/>	rango	

1 2 5 filas por página

Borrar filas seleccionadas

En la parte inferior de la lista existe otra barra para navegar entre las distintas páginas, en caso de haber más de una, así como el número de elementos que se muestra en cada una de ellas (5, 10, 15 ó 20). También se muestran los botones de acciones múltiples, como el borrado de uno o más registros.

1 2 5 filas por página

Borrar filas seleccionadas

5
10
12
15
20

En el caso de la vista modo detalle las acciones mostradas se ciñen a la navegación y edición. Dirigirse al anterior o posterior registro, al primero o el último, así como la opción de crear un nuevo registro, borrar el actual, buscar a través de un filtro o refrescar la ventana.

⏪ ⏩ 🔍 Buscar 🔄 Refrescar

2. MODULOS DEL SISTEMA

2.1. ALMACENES

El módulo de almacenes es donde se pueden visualizar los almacenes que hay en el sistema.

Modo Lista

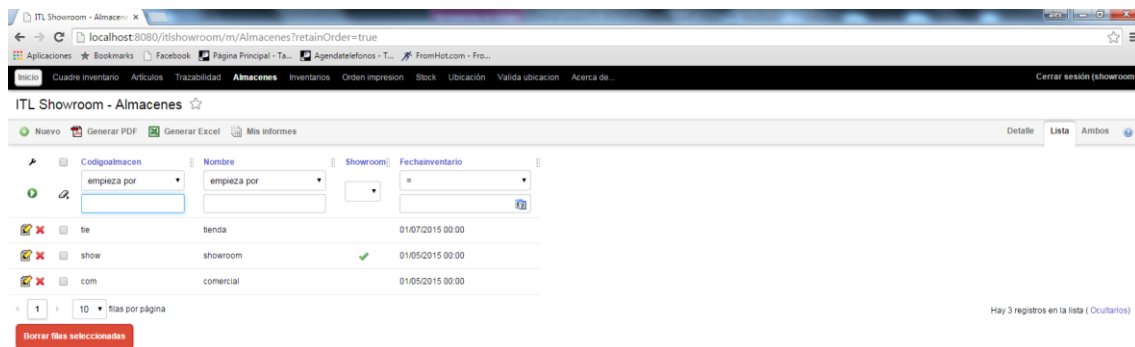
En el modo lista aparece en la barra superior el botón “nuevo” aquí es donde se da de alta un nuevo almacén en el sistema, se ve también los datos relacionados con el almacén (código, nombre, showroom, fechainventario).

El campo Código indica el código o abreviatura del almacén.

El campo Nombre indica el nombre del almacén.

El campo Showroom indica que es el almacén Showroom donde se permite ubicaciones, vemos que solo hay uno marcado ya que no se permite tener marcado más de un showroom.

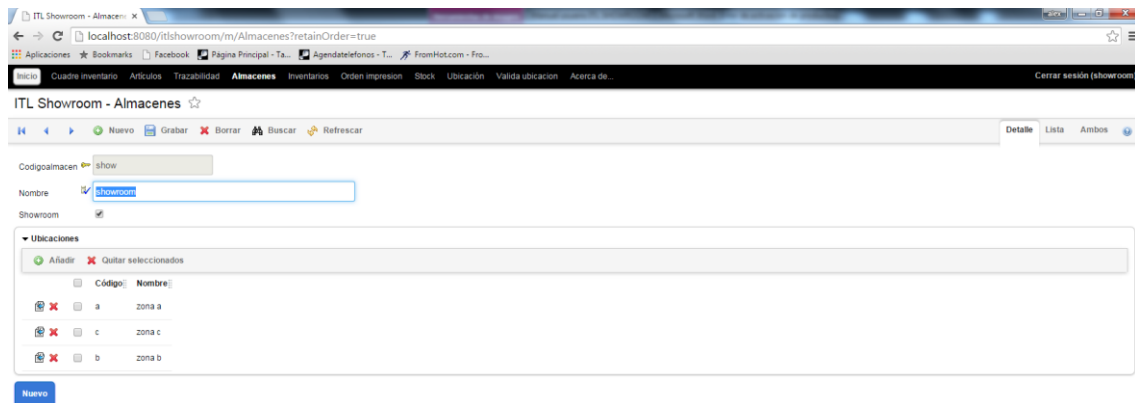
El campo Fecha Inventario indica que la fecha del último inventario realizado y procesado sobre ese almacén.



Modo Detalle

En el modo detalle se ve la opción de guardar que se utilizara para cuando se crea un nuevo almacén poder guardarlo en el sistema.

La marca de tic de Showroom solo podrá estar para un único almacén.



2.2. ARTICULOS

En este módulo se ven todos los artículos que hay en el sistema y que se van a realizar operaciones sobre ellos.

Modo Lista

En la vista modo lista se ven los datos que contiene un artículo (campos):

Barcode, Referencia, Sku, Descripción, Campolibre1, Campolibre2, Campolibre3, Futuro y Kit.

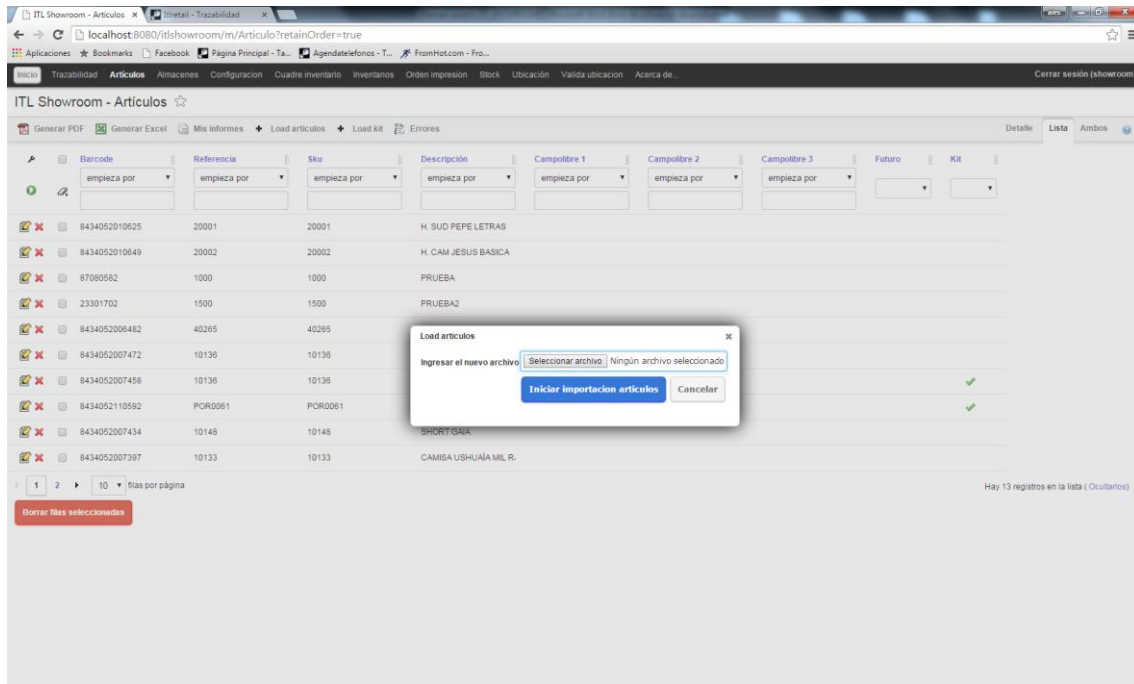
- El campo Barcode: es un campo único que identifica a cada artículo.
- Campo Referencia: es un campo que tiene una longitud de cadena de caracteres 50 donde se tiene la referencia del artículo
- Campo Sku: es un campo de código corto para identificar el artículo que tiene una longitud de cadena de caracteres 20.
- Campo Descripción: tiene una longitud de cadena de caracteres 100, se describe el producto
- Campo Código talla: tiene una longitud de cadena de caracteres 20 donde se tiene el código de la talla de artículo
- Campo Descripción de talla: que tiene una longitud de cadena de caracteres 50 donde se tiene una descripción de la talla de artículo
- Campo Código color: que tiene una longitud de cadena de caracteres 20 donde se tiene el código de color de artículo
- Campo Descripción de color: que tiene una longitud de cadena de caracteres 50 donde se tiene la descripción del color de artículo

- Los campos camposlibre1, camposlibre2 y camposlibre3: son opcionales tienen una longitud de cadena de caracteres 100. Se utilizan para albergar información adicional, necesaria para el cliente, y que no queda registrada con el resto de campos. En caso de no tener valor, debe aparecer el carácter separador de campo. Se puede tener tantos campos libres como se requiera.
- Campo Ubicación: es un campo opcional donde tiene una longitud de cadena de caracteres.
- Campo Artículo futuro: es un campo donde tiene opción de estar a valor verdadero (true) o falso (false).

