ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

"Sistema para Administrar una bodega, orientado a la organización del espacio físico, utilizando plataforma Java y soporte de dispositivo PDA"

TÉSIS DE GRADO

Previa a la obtención del Título de:
INGENIERO EN COMPUTACIÓN

ESPECIALIZACIONES:
SISTEMAS DE INFORMACIÓN Y SISTEMAS TECNOLÓGICOS

Presentado por:

José Alfredo Córdova Larrea
Juan Pompilio Moreno Velasco

GUAYAQUIL – ECUADOR

2005

AGRADECIMIENTO

A Dios por darme la perseverancia necesaria para llegar al término de mi carrera.

A mi abuelita por estar presente en todos los momentos difíciles y darme el apoyo incondicional.

A mi enamorada Verónica Álvarez, por estar a mi lado en todo el proceso de graduación.

Juan Pompilio Moreno Velasco

A Dios por poner en mi camino a tantas personas, quienes han contribuido de forma importante a este trabajo y a mi vida.

A mis padres y hermanos que siempre han estado apoyándome en el desarrollo de este proyecto.

José Alfredo Córdova Larrea

DEDICATORIA

A mis padres,

A mis hermanos.

José Córdova

A la memoria de mi madre,

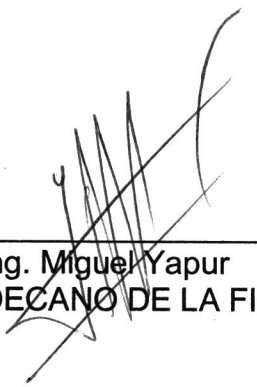
Sra. Angela Velasco.

A la memoria de mi abuelo,

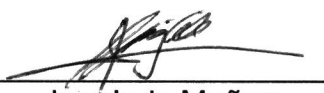
Sr. Ramón Velasco.

Juan Moreno

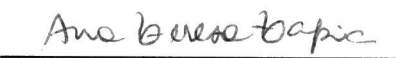
TRIBUNAL DE GRADUACIÓN



Ing. Miguel Yapur
SUB-DECANO DE LA FIEC



Ing. Luis Muñoz
DIRECTOR DEL TÓPICO



Ing. Ana Tapia
VOCAL PRINCIPAL

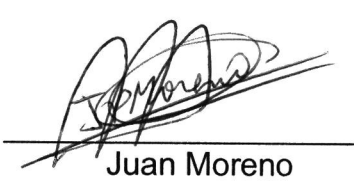
Ing. Mónica Villavicencio
VOCAL PRINCIPAL

SUB-ESPOL

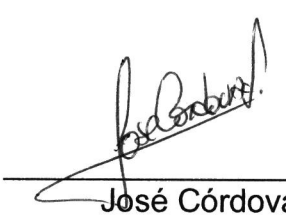
DECLARACIÓN EXPRESA

"La responsabilidad por las ideas, hechos y doctrinas expuestas en esta tesis nos corresponde exclusivamente y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL"

(Reglamento de Graduación de la ESPOL)



Juan Moreno



José Córdova

RESUMEN

Este proyecto ayuda a resolver el problema de organizar los productos en el espacio físico disponible de una Bodega. No es un manejador de inventarios ni de existencias de productos, sino un complemento.

Se realizó lo siguiente:

- ✓ En el interior del espacio físico de una Bodega se crearon lugares de almacenamiento denominados Perchas y SubPerchas; ambas tienen 3 dimensiones espaciales: alto, ancho y profundidad. Las SubPerchas se encuentran en el interior de las Perchas y, pueden haber algunas, siempre y cuando su volumen no exceda el de las Perchas.

- ✓ Los productos son manejados como "Cajas de Productos", las cuáles poseen también las dimensiones espaciales antes mencionadas.

- ✓ Las SubPerchas solo pueden contener Cajas de Productos que sean del mismo tipo y de la misma dimensión.

En el capítulo 1, se describe el proyecto con sus objetivos y se justifica teóricamente la tecnología utilizada.

En el capítulo 2, se detallan las herramientas de codificación, el servidor de aplicaciones y la base de datos utilizados en el desarrollo del proyecto.

En el capítulo 3, se presentan los requerimientos del Sistema y el diseño arquitectónico de la aplicación. Adicionalmente, se explica como se desarrolló la interfaz del Sistema.

En el capitulo 4, se detallan los problemas a los cuáles nos enfrentamos durante las etapas de la implementación, además se describe el plan de pruebas y los resultados obtenidos.

Finalmente se exponen las conclusiones y recomendaciones.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ABREVIATURAS

Abreviaturas	Descripción
WSM	Warehouse Space Manager
PDA	Personal Digital Assistant
MVC	Modelo Vista Controlador
J2EE	Java 2 Enterprise Edition
EJB	Enterprise Java Bean
CMP	Container Managment Persistency
JSP	Java Server Page
HTTP	Hiper Text Transfer Protocol.
JDBC	Java DataBase Connectivity
JVM	Java Virtual Machine
J2SE	Java 2 Standard Edition
J2ME	Java 2 Micro Edition
JDK	Java Development Kit
J2SDK	Java 2 Standard Development Kit
IDE	Integrated Development Environment
WAS	Web Application Server
MVC	Model View Controller

INTRODUCCIÓN

En la actualidad existen empresas y otras entidades, que necesitan almacenar adecuadamente sus artículos o productos y tener un ordenamiento coherente, para su futura distribución.

Llevar un control de los espacios libres y ocupados que existen en una bodega es un problema difícil de resolver para las empresas; para resolverlo, se desarrolló el proyecto: "Sistema para Administrar una Bodega, orientado a la organización del espacio físico, utilizando plataforma Java y soporte de dispositivo PDA". De ahora en adelante, para su simplicidad, denominaremos al Sistema como "WSM" por sus siglas en inglés: "Warehouse Space Manager".

La plataforma utilizada en el desarrollo del proyecto, es Java, específicamente J2EE (Java 2 Enterprise Edition) a través de EJB (Enterprise Java Beans), JSP (Java Server Pages); potenciando la arquitectura MVC (Modelo Vista Controlador) y accesando a una base de datos en MySql. Todas las herramientas de desarrollo que se utilizaron son de libre distribución y de código abierto.

CAPÍTULO 1

1. Justificación del Proyecto y teoría de la tecnología utilizada

1.1. Descripción del Proyecto

Este proyecto, denominado "WSM", permite organizar el espacio físico disponible en el interior de una bodega. Contempla la creación y mantenimiento de Perchas y SubPerchas, que poseen las siguientes dimensiones físicas para su fácil administración: alto, ancho y profundidad. WSM permite crear y mantener cajas de productos, las cuáles son almacenadas en las SubPerchas.

1.2. Objetivos del Proyecto

Objetivos Primarios:

- ✓ Utilizar la Plataforma JAVA para desarrollar aplicaciones transaccionales bajo la arquitectura de J2EE y como parte de su especificación a: EJB, JSP, SERVLET.
- ✓ Utilizar el Servidor de Aplicaciones Web: JBOSS para soportar a los EJB, JSP y SERVLET.
- ✓ Utilizar el Modelo Vista Controlador (MVC) para el correcto desarrollo del Sistema.

Objetivos Secundarios:

- ✓ Resolver el problema de organizar el espacio físico previamente definido de una Bodega.
- ✓ Organizar Cajas de Productos dentro de una bodega.
- ✓ Mantener un registro de las Cajas que se ingresan o se dan de Baja de una bodega.
- ✓ Desarrollar los módulos para Ingresar y dar de Baja a las Cajas de Productos, a través de un dispositivo PDA como la Palm Tungsten C, utilizando enlace inalámbrico Wi-Fi con el protocolo 802.11b, vía Web por medio del protocolo http.

1.3. ¿Qué es J2EE?

J2EE son las siglas de Java 2 Enterprise Edition que es la edición empresarial del paquete Java creada y distribuida por Sun Microsystems. Comprenden un conjunto de especificaciones y funcionalidades orientadas al desarrollo de aplicaciones empresariales. Debido a que J2EE no deja de ser un estándar, existen otros productos desarrollados a partir de ella aunque no exclusivamente. [4]

Algunas de sus funcionalidades más importantes son:

- ✓ Aplicaciones Web (JSP y Servlet)
- ✓ Acceso a base de datos (JDBC)
- ✓ Utilizado por BEA, IBM, Oracle, Sun, y Apache Tomcat entre otros.
- ✓ Utilización de directorios distribuidos (JNDI)
- ✓ Acceso a métodos remotos (RMI/CORBA)
- ✓ Funciones de correo electrónico (JavaMail)
- ✓ Uso de Beans.

De los valores que convierten a Java en algo único en este mercado destacan los siguientes puntos:

Capacidad Multiplataforma. Java como capa de abstracción permite que el mismo lenguaje de programación, las mismas herramientas y sobre todo el mismo know-how pueda ser utilizado para desarrollar servicios destinados a un gran ordenador de millones de dólares.

Seguridad. Si el dispositivo va a estar en la Red interactuando e intercambiando datos y código con otros sistemas era necesario que dispusiera de mecanismos robustos que impidan a esos programas actuar como virus, troyanos o cualquier otro comportamiento extraño.

Su velocidad de implantación se explica no sólo porque permite desarrollar servicios y aplicaciones una única vez, para que puedan funcionar en toda clase de sistemas operativos y dispositivos. Sino que a esta capacidad se le añade su facilidad de enviar código de un lugar a otro, lo que permite utilizar cualquier aplicación sin necesidad de que esté instalada en una máquina determinada, solo es necesario esté en la World Wide Web.

Java está teniendo un impacto más fuerte en el sector de las telecomunicaciones. Se está convirtiendo en el nuevo estándar para servicios móviles con el respaldo de más de 70 operadoras en todo

el mundo, que han puesto a disposición de sus clientes más de 200 modelos diferentes de terminales móviles capaces de soportar Java.

Java es un gran aliado tecnológico del sector de las telecomunicaciones, gracias a que los mismos programadores que han creado aplicaciones sobre Java 2 Enterprise Edition (J2EE) para el entorno empresarial de los que existen más de 3 millones en todo el mundo pueden ahora conectar estas aplicaciones o crear otras nuevas para dar servicios sobre cualquier terminal que incorpore Java 2 Micro Edition (J2ME), de una manera fácil, rápida y segura.

Con J2ME, los usuarios de estos móviles pueden acceder a nuevos servicios, sin necesidad de instalar y almacenar aplicaciones que ocupan varios Megabytes y funciones que nunca van a utilizar Java es muy dinámico, de forma que el sistema descargará lo que necesite de forma transparente y lo usará como se hace ahora en la web, de forma local y sin conexión ni costo adicional.

La plataforma J2EE añade a Java la funcionalidad necesaria para convertirse en un lenguaje orientado al desarrollo de servicios en Internet. Mediante JSP y Servlets se pueden desarrollar sitios Web bajo la tecnología Java.

J2EE pudiera ser considerado el rival de DNA ("Distributed Network Applications") ofrecido por Microsoft, conocido hoy en día como Microsoft Solution Platform, el cual esta compuesto por Windows2000, SQL Server, Internet Information Services, COM y MSMQ, como puede notar a diferencia de J2EE, DNA es un producto y esta forzado a utilizar una sola implementación, a diferencia de implementaciones J2EE donde puede elegir entre diversos productos de distintas empresas. DNA ha evolucionado a lo que hoy se conoce como .NET

Otras ventajas con que las que cuenta J2EE para ser elegido como pieza fundamental de este proyecto, son la escalabilidad, la posibilidad de tener la lógica del negocio independiente de los clientes, la alta disponibilidad de las aplicaciones y del acceso a las bases de datos, así como el manejo de clientes y del factor seguridad, entre otros.

1.4. ¿Qué son los EJBs?

Los EJBs proporcionan un modelo de componentes distribuidos estándar para el lado del servidor. El objetivo de los

Enterprise beans es brindar un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad) para centrarse en el desarrollo de la lógica de negocio en sí. El hecho de estar basado en componentes nos permite que éstos sean flexibles y sobre todo reutilizables.

No hay que confundir a los Enterprise JavaBeans con los JavaBeans. Los JavaBeans también son un modelo de componentes creado por Sun Microsystems para la construcción de aplicaciones, pero no pueden utilizarse en entornos de objetos distribuidos al no soportar nativamente la invocación remota (RMI).

1.4.1 EJB tipo Session

EJBs de Sesión (Session EJBs): gestionan el flujo de la información en el servidor. Generalmente sirven a los clientes como una fachada de los servicios proporcionados por otros componentes disponibles en el servidor. Puede haber dos tipos:

Con estado (StateFul).- Los beans de sesión con estado son objetos distribuidos que poseen un estado. El estado no es persistente, pero el acceso al bean se limita a un solo cliente.

Sin estado (Stateless).- Los beans de sesión sin estado son objetos distribuidos que carecen de estado asociado permitiendo por tanto que se los acceda concurrentemente. No se garantiza que los contenidos de las variables de instancia se conserven entre llamadas al método.

1.4.2 EJB de Entidad

EJBs de Entidad (Entity EJBs): su objetivo es encapsular los objetos de lado de servidor que almacenan los datos. Los EJBs de entidad presentan la característica fundamental de la persistencia:

Persistencia gestionada por el contenedor (CMP): el contenedor se encarga de almacenar y recuperar los datos del objeto de entidad mediante un mapeado en una tabla de una base de datos.

Persistencia gestionada por el bean (BMP): el propio objeto entidad se encarga, mediante una base de datos u otro mecanismo, de almacenar y recuperar los datos a los que se refiere.

1.5. Servlets y JSP

¿Qué son los Servlet?

Para saber que es un servlet, vamos a empezar indicando que es un objeto Java que implementa la interfaz javax.servlet.Servlet ó hereda alguna de las clases más convenientes para un protocolo específico, como por ejemplo: javax.servlet.HttpServlet).

Actualmente, cuando se habla de aplicaciones para uso del Web, comúnmente escuchamos hablar del potencial JAVA, y junto con ello sus aplicaciones mas conocidas, como son los Applets, los mismo que son programas que se pueden cargar a través de una red y que se ejecutan de igual forma en cualquier plataforma (se pueden ejecutar en cualquier sistema operativo, todo se encarga el Java Virtual Machine JVM).

Hace poco tiempo, JAVA era utilizado para facilitar las páginas Web con interactividad usando los Applets, y por tanto solo actuaba sobre

el lado cliente. Actualmente, se usa a los servlets para que en el lado del servidor puede beneficiarse todas las ventajas que nos ofrece JAVA.

Entendamos cual es la diferencia entre los Servlets y Applets, los Servlet básicamente se ejecutan en el lado del servidor y en que no presentan ningún tipo de interfaz gráfica y esto porque se encargan de hacer el trabajo oculto; mientras que los Applets se ejecutan del lado del cliente.

A continuación vamos a revisar ciertas características de porque usamos Servlet y no CGI.

- ✓ Los servlets son más rápidos que los CGI debido a que utilizan threads en lugar de procesos.

- ✓ Es el desperdicio del CGI/Perl en los recursos del sistema. Cada petición crea un nuevo proceso el cual carga el interpretador de Perl. El intérprete de Perl carga el script CGI, lo compila, y lo ejecuta. Además, si la aplicación se comunica con la BD, una nueva conexión será necesaria para ser hecha por cada CGI.

- ✓ En cambio los Servlets inician un nuevo thread (hilo) con cada petición. Cada servlet es cargado una vez y usado más y más. Nota que a diferencia de los CGI, los servlets requieren que se cuente con una Máquina Virtual de Java (JVM) corriendo sobre el servidor todo el tiempo. Para sitios ocupados, ésta permite a los servlets usar mucho menos los recursos del sistema e incrementar el desempeño.

- ✓ Los servlets son tan portables como cualquier otra aplicación de Java.

- ✓ La portabilidad que nos facilita los servlets de Java es más simple, esto porque Java fue diseñado para ser portable a través de todas las plataformas, permitiendo que las aplicaciones sean movidas fácilmente de un sistema operativo a otro.

Arquitectura de los Servlets

El componente principal de la Servlet API es la interfaz Servlet, a su vez todos los servlets implementan esta interfaz directamente, por medio de la clase que la implementa, HttpServlet. Esta interfaz está provista de métodos que manipulan a los servlets y a su vez la comunicación con sus clientes.

Cuando un cliente invoca a un servlet, este recibe dos objetos: ServletRequest y ServletResponse. La interfaz ServletRequest se encarga de la comunicación desde el cliente al servidor, mientras que la interfaz ServletResponse atiende la comunicación desde servlet al cliente.

La interfaz ServletRequest permite al servlet acceder a la información: nombres de parámetros dados por el cliente, el protocolo usado por el cliente, y los nombres de los host remotos que crean la solicitud y el servidor quien las recibe. Esta interfaz ayuda a los servlets el acceso a métodos que permiten manejar la presentación de la respuesta como salida en el navegador Web, a través de los cuales consiguen los datos desde el cliente que usa protocolos como HTTP POST, entre otros.

La interfaz ServletResponse proporciona a los servlet los métodos para contestarle al cliente. Permite al servlet configurar la forma de salida de los datos para el cliente.

¿Qué es Java Server Page?

Java Server Page es otra de las nuevas tecnologías para tratar de hacer mas eficiente el modelo cliente-servidor. Con JSP podemos

crear aplicaciones web que se ejecuten en variados servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. El motor de las páginas JSP está basado en los servlets de Java -programas en Java destinados a ejecutarse en el servidor.

Java Server Pages es un conjunto de tecnologías que permiten la generación dinámica de páginas web combinando código Java (script) con un lenguaje de marcas como HTML ó XML, para generar el contenido de la página. Una característica importante es que permite separar la interfaz del usuario de la generación del contenido dinámico, dando lugar a procesos de desarrollo más rápidos y eficientes.

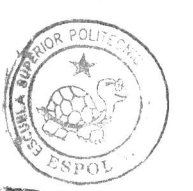
Pueden acceder directamente a componentes Java Beans ó Enterprise Java Beans (EJB), instanciandolos y estableciendo sus propiedades e invocando sus métodos directamente desde la página JSP. Esto permite desarrollar aplicaciones n-capas donde se separan en lo posible los datos, la lógica del negocio y la lógica de presentación, encapsulando, generalmente, en Beans el acceso a los datos.

Cada vez que un cliente solicita al servidor web una página JSP, este pasa la petición al motor de JSP el cual verifica si la página no se ha ejecutado antes ó fue modificada después de la última compilación, tras lo cual la compila, convirtiéndola en Servlet, la ejecuta y devuelve los resultados al cliente en formato HTML.

Es importante indicar, que la tecnología JSP es un componente clave de la plataforma Java 2 Enterprise Edition propuesta por Sun Microsystems.

En resumen, las tecnologías JSP y Servlets son una importante alternativa para la programación de web de contenido dinámico que nos permiten:

- ✓ Independencia de la plataforma.
- ✓ Rendimiento mejorado.
- ✓ Separación de la lógica de la aplicación de la presentación de los datos.
- ✓ Uso de componentes (Java Beans)
- ✓ Facilidad de administración y uso.
- ✓ El importante respaldo de la sólida tecnología Java TM.



1.6. Modelo Vista Controlador (MVC)

Model View Controller no es más que un patrón de diseño aportado originariamente por el lenguaje SmallTalk a la Ingeniería del Software. El paradigma MVC consiste en dividir las aplicaciones en tres partes:

- ✓ Controlador (Controller)
- ✓ Modelo (Model)
- ✓ Vistas. (View)

El controlador es el encargado de redirigir o asignar una aplicación a cada petición; el controlador debe poseer de algún modo, un "mapa" de correspondencias entre peticiones y respuestas que se les asignan. El modelo seria la aplicación que responde a una petición, es la lógica de negocio a fin de cuentas. [5]

Una vez realizadas las operaciones necesarias el flujo vuelve al controlador y este devuelve los resultados a una vista asignada. Vemos las diferencias que supone el modelo con los modelos convencionales.

Figura 1: MVC Modelo Convencional

Si revisamos el esquema más básico de programa, tenemos una entrada o parámetros que llegan (INPUT), se procesan y se muestra el resultado (OUTPUT).

En el caso del patrón MVC el procesamiento se lleva a cabo entre sus tres componentes. El controlador recibe una orden y decide quien la lleva a cabo en el modelo. Una vez que el modelo (la lógica de negocio) termina sus operaciones devuelve el flujo vuelve al controlador y este envía el resultado a la capa de presentación.

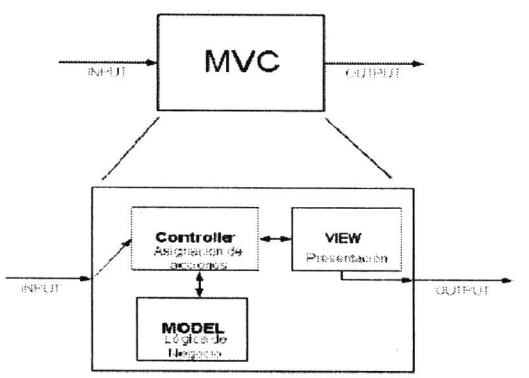


Figura 2: Patrón MVC

El Controlador en cierta forma debe tener un registro de la relación entre órdenes que le pueden llegar y la lógica de negocio que le corresponde. En el siguiente gráfico se representa ese funcionamiento:

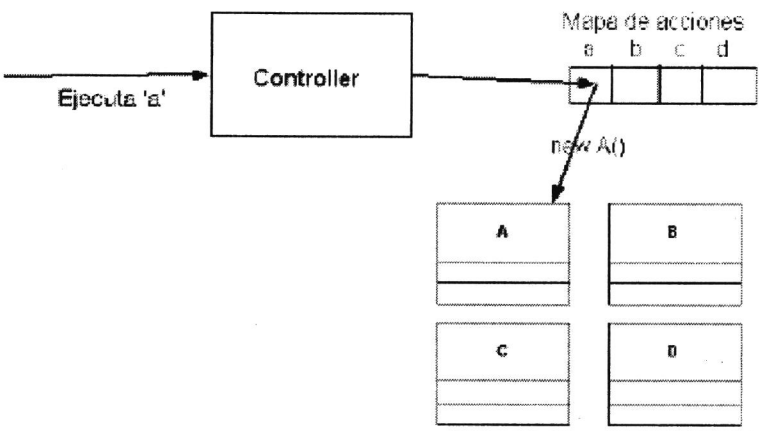


Figura 3: Modelo MVC

La ventaja que se obtiene de este modelo consiste en una separación total entre la lógica del negocio y la presentación. A esto se le pueden aplicar opciones como el multilenguaje, distintos diseños de presentación, sin alterar la lógica de negocio.

