ESCUELA POLITÉCNICA DEL EJÉRCITO

DPTO. DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA EN SISTEMAS E INFORMÁTICA

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA LA GESTIÓN Y ADMINISTRACIÓN EMPRESARIAL BASADO EN TECNOLOGÍA JEE Y PRIMEFACES

Previa a la obtención del Título de:

INGENIERO EN SISTEMAS E INFORMÁTICA

POR: **JONATHAN PATRICIO CÁRDENAS RUPERTI**

SANGOLQUÍ, Enero de 2013

CERTIFICACIÓN

Certifico que el presente trabajo fue realizado en su totalidad por el Sr. CÁRDENAS RUPERTI JONATHAN PATRICIO como requerimiento parcial a la obtención del título de INGENIERO EN SISTEMAS E INFORMÁTICA.

Sangolquí, 30 de Enero de 2013

_____

ING. ANDRÉS DE LA TORRE
DIRECTOR

ESCUELA POLITÉCNICA DEL EJÉRCITO

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

AUTORIZACIÓN

Yo, JONATHAN PATRICIO CÁRDENAS RUPERTI

Autorizo a la Escuela Politécnica del Ejército la publicación, en la biblioteca virtual de la Institución, del trabajo "DISEÑO E IMPLEMENTACIÓN DE UN SITIO WEB PARA LA GESTIÓN Y ADMINISTRACIÓN EMPRESARIAL BASADO EN TECNOLOGÍA JEE Y PRIMEFACES", cuyo contenido, ideas y criterio son de mi exclusiva responsabilidad y autoridad.

Sangolquí, Enero 2013

_____

JONATHAN PATRICIO CÁRDENAS RUPERTI

DEDICATORIA

Dedico esta tesis a mis padres Sonia y Jonathan por el apoyo incondicional durante mi carrera, fundamental para el desarrollo de la misma, mis hermanas Kathya y Fernanda con quienes siempre pude contar en esos momentos más difíciles, a todas esas personas quienes me dieron la mano en algún momento y estuvieron en los momentos buenos y malos respaldándome.

Jonathan Cárdenas R.

AGRADECIMIENTOS

De manera muy especial, quiero agradecer a mis padres por su incondicional apoyo en todo el transcurso de mi vida. A mis hermanas por su amor, tolerancia y comprensión. A mi novia Olga quien me apoyó moralmente y siempre estuvo pendiente de muchas cosas. A mi institución y mis profesores por todo lo aprendido durante el curso de mi carrera. A los ingenieros Andrés De La Torre y Tatiana Gualotuña por su adecuada guía para la realización de este proyecto. A todo el personal de ASESTRATÉGICO por las facilidades prestadas en la elaboración de este proyecto.

Jonathan Cárdenas R.

Índice de Contenidos

Listado de Tablas

Listado de Figuras

RESUMEN

Las empresas ecuatorianas requieren gestionar adecuadamente las áreas como Inventario, Nómina y Activos Fijos de forma eficaz, que permitan tener integridad en los datos, sin embargo, no cuentan con una herramienta de administración que les ayude a gestionar todos los recursos de la empresa; por ello la necesidad de automatizar totalmente estas áreas de la organización.

Se pretendió investigar y determinar las tecnologías más actuales de desarrollo con RestFul[1] (Representational State Transfer) para poder desarrollar un software de alta jerarquía. Se investigó y analizó los conceptos involucrados para cada una de las áreas antes mencionadas en función a las reglas de negocio dadas. El propósito del presente proyecto se basa en el diseño e implementación de un sistema para la gestión administrativa con JEE[2] (Java Plataform Enterprise Edition) y PrimeFaces[3] mediante el uso de la metodología XP[4] (eXtemeProgramming).

En este proyecto se describe el proceso de desarrollo utilizando la metodología XP, el diseño e implementación del sistema, y los resultados del proyecto. Es así que los módulos principales involucrados dentro de la realización del sistema son: Administración de Activos Fijos, Administración de Nómina y Administración de Inventarios, siendo todos estos módulos los más importantes que requiere la empresa que sean automatizados de acuerdo al flujo de negocio que posee. La distribución de los procesos en varios

[1]Transferencia de Estado Representacional define un set de principios arquitectónicos por los cuales se diseñan servicios web. http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.html.
[2] Java Plataform Enterprise Edition, plataforma de programación para ejecutar software de programación java con N capas. http://download.oracle.com/javaee
[3]PrimeFaces.- Es un componente para JavaServer Faces de código abierto que facilitan la creación de las aplicaciones web. http://es.wikipedia.org/wiki/PrimeFaces
[4]eXtremProgramming, una metodología de desarrollo ágil, la cual se enfoca con la comunicación con el cliente. http://www.slideshare.net/joaquin_win/extreme-programming-456979

RestFul para el desarrollo de este proyecto muestra ventajas como: modularidad, interoperabilidad y reusabilidad, que son uno de los principios de la metodología empleada. Durante la construcción del sistema con la metodología XP se elaboraron iteraciones para un avance incremental con mini versiones funcionales de forma inmediata siendo así el alcance del desarrollo ágil y por ende rápido enfocado en la construcción del sistema.