Los campos Futuro y kit son booleanos, se marcan automáticamente con la importación del fichero de artículos si tienen indicados el campo futuro como verdadero y los kits también se marcan automáticamente al realizar la importación desde el botón de “load kit” con sus valores.

Barcode	Referencia	Sku	Descripción	Campolibre 1	Campolibre 2	Campolibre 3	Futuro	Kit
8434052010625	20001	20001	H. SUD PEPE LETRAS					
8434052010649	20002	20002	H. CAM JESUS BASICA					
87080582	1000	1000	PRUEBA					
23301702	1500	1500	PRUEBA2					
8434052006482	40265	40265	ASHTRAY ESCENARIO					
8434052007472	10136	10136	BAÑADOR COLIBRI SS					
8434052007458	10136	10136	BAÑADOR COLIBRI SS					✓
8434052110592	POR0061	POR0061	PULSERA CLASICA FLEI					✓
8434052007434	10148	10148	SHORT GAIA					
8434052007397	10133	10133	CAMISA USHUJAJA MIL R.					

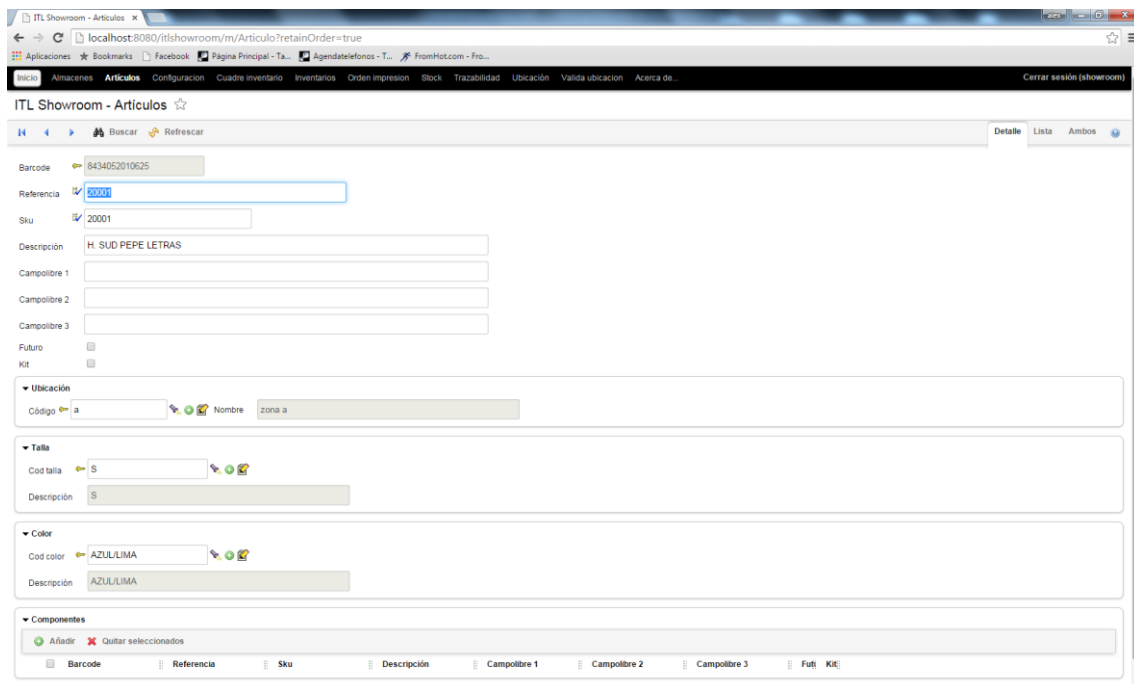
En la barra superior hay dos botones de importación de artículos y de kit, que es donde se darán de alta tanto los artículos como los kits en el sistema, para ello se tiene que seleccionar un archivo e iniciar la importación.



Vista detalle

En ella aparecen la información completa y detallada, lo único que se puede realizar en esta vista es el movimiento hacia adelante, hacia atrás y al inicio de los artículos para su visualización.

En el campo componentes se visualizaran los artículos que forman un kit en caso de que ese artículo sea un kit.



2.2.1. IMPORTACIÓN DE ARTÍCULOS

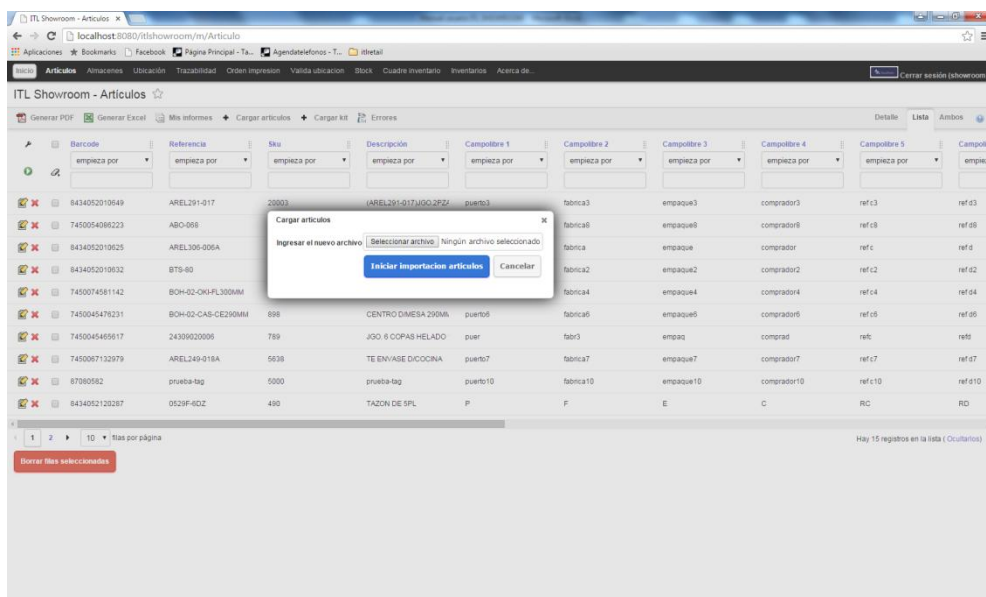
La importación de artículos no corresponde a un módulo como tal, se puede realizar por el botón de importación de artículos o por la tarea automáticamente.

TAREA

Hay una tarea de importación de artículos que se mantiene a la espera, y que comprueba el ‘ESTADO’ de los artículos y cuando lo comprueba que hay artículos pendientes los recoge de la base de datos y los inserta en el sistema. Para que se inserte en el sistema el ‘ESTADO’ debe ser ‘P’.

MÓDULO ARTÍCULOS

La importación de artículos se realiza mediante la utilización del Módulo Artículos pulsando el botón “cargar artículos” aparece una ventana indicando la posibilidad de seleccionar el fichero de artículos como se aprecia en la imagen siguiente:



2.2.2. ESTRUCTURA DE FICHERO

- Fichero txt/csv con 13 campos delimitados por el carácter “|” a excepción del últimocampo.
- Codificación UTF-8
- El carácter delimitador no puede aparecer en ningún valor de campo.
- El orden de los campos es el siguiente.

- El carácter delimitador no puede aparecer en ningún valor de campo.
- El orden de los campos es el siguiente.