Lo importante en la separación de las capas como presentación, lógica de negocio, acceso a datos es fundamental para el desarrollo de arquitecturas consistentes, reutilizables y más fácilmente

mantenibles, lo que al final resulta en un ahorro de tiempo en desarrollo en posteriores proyectos.

En el lenguaje Java disponemos de unas clases muy sencillas para implantar el modelo MVC: la clase Observer, Observable del paquete util. Aunque esa implementación del MVC con esas clases se podría hacer a un nivel muy simple de interacción entre unas pocas clases.

1.7. J2ME para desarrollo de aplicaciones móviles

La plataforma J2ME ofrece una gran gama de especificaciones que definen varias versiones minimizadas de la plataforma Java 2; estas versiones pueden ser utilizadas para programar en dispositivos electrónicos; desde teléfonos celulares, en PDAs, hasta en tarjetas inteligentes, para nuestro caso usaremos para el desarrollo de los dispositivos PDA. Estos dispositivos presentan en común que no disponen de abundante memoria ni mucha potencia en el procesamiento, ni tampoco necesitan de todo el soporte que brinda el J2SE, (la plataforma estándar de Java usada en sistemas de escritorio y servidor)

Aunque J2ME utiliza el mismo lenguaje base que las ediciones J2SE y J2EE, se reduce el tamaño del entorno de ejecución a fin de poder ejecutarse en dispositivos que tienen distintas restricciones de memoria y potencia de procesamiento.

Para saber algo de historia de J2ME, podemos comentar que al principio de los años 90, Sun Microsystems creó un nuevo lenguaje de programación llamado Oak como parte de un proyecto de investigación para construir productos electrónicos que dependan principalmente del software. El primer prototipo para Oak fue un controlador portable llamado Star7, el mismo que estaba compuesto por un pequeño dispositivo handheld con una pantalla touchscreen LCD que tenía incorporado soporte a redes inalámbricas y comunicaciones infrarojas. Lo muy importante de este dispositivo consistía en que podría ser usado como control remoto para televisores o VCR, guía de programas electrónicos, e incluso tenía algunas funciones que ahora son asociadas a los PDAs, como agenda de citas.

Lo importante que se debe de tener en cuenta es que el software para este tipo de dispositivos necesitaba ser confiable y no debía tener excesivo uso de memoria ni requerir demasiada potencia en el

procesador. Oak fue desarrollado como resultado de la experiencia del desarrollo con el lenguaje C++, el cual, a pesar de tener muchas grandes características, demostró que era un lenguaje muy complejo y ocasionaba que los programadores comentan fácilmente errores y eso afectaba la confiabilidad del software.

Oak fue diseñado para quitar o reducir la posibilidad de que los programadores comentan esos tipos de errores, detectando la mayoría de errores en tiempo de compilación y quitando algunas de las características del lenguaje C++, como por ejemplo: punteros, administración de memoria controlada por el programador.

Después de todo el trabajo realizado, el mercado para este tipo de dispositivos que el nuevo lenguaje fue creado no se desarrolló tanto como Sun Microsystems lo esperaba, y al final ningún dispositivo basado en Oak fue vendido a los clientes. Luego de esto y en base al conocimiento público de Internet produjo un mercado para el software de los navegadores Web. En respuesta a esto, Sun Microsystems renombró el lenguaje de programación Oak a Java y lo usó para desarrollar un navegador multiplataforma llamado HotJava.

En un año, las capacidades multiplataforma del lenguaje de programación Java y su potencia como plataforma de desarrollo para aplicaciones que podían ser escritas una vez y ejecutadas en diversos sistemas Unix y Windows, había despertado el interés de usuarios finales, porque vieron en ella una manera de reducir los costos del desarrollo de software.

Con el tiempo Sun Microsystems liberó la primera versión de la plataforma Java 2, había sido necesario dividirla en varias piezas. La funcionalidad principal, estimado como el mínimo soporte requerido para cualquier ambiente Java, estaba empaquetada en el Java 2 Standard Edition (J2SE).

Sun Microsystems también respondió al incremento de usar Java para el desarrollo a un nivel empresarial, y ambientes de servidores de aplicaciones con la plataforma Java 2 Enterprise Edition (J2EE), el cual incorpora nuevas tecnologías como servlets, Enterprise JavaBeans, JavaServer pages, etc.

Mientras Sun Microsystems estaba enfocado en el desarrollando Java para Internet y para la programación comercial, la demanda empezó a crecer en los dispositivos móviles. Sun Microsystems

respondió a esta demanda creando varias plataformas Java con funcionalidades reducidas.

Estas plataformas reducidas están todas basadas en el JDK 1.1, el predecesor de la plataforma Java 2, y cada una tiene una estrategia diferente al problema de reducir la plataforma para acomodarla a los recursos disponibles que se tiene. Por lo tanto, cada una de estas plataformas de funcionalidad reducida representan una solución ad hoc al problema. Por ello es que aparece la plataforma J2ME, para reemplazar todas esas plataformas reducidas basadas en el JDK 1.1 y crear una sola solución basada en Java 2.

1.8. APIs de desarrollo para equipos PDA, Palm TungstenC

Debido a la compatibilidad entre J2ME y J2SDK, no hubo necesidad de utilizar alguna API específica para el dispositivo PDA: Palm Tungsten C.

Al utilizar el protocolo http en el dispositivo PDA, solo hubo necesidad de rediseñar las interfaces para interactúa r con la Palm Tungsten C (por el limitado espacio de la pantalla: 320x320 pixeles), en lo que tiene que ver a los módulos de: "Asignación de Productos a SubPercha" y "Egreso de Productos desde SubPerchas".

CAPÍTULO 2

2. Herramientas de código abierto utilizadas en el desarrollo del Proyecto

2.1. Eclipse como IDE de Desarrollo

"Eclipse es una plataforma de integración de herramientas construida por una comunidad abierta de proveedores de herramientas. Operando dentro del paradigma de código abierto, con una licencia de uso público común que provee código fuente gratis y derechos de distribución mundial, la plataforma Eclipse provee a desarrolladores de herramientas un nivel inigualable de flexibilidad y control sobre sus tecnologías de software." [6]

Eclipse se ha convertido en unos de los entornos de programación más eficaces para el desarrollo de aplicaciones. Su soporte para múltiples lenguajes de programación y la orientación a pequeños y grandes proyectos, permiten utilizarlo para el desarrollo de cualquier programa.

Originalmente un producto de IBM, decir que Eclipse es simplemente un IDE (Integrated Development Enviroment) o ambiente de programación es subestimar el poder completo de esta herramienta. Una mejor forma de describirlo es como una plataforma donde se pueden integrar múltiples IDEs. Eclipse no esta limitado a ninguna tecnología de desarrollo, está abierto, mediante una arquitectura de plug-ins, a recibir funcionalidad para cualquier tecnología de cualquier proveedor. El hecho de que es código abierto, su increíble flexibilidad, y su excelente reputación convirtieron a Eclipse la primera y única opción como la herramienta de desarrollo del proyecto Warehouse Space Manager. Actualmente Eclipse está en su versión 3.0 la cuál utilizamos en nuestro proyecto.

Eclipse cuenta con múltiples funciones que lo convierten en el entorno perfecto tanto para desarrollo profesional como de aprendizaje. Algunas de sus características más importantes son:

Editor cómodo y con todas las características que podemos esperar de cualquier editor de código. Se suma el perfecto reconocimiento de las llaves de las funciones.

Multidesarrollo, Eclipse aunque esta programado y enfocado para java permite, mediante el uso de complementos, la programación en otros lenguajes, como PHP, C, C++ o Cobol.

Proyectos, Son la base de la programación en Eclipse y el centro de organización del programa, ya que todo fichero se almacena en un proyecto.

Módulos que gracias a la comunidad de desarrolladores, se han multiplicado para aumentar las prestaciones de Eclipse. Son uno de los factores determinantes, ya que añaden diversas funciones indispensables para desarrollos de carácter profesional.

Autocomplementado. Simplifica la creación de código fuente al visualizarse un menú con las opciones disponibles, por ejemplo: al escribir un método de una clase, también se muestra información referente al método que vamos a escribir, sus parámetros y finalidad.
Refactorización. Una de las funciones más útiles que modifica automáticamente las funciones referenciadas. Esto permite al programador olvidarse de errores en las llamadas a las mismas.

Depurador. Controla el código, tanto mientras se escribe como al ejecutar la aplicación. La información referente a cada uno de los errores es completa y permite solucionarlos rápidamente.

Programación gráfica a través de SWT, toolkit de última generación para crea interfaces gráficos en Java.

2.2. Jboss como servidor de Aplicaciones

JBoss es una implementación Open-Source de un "EJB Container"; es mediante este tipo de productos que es posible llevar acabo un desarrollo con EJB's "Enterprise Java Bean's" . Este tipo de producto ("EJB Container") generalmente no es distribuido como producto individual y por esta razón se le pudiera considerar a "JBoss" como un producto diferente mas no único. [7]

El producto JBoss es únicamente un "EJB Container" y es por esto que generalmente se utiliza en conjunción con un "Web-Container", el "Web-Container" puede ser cualquiera disponible en el mercado, para nuestro proyecto hemos usado Tomcat como "Web-Container", aunque lo anterior no restringe a JBoss para operar con otro "Web Container" como ServletExec , la única ventaja de utilizar el "Web Container" incluido con JBoss será en tiempo de coordinación/configuración entre JBoss "x" Web-Container, y siendo

que un ambiente utilizando EJB's es altamente complejo es preferible concentrarse en algo que ya ha sido utilizado y depurado.

2.3. MySql

MySQL es la base de datos más popular Open Source SQL, es desarrolla y la provee MySql AB. MySql AB es una compañía comercial que construye su proveedor servicios de negocios de base de datos MySql. [8]

MySql es un manejador de base de datos que ofrece una gama de características para garantizar la integridad de los datos almacenados. La intención de MySql Server es orientado para ambientes críticos, cargas pesados en ambiente productivos.

A continuación mostramos las principales características de MySql:

- ✓ La robustez debido a que esta escrito en C y C++.
- ✓ Trabaja en ambientes heterogéneos.
- ✓ Completamente Multihilos.
- ✓ Descarga de tablas en memoria la cual son usadas como tablas temporales.
- ✓ Ofrecen joins muy rápidos usando un optimised one-sweep (Multi-Join)

- ✓ Seguridad en el sistema de password.
- ✓ Hasta 32 índices por tabla son permitidos.
- ✓ Los clientes pueden conectarse a MySql usando sockets TCP/IP.
- ✓ Alias en tablas y columnas son permitidas como un Standard en SQL92.

Podemos seguir mostrando un sin numero de características que nos ofrece MySQL.

2.4. Integración Linux-Jboss-MySql

Linux, MySQL, y JBOSS forman un eje en el cuál se desenvuelve el ambiente de ejecución de WareHouse Space Manager (WSM). Para que WSM pueda funcionar de manera rápida y eficiente en un ambiente de producción es necesario que estos tres componentes estén perfectamente integrados.

Linux, siendo el sistema operativo elegido para el funcionamiento, tiene que cumplir ciertos requisitos para poder integrarse con MySQL y JBOSS. El primer y más importante requisito es la incorporación de J2SDK versión 1.4.2_02 al sistema operativo. La última versión de

Java debe de estar instalada correctamente, todas las variables de ambiente que éste requiera también deben de estar definidas correctamente en el sistema. El segundo requisito es que el sistema operativo pueda dar las suficientes prestaciones a MySQL y JBOSS para funcionar de una manera eficiente. Esto quiere decir que Linux no debe de estar sobrecargado con otros servicios que consuman recursos o causen conflictos de operación con JBOSS o MySQL. JBOSS requiere una gran cantidad de memoria y un gran número de puertos de red libres para poder funcionar. Los siguientes paquetes de software pueden potencialmente interferir con la aplicación: Apache, Tomcat, Oracle, Postgres, y versiones anteriores de MySQL o JBOSS en el mismo sistema.

Linux es sensible a la capitalización de los caracteres por lo tanto la versión de MySQL para Linux también lo es. Un error común al traspasar código desarrollado en herramientas ejecutadas en Windows a Linux es olvidar este aspecto. Es muy importante que el nombre de las tablas y objetos se mantengan igual durante todo el desarrollo del proyecto y a través de todos los módulos. Es importante además que MySQL y JBOSS se ejecuten e instalen como un servicio en el sistema operativo Linux.

JBOSS intenta mantener en lo más posible la sensibilidad a la capitalización mediante Java o mediante su propia funcionalidad. JBOSS 3.0 requiere por lo menos al JSDK 1.3 o mayor. Además JBOSS tiene que tener bien configuradas algunas variables en el sistema operativo, ejemplo JBOSS_HOME. JBOSS necesita un driver JDBC para conectarse con MySQL. Existen varias implementaciones de este driver. En nuestro proyecto usamos la clase org.gjt.mm.mysql.Driver en el jar MySQL Connector versión 4.0.20

En resumen JBOSS y MySQL están diseñados para integrarse con Linux sin ningún cambio grande en la configuración de ambos paquetes de software. JBOSS corre sobre Java el cuál es independiente de plataforma. La integración en nuestro proyecto es eficiente y prácticamente transparente.

Las herramientas usadas en nuestro proyecto están con más detalle en el capítulo 4, en la sección Implementación y pruebas.

CAPÍTULO 3

3. Desarrollo del Sistema

3.1. Análisis del Sistema

3.1.1 Requisitos y Casos de Uso

El proyecto se limita de acuerdo a los siguientes requisitos y alcances:

Requisitos Generales:

- ✓ Los códigos de Bodega, Percha y SubPercha son únicos.

- ✓ El Sistema debe proveer la conexión de un dispositivo PDA para asignar o dar de Baja las Cajas de Productos de las SubPerchas. Esta conexión debe ser a través de una red inalámbrica de protocolo 802.11b y vía web utilizando el protocolo http.

- ✓ El Sistema debe indicar el máximo número de Cajas de Productos que se pueden ubicar en una SubPercha determinada.

Requisitos respecto a las Bodegas:

- ✓ No pueden ser eliminadas si tienen Perchas asignadas.

Requisitos respecto a las Perchas:

- ✓ Una Percha sólo puede pertenecer a una Bodega.
- ✓ No pueden ser eliminadas si tienen SubPerchas asignadas.

Requisitos respecto a las SubPerchas:

- ✓ Solo puede pertenecer a una Percha.
- ✓ Solo pueden almacenar un mismo tipo de "Caja de producto".
- ✓ No se pueden eliminar si tienen Cajas de Productos asignadas.
- ✓ El estado de la SubPercha se mantendrá "Libre" hasta que se complete el número máximo de cajas permitidas (según el volumen máximo permitido); pasando luego al estado "ocupado".

- ✓ El Volumen máximo de todas las SubPerchas asignadas a una Percha, no puede exceder al volumen máximo de la Percha a la que pertenecen.

Requisitos respecto a las Cajas de Productos:

- ✓ El volumen máximo de todas las Cajas de productos asignadas a una subpercha no puede exceder al volumen máximo de la SubPercha a la que pertenecen.
- ✓ Solo se podrán eliminar las Cajas de Productos que no hayan sido asignadas a SubPerchas. Aquellas que están asignadas, deben darse de Baja.
- ✓ No se podrán dar de baja más productos de los que físicamente existen asignados a una SubPercha.

Requisitos respecto a los Reportes:

El Sistema provee dos Reportes que permiten conocer:

- ✓ Los productos que existen en una SubPercha determinada.
  Los productos que han sido ingresados o dados de baja, en un rango de fechas determinados.

A continuación mostramos los casos de usos que fueron desarrollados en nuestro proyecto:

Caso de Uso:	**Administrar bodegas**
Actores:	Administrador del Sistema
Datos/Atributos	Los atributos de un usuario son: • id_bodega • descripción • temperatura • ciudad
Descripción:	Un Administrador ingresa al sistema y permite realizar las siguientes operaciones: • Crear bodegas • Consultar bodegas • Modificar bodegas • Eliminar bodegas

Escenario	Crear Bodega.
Antecedentes	El administrador del sistema está conectado. La Bodega a crear no exista registrada en el sistema.
Resultados	Se ha creado la bodega al sistema.
Pasos	• Ingresar datos de la Bodega. • Aceptar el ingreso de la bodega

Escenario	Consultar Bodegas.
Antecedentes	El administrador del sistema está conectado. Hay bodegas creadas en el sistema.
Resultados	La información de la bodega consultada se ha desplegado.
Pasos	• Escoger la bodega a Consultar.

Escenario	Modificar Bodegas
Antecedentes	El administrador del sistema está conectado. Hay bodegas creadas en el sistema.
Resultados	Modificar los datos de la bodega registrada en el sistema
Pasos	• Escoger el bodega a modificar

Escenario	Eliminar Bodegas
Antecedentes	El administrador del sistema está conectado. Hay bodegas creadas en el sistema.
Resultados	Eliminar bodega registradas en el sistema
Pasos	• Escoger el bodega a eliminar

Caso de Uso:	**Administrar Perchas**
Actores:	Administrador del Sistema
Datos/Atributos	Los atributos de un usuario son: • Id_percha • Id_bodega • Alto • Ancho • Profundidad • Estado • descripcion
Descripción:	Un Administrador ingresa al sistema y permite realizar las siguientes operaciones: • Crear perchas • Consultar perchas • Modificar perchas • Eliminar perchas

Escenario	Crear Percha.
Antecedentes	El administrador del sistema está conectado. La percha a crear no exista registrada en el sistema.
Resultados	Se ha creado la bodega al sistema.
Pasos	• Ingresar datos de la Percha

	• Aceptar el ingreso de la Percha
Escenario	Consultar Percha.
Antecedentes	El administrador del sistema está conectado. Hay perchas creadas en el sistema.
Resultados	La información de la bodega consultada se ha desplegado.
Pasos	• Escoger la percha a Consultar.

Escenario	Modificar Perchas
Antecedentes	El administrador del sistema está conectado. Hay perchas creadas en el sistema.
Resultados	Modificar los datos de la percha registrada en el sistema
Pasos	• Escoger la percha a modificar

Escenario	Eliminar Percha
Antecedentes	El administrador del sistema está conectado. Hay perchas creadas en el sistema.
Resultados	Eliminar perchas registradas en el sistema
Pasos	• Escoger la percha a eliminar

Caso de Uso:	**Administrar Subpercha**
Actores:	Administrador del Sistema
Datos/Atributos	Los atributos de un usuario son: • id_subpercha • id_percha • alto • ancho • profundidad • estado • numero_cajas_asignadas • descripcion
Descripción:	Un Administrador ingresa al sistema y permite realizar las siguientes operaciones: • Crear subperchas • Consultar subperchas • Modificar subperchas

	• Eliminar subperchas
Escenario	Crear subPercha.
Antecedentes	El administrador del sistema está conectado. La percha a crear no exista registrada en el sistema.
Resultados	Se ha creado la bodega al sistema.
Pasos	• Ingresar datos de la Percha • Aceptar el ingreso de la SubPercha

Escenario	Consultar SubPercha.
Antecedentes	El administrador del sistema está conectado. Hay subperchas creadas en el sistema.
Resultados	La información de la bodega consultada se ha desplegado.
Pasos	• Escoger la subpercha a Consultar.

Escenario	Modificar subPerchas
Antecedentes	El administrador del sistema está conectado. Hay Subperchas creadas en el sistema.
Resultados	Modificar los datos de la Subpercha registrada en el sistema
Pasos	• Escoger la Subpercha a modificar

Escenario	Eliminar SubPercha
Antecedentes	El administrador del sistema está conectado. Hay Subperchas creadas en el sistema.
Resultados	Eliminar Subperchas registradas en el sistema
Pasos	• Escoger la Subpercha a eliminar

Caso de Uso:	**Administrar Cajas de Productos**
Actores:	Administrador del Sistema

Datos/Atributos	Los atributos de un usuario son: • id_caja_producto • id_cliente • id_movimiento • id_subpercha • descripcion • peso • alto • ancho • profundidad • unidades_cajas • Numero_cajas • fecha_caducidad
Descripción:	Un Administrador ingresa al sistema y permite realizar las siguientes operaciones: • Crear Cajas de Productos • Consultar Cajas de Productos • Modificar Cajas de Productos • Eliminar Cajas de Productos

Caso de Uso:	**Subir Cajas a Subperchas**
Actores:	Administrador del Sistema
Descripción:	Un Administrador ingresa al sistema y permite subir a las subperchas.

Caso de Uso:	**Bajar Cajas a Subperchas**
Actores:	Administrador del Sistema
Descripción:	Un Administrador ingresa al sistema y permite bajar a las subperchas.

Caso de Uso:	**Administrar Clientes**
Actores:	Administrador del Sistema
Datos/Atributos	Los atributos de un usuario son: • id_cliente • nombre • direccion • ruc

	• telefono • email
Descripción:	Un Administrador ingresa al sistema y permite realizar las siguientes operaciones: • Crear clientes • Consultar clientes • Modificar clientes • Eliminar clientes

Escenario	Crear Cliente.
Antecedentes	El administrador del sistema está conectado. El Cliente a crear no exista registrada en el sistema.
Resultados	Se ha creado el Cliente al sistema.
Pasos	• Ingresar datos del Cliente. • Aceptar el ingreso de la Cliente

Escenario	Consultar Cliente.
Antecedentes	El administrador del sistema está conectado. Hay Cliente creados en el sistema.
Resultados	La información del Cliente consultada se ha desplegado.
Pasos	• Escoger el Cliente a Consultar.