CAPÍTULO 1

1. DESCRIPCIÓN DE LA SITUACIÓN ACTUAL

1.1. Introducción

En la actualidad, los procesos administrativos de las instituciones, organizaciones y empresas son cada vez más exigentes, por lo que se requiere la automatización de los mismos para mejorar la realización de éstos. Es por ello, que los nuevos sistemas de gestión y administración empresarial deben ser completamente estables, fiables y a la vez seguros, para mantener íntegra la información confidencial de las empresas.

La empresa ASESTRATÉGICO CÍA. LTDA[5]. presta servicios de asesoría financiera, contable y tributaria con una trayectoria de aproximadamente 20 años de experiencia en el mencionado campo profesional. ASESTRATÉGICO CÍA. LTDA. tiene implementado un sistema con muchas falencias en cuanto a estabilidad y seguridad de los datos, por lo cual se lo podría catalogar como un aplicativo ya obsoleto, que procesa incorrectamente la información y sin mucha utilidad para el usuario.

ASESTRATÉGICO CÍA. LTDA. está dispuesta a proveer toda la información necesaria para que se lleve a cabo el presente proyecto y poder solucionar todos las posibles contrariedades y complicaciones que se dan en este momento con el sistema actual.

[5]ASESTRATÉGICO CIA. LTDA.- Es la empresa patrocinadora de la tesis, formada hace más de 20 años en el campo contable, financiero, legal, tributario y de capacitación personal.

El desarrollo permitirá tener un sistema completamente nuevo y orientado a la Web, para así suplir con todos los problemas de accesibilidad, confiabilidad y disponibilidad que se presentan hoy en día.

1.2. Justificación

Los sistemas informáticos van progresando de acuerdo a los requerimientos de los usuarios y las necesidades que se presentan, y es por ello que es completamente necesario crear software estable y seguro en su totalidad.

El sistema actual presenta problemas como:

- Bajo nivel de consistencia de la información en las diferentes transacciones.
- Bajo nivel de trascendencia del sistema con la actualidad (no se ajusta a las necesidades actuales).
- Alto nivel de inseguridad al momento de limitar a los usuarios del sistema.

Para el caso específico propuesto, se realizará un sistema Web para que la información de la empresa ASESTRATÉGICOCÍA. LTDA. con los usuarios autorizados realicen la gestión administrativa en todo momento y en cualquier ubicación geográfica.

El objetivo principal del presente proyecto es lograr satisfacer las necesidades de ASESTRATÉGICOCÍA. LTDA. y conseguir reducir los problemas y contrariedades que se están evidenciando, los cuales son causados por el uso de un sistema de baja calidad. Por tanto es considerable y de importancia el solucionar todos aquellos problemas con un sistema estable y que se ajuste a los requerimientos de la mencionada empresa.

1.3. Objetivos

1.3.1. Objetivo General

Diseñar e implementar un sistema orientado a la Web basado en tecnologías de JEE a nivel de servidor y PrimeFaces a nivel de cliente, para la gestión administrativa de la empresa ASESTRATÉGICOCÍA. LTDA.

1.3.2. Objetivos Específicos

- Analizar las "historias de usuario" para planificar la realización de las tareas que conllevan a generar pequeñas versiones correspondientes a los módulos Activos Fijos, Inventarios y Nómina.
- Diseñar versiones pequeñas de acuerdo al plan de iteraciones bajo el marco de la arquitectura EJB3 para garantizar el buen desarrollo del sistema y proporcionar un valor claro con referencia a las reglas del negocio.
- Construir cada una de las miniversiones con la ayuda de RESTFul de forma incremental para asegurar el desarrollo de forma reutilizable y completar la totalidad del sistema.
- Realizar pruebas a cada una de estas versiones para respaldar el buen funcionamiento completo del sistema.