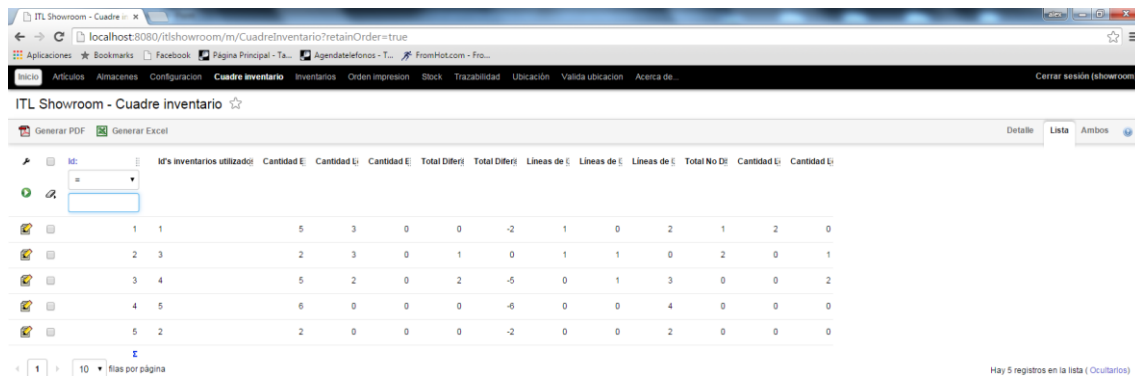
Ejemplo:

```
8434052118542|8434052006963;8434052007741;8434052007854
8434052007397|8434052007434;8434052007434
8434052110592|8434052006482;8434052007472;8434052007519
8434052118556|8434052006979;8434052007975;8434052007691;8434052007324
```

2.3. CUADRE INVENTARIO

A partir de un inventario, podemos generar un cuadro con las cantidades leídas y comparar con las cantidades que nos esperamos. Para acceder a esta operación lo realizamos desde el módulo de inventarios y la acción Cuadre (previa selección del inventario que queremos contemplar en el cuadro)

Tras generar el cuadro, iremos al módulo Cuadre. En la vista modo de lista, nos aparecerán todos los cuadros creados en el sistema



Vista modo detalle

ITL Showroom - Cuadre inventario

Cargar stock

Detalle Lista Ambos

Cuadre inventario

Id: 1 Id's inventarios utilizados: 1

Cantidad Esperada: 5 Cantidad Leída: 3 Cantidad Esperada Negativa (no se contempla en la diferencia): 0

Total No Diferencias: 1 Líneas de Cuadre con Diferencia=0: 1

Cantidad Leída con Diferencia Positiva: 0 Total Diferencias Positivas: 0 Líneas de Cuadre con Diferencia Positiva: 0

Cantidad Leída con Diferencia Negativa: 2 Total Diferencias Negativas: -2 Líneas de Cuadre con Diferencia Negativa: 2

Lineas cuadro

Añadir Quitar seleccionados Generar PDF Generar Excel

Ean	Sku	Alu	Cantidad e	Cantidad R	Diferencia R
empieza por	empieza por	empieza por	=	=	=
9999000113791	BUDMRCH06	BUDMRCH06	2	1	-1
9999000113807	BUDFD3H02	BUDFD3H02	2	1	-1
9999000113784	BUDMRCH04	BUDMRCH04	1	1	0

Hay 3 registros en la lista

Un cuadro está compuesto por una cabecera y líneas de cuadro.

La cabecera dispone de los siguientes registros:

- Id inventario: contiene el id del inventario utilizado para generar el cuadro.
- Total esperada: corresponde a la suma de las cantidades positivas cargadas en el inventario teórico.
- Total no diferencias: corresponde con el total de líneas de cuadro, cuya diferencia es cero.
- Total leídas con diferencia positiva: corresponde a aquellas líneas de cuadro que se registraron lectura en los inventarios (cantidad leída>0), para las que se leyó de más con respecto al inventario teórico.
- Total leídas con diferencia negativa: corresponde a aquellas líneas de cuadro que se leyeron en los inventarios (cantidad leída>0), para las que su diferencia es negativa. Es decir, aquellas lecturas en las que se leyó de menos con respecto al inventario teórico.
- Total leída: corresponde a la suma de las cantidades leídas entre los dos inventarios utilizados para generar el cuadro.
- Total líneas no diferencias: entendiendo como diferencia = cantidad leída – cantidad esperada, este campo almacena la cantidad de líneas de cuadro en las que diferencia tiene valor 0, es decir, aquellos artículos cuya stock teórico coincide con el real.

- Total diferencia positivos: entendiendo como diferencia = cantidad leída- cantidad esperada, este campo almacena la suma de aquellas diferencias con valor positivo. Es decir, aquellos artículos para los cuales se han registrado lecturas que no se esperaban en el inventario teórico.
- Total diferencia negativas: entendiendo como diferencia = cantidad leída- cantidad esperada, este campo almacena la suma de aquellas diferencias con valor negativo, es decir, aquellos artículos para los se han registrado menos lecturas de las que se esperaban en el inventario teórico.
- Total negativos: corresponde a la suma de las cantidades negativas cargadas en el inventario teórico, es decir, la suma de cantidad esperada de las líneas de cuadro con cantidad esperada<0.
- Total líneas positivas: corresponde a la cantidad total de líneas de cuadro con cantidad esperada positiva.
- Total líneas diferencias negativas: corresponde con el total de líneas de cuadro cuya diferencia es negativa.
- Total líneas negativas: corresponde con el total de líneas de cuadro cuya cantidad esperada es negativa.

Una línea de cuadro dispone de:

- Ean, Sku y Alu.
- Cantidad esperada: inicialmente a cero. Cuando se carga el fichero de inventario teórico, en este campo se almacena la cantidad indicada en dicho fichero.
- Cantidad leída: almacena la cantidad leída en los inventarios para ese ean.
- Diferencia: indica la diferencia entre la cantidad leída y la esperada. Si es cero, el stock leído coincide con el stock teórico. Si es negativa, indica que no se leyeron en el inventario la cantidad que se esperaba. Si es positiva, indica que se leyó más cantidad de la que se esperaba tener.

En la vista destalle disponemos de dos acciones:

- Carga fichero inventario teórico: para cargar un fichero con el inventario teórico de su sistema de gestión. El fichero deberá seguir la estructura EAN|CANTIDAD. No podrán haber repetidos. Cuando finalice la carga del fichero, tendremos que cambiar a la vista de lista, y hacer clic sobre el cuadro correspondiente, para que se carguen en pantalla los cambios introducidos en el cuadro.

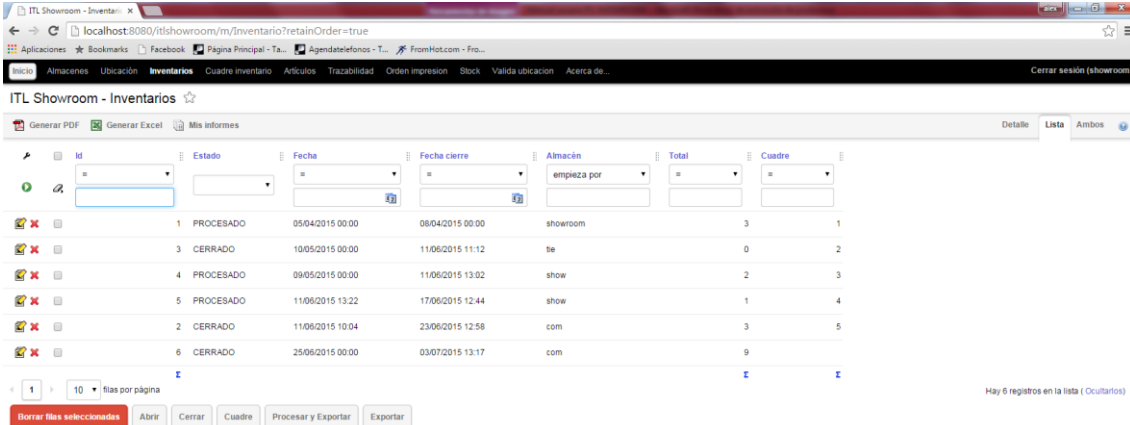
2.4. INVENTARIOS

Los inventarios se crean mediante la PDA, luego se trabaja sobre ellos en la aplicación web en el modulo inventario.

Desde este módulo podemos visualizar los inventarios creados, el estado en el que se encuentran, cuando se crearon y se cerraron los inventarios, en el almacén que se realizó, el total de lecturas realizado durante dicho inventario con la PDA, también se indica el cuadro que está asociado a el inventario y se tiene disponible una serie de acciones que se aplican sobre él.