Escenario	Modificar Cliente
Antecedentes	El administrador del sistema está conectado. Hay Cliente creados en el sistema.
Resultados	Modificar los datos del Cliente registrado en el sistema
Pasos	• Escoger el Cliente a modificar

Escenario	Eliminar Cliente
Antecedentes	El administrador del sistema está conectado. Hay Clientes creados en el sistema.
Resultados	Eliminar Cliente registrados en el

	sistema
Pasos	• Escoger el Cliente a eliminar

Caso de uso:	**Administrar usuarios**
Actores:	**Administrador del Sistema**
Datos/Atributos	Los atributos de un usuario son: • Nombre • Apellido • usuario • contraseña • tipo_usuario • cargo • telefono • rol • email
Descripción:	Un Administrador ingresa al sistema y permite realizar las siguientes operaciones: • Crear • Consultar • Modificar • Eliminar

Escenario	Crear Usuarios.
Antecedentes	El administrador del sistema está conectado. Los Usuarios a crear no existan registrados en el sistema.
Resultados	Se ha creado el Usuarios en el sistema.
Pasos	• Ingresar datos del Usuarios • Aceptar el ingreso de los Usuarios

Escenario	Consultar Usuarios
Antecedentes	El administrador del sistema está conectado. Hay Usuarios creados en el sistema.
Resultados	La información del Usuario

	consultada se ha desplegado.
Pasos	• Escoger los usuarios a Consultar.

Escenario	Modificar Usuarios
Antecedentes	El administrador del sistema está conectado. Hay Usuarios creados en el sistema.
Resultados	Modificar los datos del Usuario registrado en el sistema
Pasos	• Escoger los usuarios a modificar

Escenario	Eliminar Usuarios
Antecedentes	El administrador del sistema está conectado. Hay Usuarios creados en el sistema.
Resultados	Eliminar Usuarios registrados en el sistema
Pasos	• Escoger los Usuarios a eliminar

3.1.2 Diagramas de Estado

Estados de una Caja de Producto:

Una caja de producto puede estar en tres estados: Libre, Asignada o Dada de Baja.

- ✓ Libre cuando no se la haya asignado a SubPercha.
- ✓ Asignado cuando esté asociada a una SubPercha.
- ✓ Dada de Baja cuando se la haya egresado o quitado de la SubPercha.

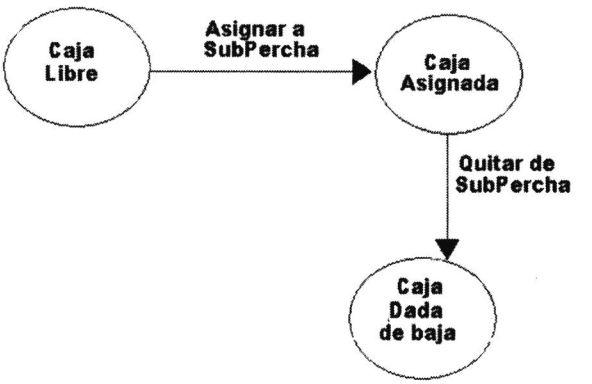


Figura 4: Estados de una caja de Productos

Estados de una SubPercha:

Una SubPercha puede estar en dos estados: Libre u Ocupada.

- ✓ Libre cuando no se le hayan asignado productos y hasta que su volumen sea ocupado por completo.
- ✓ Ocupado cuando se le hayan asignado cajas de productos y el volumen de la SubPercha esté completamente lleno.

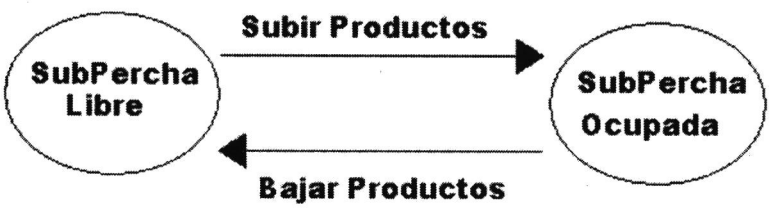


Figura 5: Estado de una subpercha

3.1.3 Proceso de Asignación de Productos a SubPerchas

Para asignar los productos a SubPerchas se deben considerar los siguientes pasos:

Verificar que:

- ✓ Exista acceso al Sistema.
- ✓ La SubPercha se encuentre en estado libre.
- ✓ El producto por asignar a la SubPercha, no exceda el volumen máximo permitido de la misma.
- ✓ Las dimensiones de la caja de producto por asignar, no excedan las dimensiones de la SubPercha.

Registrar:

La transacción como un movimiento tipo Ingreso.

3.1.4 Proceso de Egreso de Productos desde SubPerchas

Para egresar o dar de baja a productos desde SubPerchas, se deben considerar los siguientes pasos:

Verificar que:

- ✓ Exista acceso al Sistema.
- ✓ Existen productos asignados a una SubPercha determinada.

- ✓ El número de cajas que se desean egresar o dar de baja, sea menor o igual que el número de cajas que existen en la SubPercha.

Registrar:

La transacción como un movimiento tipo Egreso.

3.1.5 Proceso de Asignación de Productos a SubPerchas a través de dispositivo PDA.

Para asignar los productos a SubPerchas, a través de un dispositivo PDA se deben considerar los siguientes pasos:

Verificar que:

Exista el acceso de red Wireless al Sistema

- ✓ La SubPercha esté en estado libre, es decir, que su volumen no se haya completado.
- ✓ El producto asignar a la SubPercha, no exceda el volumen máximo permitido de la misma.
- ✓ Las dimensiones de la caja de producto asignar a la SubPercha, no excedan las dimensiones de la SubPercha.

Registrar:

La transacción como un movimiento tipo Ingreso.

3.1.6 Proceso de Egreso de Productos desde SubPerchas a través de dispositivos PDA.

Para egresar o dar de baja a productos desde SubPerchas, se deben considerar los siguientes pasos:

Verificar que:

- ✓ Exista el acceso de red Wireless al Sistema.
- ✓ Existen productos asignados a una SubPercha determinada.
- ✓ El número de cajas que se desean egresar o dar de baja, sea menor o igual que el número de cajas que existen en la SubPercha.

Registrar:

La transacción como un movimiento tipo Egreso.

3.2. Diseño de la Arquitectura del Sistema

3.2.1 Arquitectura del Sistema

El Sistema se divide en 3 partes: Cliente, Servidor de Aplicaciones Web y Base de Datos.

Cliente.- Corresponde al Browser desde donde se acceden a los JSP y Servlets que están ubicados en el WAS (Web Application Server ó Servidor de Aplicaciones Web)

Servidor de Aplicaciones Web.- Corresponde al contenedor de EJB, donde se ejecuta la lógica del negocio que está implementada en los Entity Bean del Tipo CMP y SESSION. También se ejecutan en el WAS los JSP y los SERVLET.

Base de Datos.- Las diferentes tablas del modelo entidad-relación, donde está contenida la información.

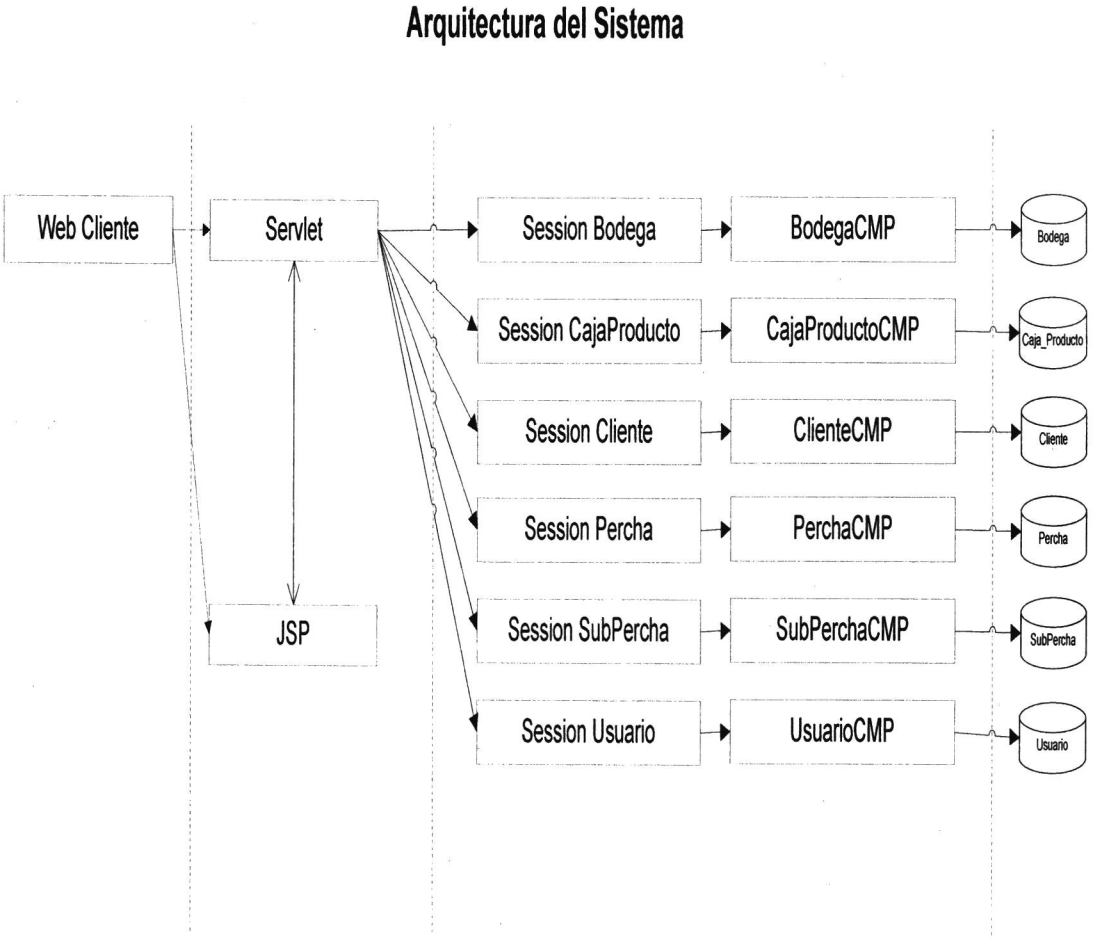


Figura 6: Arquitectura del Sistema

3.2.2 Base de Datos y Esquema Relacional

Diagrama Entidad-Relación

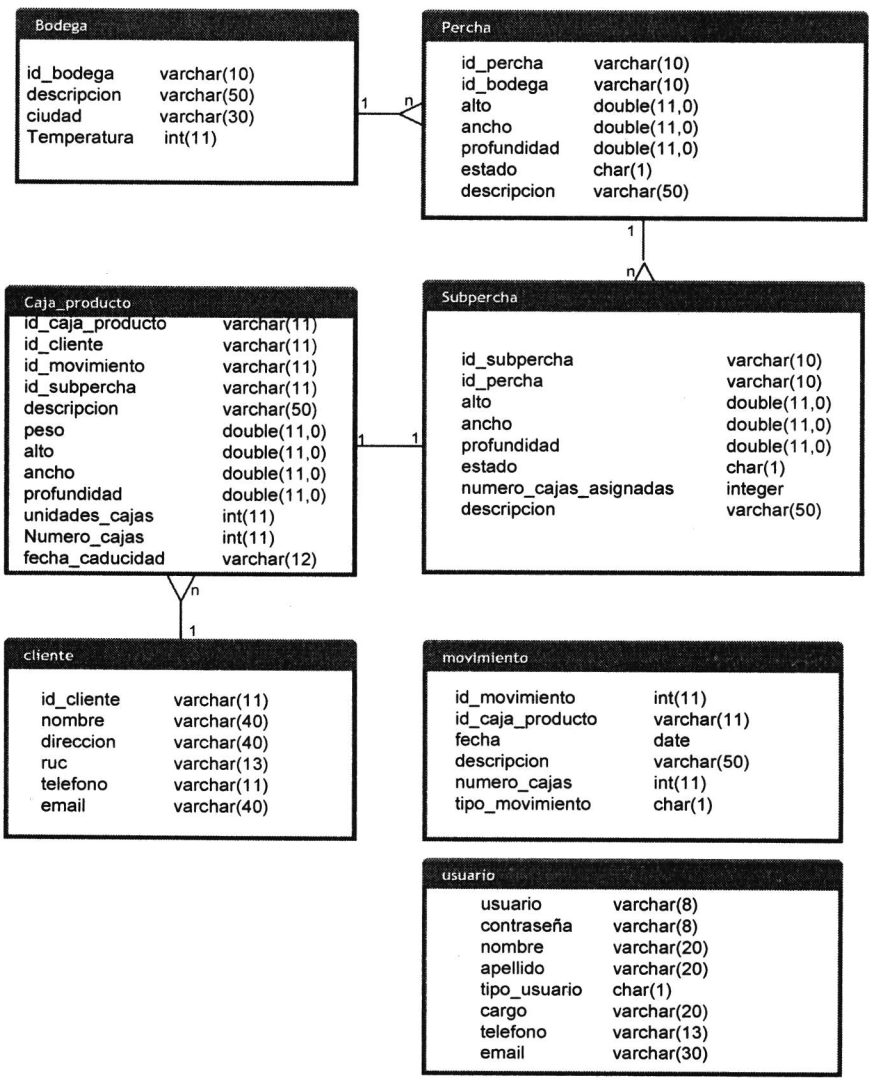


Figura 7: Diagrama Entidad-Relacion

A continuación vamos a visualizar las tablas que conforman la base de datos que se usaron en nuestro sistema:

Bodega	
id_bodega	varchar(10)
descripcion	varchar(50)
ciudad	varchar(30)
temperatura	int(11)

Tabla 1: Tabla Bodega

La tabla Bodega contiene los atributos id_bodega, descripción, ciudad y temperatura. En donde se almacenará la información de la bodega.

Campo	Tipo	Descripción
Id_bodega	varchar(10)	Almacena el identificador de la bodega.
descripción	varchar(50)	Almacena una descripción de la bodega.
ciudad	varchar(30)	Almacena la ciudad de la ciudad a la que pertenece la bodega.
temperatura	int(11)	Almacena la temperatura de la bodega.

Percha	
id_percha	varchar(10)
id_bodega	varchar(10)
alto	double(11,0)
ancho	double(11,0)
profundidad	double(11,0)
estado	char(1)
descripcion	varchar(50)

Tabla 2: Tabla Percha

La tabla Percha contiene los atributos id_percha, id_bodega, alto, ancho, profundidad, estado y descripcion. En donde se almacenará la información de las perchas.

Campo	Tipo	Descripción
id_percha	varchar(10)	Almacena el identificador de la percha.
id_bodega	varchar(10)	Almacena el identificador de la bodega.
alto	double(11,0)	Almacena el alto de la percha.
ancho	double(11,0)	Almacena el ancho de la percha.
profundidad	double(11,0)	Almacena la profundidad de la percha.
estado	char(1)	Almacena el estado de la percha, que puede ser libre u ocupada.
descripción	varchar(50)	Almacena descripción de la percha.

Subpercha	
id_subpercha	varchar(10)
id_percha	varchar(10)
alto	double(11,0)
ancho	double(11,0)
profundidad	double(11,0)
estado	char(1)
numero_cajas_asignadas	integer
descripcion	varchar(50)

Tabla 3: Tabla Subpercha

La tabla SubPercha contiene los atributos id_subpercha, id_percha, alto, ancho, profundidad, estado, numero_cajas_asignadas y descripcion. En donde se almacenará la información de las subperchas.

Campo	Tipo	Descripción
id_subpercha	varchar(10)	Almacena el identificador de la subpercha.
id_percha	varchar(10)	Almacena el identificador de la percha.
alto	double(11,0)	Almacena el alto de la subpercha.
ancho	double(11,0)	Almacena el ancho de la subpercha.
profundidad	double(11,0)	Almacena la profundidad de la subpercha.

estado	char(1)	Almacena el estado de la subpercha, que puede ser libre o ocupada
numero_cajas_asignadas	integer	Almacena descripción de la percha.
descripción	varchar(50)	Almacena descripción de la percha.

Caja_producto	
id_caja_producto	varchar(11)
id_cliente	varchar(11)
id_movimiento	varchar(11)
id_subpercha	varchar(11)
descripcion	varchar(50)
peso	double(11,0)
alto	double(11,0)
ancho	double(11,0)
profundidad	double(11,0)
unidades_cajas	int(11)
Numero_cajas	int(11)
fecha_caducidad	varchar(12)

Tabla 4: Tabla Caja_producto

La tabla Caja_producto contiene los atributos id_caja_producto, id_cliente, id_movimiento, id_subpercha,descripción, peso, alto, ancho, profundidad, unidad_cajas, numero_cajas, fecha_caducidad. En donde se almacenará la información de la caja_productos.

Campo	Tipo	Descripción
id_caja_producto	varchar(11)	Almacena el identificador de caja de producto.
id_cliente	varchar(11)	Almacena el identificador de clientes.
id_movimiento	varchar(11)	Almacena el identificador de movimiento.
id_subpercha	varchar(11)	Almacena el identificador de subperchas.
descripcion	varchar(50)	Almacena una descipción de caja de producto.
peso	double(11,0)	Almacena el peso de la caja producto.
Alto	double(11,0)	Almacena el alto de la caja de producto.
ancho	double(11,0)	Almacena el ancho de la caja de producto.
profundidad	double(11,0)	Almacena la profundidad de la caja de producto.
unidades_cajas	int(11)	Almacena la cantidad de unidades en una caja.
numero_cajas	int(11)	Almacena el numero de cajas de una caja de producto.
fecha_caducidad	varchar(12)	Almacena fecha de caducidad de la caja de producto.

cliente	
id_cliente	varchar(11)
nombre	varchar(40)
direccion	varchar(40)
ruc	varchar(13)
telefono	varchar(11)
email	varchar(40)

Tabla 5: Tabla Cliente

La tabla cliente contiene los atributos id_cliente, nombre, direccion, ruc, telefono, email. En donde se almacenará la información de los clientes.

Campo	Tipo	Descripción
id_cliente	varchar(11)	Almacena el identificador de cliente
nombre	varchar(40)	Almacena el nombre del cliente.
direccion	varchar(40)	Almacena la dirección del cliente.
ruc	varchar(13)	Almacena el ruc del cliente.
telefono	varchar(11)	Almacena el número de telefono del cliente.
email	varchar(40)	Almacena la dirección d correo del cliente.

movimiento	
id_movimiento	int(11)
id_caja_producto	varchar(11)
fecha	date
descripcion	varchar(50)
numero_cajas	int(11)
tipo_movimiento	char(1)

Tabla 6: Tabla Movimiento

La tabla movimiento contiene los atributos id_movimiento, id_caja_producto, fecha, descripción, numero_cajas, tipo_movimiento. En donde se almacenará la información de los movimientos realizados con las caja de productos.

Campo	Tipo	Descripción
id_movimiento	int(11)	Almacena el identificador del movimiento.
id_caja_producto	varchar(11)	Almacena el identificador de la caja de producto.
fecha	date	Almacena la fecha del movimiento.
descripcion	varchar(50)	Almacena la descripción del movimiento realizado.
numero_cajas	int(11)	Almacena el número de cajas en el movimiento.
tipo_movimiento	char(1)	Almacena el tipo de movimiento.

usuario	
usuario	varchar(8)
contraseña	varchar(8)
nombre	varchar(20)
apellido	varchar(20)
tipo_usuario	char(1)
cargo	varchar(20)
telefono	varchar(13)
email	varchar(30)

Tabla 7: Tabla Usuario

La tabla usuario contiene los atributos usuario, contraseña, nombre, apellido, tipo_usuario, cargo, telefono, email. En donde se almacenará la información de los usuarios del sistema.

Campo	Tipo	Descripción
usuario	varchar(8)	Almacena el identificador del usuario.
contraseña	varchar(8)	Almacena la contraseña del usuario.
nombre	varchar(20)	Almacena el nombre registrado para el usuario.
apellido	varchar(20)	Almacena el apellido registrado para el usuario.
tipo_usuario	char(1)	Almacena tipo de usuario, que puede ser limitado o administrador.
cargo	varchar(20)	Almacena el cargo del usuario.

telefono	varchar(13)	Almacena el número de teléfono del usuario.
email	varchar(30)	Almacena la dirección de correo del usuario.