1.4. Alcance

Este documento presenta una propuesta completa para el diseño, desarrollo e implementación de un sistema de gestión administrativa que estará provisto de los módulos de: Activos Fijos, Nómina e Inventarios.

El sistema a implementarse va a estar orientado a la web, lo que permitirá acceder al mismo desde cualquier equipo conectado a la red de la empresa con tan solo tener un navegador para ingresar al sistema. Además el sistema contará con permisos de acceso dependiendo del usuario con el que inicia sesión para tener habilitados las opciones del menú principal y entrar a las pantallas.

Con la ayuda de la tecnología PrimeFaces, basada en el lenguaje de programación Java, se tendrá un mejor avance en el desarrollo de la aplicación, ya que se puede realizar programación en el Front-End[6] con manipulación de muchos objetos visuales pero con poco código que programar, esto se debe a que por detrás implementa JSF2[7] y Ajax[8] que son los principales pilares de esta herramienta para el desarrollo de aplicaciones JEE permitiendo tener una interfaz amigable con el usuario.

El sistema deberá poseer las siguientes funcionalidades de acuerdo a cada módulo.

El módulo de inventarios ayudará al cálculo de la existencia en bodega; para ello se tiene las siguientes opciones y pantallas:

- Ingreso de productos
- Gestión de transacciones
- Ingreso de bodega
- Reporte de inventario
- Gestor de precios y de costos

[6] Front-End.- Parte del software que interactúa con el usuario. http://es.wikipedia.org/wiki/Front-end_y_back-end
[7] JSF2.- Java Server Faces 2 es una tecnología y framework para ejecutar aplicaciones Java de tipo Web. http://es.wikipedia.org/wiki/JavaServer_Faces
[8] Ajax.- Asynchronous JavaScript And XML, crea aplicaciones interactivas. http://es.wikipedia.org/wiki/AJAX

El módulo de nómina se manejará la información correspondiente al cálculo de rubros y manejo de roles de pago del personal que labora, en la cual se tiene las siguientes opciones y pantallas:

- Gestor de empleados
- Gestor de roles de pago
- Reporte de rol de pagos individual
- Reporte de rol de pagos general
- Gestor de rubros
- Gestor tipo de roles de pago

En el módulo de activos fijos se manejará el cálculo de la depreciación anual de los artículos de la empresa, en la cual se tiene las siguientes opciones y pantallas:

- Gestor de activos fijos
- Mantenimiento de activos fijos
- Gestor de tipo de activos fijos
- Reporte de depreciación de activos fijos

CAPÍTULO 2

2. MARCO TEÓRICO

2.1. Descripción y Conceptos de Sistemas de Gestión.

2.1.1. Administración de Activos fijos

Los activos fijos, son aquellos que no varían durante el tiempo de vida de una empresa, también se consideran que son complicados para su venta en el caso de que la empresa desee conseguir dinero, debido a que el propósito es permanecer durante mucho tiempo en la empresa.

Si bien es cierto que estos activos no son eternos, la contabilidad obliga a que estos se deprecien cada tiempo, debido a que influyen varios factores sobre ellos, como el uso, el desgaste, el paso del tiempo, por obsolescencia, de forma que se refleja su valor más ajustado posible o a amortizar los gastos a largo plazo.

Tipos de depreciación

Existen dos tipos de depreciación que afectan a los Activos Fijos de manera directa, a continuación se describe:

Depreciación Física.- Relaciona el uso de un activo depreciable a lo largo del tiempo de vida que ha tenido.

Depreciación Funcional.- Surge de la obsolescencia o inadecuación del activo con respecto a sus funciones en optimo estado.

Toda empresa necesita inventariar los activos fijos que posee, donde se procede a identificar las áreas en donde se encuentra estos activos.

Las ventajas de automatizar esto, son las siguientes:

- Agilidad para la captura de datos.
- Eliminación de tiempos muertos durante el proceso de inventarios.
- Activos fijos localizables en cualquier momento.
- Permite realizar inventarios con más frecuencia.
- Aumenta la conciencia de usuarios para no realizar movimientos de equipo sin dar aviso al departamento correspondiente.
- Clasificación de sus activos fijos en un máximo de tres niveles según las necesidades que se presenten.
- Le permite planear sus actividades de mantenimiento de activos conociendo los gastos bajo este concepto.[9]

El módulo de Activos Fijos permite obtener información actualizada de depreciación y el valor del mismo en cualquier momento que el usuario lo requiera. El usuario puede consultar o imprimir un Estado de Cuenta de cada uno de sus Activo, para tener información sobre el valor de adquisición, las depreciaciones contabilizadas y el Valor en Libros de cada Activo.