En vista modo lista:

- **Abrir:** para cambiar a estado abierto un inventario en estado cerrado. En este estado podemos añadir líneas al inventario mediante la lectura de artículos en la PDA.
- **Cerrar:** un inventario en este estado no permite añadir más artículos al mismo.
- **Procesar y Exportar:** mediante esta acción se genera la trazabilidad asociada y un fichero con el contenido del inventario.
- **Exportar:** para generar de nuevo el fichero que contiene los artículos inventariados.
- **Borrar filas seleccionadas:** para borrar los inventarios seleccionados. Tan solo se pueden borrar inventarios en estado abierto, es decir, un inventario en estado cerrado o procesado no podrá ser borrado. Si queremos borrar un inventario en estado cerrado, previamente deberíamos cambiar su estado a abierto. En el caso de un inventario procesado, éste no podría ser borrado ya que contiene una trazabilidad asociada.

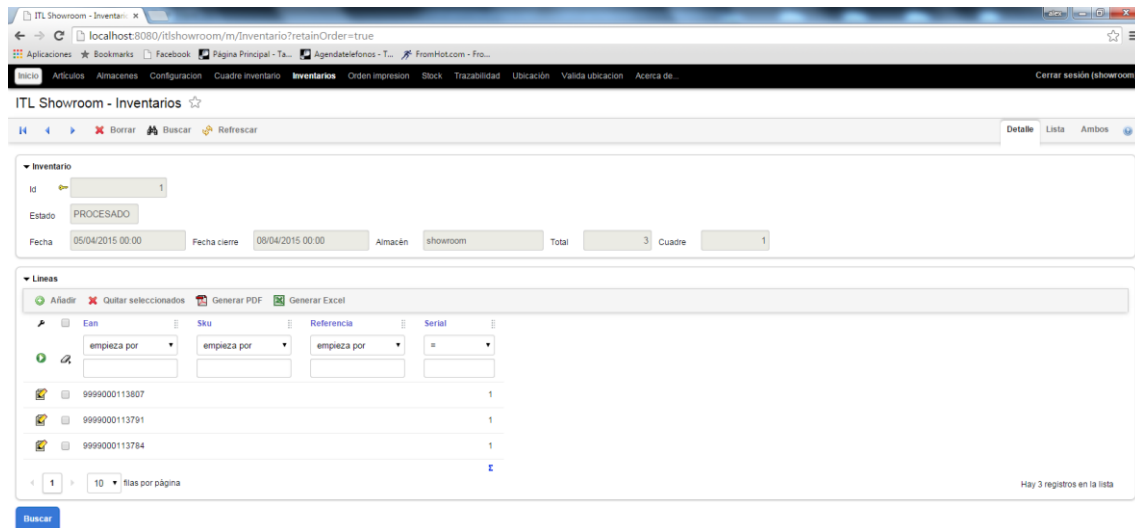


The screenshot shows the 'ITL Showroom - Inventarios' web application. The interface includes a navigation menu with options like 'Almacenes', 'Utilización', 'Inventarios', 'Cuadro inventario', 'Artículos', 'Trazabilidad', 'Orden impresión', 'Stock', 'Valida ubicación', and 'Acercas de...'. Below the menu, there are filters for 'Id', 'Estado', 'Fecha', 'Fecha cierre', 'Almacén', 'Total', and 'Cuadro'. The main area displays a table with 6 rows of inventory data. At the bottom, there are buttons for 'Borrar filas seleccionadas', 'Abrir', 'Cerrar', 'Cuadro', 'Procesar y Exportar', and 'Exportar'. A status message at the bottom right indicates 'Hay 6 registros en la lista (Ocultarlos)'.

	Id	Estado	Fecha	Fecha cierre	Almacén	Total	Cuadro
	1	PROCESADO	05/04/2015 00:00	06/04/2015 00:00	showroom	3	1
	3	CERRADO	10/05/2015 00:00	11/06/2015 11:12	te	0	2
	4	PROCESADO	09/05/2015 00:00	11/06/2015 13:02	show	2	3
	5	PROCESADO	11/06/2015 13:22	17/06/2015 12:44	show	1	4
	2	CERRADO	11/06/2015 10:04	23/06/2015 12:58	com	3	5
	6	CERRADO	25/06/2015 00:00	03/07/2015 13:17	com	9	

En vista modo detalle:

- **Borrar:** para borrar un inventario. Tener en cuenta las mismas consideraciones que en ‘Borrar filas seleccionadas’.
- **Añadir líneas:** no disponible, porque las líneas deben añadirse desde la PDA.
- **Borrar líneas:** sólo se pueden borrar líneas en un inventario en estado abierto.



2.5. INVENTARIO PARCIAL

Los inventarios parciales se crean mediante la PDA, luego se trabaja sobre ellos en la aplicación web en el modulo inventario.

Desde este módulo podemos visualizar los inventarios parciales creados, el estado en el que se encuentran, cuando se crearon y se cerraron los inventarios, en el almacén (ubicación) que se realizó, el total de lecturas realizado durante dicho inventario con la PDA, también se indica el cuadro que está asociado a el inventario y se tiene disponible una seria de acciones que se aplican sobre él.

En vista MODO LISTA:

- **Abrir:** para cambiar a estado abierto un inventario en estado cerrado. En este estado podemos añadir líneas al inventario mediante la lectura de artículos en la PDA.
- **Cerrar:** un inventario en este estado no permite añadir más artículos al mismo.

- Procesar y Exportar: mediante esta acción se genera la trazabilidad asociada y un fichero con el contenido del inventario.

ITL Retail - Inventarios Parciales

Generar PDF Generar Excel

Detalle Lista Ambos

Id	Sucursal	Estado	Ubicación	Fecha	Fecha de Cierre	total	Descripción Sucursal
1	T0001	ABIERTO	T0001TND	09/09/2015 17:15		16	Tienda test 2
3	T001	ABIERTO	TIENDA	28/09/2015 00:00			
2	T0001	ABIERTO	TIENDA	17/09/2015 15:55		1	Tienda test 2
4	T0001	ABIERTO	TIENDA	29/09/2015 18:14		0	Tienda test 2
5	T0001	ABIERTO	TIENDA	29/09/2015 17:35		4	Tienda test 2
6	T0001	ABIERTO	TIENDA	29/09/2015 17:48		5	Tienda test 2
7	T0001	ABIERTO	TIENDA	29/09/2015 17:50		5	Tienda test 2
8	T0001	PROCESAR	TIENDA	29/09/2015 18:00	30/09/2015 14:00	4	Tienda test 2
9	T0001	PROCESAR	TIENDA	29/09/2015 18:11	30/09/2015 14:03	6	Tienda test 2
10	T0001	PROCESAR	TIENDA	30/09/2015 15:18	30/09/2015 15:20	3	Tienda test 2

Hay 23 registros en la lista (Ocultarlos)

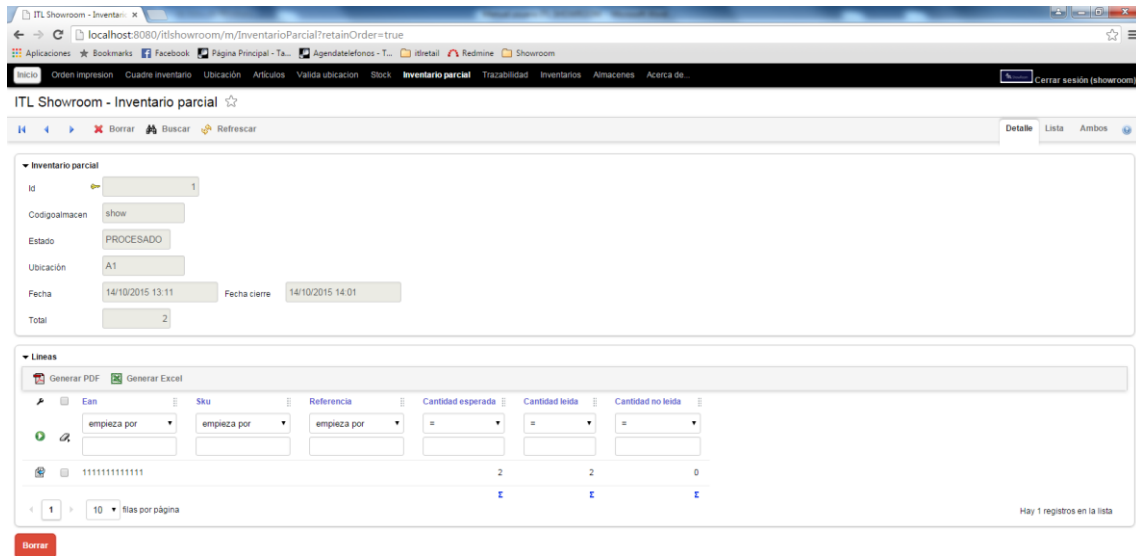
1 2 3 10 filas por página

Abrir Cerrar Procesar

En vista MODO DETALLE:

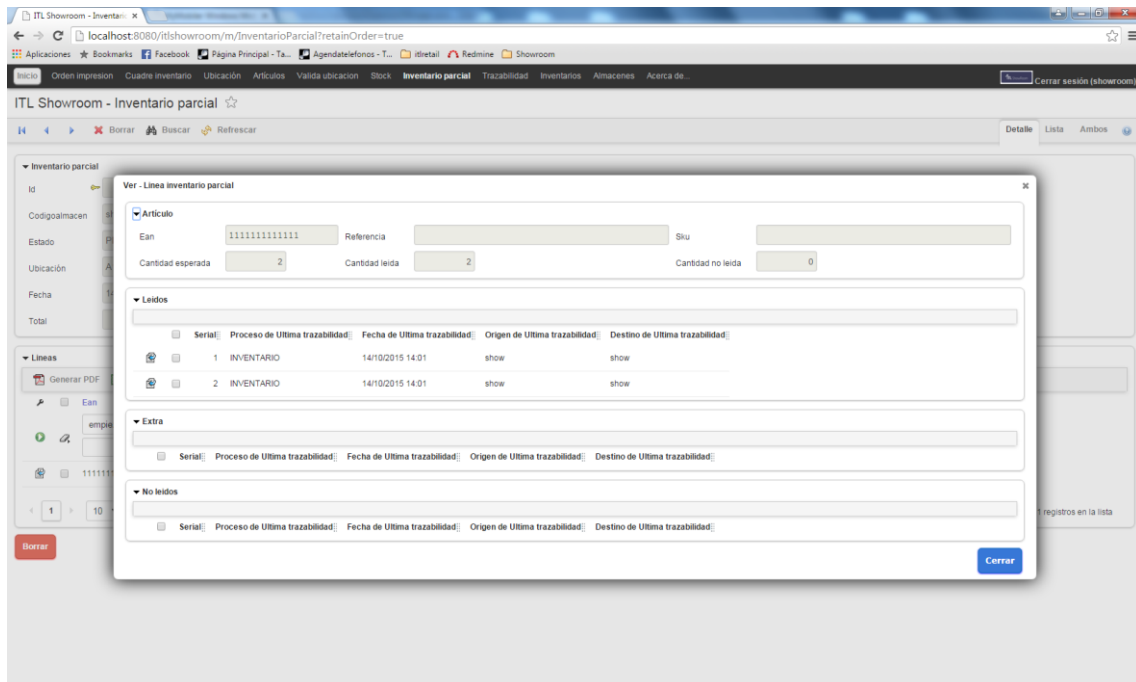
- No se tiene ningún botón para realizar búsquedas, añadir o quitar líneas. Se puede generar un PDF o un Excel del inventario parcial, también se puede realizar filtros con los campos que se quiera.
- En esta vista se visualiza el detalle de cada línea del inventario parcial que se componen por la selección de Referencia, Sku, ean o familia que se ha seleccionado desde la PDA. Aquí se muestra la cantidad que hay en stock columna “Stock”, la cantidad de artículos leídos de la selección (Referencia,Sku, ean o familia) que se haya realizado en la PDA columna “Leídos” y la cantidad de artículos no leídos y que se esperaba leer columna “No Leídos”.

Modo detalle



DETALLE LINEA INVENTARIO PARCIAL

En esta vista se ven los datos de los campos de Artículos, Leídos, Extra y No Leídos, del artículo seleccionado en el modo detalle del inventario parcial, la imagen siguiente se puede ver un ejemplo de cómo se visualiza dichos campos.



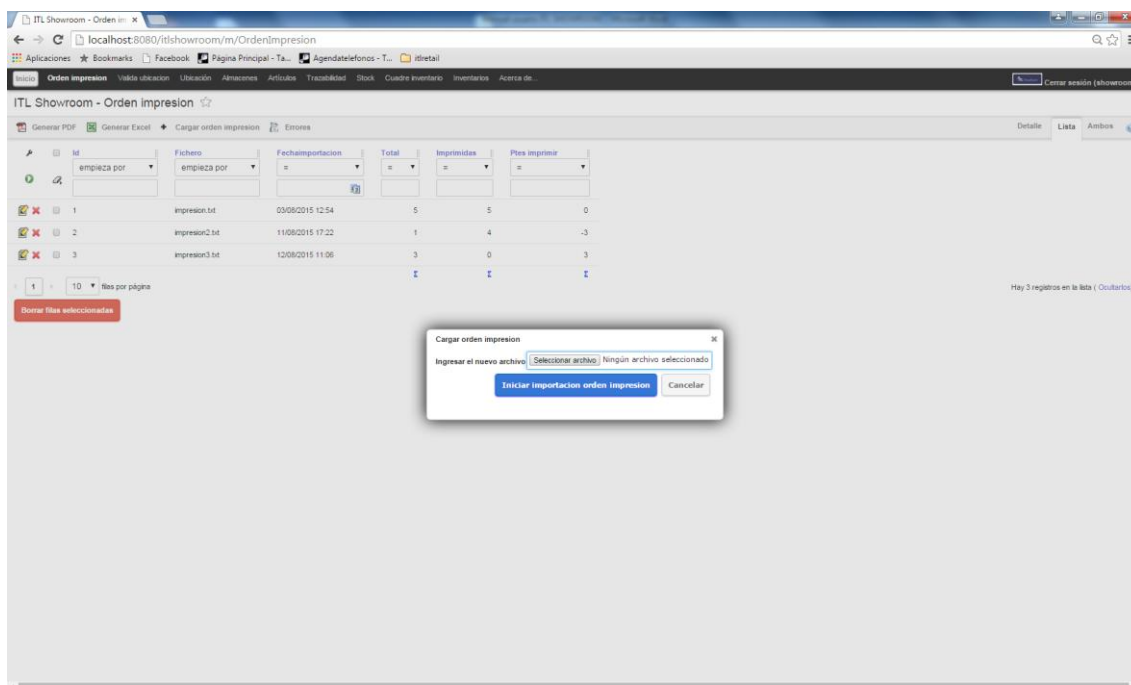
2.6. ÓRDEN IMPRESIÓN

En este módulo están los botones de “Generar PDF”, “Generar Excel”, “Errores” y “cargar orden impresión”. Los dos primeros son para generar documentos PDF y Excel, mientras que el botón de “Errores” esta para cuando al importar un archivo de orden de impresión que no esté correcto quede reflejado en un archivo y se puede visualizar pulsando en el botón “Errores” para la descarga del archivo. El botón de “cargar orden de impresión” es donde se tiene que pulsar para poder cargar la orden de impresión en el sistema, en la parte de importación de órdenes de impresión a continuación comentada queda explicado cómo realizar la importación.

Hay que tener en cuenta que a veces no descarga el archivo, cuando sucede esto hay que comprobar en el navegador si hay algún pop-ups activado que este bloqueando la descarga.

2.6.1. IMPORTACIÓN ÓRDENES DE IMPRESIÓN

La orden de impresión se carga en el sistema desde el modulo “Orden de impresión“ se tiene que pulsar en el botón “cargar orden impresión” y seleccionar el fichero que se donde se encuentra la orden de impresión que se quiera imprimir.



2.6.2. ESTRUCTURA DE FICHERO

- Fichero txt/csv con 3 campos delimitados por el carácter “|” a excepción del último campo.
- Codificación UTF-8
- El carácter delimitador no puede aparecer en ningún valor de campo.
- El orden de los campos es el siguiente.

Ejemplo:

```
2|8434052110592|2
2|8434052007755|1
2|8434052110873|1
2|8434052007412|3
```

Vista en moda lista

Se visualiza los datos del identificador de la orden de impresión, el nombre del fichero que se importa, también la fecha y hora de la importación del fichero, el total indica el número total de etiquetas que a imprimir, en la columna de imprimidas aparece el número de etiquetas imprimidas para esa orden de impresión y en la columna “Ptes imprimir” aparecen valores negativos positivos o cero según se haya realizado la impresión, si se ha imprimido una etiqueta más de una vez y están todas las demás imprimidas aparece un valor negativo.