3.2.3 Diagramas de Interacción de Objetos

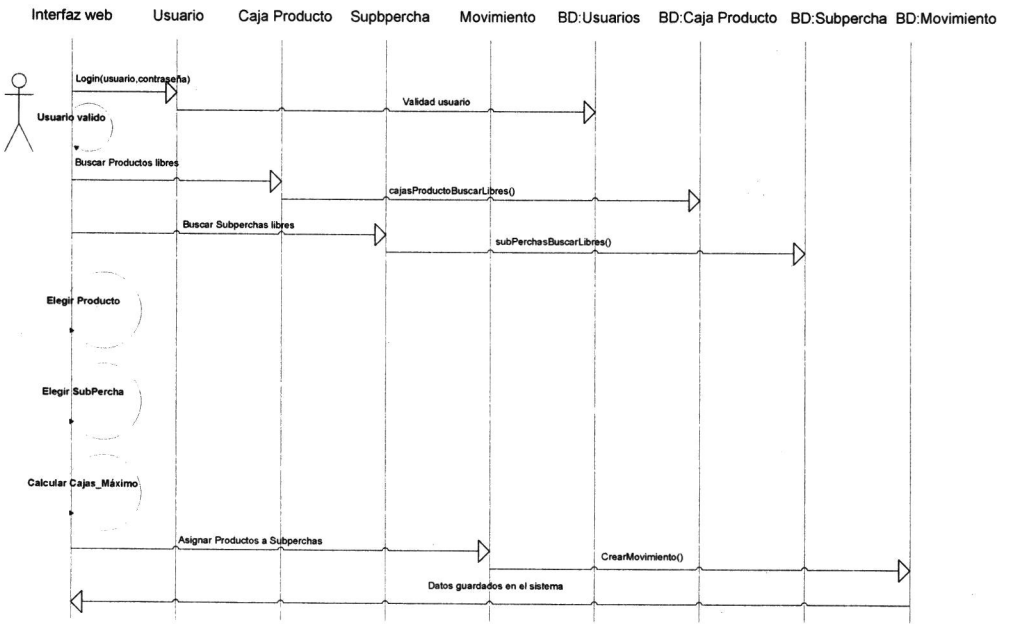


Figura 8: DIO: Asignación de productos a SubPerchas

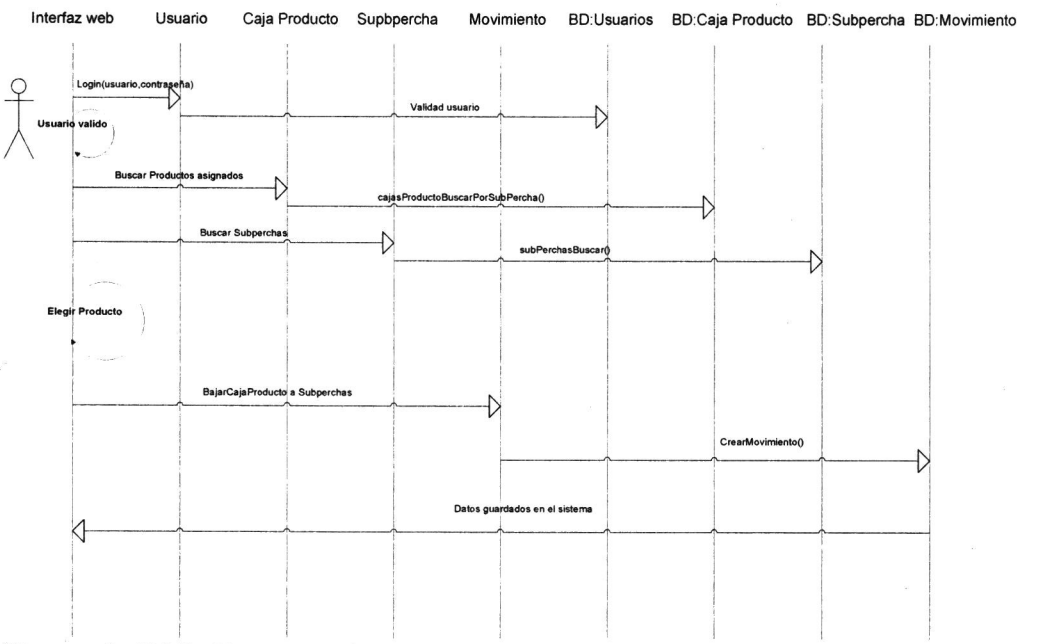


Figura 9: DIO: Proceso de egreso de productos desde subperchas a través de dispositivos PDA.

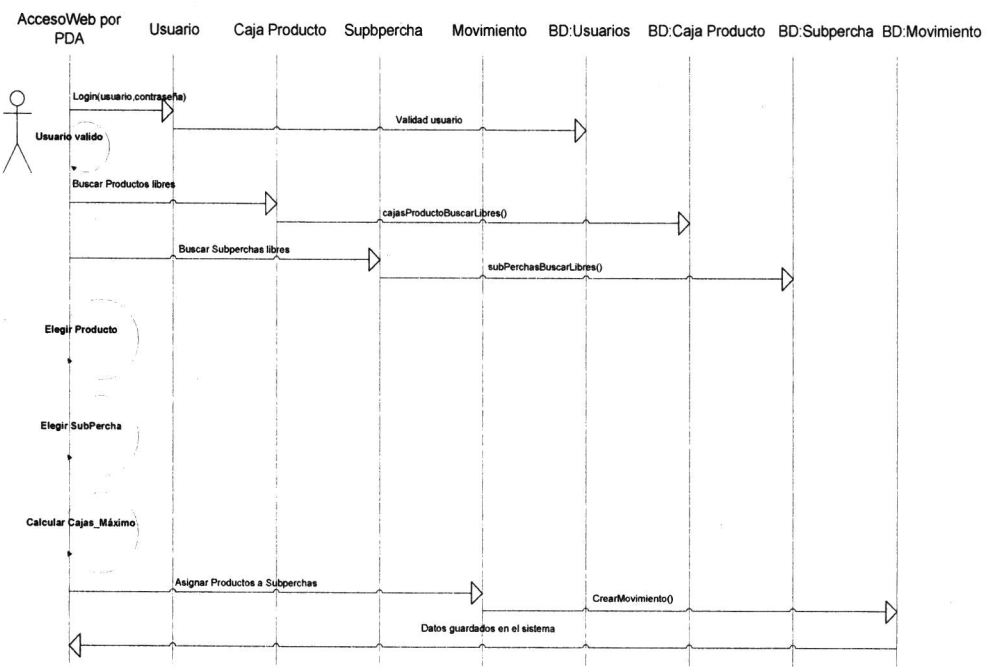


Figura 10: DIO:Proceso de Asignación de Productos a SubPerchas a través de dispositivo PDA.

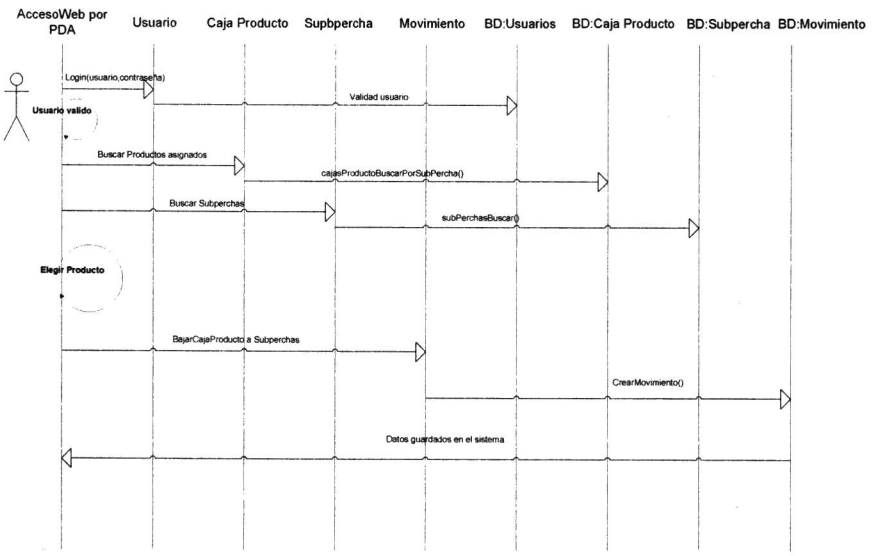


Figura 11: DIO: Proceso de Egreso de Productos desde Subperchas a través de dispositivos PDA.

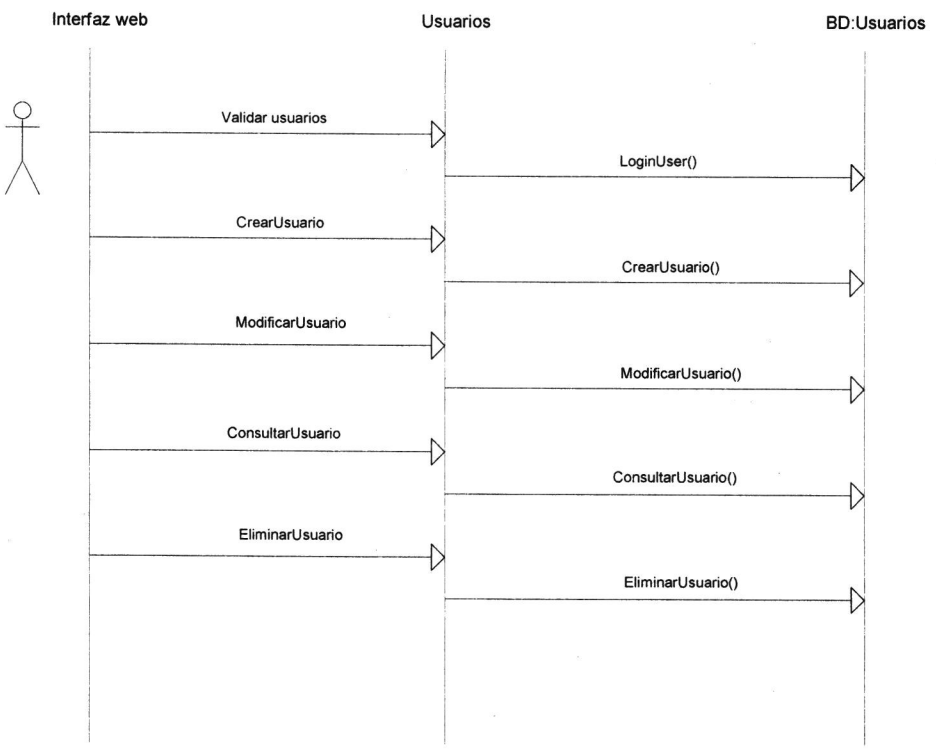


Figura 12: DIO: Mantenimiento de usuario.

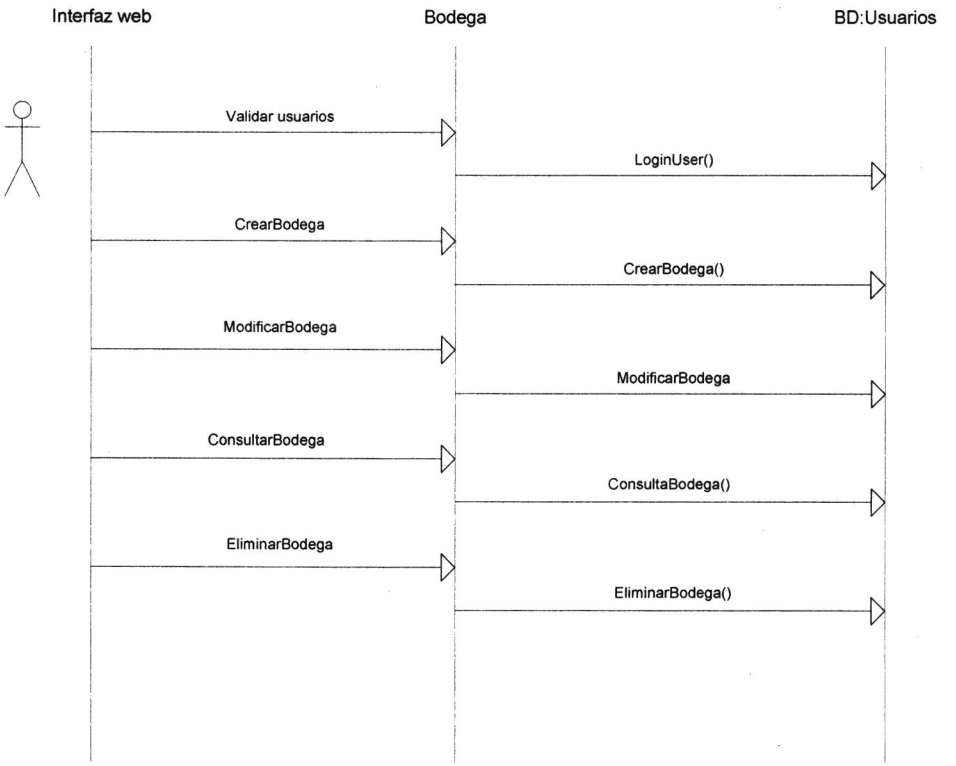


Figura 13: DIO: Mantenimiento de Bodega

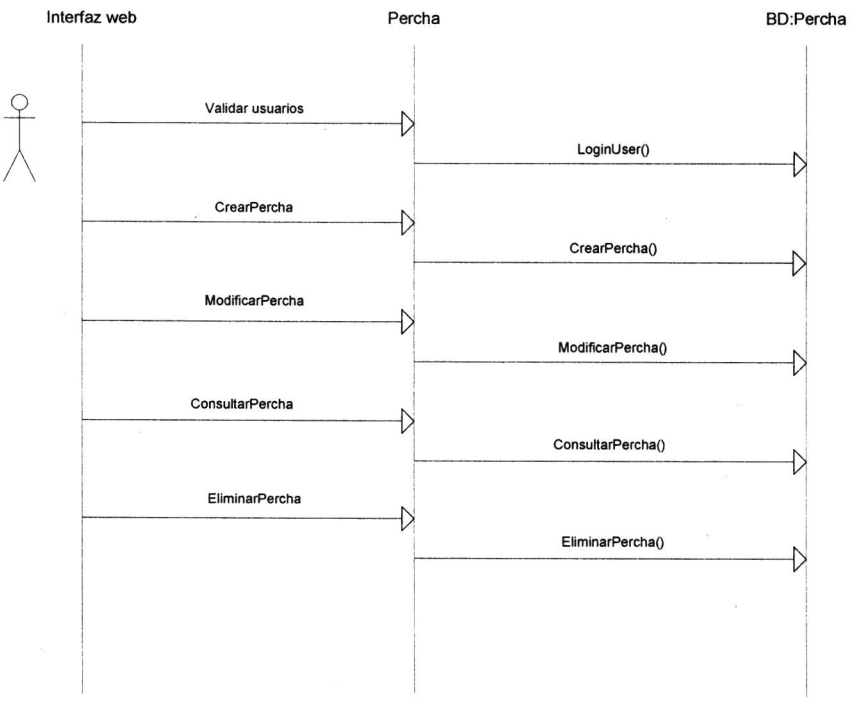


Figura 14: DIO: Mantenimiento de Percha

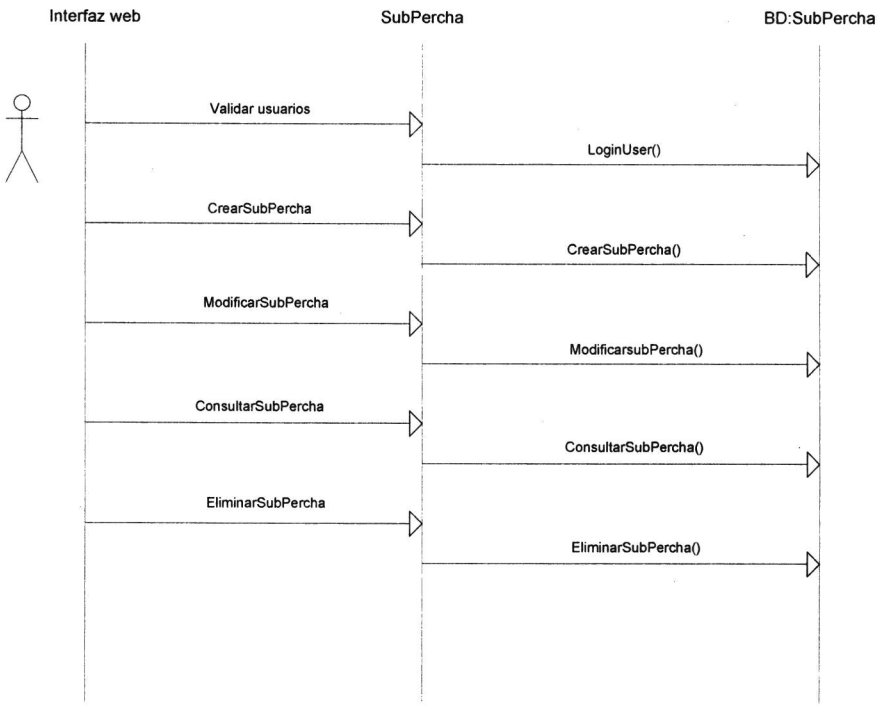


Figura 15: DIO: Mantenimiento de SubPercha

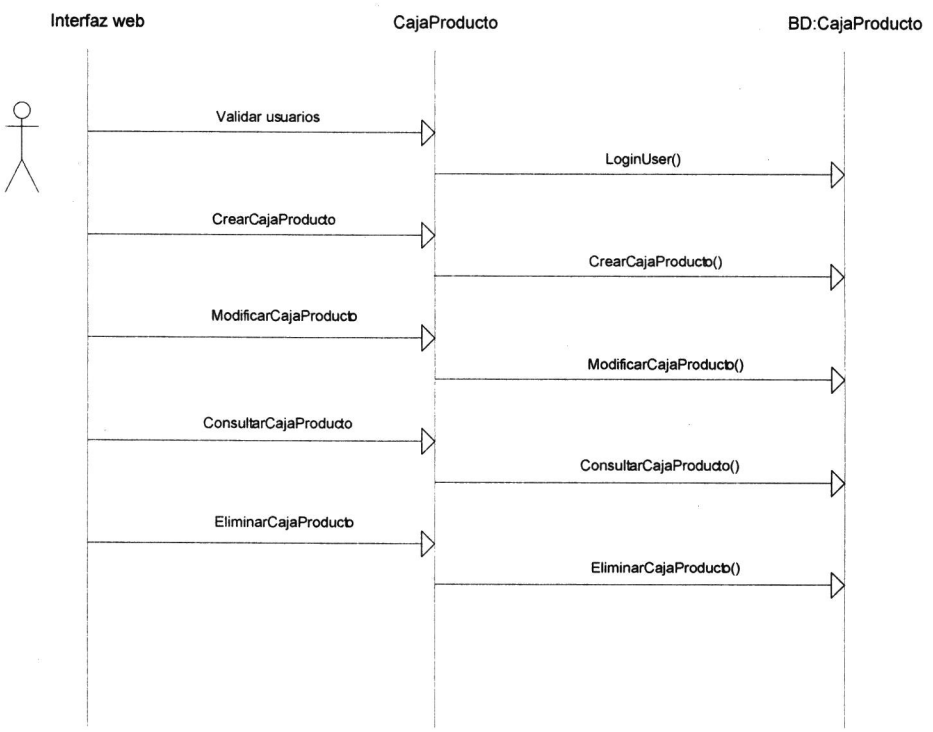


Figura 16: DIO: Mantenimiento de Producto.

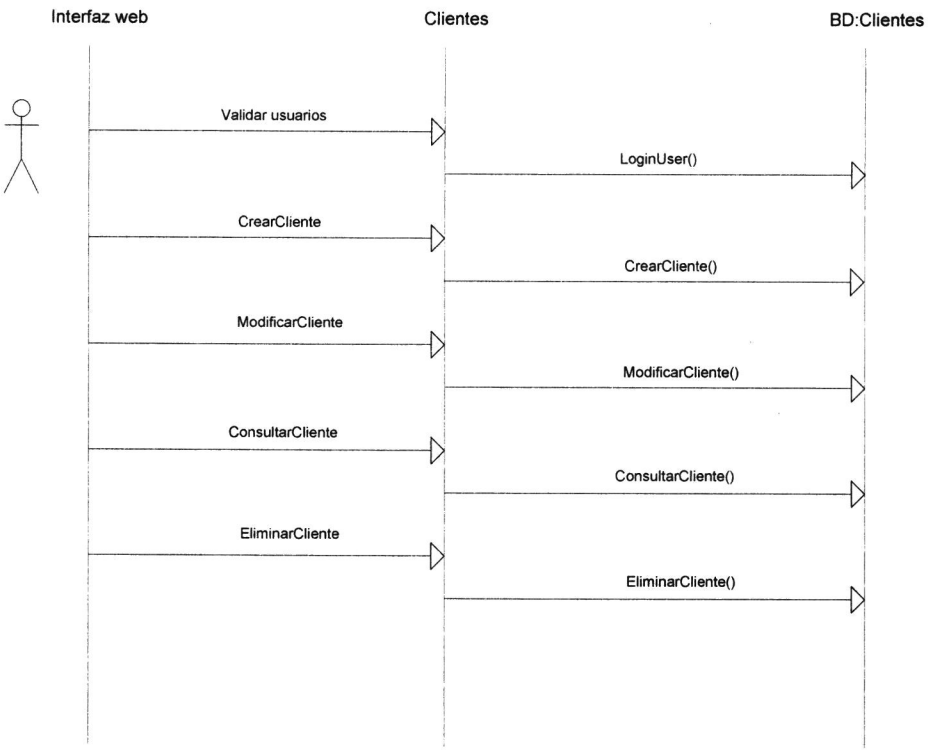


Figura 17: DIO: Mantenimiento de cliente.

3.2.4. Módulos del Sistema

El Sistema fue dividido en 9 módulos para su mejor desarrollo, módulos que están compuestos por varios Enterprise Java Beans (EJB), Servlets y Java Server Pages (JSP); la lista se puede encontrar en el **ANEXO 1: LISTA DE EJBs, SERVLETS Y JSPs**

3.2.4.1 Módulo de Mantenimiento de Usuarios

Este módulo permite crear, modificar, consultar o eliminar información de los usuarios que utilizan el Sistema, dicha información está contenida en la tabla USUARIO de la Base de Datos BODEGAPALM.

El Sistema a través del Módulo de Mantenimiento de Usuarios puede manejar dos perfiles de usuarios:

Perfil Administrador: Se pueden acceder a todas las opciones de los menús del Sistema

Perfil Limitado: Se pueden acceder sólo a las opciones del menú de Productos, así como a las opciones de Subir Cajas a SubPerchas y a Bajar Cajas desde SubPerchas

El módulo de mantenimiento de usuarios está compuesto por las siguientes clases que representan a los EJB y Servlet:

EJB tipo CMP: UsuarioCMPBean.java

Este Enterprise Java Bean interactúa con la información contenida en la tabla USUARIO; es el que hace todo el trabajo de inserción,

actualización o eliminación desde o hacia la tabla USUARIO, a través de Container proveído por el servidor de Aplicaciones Jboss.

Está formado por los métodos "get" y "set" correspondiente a cada campo de la tabla USUARIO.

Como se explicó en el capítulo 1 apartado 1.4.2, en los EJBs tipo CMP, todo el trabajo de conexión hacia las tablas de la base de datos, es realizado por el Contenedor de EJBs, de forma transparente para el programador.

EJB tipo Session StateFul: SessionUsuarioBean.java

Este Session Bean interactúa con el Bean tipo CMP llamado UsuarioCMPBean; a través del Bean de Session se envía a consultar, crear, modificar ó eliminar un registro correspondiente a un Usuario.