[9] http://www.mobileware-mx.com/controlactivosfijos.asp

2.1.2. Administración de Nómina

La nómina o rol de pagos es por medio de la cual el trabajador recibe principalmente su remuneración por el trabajo realizado durante cierto tiempo que por lo general es cada mes, dependiendo del beneficio que se le entrega; en otro sentido, la nómina no es más que darle un recibo en donde se informa al trabajador sus ingresos, sus descuentos, bonos, entre otros, estructurando el salario que en su día pactó con la compañía que lo emplea.

El principal beneficio que recibe el trabajador es el sueldo que recibe mes a mes, es por eso que cabe aclarar la diferencia entre el sueldo neto y el sueldo bruto. A groso modo, el sueldo neto es aquel que realmente recibe cada empleado, es decir, son los valores que le entrega el empleador al empleado, ya sea por transferencia, cheque o en efectivo; por otro lado el sueldo bruto se refiere al monto total que figura en el contrato del trabajador con todos los complementos salariales, retenciones y aportes al Seguro Social[10].

La administración de la nómina de una empresa a través de un sistema, constituye un proceso repetitivo en cierta forma, tomando en cuenta datos cambiantes dependiendo de algún evento ocurrido, ya sea en el tiempo que se encuentra, como por ejemplo los décimos, o por algún otro motivo, como por ejemplo, la salida de un empleado o la realización de un préstamo por parte de la empresa hacia el empleado, pero estos eventos son pocos con relación a muchos que siempre se tienen que calcular y que es algo tedioso para la persona encargada de esta actividad.

El control de esta actividad, es de suma importancia, manejarlo computacionalmente, es el objetivo de este módulo, dando al usuario la facilidad desde la

[10] http://www.finanzzas.com/la-nomina

creación de nuevas actividades como un nuevo rubro, cambiar fórmulas, entre otros, hasta la impresión de reportes, como el rol de pagos, rol individual, reporte de pagos al IESS, entre otros.

El módulo de nómina maneja los sueldos de cada empleado, tomando en cuenta todos los datos relacionados con el trabajador, ya sea, de tipo informativo, de descuento, o de incremento sobre el sueldo que recibe el empleado, así como también puede generar datos de tipo estadísticos[11].

A continuación se nombra alguno de los diferentes rubros que puede constituir la nómina:

Sueldo o Salario.- Indica la remuneración económica que recibe un trabajador por los servicios prestados a la empresa.

Décimo cuarto sueldo.- Es un bono escolar que es equivalente a un salario mínimo vital y se lo paga anualmente dependiendo que región este le empresa, si es costa se lo paga en el mes de febrero y si es sierra es en el mes de julio.

Décimo Tercer Sueldo.- Es considerado como para algunos como bono navideño. Es la suma de todos sus ingresos durante el año como bono, sueldo, horas extras, entre otras; y ese total se divide para los 12 mese del año.

Fondos de reserva.- Son valores que el IESS recauda a los trabajadores mes a mes y que a partir del año, según la nueva ley el empleador pagará con concepto de fondo de reserva de manera mensualizada y de forma directa a sus trabajadores el 8.33% de la remuneración de aportación.

[11] http://es.wikipedia.org/wiki/Sistema_de_administración_de_recursos_humanos

La versatilidad al cambio de parámetros, es una de las características que poseerá el módulo de nómina, de acuerdo a la situación actual en la que se encuentre la empresa, dependiendo de un evento interno por ejemplo, el gerente decide que todos los empleados se les incremente el sueldo en un 10%, o un evento externo.

2.1.3. Administración de Inventario

El inventario es el registro de los bienes y demás cosas pertenecientes a una organización de cualquier tipo, con el propósito de evaluar su existencia y fijar el volumen de producción y ventas, esto reflejará la situación actual de la producción de la organización para tomar decisiones importantes que pueden salvar o mantener la producción y cumplir objetivo propuesto.

Las razones por las cuales se requiere inventariar los bienes, son las siguientes:

- Reducir costos de pérdida.- Al realizar un pedido voluminoso sin tener la necesidad, influye varios inconvenientes, como la capacidad de almacenamiento, dependiendo del producto, se puede dañar, el recurso económico, el dar seguimiento y si no es la época del producto, se procede a tardar en recuperar la inversión y la ganancia.
- Reducir costos por material faltante.- El no tener el material suficiente para satisfacer la demanda, surgen varios problemas como clientes insatisfechos, adquirir a mayor precio los recursos, entre otros problemas.
- Reducir costos de adquisición.- Cuando existe la oportunidad de que un producto se pueda comprar por mayor a menor precio, puede ser favorable

para la organización, debido a que no se invierte mucho capital en la adquisición del producto[12].