The screenshot shows a web application interface for 'ITL Showroom - Orden impresión'. It features a table with columns for 'ID', 'Fichero', 'FechaImportación', 'Total', 'Imprimidas', and 'Ptes Impri'. The table contains four rows of data:

ID	Fichero	FechaImportación	Total	Imprimidas	Ptes Impri
23	impresion.txt	02/06/2015 13:40	4	2	2
1	impresion.txt	02/06/2015 13:53	4	1	3
2	impresion2.txt	02/06/2015 13:55	3	0	3
5	prueba_impresion3.txt	01/07/2015 09:45	3	0	3
6	prueba_impresion4.txt	02/07/2015 13:38	21	0	21

Vista en modo detalle

En esta vista se visualiza la información que contiene una orden de impresión más detalladamente, los datos son los siguientes:

Clave, ean, serial, ordendeimpresion, numdevecesimpreso, sku, referencia, descripción, talla, color.

The screenshot displays the 'Orden impresión' detail view in the ITL Showroom application. The browser address bar shows the URL: `localhost:8080/itlshowroom/m/OrdenImpresion?retainOrder=true`. The page title is 'ITL Showroom - Orden impresión'. The interface includes a navigation menu with options like 'Cuadro inventario', 'Artículos', 'Almacenes', 'Ubicación', 'Inventarios', 'Trazabilidad', 'Orden impresión', 'Stock', 'Valida ubicación', and 'Cerrar sesión (showroom)'. The main content area is divided into two sections: 'Orden' and 'Lineas'.

Orden Summary:

- Id: 23
- Fichero: impresion.txt
- Fechaimportacion: 02/06/2015 13:43
- Total: 4
- Imprimidas: 2
- Ples imprimir: 2

Lineas Table:

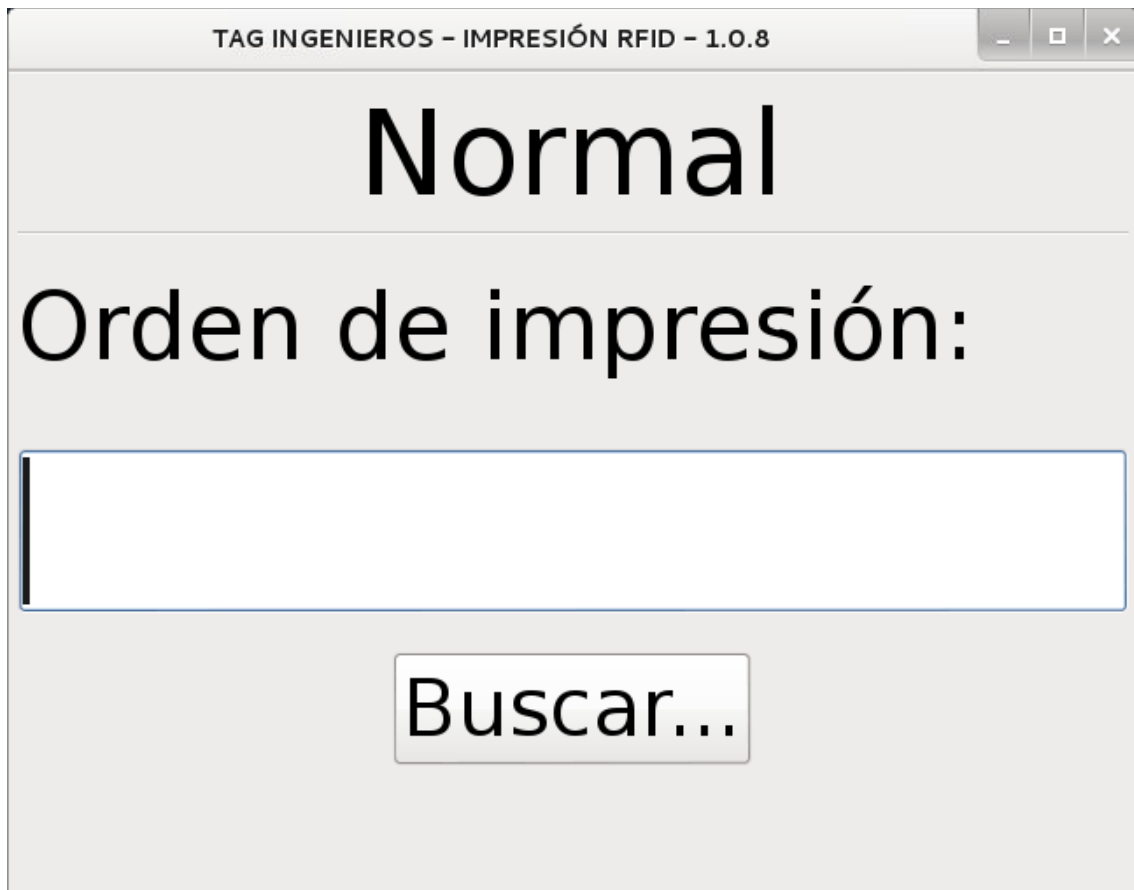
Clave	Ean	Serial	Ordendeimpresion	Numvecesimpreso	Sku	Referencia	Descripción	Talla	Color	
empieza por	empieza por									
843405200743411	8434052007434		1	1	1	10148	10148	SHORT GAIA	M	ROSA
843405200743412	8434052007434		2	2	1	10148	10148	SHORT GAIA	M	ROSA
843405200743413	8434052007434		3	3	0	10148	10148	SHORT GAIA	M	ROSA
843405200743414	8434052007434		4	4	0	10148	10148	SHORT GAIA	M	ROSA

At the bottom of the table, there is a pagination control showing '1' of 10 items per page and a note 'Hay 4 registros en la lista'.

2.7. IMPRESIÓN DE ÓRDENES

Una vez las órdenes han sido importadas, pueden ser lanzadas para su impresión desde una aplicación específica. El acceso directo se encuentra en el escritorio del servidor, bajo el nombre de “Impresión”.

La aplicación tan solo está formada por una vista sencilla, como la siguiente:



En el campo de orden de impresión se debe escribir el código de la orden importada anteriormente. Cuando se clic el botón “buscar” se cargan las etiquetas a imprimir, previamente si la empresa es multipaís (tiene más de una tarifa) se muestra una ventana de selección de país.

En caso de que se haya impreso alguna etiqueta de la orden, la aplicación pregunta si tan solo se quieren imprimir las etiquetas pendientes o toda la orden. Si se hubiera impreso todas las etiquetas también avisa al usuario, preguntando si se quiere reimprimir la orden completa.

Una vez se acepta, aparece una ventana de impresión que va mostrando el transcurso del proceso. Aparecen los datos de la etiqueta, una a una, y las opciones para cancelar el proceso por donde se ha quedado, pausar la impresión o retomar.

En caso de que la impresora informe de algún error (falta de etiquetas, de ribbon o problema de comunicación) el proceso queda pausado y el aviso se queda en la ventana con la opción de reanudar la impresión cuando se haya resuelto.

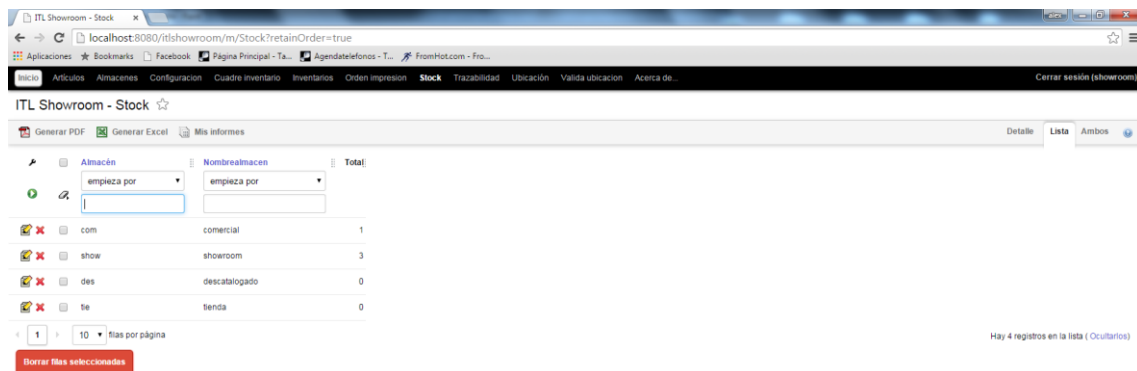
2.8. STOCK

Se trata de un módulo de cálculo, mediante el cual podemos obtener el stock TOTAL, a partir de la trazabilidad registrada en el sistema.