Está formado por los siguientes métodos:

usuarioBuscar(String usuario).- Consulta los datos correspondientes al registro de un "usuario" dentro de la tabla USUARIO. Recibe como parámetro el código del usuario

usuarioBuscarTodos( ).- Consulta los datos de todos los registros contenidos en la tabla USUARIO.

usuarioCrear(String usuario, String contrasena, String nombre, String apellido, Character tipo_usuario, String cargo, String telefono, String email).- Crea un registro de "usuario nuevo" o modifica algún registro de usuario existente. Recibe como parámetros: el código de usuario, la clave o contraseña, el nombre, el apellido, el tipo de usuario, el cargo, el teléfono y el email.

Tanto el EJB tipo CMP como el EJB tipo SESSION, forman parte del "Modelo" dentro de la arquitectura MVC (Model View Controler)

Servlet: usuarioServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos de:

Búsqueda de un Usuario existente

Creación de un Nuevo Usuario

Modificación de un Usuario existente

Eliminación de un Usuario existente

Limpiar valores del Java Server Page: Usuarios

Java Server Page (Jsp): Usuarios

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: usuarioServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

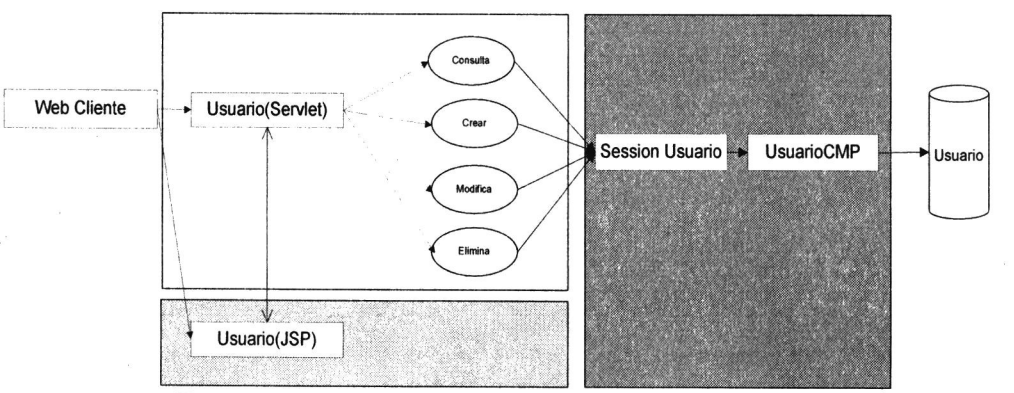


Figura 18: Módulo de Mantenimiento de Usuarios

3.2.4.2 Módulo de Mantenimiento de Bodegas

Este módulo permite crear, modificar, consultar o eliminar información de las Bodegas existentes en el Sistema; dicha información está contenida en la tabla BODEGA de la Base de Datos BODEGAPALM.

El módulo de mantenimiento de Bodegas está compuesto por las siguientes clases que representan a los EJB y Servlet:

EJB tipo CMP: BodegaCMPBean.java

Este Enterprise Java Bean interactúa con la información contenida en la tabla BODEGA; es el que hace todo el trabajo de inserción, actualización o eliminación desde o hacia la tabla BODEGA, a través de Container proveído por el servidor de Aplicaciones Jboss.

Está formado por los métodos "get" y "set" correspondiente a cada campo de la tabla BODEGA.

EJB tipo Session StateFul: SessionBodegaBean.java

Este Session Bean interactúa con el Bean tipo CMP llamado BodegaCMPBean; a través del Bean de Session se envía a consultar, crear, modificar ó eliminar un registro correspondiente a una Bodega.

Está formado por los siguientes métodos:

bodegaBuscar(String id_bodega).- Consulta los datos correspondientes al registro de una "bodega" dentro de la tabla BODEGA. Recibe como parámetro el código de la bodega

bodegaBuscarTodos( ).- Consulta los datos de todos los registros contenidos en la tabla BODEGA.

bodegaCrear(String id_bodega, String descripcion, Integer temperatura, String ciudad).- Crea un registro de "bodega nueva" o modifica algún registro de bodega existente. Recibe como parámetros: el código de la bodega, descripcion, temperatura, y la ciudad.

Tanto el EJB tipo CMP como el EJB tipo SESSION, forman parte del "Modelo" dentro de la arquitectura MVC (Model View Controler)

Servlet: bodegaServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos de:

- ✓ Búsqueda de una Bodega existente
- ✓ Creación de una Nueva Bodega
- ✓ Modificación de una Bodega existente
- ✓ Eliminación de una Bodega existente
- ✓ Limpiar valores del Java Server Page: Bodegas

Java Server Page (Jsp): Bodegas

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: bodegaServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

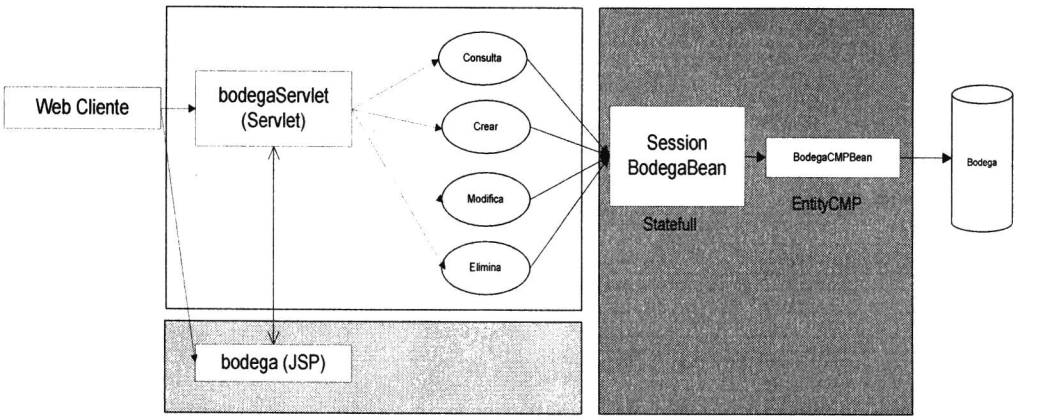


Figura 19: Módulo de Mantenimiento de Bodegas

3.2.4.3 Módulo de Mantenimiento de Perchas

Este módulo permite crear, modificar, consultar o eliminar información de las Perchas existentes en el Sistema; dicha información está contenida en la tabla PERCHA de la Base de Datos BODEGAPALM.

El módulo de mantenimiento de Perchas está compuesto por las siguientes clases que representan a los EJB y Servlet:

EJB tipo CMP: PerchaCMPBean.java

Este Enterprise Java Bean interactúa con la información contenida en la tabla PERCHA; es el que hace todo el trabajo de inserción, actualización o eliminación desde o hacia la tabla PERCHA, a través de Container proveído por el servidor de Aplicaciones Jboss.

Está formado por los métodos "get" y "set" correspondiente a cada campo de la tabla PERCHA.

EJB tipo Session StateFul: SessionPerchaBean.java

Este Session Bean interactúa con el Bean tipo CMP llamado PerchaCMPBean; a través del Bean de Session se envía a crear, modificar ó eliminar el registro correspondiente a una Percha.

Está formado por los siguientes métodos:

perchaBuscar(String id_percha).- Consulta los datos correspondientes al registro de una "percha" dentro de la tabla PERCHA. Recibe como parámetro el código de la percha.

perchaBuscarTodos().- Consulta los datos de todos los registros contenidos en la tabla PERCHA.

perchaBuscarPorBodega(String id_bodega).- Busca las bodegas que pertenecen a una Percha. Recibe como parámetro, el código de la bodega.

perchaCrear(String id_percha, String id_bodega, Double alto, Double ancho, Double profundidad, Character estado, String descripcion).- Crea un registro de "percha nueva" o modifica algún registro de percha existente. Recibe como parámetros: el código de la percha, codigo de la bodega a la que pertenece, el alto, el ancho, la profundidad, el estado inicialmente es Libre, la descripcion.

Tanto el EJB tipo CMP como el EJB tipo SESSION, forman parte del "Modelo" dentro de la arquitectura MVC (Model View Controler)

Servlet: perchaServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos de:

- ✓ Búsqueda de una Percha existente
- ✓ Creación de una Nueva Percha
- ✓ Modificación de una Percha existente
- ✓ Eliminación de una Percha existente
- ✓ Limpiar valores del Java Server Page: Perchas

Java Server Page (Jsp): Percha

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: perchaServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

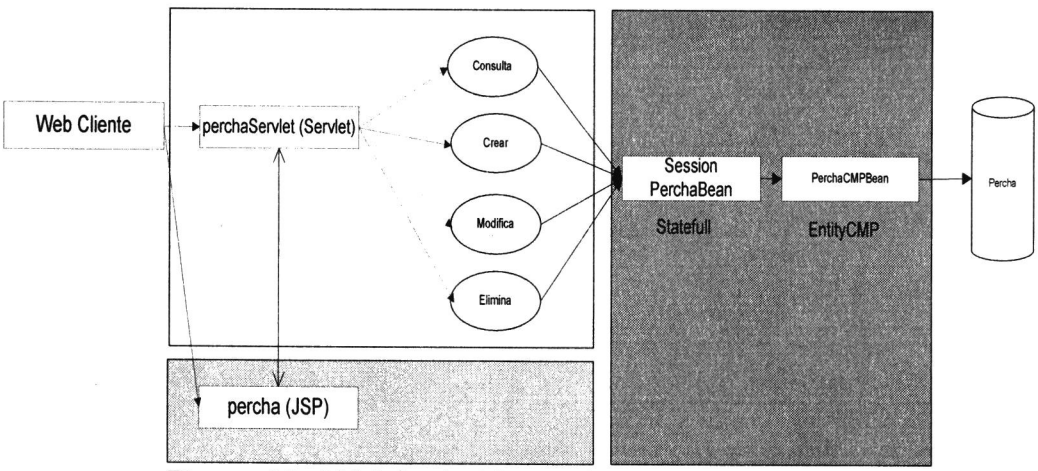


Figura 20: Módulo de Mantenimiento de Perchas

3.2.4.4 Módulo de Mantenimiento de SubPerchas

Este módulo permite crear, modificar, consultar o eliminar información de las SubPerchas existentes en el Sistema; dicha información está contenida en la tabla SUBPERCHA de la Base de Datos BODEGAPALM.

El módulo de mantenimiento de SubPerchas está compuesto por las siguientes clases que representan a los EJB y Servlet:

EJB tipo CMP: SubPerchaCMPBean.java

Este Enterprise Java Bean interactúa con la información contenida en la tabla SUBPERCHA; es el que hace todo el trabajo de inserción, actualización o eliminación desde o hacia la tabla SUBPERCHA, a través de Container proveído por el servidor de Aplicaciones Jboss. Está formado por los métodos "get" y "set" correspondiente a cada campo de la tabla SUBPERCHA.

EJB tipo Session StateFul: SessionSubPerchaBean.java

Este Session Bean sirve interactúa con el Bean tipo CMP llamado SubPerchaCMPBean; a través del Bean de Session se envía a consultar, crear, modificar ó eliminar un registro correspondiente a una SubPercha.

Está formado por los siguientes métodos:

subperchaBuscar(String id_subpercha).- Consulta los datos correspondientes al registro de una "subpercha" dentro de la tabla SUBPERCHA. Recibe como parámetro el código de la subpercha.

subperchaBuscarTodos( ).- Consulta los datos de todos los registros contenidos en la tabla SUBPERCHA.

subperchaBuscarPorPercha(String id_percha).- Busca las Perchas que pertenecen a una SubPercha. Recibe como parámetro, el código de la percha.

subperchaBuscarLibres( ).- Busca todas las subperchas que están libres, es decir, que no se le han asignado cajas de productos.

subperchaCrear(String id_subpercha, String id_percha, Double alto, Double ancho, Double profundidad, Character estado, Integer numero_cajas_asignadas, String descripcion).- Crea un registro de "subpercha nueva" o modifica algún registro de subpercha existente. Recibe como parámetros: el código de la subpercha, código de la percha a la que pertenece, el alto, el ancho, la profundidad, el estado inicialmente es Libre, el número de cajas que se asignarán y, la descripcion.

Tanto el EJB tipo CMP como el EJB tipo SESSION, forman parte del "Modelo" dentro de la arquitectura MVC (Model View Controler)

Servlet: subperchaServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos de:

- ✓ Búsqueda de una SubPercha existente
- ✓ Creación de una Nueva SubPercha
- ✓ Modificación de una SubPercha existente
- ✓ Eliminación de una SubPercha existente
- ✓ Limpiar valores del Java Server Page: SubPercha

Java Server Page (Jsp): SubPercha

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: subperchaServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

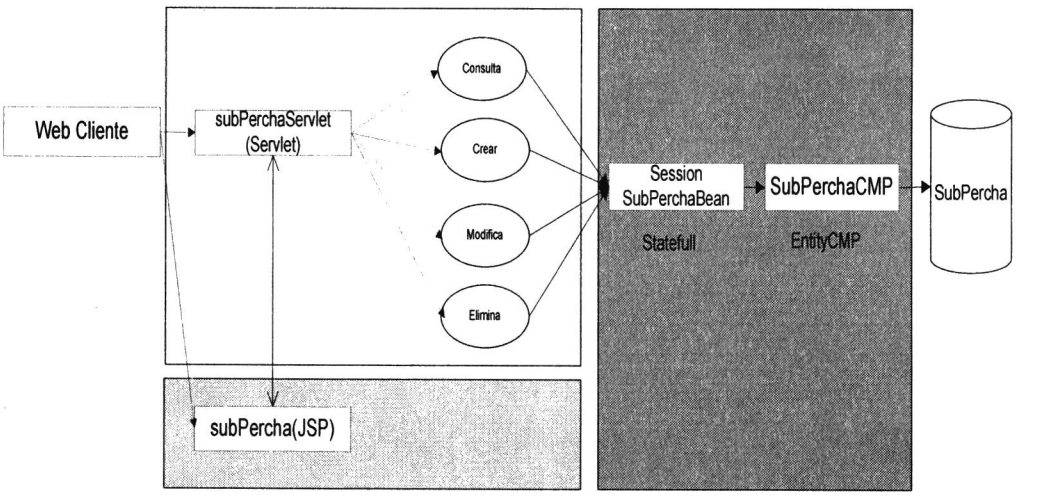


Figura 21: Módulo de Mantenimiento de SubPerchas

3.2.4.5 Módulo de Mantenimiento de Productos

Este módulo permite crear, modificar, consultar o eliminar información de los Productos existentes en el Sistema, los cuáles son representados como Cajas de Productos; dicha información está contenida en la tabla CAJA_PRODUCTO de la Base de Datos BODEGAPALM.

El módulo de mantenimiento de Productos está compuesto por las siguientes clases que representan a los EJB y Servlet:

EJB tipo CMP: CajaProductoCMPBean.java

Este Enterprise Java Bean interactúa con la información contenida en la tabla CAJA_PRODUCTO; es el que hace todo el trabajo de inserción, actualización o eliminación desde o hacia la tabla

CAJA_PRODUCTO, a través de Container proveído por el servidor de Aplicaciones Jboss.

Está formado por los métodos "get" y "set" correspondiente a cada campo de la tabla CAJA_PRODUCTO.

EJB tipo Session StateFul: SessionCajaProductoBean.java

Este Session Bean interactúa con el Bean tipo CMP llamado CajaProductoCMPBean; a través del Bean de Session se envía a consultar, crear, modificar ó eliminar el registro correspondiente a una Caja de Producto.

Está formado por los siguientes métodos:

cajaproductoBuscar(String id_caja_producto).- Consulta los datos correspondientes al registro de una "caja de producto" dentro de la tabla CAJA_PRODUCTO. Recibe como parámetro el código de la caja de producto.

cajaproductoBuscarTodos( ).- Consulta los datos de todos los registros contenidos en la tabla CAJA_PRODUCTO.

cajaproductoBuscarPorSubpercha(String id_subpercha).- Busca las Cajas de Productos que pertenecen a una SubPercha. Recibe como parámetro, el código de la subpercha.

cajaproductoBuscarLibres( ).- Busca todas las cajas de productos que están libres, es decir, que no se han asignado a subpercha.

cajaproductoBuscarAsignadas( ).- Busca todas las cajas de productos que están asignadas, es decir, que si se han asignado a subpercha.

cajaproductoCrear(String id_caja_producto, String id_cliente, String id_movimiento, String id_subpercha, String descripcion, Double peso, Double alto, Double ancho, Double profundidad, Integer unidades_cajas, Integer numero_cajas, String fecha_caducidad).- Crea un registro de "caja de producto nueva" o modifica algún registro de caja de producto existente. Recibe como parámetros: el código de la caja de producto, código del cliente al que pertenece, el código de movimiento, el código de la subpercha a la que pertenece, la descripción de la caja de producto, el peso de la caja, el alto, el ancho, la profundidad, las unidades por caja, el

número de cajas que existen como disponibles y, la fecha de caducidad.

Tanto el EJB tipo CMP como el EJB tipo SESSION, forman parte del "Modelo" dentro de la arquitectura MVC (Model View Controler)

Servlet: cajaproductoServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos de:

- ✓ Búsqueda de una Caja de Producto existente
- ✓ Creación de una Nueva Caja de Producto
- ✓ Modificación de una Caja de Producto existente
- ✓ Eliminación de una Caja de Producto existente
- ✓ Limpiar valores del Java Server Page: Productos

Java Server Page (Jsp): Productos

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: cajaproductoServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

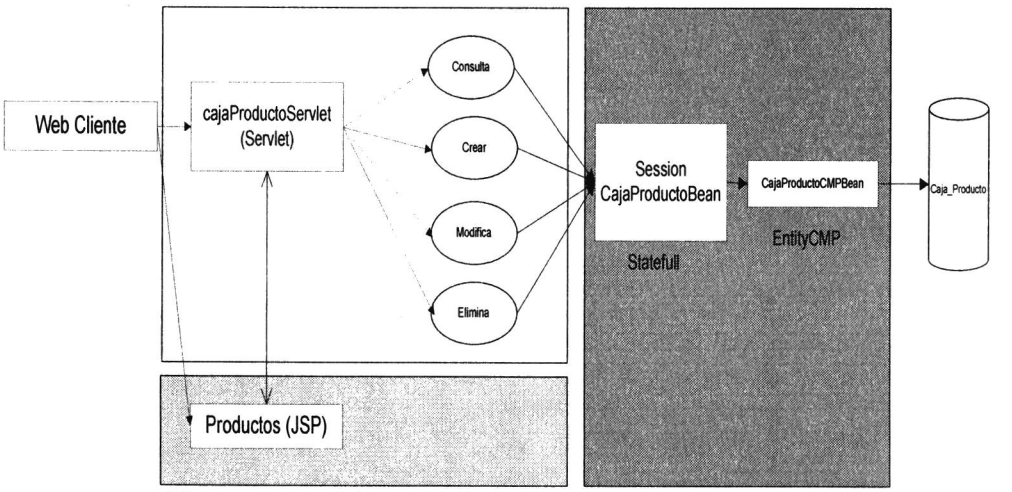


Figura 22: Módulo de Mantenimiento de Productos

3.2.4.6 Módulo de Mantenimiento de Clientes

Este módulo permite crear, modificar, consultar o eliminar información de los Clientes existentes en el Sistema; dicha información está contenida en la tabla CLIENTE de la Base de Datos BODEGAPALM.

El módulo de mantenimiento de Clientes está compuesto por las siguientes clases que representan a los EJB y Servlet:

EJB tipo CMP: ClienteCMPBean.java

Este Enterprise Java Bean interactúa con la información contenida en la tabla CLIENTE; es el que hace todo el trabajo de inserción,

actualización o eliminación desde o hacia la tabla CLIENTE, a través de Container proveído por el servidor de Aplicaciones Jboss.

Está formado por los métodos "get" y "set" correspondiente a cada campo de la tabla CLIENTE.

EJB typo Session Tasteful: SessionClienteBean.java

Este Session Bean interactúa con el Bean tipo CMP llamado ClienteCMPBean; a través del Bean de Session se envía a consultar, crear, modificar ó eliminar el registro correspondiente a un Cliente.

Está formado por los siguientes métodos:

clienteBuscar(String id_cliente).- Consulta los datos correspondientes al registro de un "cliente" dentro de la tabla CLIENTE. Recibe como parámetro el código del cliente.

clienteBuscarTodos().- Consulta los datos de todos los registros contenidos en la tabla CLIENTE.

clienteCrear(String id_cliente, String nombre, String direccion, String ruc, String telefono, String email).- Crea un registro de "cliente nuevo" o modifica algún registro de cliente existente. Recibe

como parámetros: el código del cliente, nombre del cliente, la direccion, el número de Ruc, el número de teléfono, y el email.

Tanto el EJB tipo CMP como el EJB tipo SESSION, forman parte del "Modelo" dentro de la arquitectura MVC (Model View Controler)

Servlet: clienteServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos de:

- ✓ Búsqueda de un Cliente existente
- ✓ Creación de un Nuevo Cliente
- ✓ Modificación de un Cliente existente
- ✓ Eliminación de un Cliente existente
- ✓ Limpiar valores del Java Server Page: Clientes

Java Server Page (Jsp): Clientes

Esta página JSP es la que sirve de interacción entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: clienteServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

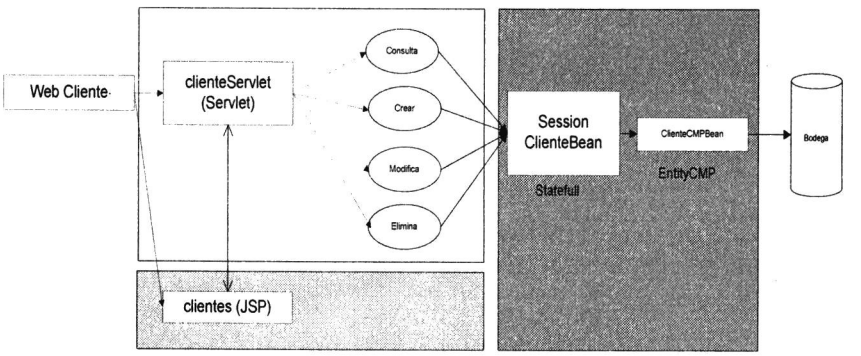


Figura 23: Módulo de Mantenimiento de Clientes

3.2.4.7 Módulo de Asignación de Productos a SubPerchas

Este módulo efectúa la asignación de cajas de productos a las SubPerchas, respetando los valores máximos de cajas de productos que pueden almacenar las mismas.

Está compuesto por las siguientes clases que representan a los EJB y Servlet:

EJB tipo CMP: MovimientoCajasBean.java

Este EJB es utilizado adicionalmente por los módulos: de "Egreso de Productos desde SubPerchas" y de "Reportes".

Registra la transacción de ingreso de Cajas de Productos hacia las SubPerchas. Dichas transacciones son ingresadas en la tabla MOVIMIENTO.

EJB tipo Session StateFul: SessionMovimientoCajasBean.java

Interactúa con el Bean tipo CMP llamado MovimientoCajasBean; a través del Bean de Session se registran los ingresos o asignaciones de Cajas de Productos en la tabla MOVIMIENTO.

Está formado por los siguientes métodos:

movimientocajasCrear(Integer id_movimiento, String id_caja_producto, Date fecha, String descripcion, Integer numero_cajas, Character tipo_movimiento).- Crea un registro de "movimiento nuevo" . Recibe como parámetros: el código de movimiento, el código de la caja de producto, la fecha, la descripción de la caja de producto, el numero de cajas que se mueven, y el tipo de movimiento que puede ser de ingreso "I" o de egreso "E".

movimientoBuscarTodos( ).- Busca todos los registros de movimientos de cajas de productos que se han realizado, ya sea de Ingresos como de Egresos.

movimientoBuscarIngreso(Character tipo).- Busca los registros de movimientos realizados. Recibe como parámetro el tipo de movimiento, puede ser de Ingreso "I" o de Egreso "E".

movIngrEgrFecha(Character tipo, Date fecha1, Date fecha2).- Busca los registros de movimientos realizados ya sean de ingresos como de egresos, en un rango de fechas determinado. Recibe como parámetos: el tipo de movimiento, la fecha inicial y la fecha final.

movIngrEgrProd(Character tipo, String id_caja_producto).- Busca los registros de los movimientos de una caja de producto específica, ya sea de ingreso o de egreso. Recibe como parámetros, el tipo de movimiento y el código de la caja de producto.

Este EJB es utilizado adicionalmente por los módulos de "Egreso de Productos desde SubPerchas" y de "Reportes"

Tanto el EJB tipo CMP como el EJB tipo SESSION, forman parte del "Modelo" dentro de la arquitectura MVC (Model View Controler)

Servlet: asignarcajasServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos de:

- ✓ Búsqueda de Cajas de Productos a subir a SubPerchas
- ✓ Cálculo de número de cajas permitidas para subir a las SubPerchas
- ✓ Asignar Cajas de Productos a SubPerchas, respetando el cálculo de cajas máximas permitidas.

Ver ANEXO 2: PSEUDOCÓDIGO PARA ASIGNAR CAJAS DE PRODUCTOS A SUBPERCHAS.

Servlet: asignarpalmServlet.java (Válido solo para acceso desde PDA)

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos desde el dispositivo PDA para:

- ✓ Búsqueda de Cajas de Productos a subir a SubPerchas.
- ✓ Cálculo de número de cajas permitidas para subir a las SubPerchas.
- ✓ Asignar Cajas de Productos a SubPerchas, respetando el cálculo de cajas máximas permitidas.

Java Server Page (Jsp): AsignarProductos

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: asignarcajasServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

Java Server Page (Jsp): AsignarProductosPalm (Válido solo para acceso desde PDA)

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema, desde dispositivo PDA) y el Servlet. Desde esta página se envían los requerimientos al Servlet: asignarpalmServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

3.2.4.8 Módulo de Egreso de Productos desde SubPerchas

Este módulo efectúa el egreso de las cajas de productos desde las SubPerchas, considerando el número de cajas existentes en las mismas.

Está compuesto por las siguientes clases que representan a los EJB y Servlet:

EJB tipo CMP: MovimientoCajasBean.java

Este EJB es utilizado adicionalmente por los módulos: de "Asignación de Productos a SubPerchas" y de "Reportes".

Sirve para registrar la transacción de egreso de Cajas de Productos desde las SubPerchas. Tales transacciones son ingresadas en la tabla MOVIMIENTO.

EJB tipo Session StateFul: SessionMovimientoCajasBean.java

Este Session Bean interactúa con el Bean tipo CMP llamado MovimientoCajasBean; a través del Bean de Session se registran los egresos o de Cajas de Productos en la tabla MOVIMIENTO.

También es utilizado adicionalmente por los módulos de "Asignación de Productos a SubPerchas" y de "Reportes"

Tanto el EJB tipo CMP como el EJB tipo SESSION, forman parte del "Modelo" dentro de la arquitectura MVC (Model View Controler)

Servlet: bajarcajasServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos de:

- ✓ Búsqueda de Cajas de Productos a bajar desde SubPerchas.
- ✓ Egreso de Cajas de Productos, considerando el número de Cajas existentes en las SubPerchas.

Ver ANEXO 3: PSEUDOCÓDIGO PARA BAJAR CAJAS DESDE SUBPERCHAS.

Servlet: bajarpalmServlet.java (Válido solo para acceso desde PDA)

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION. A través del Servlet, se envían los requerimientos desde el dispositivo PDA para:

- ✓ Búsqueda de Cajas de Productos a bajar desde SubPerchas
- ✓ Egreso de Cajas de Productos, considerando el número de Cajas existentes en las SubPerchas.

Java Server Page (Jsp): QuitarProductos

Esta página JSP es la que sirve de interacción entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: bajarcajasServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

Java Server Page (Jsp): QuitarProductosPalm (Válido solo para acceso desde PDA)

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema, desde dispositivo PDA) y el Servlet. Desde esta página se envían los requerimientos al Servlet: bajarpalmServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

3.2.4.9 Módulo de Reportes

Permite desplegar los movimientos de los registros de Cajas de Productos, realizados en los procesos de asignación y egreso de los mismos en las SubPerchas.

Está compuesto por dos Servelets:"reporteProdServlet" y "reporteMovServlet", que son utilizados para obtener la información a

través del SessionBean:"SessionMovimientoCajasBean" descrito anteriormente.

Servlet: reporteProdServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION: MovimientoCajasBean y SessionMovimientoCajasBean respectivamente.

A través del Servlet, se envían los requerimientos para:

Buscar las Cajas de Productos que se han Ingresado o se han dado de Baja en una SubPercha específica; a través del método: movIngrEgrProd que se encuentra en el Bean de Session: SessionMovimientoCajasBean

Servlet: reporteMovServlet.java

Este Servlet sirve de controlador de requerimientos y respuestas que se hacen desde y hacia los EJBs tipo CMP y SESSION: MovimientoCajasBean y SessionMovimientoCajasBean respectivamente.

A través del Servlet, se envían los requerimientos para:

Buscar las Cajas de Productos que se han Ingresado o se han dado de Baja en una SubPercha específica, en una fecha (ó rango de fechas) determinada; a través del método: movIngrEgrFecha que se encuentra en el Bean de Session: SessionMovimientoCajasBean

Java Server Page (Jsp): ReporteProdSub

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: reporteProdServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

Java Server Page (Jsp): ReporteIngrEgrSub

Esta página JSP es la que sirve de nexo entre el usuario (actor que utiliza el sistema) y el Servlet. Desde esta página se envían los requerimientos al Servlet: reporteMovServlet. Dentro de la arquitectura MVC, esta página es la que sirve de "Vista" o "View"

3.2.5 Interacción Hombre – Máquina

En esta sección se detalla la forma en que el usuario interactúa con el Sistema; describiéndose cada pantalla y su funcionalidad.

Pantalla de Login

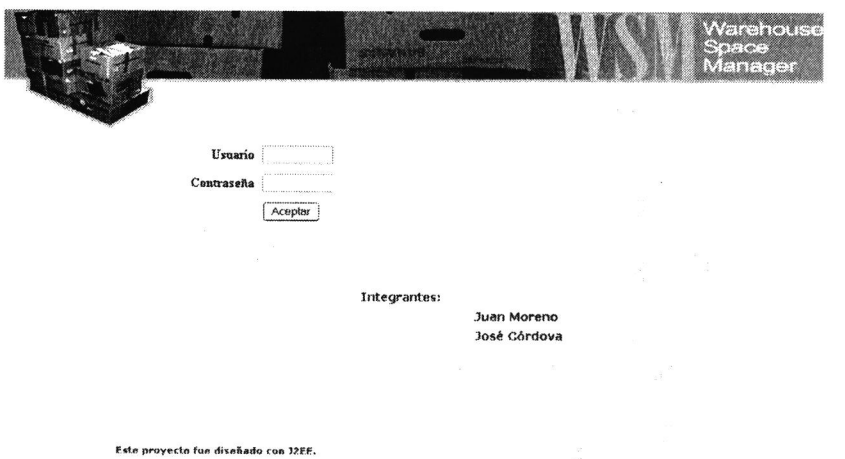


Figura 24: Login al Sistema

La pantalla de Login accede al Sistema; validándose el usuario y contraseña. Está formada por los siguientes controles:

TextBox de usuario.- Permite ingresar el código o username

TextBox de Contraseña.- Permite ingresar la clave o password

Botón Submit Aceptar: Permite enviar a validar el código (username) y clave de acceso, para ingresar a las opciones del Sistema.

Pantalla de Login para dispositivo PDA

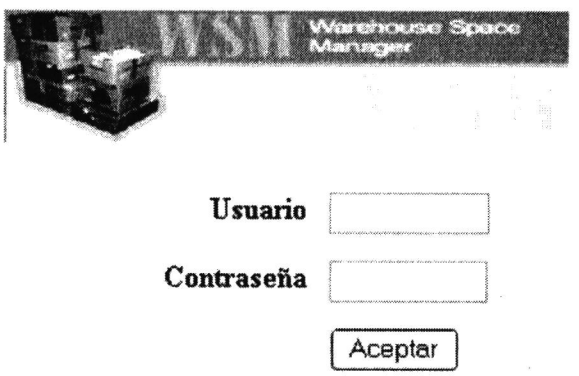


Figura 25: Login al Sistema para dispositivos PDA

Debido a que el PDA: Palm Tungsten C, tiene una resolución máxima de 320x320 pixeles, se rediseñó la pantalla de Login para que pueda ingresar en el tamaño de la pantalla del dispositivo móvil antes indicado.

La pantalla de Login sirve accede al Sistema; validándose el usuario y contraseña. Está formada por los siguientes controles:

TextBox de usuario.- Permite ingresar el código o username

TextBox de Contraseña.- Permite ingresar la clave o password.

Botón Submit Aceptar.- Permite enviar a validar el código (username) y clave de acceso, para ingresar a las opciones del Sistema.

Pantalla de Selección de Opciones (Perfil Administrador)

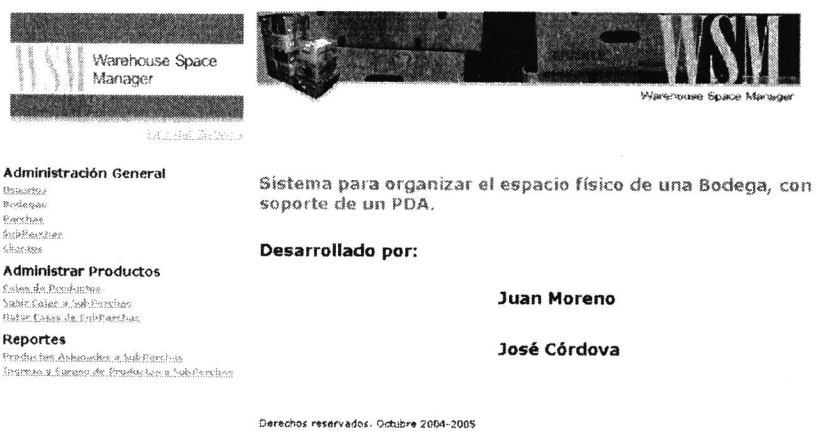


Figura 26: Selección de Opciones (Perfil Administrador)

Luego de que se ha validado el usuario y contraseña; si este usuario tiene perfil de administrador, se despliega la pantalla de selección de opciones con todas las opciones habilitadas por ser perfil de Administrador. Esta pantalla está dividida en tres partes llamadas Frames:

Frame Izquierdo, donde se encuentran las opciones del sistema.

En este Frame, las opciones se encuentran divididas en tres secciones que son:

Administración General.- Se encuentran las opciones que permiten la administración de: Usuarios, Bodegas, Perchas, SubPerchas y Clientes.

Administrar Productos.- Se encuentran las opciones que permiten: el mantenimiento de las Cajas de Productos, Subir Cajas a SubPerchas y Bajar Cajas de SubPerchas.

Reportes.- Se encuentran las opciones correspondientes a los reportes existentes en el Sistema, mismos que son: Productos Asignados a SubPerchas e, Ingreso y Egreso de Productos a SubPerchas.

Frame Superior, donde se encuentra el Título pseudónimo del Sistema, que es: Warehouse Space Manager (Administrador de Espacios de Bodega)

Frame Central, donde se despliegan el contenido o páginas de cada una de las opciones del Sistema. Inicialmente se muestra una página de inicio que tiene sólo la descripción del Sistema y por quiénes fue desarrollado.

Pantalla de Selección de Opciones (Perfil Limitado)

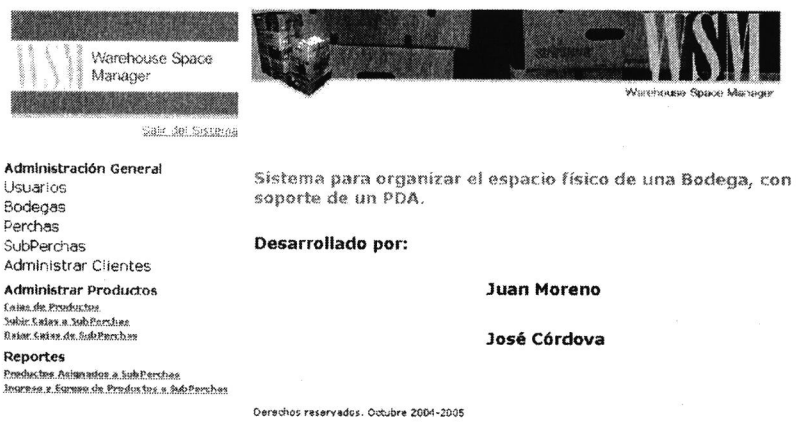


Figura 27: Selección de Opciones (Perfil Limitado)

Luego de que se ha validado el usuario y contraseña; si este usuario tiene perfil de usuario Limitado, se despliega la pantalla de selección de opciones solo con las opciones de administración de Productos y Reportes habilitadas, las demás opciones aparecen deshabilitadas, por ser perfil Limitado. Esta pantalla está dividida en tres partes llamadas Frames:

Frame Izquierdo, donde se encuentran las opciones del sistema.

En este Frame, las opciones se encuentran divididas en tres secciones que son:

Administración General (Desahabilitado).- Se encuentran las opciones que permiten la administración de: Usuarios, Bodegas, Perchas, SubPerchas y Clientes. En el perfil Limitado, aparecen deshabilitadas.

Administrar Productos (Habilitado).- Se encuentran las opciones que permiten: el mantenimiento de las Cajas de Productos, Subir Cajas a SubPerchas y Bajar Cajas de SubPerchas.

Reportes (Habilitado).- Se encuentran las opciones correspondientes a los reportes existentes en el Sistema, mismos que son: Productos Asignados a SubPerchas e, Ingreso y Egreso de Productos a SubPerchas.

Frame Superior, donde se encuentra el Título pseudónimo del Sistema, que es: Warehouse Space Manager (Administrador de Espacios de Bodega)

Frame Central, donde se muestra el contenido o páginas de cada una de las opciones del Sistema, que se encuentran en el Frame Izquierdo. Inicialmente se muestra una página de inicio que tiene sólo la descripción del Sistema y por quiénes fue desarrollado.

Pantalla de Selección de Opciones para Dispositivo PDA

Figura 28: Selección de Opciones para Dispositivos PDA

Debido a que el PDA: Palm Tungsten C, tiene una resolución máxima de 320x320 pixeles, se rediseñó la pantalla de Selección de Opciones para que pueda ingresar en el tamaño de la pantalla del dispositivo móvil antes indicado.

Esta pantalla está dividida en dos partes llamadas Frames:

Frame Izquierdo, donde se encuentran las opciones del sistema.

En este Frame, las opciones se encuentran habilitadas son las siguientes: Administrar Cajas, Subir Cajas y Bajar Cajas

Frame Central, donde se muestra el contenido o páginas de cada una de las opciones del Sistema, existentes en el Frame Izquierdo. Inicialmente se despliega una página de inicio que tiene sólo las iniciales del Sistema y por quiénes fue desarrollado.

Pantalla de Administración de Usuarios

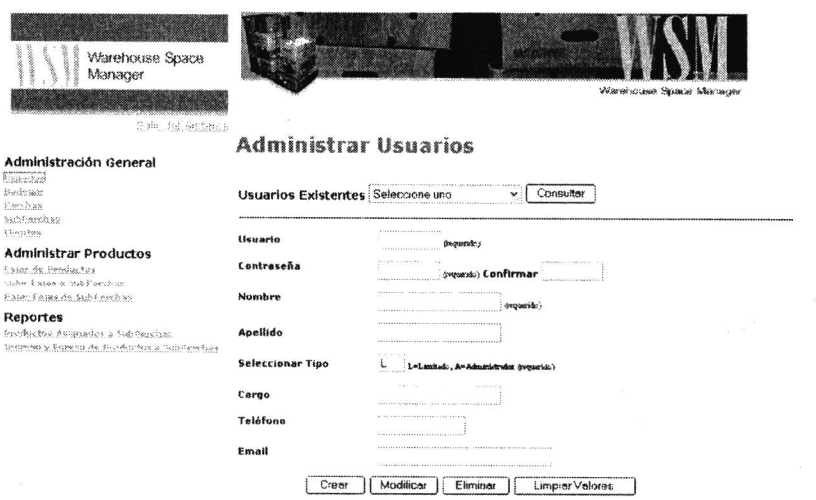


Figura 29: Administración de usuarios

La pantalla de Administración de Usuarios, denominada "Administrar Usuarios", está formada por los siguientes controles:

PullDown de Usuarios Existentes.- En este control se almacenan todos los códigos de usuarios existentes en el Sistema, para poder hacer las consultas de la información de los mismos.

Botón Submit para Consultar.- Este control permite consultar los Datos de un usuario existente en la tabla USUARIO.

TextBox de Usuario.- Permite ingresar o visualizar el código o username del Usuario

TextBox de Contraseña.- Permite ingresar la contraseña o password del Usuario.

TextBox de Confirmar.- Permite ingresar la confirmacion de la contraseña.

TextBox de Nombre.- Permite ingresar o visualizar el nombre del Usuario.

TextBox de Apellido.- Permite ingresar o visualizar el apellido del Usuario.

TextBox de Tipo.- Permite ingresar el tipo del Usuario, puede ser "A" que se refiere a "Administrador" ó "L" que se refiere a "Limitado". No se acepta ningún otro carácter.

TextBox de Cargo.- Permite ingresar o visualizar el cargo que tiene el Usuario.

TextBox de Teléfono.- Permite ingresar o visualizar el teléfono que tiene el Usuario.

TextBox de Email.- Permite ingresar o visualizar el email que tiene el Usuario.

Botón Submit para Crear.- Permite Crear un registro de Usuario nuevo en la tabla USUARIO. Se envía el requerimiento al servlet usuarioServlet.

Botón Submit para Modificar.- Permite Modificar el contenido de un registro de usuario existente en la tabla USUARIO. Se envía el requerimiento al servlet usuarioServlet.

Botón Submit para Eliminar.- Permite Eliminar el registro de un usuario existente en la tabla USUARIO. Se envía el requerimiento al servlet usuarioServlet.

Botón Submir para Limpiar Valores.- Permite borrar los valores de los controles de esta pantalla. Se envía el requerimiento al servlet usuarioServlet.

Pantalla de Administración de Bodegas.

Figura 30: Administración de Bodegas

La pantalla de Administración de Bodegas, denominada "Administrar Bodega", está formada por los siguientes controles:

PullDown de Bodegas Existentes.- En este control se almacenan todos los códigos de bodegas existentes en el Sistema, para poder hacer las consultas de la información de los mismos.

Botón Submit para Consultar.- Permite consultar los datos de una bodega existente en la tabla BODEGA del Sistema.

TextBox de Código de Bodega.- Permite ingresar o visualizar el código de una Bodega.

TextBox de Temperatura.- Permite visualizar, ingresar ó modificar la temperatura promedio que tendrá la Bodega. Solo se aceptan datos numéricos.

TextBox de Ciudad.- Permite visualizar, ingresar ó modificar la ciudad a la que pertenece la Bodega.

TextBox de Descripcion.- Permite visualizar, ingresar ó modificar la descripción de la Bodega, es decir, un detalle de qué va almacenar.

Botón Submit para Crear.- Permite Crear un registro de una Bodega nueva en la tabla BODEGA. Se envía el requerimiento al servlet bodegaServlet.

Botón Submit para Modificar.- Permite Modificar el contenido de un registro de una Bodega existente en la tabla BODEGA. Se envía el requerimiento al servlet bodegaServlet.

Botón Submit para Eliminar.- Permite Eliminar el registro de una Bodega existente en la tabla BODEGA. Se envía el requerimiento al servlet bodegaServlet.

Botón Submir para Limpiar Valores.- Permite borrar los valores de los controles de esta pantalla. Se envía el requerimiento al servlet bodegaServlet.

Pantalla de Administración de Perchas.