Necesita partir de un inventario de almacén que sea procesado, a partir de ahí ese inventario pasa a ser el stock del sistema y se irá modificando el stock con cada movimiento, venta o alta en el sistema de RFID.

Vista modo lista solo están los botones de generar pdf, generar Excel y mis informes.

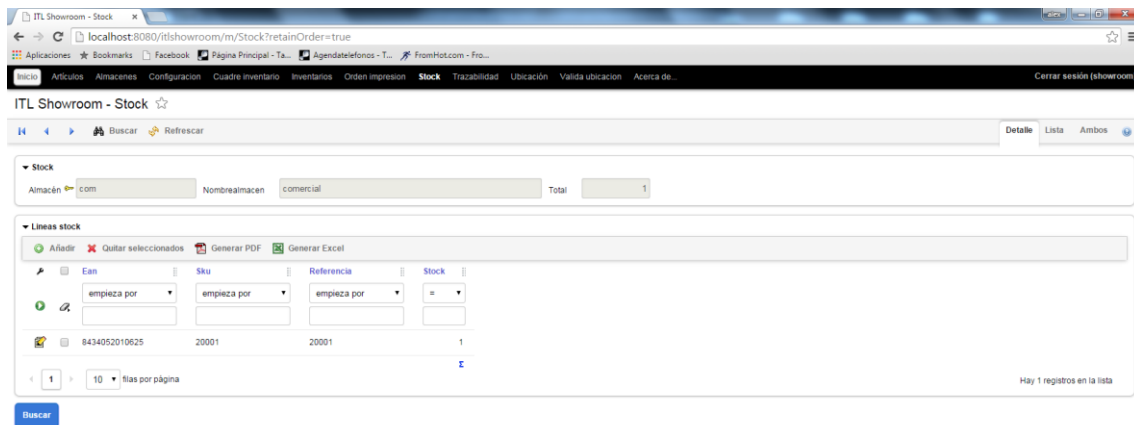
En esta vista se ve el Stock total que se tiene en cada almacén que hay registrado en el sistema.



Vista modo detalle

Aquí se ven los datos de stock que hay en un almacén que se ha seleccionado en la vista modo lista.

Los datos son los relativos a los ean, sku, referencia y stock, de cada línea de stock que hay en el almacén, el almacén es el que aparece en la ventana de stock en la parte de arriba.



2.9. TRAZABILIDAD

En este módulo podemos visualizar toda la trazabilidad registrada en el sistema para la tienda en cuestión. Se trata de un módulo de sólo lectura, por lo tanto, las acciones disponibles serán

“Generar PDF” y “Generar Excel”

Campos:

- Fecha: momento en el cuál se registra el proceso de trazabilidad.
- Proceso: tipo de proceso de trazabilidad. Puede tomar los siguiente valores **ALTA** (cuando se genera el fichero de cajas), **IMPRESIÓN** (cuando se realiza el proceso de impresión de un tag), **INVENTARIO** (cuando el tag se encuentra en un inventario que ha sidoprosesado), **MOV_ALMACEN**(cuando el tag se ha movido de un almacén a otro),**MOV_UBICACION**(cuando el tag se ha movido de una ubicación a otra),**BAJA** (cuando el tag está dado de baja en el sistema).
- Ean, Sku, Referencia, Serial: son datos que hacen mención a un tag concreto.

Vista modo lista

Se ve todos los datos relacionados con cada trazabilidad que hay en el sistema.

Fecha	Proceso	Ean	Sku	Referencia	Origen	Destino
11/06/2015 00:00	MOV ALMACEN	8434052007472	10136	10136	show	show
17/06/2015 12:44	INVENTARIO	8434052007519	10150	10150	show	te
03/07/2015 00:00	MOV ALMACEN	8434052010625	20001	20001	show	com

Vista modo detalle

Se visualiza el detalle del proceso que se ha seleccionado en el modo lista.

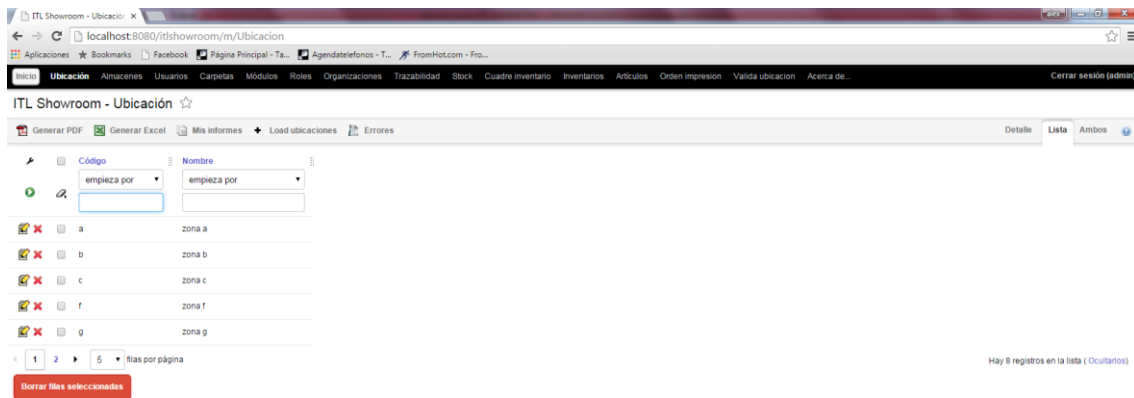
Id	1
Fecha	11/06/2015 00:00
Proceso	MOV ALMACEN
Ean	8434052007472
Serial	1
Origen	show
Destino	show
Stock procesado	<input checked="" type="checkbox"/>
Referencia	10136
Sku	10136

2.10. UBICACIÓN

En este módulo se ven las ubicaciones que están en el sistema.

En la vista modo lista aparecen el código y el nombre de cada ubicación.

En la barra superior hay un botón de importación de ubicación donde se tiene que seleccionar un archivo que estarán las ubicaciones que se quieren importar con los artículos que contendrá cada ubicación, esta es la opción de dar de alta las ubicaciones.



2.10.1. IMPORTACIÓN DE UBICACIÓN

En el fichero contiene los datos de la ubicación con los artículos que hay en cada una y el almacén que esta dicha ubicación.

2.10.2. ESTRUCTURA DE FICHERO

- Fichero txt/csv con 3 campos delimitados por el carácter “|” a excepción del último campo.
- El ultimo campo está compuesto por los artículos que están en la ubicación, se delimitan entre ellos con “;”.
- Codificación UTF-8
- El carácter delimitador no puede aparecer en ningún valor de campo.
- El orden de los campos es el siguiente.

Ejemplo:

```

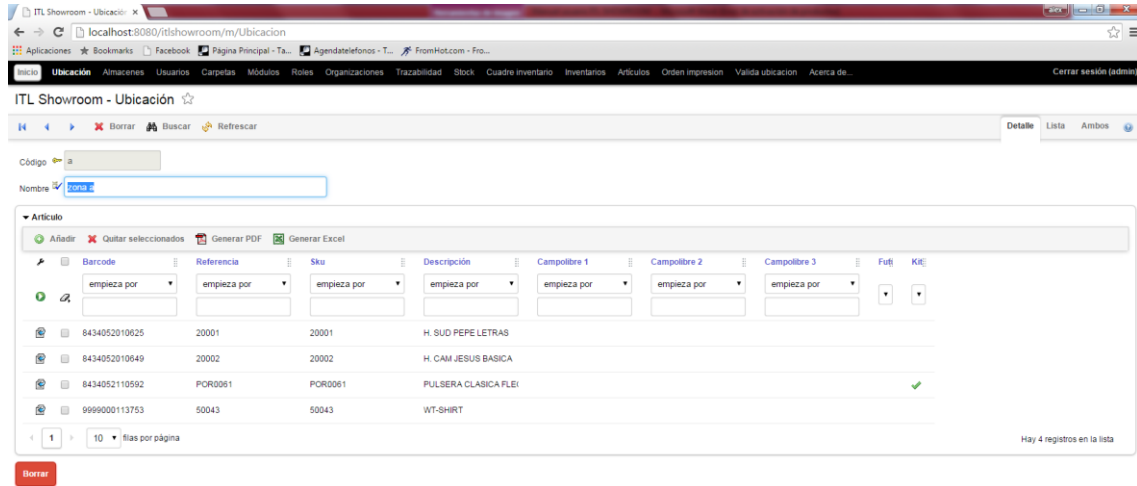
a | zona a | 8434052007741; 8434052007397
b | zona b | 8434052007753; 8434052010368
c | zona c | 8434052007434; 8434052009865
d | zona d | 8434052007519; 8434052010632

```

Vista modo detalle

Aquí se ve en cada ubicación los artículos que tiene relacionados.