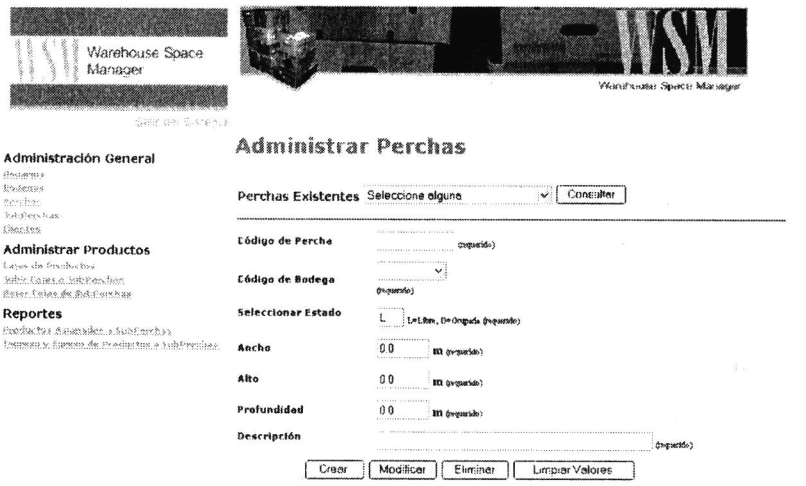


Figura 31: Administración de Perchas

La pantalla de Administración de Perchas, denominada "Administrar Perchas", está formada por los siguientes controles:

PullDown de Perchas Existentes.- En este control se almacenan todos los códigos de Pechas existentes en el Sistema, para poder hacer las consultas de la información de las mismos.

Botón Submit para Consultar.- Permite consultar los Datos de una Percha existente en la tabla PERCHA del Sistema

TextBox de Código de Percha.- Permite ingresar o visualizar el código de una Percha.

PullDown de Bodegas Existentes.- En este control se almacenan todos los códigos de Bodegas existentes en el Sistema. Sirve para seleccionar la Bodega a la que pertenecerá la Percha cuando se cree un registro nuevo de Percha o se modifique un registro de Percha existente.

TextBox de Estado.- Permite visualizar, ingresar ó modificar el estado que tiene una Percha, puede ser "L" , que se refiere a "Libre" ó puede ser "O", que se refiere a "Ocupado". No se acepta ningún otro carácter.

TextBox de Ancho.- Permite visualizar, ingresar ó modificar el Ancho que tiene la Percha. Solo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Alto.- Permite visualizar, ingresar ó modificar el Alto que tiene la Percha. Sólo se aceptan valores numéricos

con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Profundidad.- Permite visualizar, ingresar ó modificar la Profundidad que tiene la Percha. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Descripcion.- Permite visualizar o desplegar la descripción de la Percha, es decir, un detalle de qué va almacenar.

Botón Submit para Crear.- Permite Crear un registro de una Percha nueva en la tabla PERCHA. Se envía el requerimiento al servlet perchaServlet.

Botón Submit para Modificar.- Permite Modificar el contenido de un registro de una Percha existente en la tabla PERCHA. Se envía el requerimiento al servlet perchaServlet.

Botón Submit para Eliminar.- Permite Eliminar el registro de una Percha existente en la tabla PERCHA. Se envía el requerimiento al servlet perchaServlet.

Botón Submir para Limpiar Valores.- Permite borrar los valores de los controles de esta pantalla. Se envía el requerimiento al servlet perchaServlet.

Pantalla de Administración de SubPerchas.

Figura 32: Administración de Subperchas

La pantalla de Administración de SubPerchas, denominada "Administrar SubPerchas", está formada por los siguientes controles:

PullDown de SubPerchas Existentes.- En este control se almacenan todos los códigos de SubPechas existentes en el Sistema, para poder hacer las consultas de la información de las mismos.

Botón Submit para Consultar.- Permite consultar los Datos de una SubPercha existente en la tabla SUBPERCHA del Sistema

TextBox de Código de SubPercha.- Permite ingresar o visualizar el código de una SubPercha.

PullDown de Perchas Existentes.- En este control se almacenan todos los códigos de Perchas existentes en el Sistema. Sirve para seleccionar la Percha a la que pertenecerá la SubPercha cuando se cree un registro nuevo de SubPercha o se modifique un registro de SubPercha existente.

TextBox de Ancho.- Permite visualizar, ingresar ó modificar el Ancho que tiene la SubPercha. Sólo se aceptan valores

numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Alto.- Permite visualizar, ingresar ó modificar el Alto que tiene la SubPercha. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Profundidad.- Permite visualizar, ingresar ó modificar la Profundidad que tiene la SubPercha. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Descripcion.- Permite ingresar o visualizar la descripción de la SubPercha, es decir, un detalle de qué va almacenar.

Botón Submit para Crear.- Permite Crear un registro de una SubPercha nueva en la tabla SUBPERCHA. Se envía el requerimiento al servlet subperchaServlet.

Botón Submit para Modificar.- Permite Modificar el contenido de un registro de una SubPercha existente en la tabla SUBPERCHA. Se envía el requerimiento al servlet subperchaServlet.

Botón Submit para Eliminar.- Permite Eliminar el registro de una SubPercha existente en la tabla SUBPERCHA. Se envía el requerimiento al servlet subperchaServlet.

Botón Submir para Limpiar Valores.- Permite borrar los valores de los controles de esta pantalla. Se envía el requerimiento al servlet subperchaServlet.

Pantalla de Administración de Clientes.

Figura 33: Administración de Clientes

La pantalla de Administración de Clientes, denominada "Administrar Clientes", está formada por los siguientes controles:

PullDown de Clientes Existentes.- En este control se almacenan todos los códigos de Clientes existentes en el Sistema, para poder hacer las consultas de la información de las mismos.

Botón Submit para Consultar.- Permite Consultar los Datos de un Cliente existente en la tabla CLIENTE.

TextBox de Código de Cliente.- Permite ingresar o visualizar el código de un Cliente.

TextBox de RUC.- Permite visualizar, ingresar ó modificar el RUC que tiene el Cliente. Sólo se aceptan valores numéricos.

TextBox de Nombre.- Permite visualizar, ingresar ó modificar el Nombre que tiene un Cliente.

TextBox de Dirección.- Permite visualizar, ingresar ó modificar la Dirección domiciliaria o de negocio que tiene un Cliente.

TextBox de Teléfono.- Permite visualizar, ingresar ó modificar el teléfono que tiene un Cliente.

TextBox de email.- Permite visualizar, ingresar ó modificar el email que tiene un Cliente.

Botón Submit para Crear.- Permite Crear un registro de un Cliente nuevo en la tabla CLIENTE. Se envía el requerimiento al servlet clienteServlet.

Botón Submit para Modificar.- Permite Modificar el contenido de un registro de un Cliente existente en la tabla CLIENTE. Se envía el requerimiento al servlet clienteServlet.

Botón Submit para Eliminar.- Permite Eliminar el registro de un Cliente existente en la tabla CLIENTE. Se envía el requerimiento al servlet clienteServlet.

Botón Submir para Limpiar Valores.- Permite borrar los valores de los controles de esta pantalla. Se envía el requerimiento al servlet clienteServlet.

Pantalla de Administración de Cajas de Productos.

Figura 34: Administración de Cajas de Producto

La pantalla de Administración de Cajas de Productos, denominada "Administrar Cajas de Productos", está formada por los siguientes controles:

PullDown de Cajas de Productos Existentes.- En este control se almacenan todos los códigos de Cajas de Productos existentes en el Sistema, para poder hacer las consultas de la información de las mismos.

Botón Submit para Consultar.- Permite consultar los Datos de una Caja de Producto existente en la tabla CAJA_PRODUCTO.

TextBox de Código de Caja de Producto.- Permite ingresar o visualizar el código de una Caja de Producto.

PullDown de Clientes Existentes.- En este control se almacenan todos los códigos de los Clientes existentes en el Sistema. Sirve para seleccionar el Cliente al que pertenece la Caja de Producto, cuando se cree un registro nuevo de Caja de Producto o se modifique un registro de existente.

TextBox de Descripción.- Permite ingresar o visualizar la descripción de la Caja de Producto, es decir, un detalle de qué producto es.

TextBox de Peso.- Permite visualizar, ingresar ó modificar el Peso que tiene la Caja de Producto. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de masa en kilogramos.

TextBox de Número de Cajas.- Permite visualizar, ingresar ó modificar el número de cajas que existe en stock de la Caja de Producto.

TextBox de Alto.- Permite visualizar, ingresar ó modificar el Alto que tiene la Caja de Producto. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Ancho.- Permite visualizar, ingresar ó modificar el Ancho que tiene la Caja de Producto. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Profundidad.- Permite visualizar, ingresar ó modificar la Profundidad que tiene la Caja de Producto. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Unidades por Caja.- Permite visualizar, ingresar ó modificar el número de unidades de producto que tiene cada Caja de Producto.

PullDown de día de caducidad.- En este control se almacenan los días del mes, numerados del 1 al 31; sirve para seleccionar el día en que expira la Caja de Producto.

PullDown de mes de caducidad.- En este control se almacenan los meses del año, numerados del 1 al 12; sirve para seleccionar el mes en que expira la Caja de Producto.

PullDown de año de caducidad.- En este control se almacenan los años, numerados del 2005 al 2070; sirve para seleccionar el año en que expira la Caja de Producto.

Botón Submit para Crear.- Permite Crear un registro de una Caja de Producto nueva en la tabla CAJA_PRODUCTO. Se envía el requerimiento al servlet cajaproductoServlet.

Botón Submit para Modificar.- Permite Modificar el contenido de un registro de una Caja de Producto existente en la tabla CAJA_PRODUCTO. Se envía el requerimiento al servlet cajaproductoServlet.

Botón Submit para Eliminar.- Permite Eliminar el registro de una Caja de Producto existente en la tabla CAJA_PRODUCTO. Se envía el requerimiento al servlet cajaproductoServlet.

Botón Submir para Limpiar Valores.- Permite borrar los valores de los controles de esta pantalla. Se envía el requerimiento al servlet subperchaServlet.

Pantalla para Subir Cajas de Productos a SubPerchas.

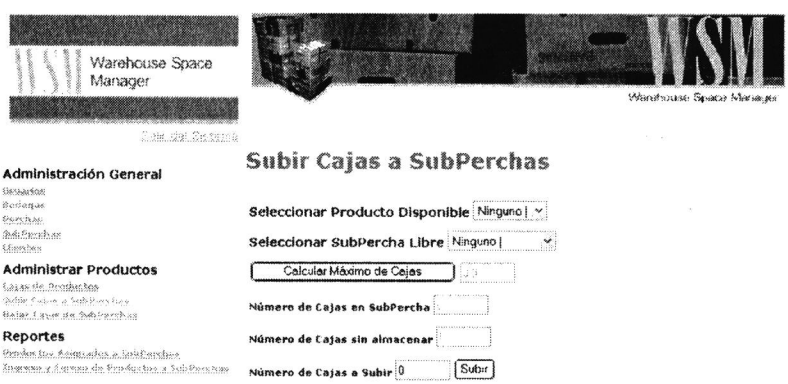


Figura 35: Subir cajas de producto a subperchas

La pantalla denominada "Subir Cajas a SubPerchas", está formada por los siguientes controles:

PullDown de Seleccionar Producto.- En este control se almacenan todos los códigos de Cajas de Productos que

están libres, es decir, que no se han asignado aún a SubPerchas.

PullDown de Seleccionar SubPercha Libre.- En este control se almacenan todos los códigos de las SubPerchas que se encuentran Libres, es decir, que aún no se le han asignado Productos o que la capacidad de la SubPercha aún no se excede.

Botón Submit para Calcular Máximo de Cajas.- Este botón sirve para enviar un requerimiento al Servlet asignarcajasServlet, de tal manera que se calcule el máximo número de cajas de producto que son posibles almacenar en la SubPercha, considerando las dimensiones tanto de la Caja de Producto como de la SubPercha.

TextBox de Número de Cajas en SubPercha.- Permite desplegar el número de cajas que actualmente existen físicamente en la SubPercha seleccionada.

TextBox de Número de Cajas sin almacenar.- Permite desplegar el número de cajas aún existentes en Stock, que no han sido almacenadas en alguna SubPercha.

TextBox de Número de Cajas a Subir.- Permite ingresar o modificar el número de cajas que se desean subir a la subpercha seleccionada.

Botón Submit de Subir Cajas.- Permite Subir o Asignar a la SubPercha seleccionada, el número de cajas ingresadas en "Numero de Cajas a Subir", referente a la Caja de Producto seleccionada. Se envía el requerimiento al servlet asignarcajasServlet.

Pantalla para Bajar Cajas de Productos desde SubPerchas.

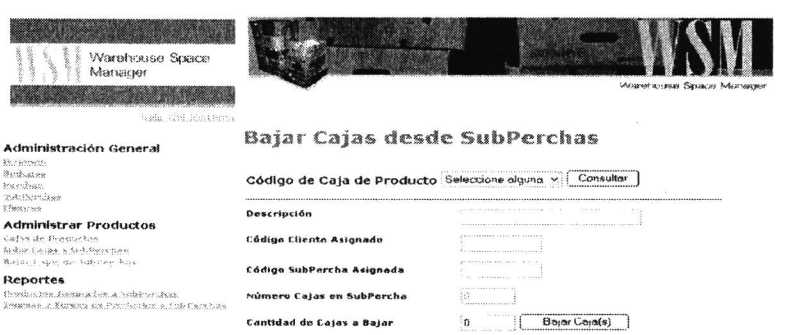


Figura 36: Bajar Cajas de productos desde SubPercha

La pantalla denominada "Subir Cajas a SubPerchas", está formada por los siguientes controles:

PullDown de Seleccionar Producto.- En este control se almacenan todos los códigos de Cajas de Productos que están asignados a SubPerchas.

Botón Submit para Consultar.- Permite consultar datos de la Caja de Producto seleccionada, para decidir cuántas cajas se desean bajar o egresar de la SubPercha a la que pertenece.

TextBox de Descripcion.- Permite visualizar la descripción de la caja de producto seleccionada.

TextBox de Código de Cliente Asignado.- Permite visualizar el código de cliente al que pertenece la caja de producto seleccionada.

TextBox de Código de SubPercha Asignada.- Permite visualizar el código de la subpercha a la que pertenece la caja de producto seleccionada.

TextBox de Número de Cajas en SubPercha.- Permite visualizar el número de Cajas que actualmente están asignadas en la SubPercha.

TextBox de Cantidad de Cajas a Bajar.- Permite ingresar o modificar el número de cajas de producto que se desean dar de baja o egresar..

Botón Submit de Bajar Cajas.- Permite dar de Baja o egresar cajas de productos. Se envía el requerimiento al servlet bajarcajasServlet.

Pantalla de Reporte de Productos Asignados a SubPerchas.

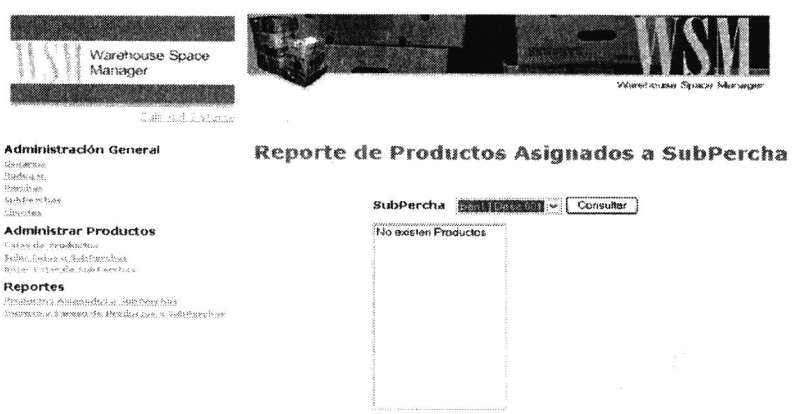


Figura 37: Reporte de productos asignados a subpercha

La pantalla denominada "Reporte: Productos Asignados a SubPercha", está formada por los siguientes controles:

PullDown de SubPercha.- En este control se almacenan los códigos de las SubPerchas que tienen cajas de productos asignadas

Botón Submit de Consultar.- Permite enviar al servlet reporteProdServlet, el requerimiento de consulta de los registros existentes en la tabla MOVIMIENTO, que pertenezcan a la SubPercha seleccionada.

ListBox de muestra registros.- Este control almacena y despliega los registros que se encuentran en la tabla MOVIMIENTO y, que se refieren a los productos que se encuentra asignados a la SubPercha

Pantalla de Reporte de Ingreso y Egreso de Productos desde SubPerchas.

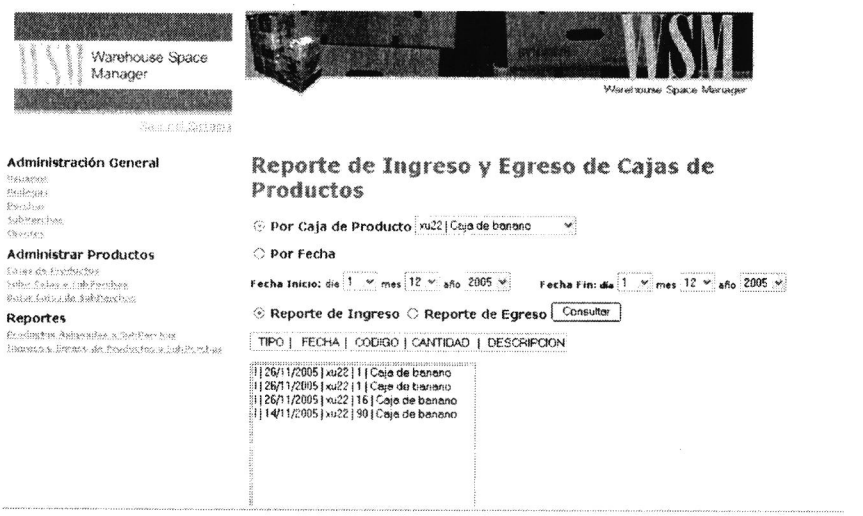


Figura 38: Reporte de Ingreso y Egreso de productos desde SubPercha

La pantalla denominada "Reporte de Ingreso y Egreso de Cajas de Productos", está formada por los siguientes controles:

RadioBotón de Caja de Producto.- Permite seleccionar si el reporte será por caja de Producto.

PullDown de Caja de Producto.- En este control se almacenan los códigos de las Cajas de Productos existentes, que están asignadas a SubPerchas.

RadioBotón de Fecha.- Permite seleccionar si el reporte será por rango de fechas.

PullDown de Fecha de Inicio.- Existen 3 controles tipo PullDown para seleccionar el día, mes y año de la Fecha de Incio.

PullDown de Fecha Fin.- Existen 3 controles tipo PullDown para seleccionar el día, mes y año de la Fecha Fin .

RadioBotón de Reporte de Ingreso.- Permite seleccionar si el reporte será por Ingreso de Cajas de Productos.

RadioBotón de Reporte de Egreso.- Permite seleccionar si el reporte será por Egreso de Cajas de Productos.

Botón Submit de Consultar.- Permite enviar al servlet reporteMovServlet, el requerimiento de consulta de los

registros existentes en la tabla MOVIMIENTO, que cumplan con los criterios de consulta seleccionadas.

Pantalla de Administración de Productos, para dispositivo PDA.

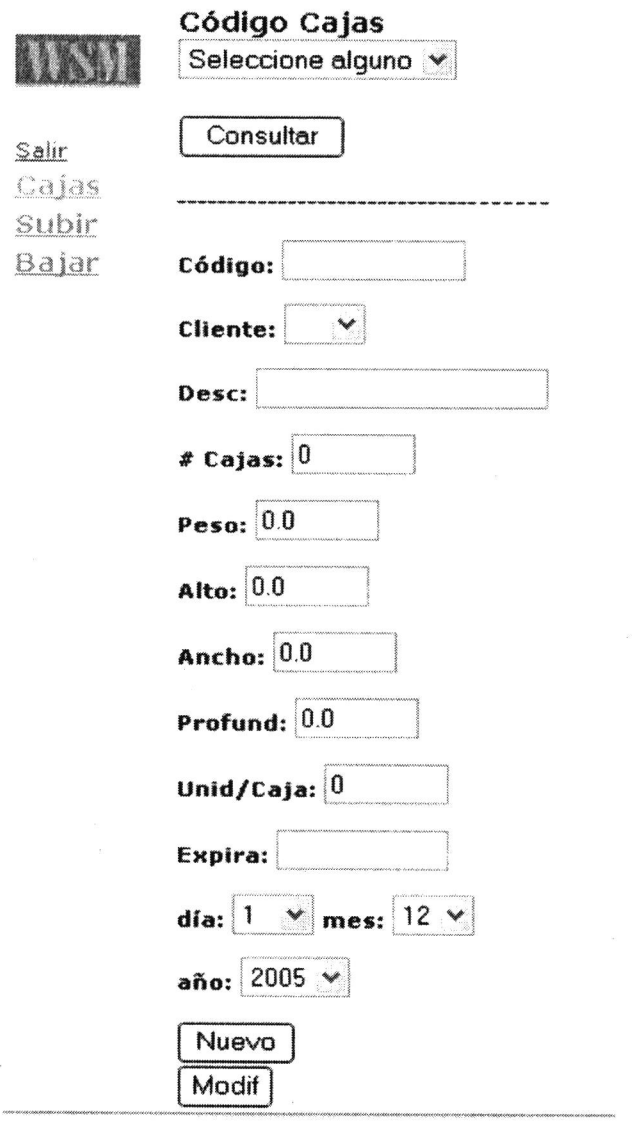


Figura 39: Administración de productos para dispositivos PDA

Debido a que el PDA: Palm Tungsten C, tiene una resolución máxima de 320x320 pixeles, se rediseñó la pantalla de Administración de Productos para que pueda ingresar en el tamaño de la pantalla del dispositivo móvil antes indicado.

PullDown de Cajas de Productos Existentes.- En este control se almacenan todos los códigos de Cajas de Productos existentes en el Sistema, para poder hacer las consultas de la información de las mismos.

Botón Submit para Consultar.- Permite consultar los Datos de una Caja de Producto existente en la tabla CAJA_PRODUCTO.

TextBox de Código de Caja de Producto.- Permite ingresar o visualizar el código de una Caja de Producto.

PullDown de Clientes Existentes.- En este control se almacenan todos los códigos de los Clientes existentes en el Sistema. Sirve para seleccionar el Cliente al que pertenece la

Caja de Producto, cuando se cree un registro nuevo de Caja de Producto o se modifique un registro de existente.

TextBox de Descripción.- Permite ingresar o visualizar la descripción de la Caja de Producto, es decir, un detalle de qué producto es.

TextBox de Peso.- Permite visualizar, ingresar ó modificar el Peso que tiene la Caja de Producto. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de masa en kilogramos.

TextBox de Número de Cajas.- Permite visualizar, ingresar ó modificar el número de cajas que existe en stock de la Caja de Producto.

TextBox de Alto.- Permite visualizar, ingresar ó modificar el Alto que tiene la Caja de Producto. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Ancho.- Permite visualizar, ingresar ó modificar el Ancho que tiene la Caja de Producto. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Profundidad.- Permite visualizar, ingresar ó modificar la Profundidad que tiene la Caja de Producto. Sólo se aceptan valores numéricos con dos decimales, mismos que se refieren a la unidad de medida en metros lineales.

TextBox de Unidades por Caja.- Permite visualizar, ingresar ó modificar el número de unidades de producto que tiene cada Caja de Producto.

PullDown de día de caducidad.- En este control se almacenan los días del mes, numerados del 1 al 31; sirve para seleccionar el día en que expira la Caja de Producto.

PullDown de mes de caducidad.- En este control se almacenan los meses del año, numerados del 1 al 12; sirve para seleccionar el mes en que expira la Caja de Producto.

PullDown de año de caducidad.- En este control se almacenan los años, numerados del 2005 al 2070; sirve para seleccionar el año en que expira la Caja de Producto.

Botón Submit para Crear.- Permite Crear un registro de una Caja de Producto nueva en la tabla CAJA_PRODUCTO. Se envía el requerimiento al servlet cajaproductoServlet.

Botón Submit para Modificar.- Permite Modificar el contenido de un registro de una Caja de Producto existente en la tabla CAJA_PRODUCTO. Se envía el requerimiento al servlet cajaproductoServlet.

Botón Submit para Eliminar.- Permite Eliminar el registro de una Caja de Producto existente en la tabla CAJA_PRODUCTO. Se envía el requerimiento al servlet cajaproductoServlet.

Botón Submir para Limpiar Valores.- Permite borrar los valores de los controles de esta pantalla. Se envía el requerimiento al servlet subperchaServlet.

Pantalla para Subir Cajas de Productos a SubPerchas, a través de Dispositivo PDA.

Figura 40: Subir Cajas de productos a SubPerchas a través de Dispositivos PDA

Debido a que el PDA: Palm Tungsten C, tiene una resolución máxima de 320x320 pixeles, se rediseñó la pantalla de Subir

Cajas de Productos para que pueda ingresar en el tamaño de la pantalla del dispositivo móvil antes indicado.

PullDown de Seleccionar Producto.- En este control se almacenan todos los códigos de Cajas de Productos que están libres, es decir, que no se han asignado aún a SubPerchas.

PullDown de Seleccionar SubPercha Libre.- En este control se almacenan todos los códigos de las SubPerchas que se encuentran Libres, es decir, que aún no se le han asignado Productos o que la capacidad de la SubPercha aún no se excede.

Botón Submit para Calcular Máximo de Cajas.- Permite calcular el máximo número de cajas de producto que son posibles almacenar en la SubPercha, considerando las dimensiones tanto de la Caja de Producto como de la SubPercha. El requerimiento es enviado al Servelt: asignarcajasServlet

TextBox de Número de Cajas en SubPercha.- Permite visualizar el número de cajas que actualmente existen físicamente en la SubPercha seleccionada.

TextBox de Número de Cajas sin almacenar.- Permite visualizar el número de cajas aún existentes en Stock, que no han sido almacenadas en alguna SubPercha.

TextBox de Número de Cajas a Subir.- Permite ingresar o modificar el número de cajas que se desean subir a la subpercha seleccionada.

Botón Submit de Subir Cajas.- Permite Subir o Asignar a la SubPercha seleccionada, el número de cajas ingresadas en "Numero de Cajas a Subir", referente a la Caja de Producto seleccionada. Se envía el requerimiento al servlet asignarcajasServlet.

Pantalla para Bajar Cajas de Productos desde SubPerchas, a través de Dispositivo PDA.

Figura 41: Bajar Cajas de productos desde SubPerchas, a través de Dispositivos PDA

Debido a que el PDA: Palm Tungsten C, tiene una resolución máxima de 320x320 pixeles, se rediseñó la pantalla de Bajar Cajas de Productos para que pueda ingresar en el tamaño de la pantalla del dispositivo móvil antes indicado.

PullDown de Seleccionar Producto.- En este control se almacenan todos los códigos de Cajas de Productos que están asignados a SubPerchas.

Botón Submit para Consultar.- Permite consultar datos de la Caja de Producto seleccionada, para decidir cuántas cajas se desean bajar o egresar de la SubPercha a la que pertenece.

TextBox de Descripcion.- Permite visualizar la descripción de la caja de producto seleccionada.

TextBox de Código de Cliente Asignado.- Permite visualizar el código de cliente al que pertenece la caja de producto seleccionada.

TextBox de Código de SubPercha Asignada.- Permite visualizar el código de la subpercha a la que pertenece la caja de producto seleccionada.

TextBox de Número de Cajas en SubPercha.- Permite visualizar el número de Cajas que actualmente están asignadas en la SubPercha.

TextBox de Cantidad de Cajas a Bajar.- Permite ingresar o modificar el número de cajas de producto que se desean dar de baja o egresar..

Botón Submit de Bajar Cajas.- Permite dar de Baja o egresar cajas de productos. Se envía el requerimiento al servlet bajarcajasServlet.

CAPÍTULO 4

4.- Implementación y Pruebas

El Sistema fue implementado utilizando las siguientes herramientas de Desarrollo:

- ✓ IDE Eclipse version 3.0 (libre distribución)
- ✓ Especificaciones API de EJB 2.1
- ✓ J2SDK version 1.4.2_02
- ✓ Servidor de Aplicaciones Jboss version 3.2.2 (libre distribución)
- ✓ RDBMS MySql 4.0.20 a (libre distribución)

El Sistema fue desarrollado sobre plataforma windows (XP Professional), con el siguiente hardware:

- ✓ CPU AthlonXP2000+ (análogo a PIV de 2.0Ghz)
- ✓ 512MB Ram
- ✓ Disco de 80GB
- ✓ Tarjeta de Red10/100Mbps
- ✓ Tarjeta video 64MB Agp

Para desarrollar los módulos para el dispositivo PDA, se utilizó el dispositivo Palm Tungsten C que tiene 64Mb ram, wireless 802.11b, browser WebBro2.0.1.1

4.1 Problemas de Implementación

Para poder desarrollar este Sistema, tuvimos que resolver algunos inconvenientes, los cuáles se han divido en dos tipos que son: problemas intrinsecos y problemas extrinsecos

Problemas intrinsecos al desarrollo del Sistema

Aprender a utilizar el IDE de Desarrollo Eclipse

Se tuvieron que hacer algunas pruebas antes de emprender el desarrollo utilizando este IDE, puesto que se debía estar "familiarizado" con la forma que tiene Eclipse para distribuir los componentes y ejecutar el código de programación.

Aprender a utilizar el Servidor de Aplicaciones Jboss

De igual forma que aprender a utilizar Eclipse, se tenía que aprender a utilizar el Servidor de Aplicaciones Jboss, mismo que sirve para deplegarlos EJB y Servlet.

Entender la especificación EJB 2.1

Antes de empezar el modelamiento del Sistema y, recordando las clases impartidas durante el desarrollo del Tópico, se tuvo que comprender como funcionaban los EJB tipo CMP y SESSION, esenciales para el desarrollo del Sistema.

Modelamiento y alcance del Sistema

Esta parte fue muy complicada, pues se debía limitar el Sistema para que sea posible su desarrollo; de tal forma que se especificaron los alcances detallados en el capitulo 1.

Luego de definir el alcance del Sistema, se procedió a su modelamiento para poder desarrollarlo.

Encontrar el dispositivo PDA

Debido a que el desarrollo incluía una sección que debía funcionar con un dispositivo PDA, nos empeñamos en desarrollar algo real y no ficticio; es decir, un producto que sea probado directamente en una PALM y no en un EMULADOR. De esta forma hicimos todo el esfuerzo, de investigación y de dinero para poder adquirir una PDA que permita nuestro desarrollo y futuras pruebas reales del Sistema en funcionamiento.

Problemas extrínsecos al desarrollo del Sistema

Tiempo disponible para el desarrollo

Debido a multiples factores personales y laborales, el proyecto no pudo ser terminado en el tiempo previsto y, hubo que hacer continuas paras hasta que finalmente se llegó a terminar el Proyecto fines de Octubre del 2005.

Inconvenientes de Salud y Viajes

Hubo una discontinuidad fuerte de casi 2 meses (enero y febrero del 2005) debido a una operación de hernias discales en la columna lumbar a la que fue sometido el suscrito co-autor de este proyecto de graduación, Sr. Juan Moreno.

Durante todo el desarrollo del proyecto hubieron varias discontinuidades, por motivos de viajes fuera del país por parte del co-autor de este proyecto de graduación, Sr. José Córdova, quién trabaja desde el año 2002 en la CIA. MAINT

Disponibilidad de dispositivo PDA

Debido al costo que representaba adquirir la palm Tungsten C, se la compró en Julio del 2005 a un costo de $430 dólares. Finalmente el

proyecto funciona a la perfección bajo el esquema para el que fue desarrollado.

4.2 Plan de Pruebas

Nuestro plan de pruebas consistió en ir verificando lo que se desarrollaba, a medida que se avanzaba; de tal manera que se hicieron pruebas de ejecución en :

- ✓ Consulta de Registros en módulo de Mantenimiento de Usuarios
- ✓ Inserción de Registros en módulo de Mantenimiento de Usuarios
- ✓ Modificación de Registros en módulo de Mantenimiento de Usuarios
- ✓ Eliminación de Registros en módulo de Mantenimiento de Usuarios

- ✓ Consulta de Registros en módulo de Mantenimiento de Bodegas
- ✓ Inserción de Registros en módulo de Mantenimiento de Bodegas
- ✓ Modificación de Registros en módulo de Mantenimiento de Bodegas
- ✓ Eliminación de Registros en módulo de Mantenimiento de Bodegas

- ✓ Consulta de Registros en módulo de Mantenimiento de Perchas
- ✓ Inserción de Registros en módulo de Mantenimiento de Perchas
- ✓ Modificación de Registros en módulo de Mantenimiento de Perchas
- ✓ Eliminación de Registros en módulo de Mantenimiento de Perchas

- ✓ Consulta de Registros en módulo de Mantenimiento de SubPerchas
- ✓ Inserción de Registros en módulo de Mantenimiento de SubPerchas
- ✓ Modificación de Registros en módulo de Mantenimiento de SubPerchas
- ✓ Eliminación de Registros en módulo de Mantenimiento de SubPerchas

- ✓ Consulta de Registros en módulo de Mantenimiento de Productos
- ✓ Inserción de Registros en módulo de Mantenimiento de Productos
- ✓ Modificación de Registros en módulo de Mantenimiento de Productos
- ✓ Eliminación de Registros en módulo de Mantenimiento de Productos

- ✓ Consulta de Registros en módulo de Mantenimiento de Clientes
- ✓ Inserción de Registros en módulo de Mantenimiento de Clientes
- ✓ Modificación de Registros en módulo de Mantenimiento de Clientes
- ✓ Eliminación de Registros en módulo de Mantenimiento de Clientes

- ✓ Asignación de productos a SubPerchas en el módulo del mismo nombre.

- ✓ Dar de Baja a productos existentes en SubPerchas en el módulo de Egreso de Productos desde SubPerchas.
- ✓ Asignación de productos a SubPerchas en el módulo del mismo nombre, utilizando el dispositivo PDA
- ✓ Dar de Baja a productos existentes en SubPerchas en el módulo de Egreso de Productos desde SubPerchas, utilizando el dispositivo PDA.

- ✓ Presentación de información utilizando el módulo de Reportes, en lo que se refiere consulta de productos existentes en Subperchas y, consulta de ingresos/egresos de productos en las SubPerchas en un rango de fechas.

4.3 Resultado de las Pruebas

Los resultados luego de las pruebas realizadas en cada módulo del proyecto fueron muy satisfactorios, pues se ajustaban al modelamiento del Sistema y a la descripción y alcances detallados en el Capítulo 1. De esta manera se verificaron los mensajes de éxitos o de error previamente codificados en el Sistema.

CONCLUSIONES

Luego de finalizar el desarrollo del proyecto, podemos clasificar nuestras conclusiones en tres niveles: Tecnológico, Teórico-Práctico y Humano

De nivel Tecnológico

- ✓ La tecnología Java es muy importante y actualmente es muy útil para el desarrollo de aplicaciones transaccionales en Sistemas de ambiente Web como en ambientes de microcódigo para dispositivos móviles.

- ✓ La arquitectura MVC es muy buena para dividir el modelo del negocio, en cuanto a su lógica (a través de los EJBs), de la parte de presentación y del formato de la información (a través de los JSPs y HTMLs) utilizando unos controladores como son los SERVLETs.

- ✓ El Desarrollo de Sistemas utilizando herramientas de libre distribución, es una forma económica de encontrar soluciones funcionales a problemas existentes en la vida real.

De nivel Teórico-Práctico

- ✓ El aprendizaje de la tecnología JAVA ha sido muy importante pues durante toda nuestra carrera previo al título de Ingenieros en Computación, solo se desarrollaron sistemas que funcionaban en ambientes de escritorio y, no en un ambiente 100% web.

- ✓ Haber desarrollado el "Sitema para Administrar una Bodega, orientado a la organización del espacio físico, utilizando plataforma Java y soporte de dispositivo PDA", es una muestra real de la practicidad y de la potencia de la tecnología JAVA, misma que está continuamente siendo actualizada y mejorada para simplificar el desarrollo de Sistemas.

De nivel humano

- ✓ Durante el desarrollo de este Sistema, hemos adquirido muchas destrezas, las cuáles serán puestas al servicio de la sociedad para el mejoramiento de procesos públicos y privados, en los que el

tratamiento de la información es fundamental para el desarrollo del país.

- ✓ Se hicieron muchos contactos interpersonales para poder sacar adelante algunas ideas, una de ellas fue el actual Sistema. En todo este tiempo de Desarrollo se afianzó la amistad de un grupo que se mantuvo firme a pesar de tantos problemas.

RECOMENDACIONES

A continuación se detallan algunas mejoras que se pueden realizar al proyecto WSM.

Estas mejoras no han sido consideradas en la actual versión, puesto que no estaban contempladas en el alcance ni definición del Proyecto; sin embargo en alguna futura versión del WSM si se las puede incluir.

Posibles Mejoras al Sistema

- ✓ Utilizar un dispositivo lector de código de barras para evitar el ingreso de los códigos de cajas de productos.

- ✓ Intentar colocar un gráfico (a escala) en tiempo real a medida que se van creando las perchas y subperchas para mejor visualización de la bodega y sus espacios físicos disponibles.

- ✓ Implementar un adaptador que permita conectarse a algún sistema de inventarios existente en la empresa, sin embargo, este sistema provee intrinsecamente una manera de emular al Sistema de Inventarios, pues puede llevar registros de ingresos y egresos de cajas de productos existentes en una Bodega.

ANEXOS

ANEXO 1

LISTA DE EJBs, SERVLETS Y JSPs

EJBs tipo CMP desarrollados en el Sistema

- ✓ UsuarioCMPBean
- ✓ BodegaCMPBean
- ✓ PerchaCMPBean
- ✓ SubPerchaCMPBean
- ✓ CajaProductoCMPBean
- ✓ ClienteCMPBean
- ✓ MovimientoCajasBean

EJBs tipo SESSION Statefull desarrollados en el Sistema

- ✓ SessionUsuarioCMPBean
- ✓ SessionBodegaCMPBean
- ✓ SessionPerchaCMPBean
- ✓ SessionSubPerchaCMPBean
- ✓ SessionCajaProductoCMPBean
- ✓ SessionClienteCMPBean
- ✓ SessionMovimientoCajasBean

Servlets desarrollados en el Sistema

- ✓ usuarioServlet
- ✓ bodegaServlet
- ✓ perchaServlet
- ✓ subperchaServlet
- ✓ cajaproductoServlet
- ✓ clienteServlet
- ✓ asignarcajasServlet
- ✓ asignarpalmServlet
- ✓ bajarcajasServlet
- ✓ bajarpalmServlet
- ✓ reporteMovServlet
- ✓ reporteProdServlet

Java Server Pages desarrollados en el Sistema

- ✓ Bodega
- ✓ Percha
- ✓ SubPercha
- ✓ Productos
- ✓ ProductosPalm
- ✓ LoginBodegaPalm
- ✓ LoginPalm

- ✓ Clientes
- ✓ Usuarios
- ✓ AsignarProductos
- ✓ AsignarProductosPalm
- ✓ QuitarProductos
- ✓ QuitarProductosPalm
- ✓ reporteIngrEgrSub
- ✓ reporteProdSub

ANEXO 2

PSEUDOCÓDIGO PARA ASIGNAR CAJAS DE PRODUCTOS A SUBPERCHAS

1. Buscar Cajas de Productos que estén Libres
    a. Seleccionar un tipo de Caja de Producto
2. Buscar SubPerchas que estén Libres.
    a. Seleccionar una SubPercha
3. Calcular Volumen de la Caja de Producto Seleccionada
4. Calcular Volumen de la SubPercha Seleccionada
5. Calcular el número maximo de Cajas que la SubPercha puede sopotar, dividiendo el Volumen de SubPercha para el Volumen de la Caja de Producto
6. Obtener el número de cajas asignadas en la SubPercha.
7. Ingresar número de cajas que se desea subir a la Subpercha
8. Sumar el numero de cajas asignadas en subpercha, mas el número de cajas que se desean subir a la Subpercha. Esto es el numero de cajas actuales.
9. Preguntar Si el numero de cajas actuales no excede el maximo numero de cajas permitidas en SubPercha.
    a. Preguntar Si el número de cajas a ingresar excede el número de cajas disponibles en Stock

i. Desplegar mensaje de error

b. Caso Contrario

i. Guardar el movimiento, como Ingreso.

ii. Restar del numero de cajas en Stock, el numero de cajas ingresadas.

iii. Incrementar el valor del número de Cajas en la SubPercha, sumando el valor del numero de cajas ingresadas

10. Caso contrario

a. Desplegar mensaje de error.

11. Fin.

ANEXO 3

PSEUDOCÓDIGO PARA BAJAR CAJAS DESDE SUBPERCHAS.

Buscar Cajas de Productos que estén asignadas a SubPerchas

a. Seleccionar un tipo de Caja de Producto

2. Obtener la SubPercha a la que pertenece la Caja de Producto seleccionada.
3. Obtener el número de cajas que están asignadas en dicha SubPercha.
4. Ingresar el número de cajas que se desean bajar
5. Preguntar Si el numero de cajas que se desean bajar es menor o igual que el número de cajas que están asignadas
   a. Guardar el movimiento, como Egreso
   b. Decrementar el numero de cajas asignadas en la subpercha, en el valor del número de cajas que se egresaron
6. Caso contrario
   a. Desplegar mensaje de error.
7. Fin

GLOSARIO

Botón Submit.- Control del tipo entrada que se emplea para enviar requerimientos desde un fragmento de código a otro, dentro ó fuera del mismo módulo de un sistema. También se lo utiliza para pasar el control desde una parte del Sistema a otra.

Control.- Interface que permite realizar acciones dentro de la codificacion de un Sistema

Frame.- Sección de una página web, que posee dimensiones específicas, en donde se pueden colocar enlaces hacia otras páginas, controles varios, graficos, tablas, etc, etc.

ListBox.- Control que sirve para desplegar información o para seleccionar ítems en un orden específico. Se pueden visualizar todos los ítems al mismo tiempo.

PullDown.- Control que permite la selección de ítems, en un orden específico. Siempre se muestra un solo item a la vez.

Radio Botón.- Cotrol que permite habilitar o deshabilitar alguna opción en la interfaz de un sistema.

Servlet.- Un objeto en Java que se ejecuta en el lado del Servidor de Aplicaciones Web.

TextBox.- Control del tipo entrada/salida que se emplea para la presentación o ingreso de caracteres dentro de un Sistema.

802.11b/g.- Protocolo de acceso para redes inalámbricas.

Persistencia.- almacenar y recuperar datos de un objeto, mediante el mapeado en una tabla de una base de datos.

BIBLIOGRAFIA

1.- Ed Roman, Mastering Enterprise JavaBeans (2da. Edición, EUA, John Wiley & Sons, Inc., 2002), pp.3-200, 569-583

2.- Ángel Esteban, Acceso a Bases de Datos con Java-JDBC 2.0 (Grupo EIDOS, 2003), pp.55-142

3.- Ángel Esteban, Tecnologías de Servidor con Java: Servlets, JavaBeans, JSP (Grupo EIDOS 2003), pp 41-365

4.- J2EE, Octubre 2005, Construir aplicaciones EJB con Jboss, Lomboz y Eclipse, http://www.programacion.com/java/tutoriales/J2EE

5.- Steve Burbeck, July 2004, How to use Model-View-Controller, http://www.object-arts.com/EducationCentre/Overviews/MVC.htm

6.- Eclipse.org, Octubre 2005, Eclipse.org, http://www.eclipse.org/

7.- Jboss.org,Octubre 2005, Jboss.org, http://www.osmosislatina.com/jboss

8.- MySQL, Octubre 2005, MySql Profesional, http://www.mysql.com/doc/