En la barra superior se puede recorrer las ubicaciones hacia delante o hacia atrás. En el campo de Artículos los botones de “Añadir” y “Quitar seleccionados” están deshabilitados porque no se puede añadir artículos o quitar los desde este módulo.



2.11. VALIDA UBICACIÓN

Es un modulo donde se validan los artículos que se encuentran en las ubicaciones seleccionadas para dicha validación.

Modo Lista

Aquí están las validaciones realizadas de las ubicaciones, en el estado que se encuentran, la fecha que se empezaron a realizar y la que se cerraron y también el total de ubicaciones elegidas para validar.

Los estados son los mismos que los mencionados en el inventario anteriormente.

ITL Showroom - Valida ubicación

Generar PDF | Generar Excel | Mis informes

Detalle | Lista | Ambos

	Almacén	Estado	Fecha	Fecha cierre	Total ubicaciones
1	show	PROCESADO	23/06/2015 00:00	23/06/2015 13:42	2
2	show	CERRADO	25/06/2015 00:00	26/06/2015 13:21	1
13	show	ABIERTO	01/07/2015 16:03		3
12	show	CERRADO	01/07/2015 16:01	09/07/2015 11:55	3
3	show	CERRADO	26/06/2015 00:00	09/07/2015 13:50	3

Hay 5 registros en la lista (Ocultarlos)

10 filas por página

Buscar las seleccionadas | Abrir | Cerrar | Procesar y Exportar | Exportar

Modo Detalle

En esta vista se visualiza los datos que se tienen al realizar una validación de una ubicación seleccionada en la PDA. En la vista se tiene los datos del código de la ubicación que has realizado la validación el nombre y el total de artículos que se leyeron en la ubicación.

ITL Showroom - Valida ubicación

Estado: CERRADO | Almacén: show

Fecha: 13/08/2015 10:42 | Fecha cierre: 13/08/2015 12:47

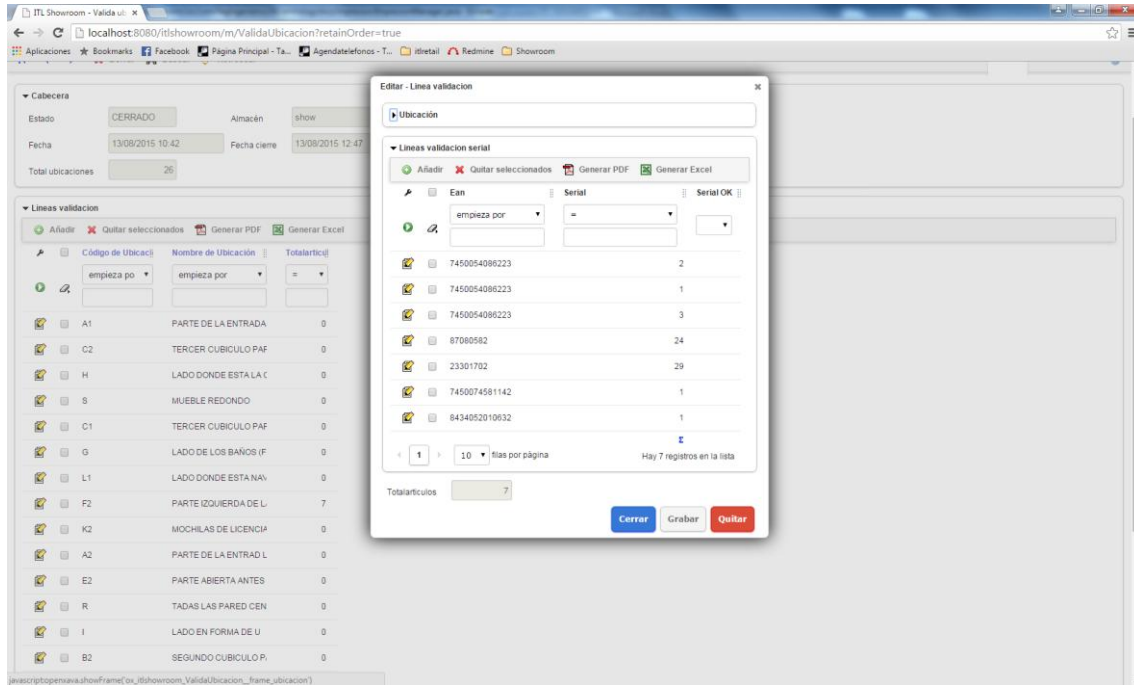
Total ubicaciones: 26

Lineas validacion

Alfadir | Quitar seleccionados | Generar PDF | Generar Excel

Código de Ubicaci	Nombre de Ubicación	TotalArtic
A1	PARTE DE LA ENTRADA	0
C2	TERCER CUBICULO PAF	0
H	LADO DONDE ESTA LA C	0
S	MUEBLE REDONDO	0
C1	TERCER CUBICULO PAF	0
G	LADO DE LOS BANOS (F	0
L1	LADO DONDE ESTANAV	0
F2	PARTE IZQUIERDA DE L	7
K2	MOCHILAS DE LICENCIA	0
A2	PARTE DE LA ENTRAD L	0
E2	PARTE ABIERTA ANTES	0
R	TADAS LAS PARED CEN	0
I	LADO EN FORMA DE U	0
B2	SEGUNDO CUBICULO P.	0
K1	LADO DERECHA DONDE	0

Se puede pulsar sobre una ubicación y se accederá a ver una ventana donde se visualizarán todos los artículos que hay y con un campo de verificación para ver si el serial del artículo es correcto en la ubicación o no.





Confidencialidad

Esta propuesta contiene información confidencial que pertenece a **TAG INGENIEROS CONSULTORES, S.L.** Esta información se entrega únicamente para permitir al destinatario poder valorar la oferta.

El destinatario no podrá difundir a terceros sin el consentimiento explícito y por escrito de TAG Ingenieros Consultores, S.L. todo o parte del presente documento.

Derechos de Propiedad Intelectual e Industrial

©2016, TAG Ingenieros Consultores, S.L. Todos los derechos reservados.



UPV

Empresa asociada a:

 INNOVALL

 estic

 ADL

 aitex®

 COE

 ITENS

 tag ingenieros
tecnología RFID

TAG INGENIEROS CONSULTORES S.L
Poligono industrial "El Plá"
C/De la trama, 5-C 1º, 1ª
46870 ONTINYENT. Apdo. Correos 513
Tel-96-238 95 85 Fax-96 291 50 12
www.tagingenieros.com

BIBLIOGRAFÍA

ABC. (2016). *Definición ABC: Tu diccionario hecho fácil*. Recuperado el 3 Mayo del 2016, desde:

<http://www.definicionabc.com/economia/inventario.php>

Equipo OpenXava. (2005). *OpenXava*. Recuperado el 5 Mayo del 2016, desde:

<http://www.openxava.org>

Paniza, J. (2011). *Aprende OpenXava con ejemplos*. (1.1 ed.)

Syed A. Ahson y Mohammed Ilyas. (2008). *RFID Handbook: Applications, Technology, Security and Privacy*.

TAG Ingenieros consultores S.L. (2011). *TAG Ingenieros: Tecnología RFID*. Recuperado el 25 Marzo del 2016, desde:

<http://www.tagingenieros.com/>

Nordic ID. (2004). *Nordic ID: RETAIL*. Recuperado el 6 Mayo del 2016, desde:

<http://www.nordicid.com/en/>