Este módulo maneja proveedores, productos, categorías de productos, entre los más importantes, el cual ayudará a conocer en tiempo real los productos que se encuentran en bodega, que productos tiene órdenes de pedido pendientes, que usuarios han solicitado productos entre otra funcionalidad[13].

El control se lleva mediante tarjetas kárdex[14], en donde se registran cada producto, su valor de compra, la fecha de adquisición, el valor de la salida de cada producto y la fecha en que se sale de la bodega. De esta forma se puede conocer el saldo que se encuentra en bodega (la existencia) y el costo de cada producto.

2.2. Herramientas de diseño y desarrollo.

2.2.1. Herramientas de desarrollo.

2.2.1.1. Netbeans 6.9

Netbeans es un entorno de desarrollo integrado que principalmente fue hecho para el lenguaje de programación Java[15]. Cabe recalcar que Netbeans es un IDE (IntegratedDevelopmentEnvironments)[16] gratuito.

La plataforma Netbeans permite que los proyectos nazcan desde módulos, en donde contienen clases de Java escritas para interactuar con las API (Aplication Programming

[12] http://es.wikipedia.org/wiki/Inventario
[13] http://www.desarrollandoweb.com/demos/sistema-de-inventario-3.php
[14] Tarjetas Kárdex.- Documento para mantener el control de la mercadería para llevar el registro de entrada y salida de la mercadería de la empresa. http://www.mailxmail.com/curso-estados-financieros-ejercicios-herramientas-auxiliares/kardex-metodos-valoracion-inventarios-1
[15] Java.- Es un lenguaje de programación que permite el uso de programas punteros, como herramientas, juegos y aplicaciones de negocios. http://www.java.com/es/faq/whatis_java.xml
[16] IDE.- Son editores de código que además puede servir para depurar y facilitar las diferentes tareas necesarias en el desarrollo de una aplicación. http://aprender-java.blogspot.com/ide.html

Interface)[17] de NetBeans y un archivo especial que lo identifica como módulo. Estos módulos son fáciles de extender a lo largo del proyecto, ya sea por el código o por desarrolladores programando en el proyecto.

Esta herramienta, está escrita en java la misma que soporta cualquier tipo de lenguaje de programación, además de soportar el desarrollo de Aplicaciones empresariales con Java EE 5, incluyendo herramientas de desarrollo visuales de SOA (Arquitectura Orientada a Servicios)[18], herramientas de esquema XML (Extensible Markup Language)[19], orientación a web servicies, orientación a RESTFUL y modelado UML (Lenguaje Unificado de Modelado)[20].

Existen varios packs, como el NetBeans Enterprise Pack, NetBeans C/C++ Pack, PHP Pack, entre otros, los cuales hacen que este IDE, no solo se pueda desarrollar utilizando el lenguaje Java sino otros tipos de lenguaje.

Características destacadas de Netbeans 6.9

- Soporte JavaScript.- Ayuda en la escritura del código con la autocompletación de este y el resalte con colores con la refactorización.
- Mejoras en el Desempeño.- En esta versión se a aumentado el tiempo de inicio en un 40% destacando un menor consumo de memoria gracias a sus promociones más inteligentes.

[17] API.- Es una "llave de acceso" a funciones que nos permiten hacer uso de un servicio web por un tercero dentro de una aplicación web propia, de manera segura. http://desarrolloweb.com.ar/API
[18] SOA.- Diseño para la integración de aplicaciones independientes de manera que desde la red pueda accederse a sus funcionalidades. http://download.microsoft.com/0701-Real_World_SOA.pdf
[19] XML.- Permite diseñar etiquetas, permitiendo la definición, transmisión, validación e interpretación de datos entre aplicaciones. http://www.masadelante.com/flags/xml
[20] UML.- Es un conjunto de herramientas, que permite modelar sistemas orientados a objetos. http://www.ingenierosoftware.com/analisisydiseno/uml.php

- Soporte de Estructuras Spring.- Esto se debe a que tiene incorporado la librería Spring Framework 2.5 con controladores Spring Web MVC (Modelo Vista Controlador)[21]y un Asistente para la configuración de archivos XML.
- Nuevo Sporte MySQL en Exploración de Base de Datos.- Se puede registrar servidores MySQL que permiten gestionar la base de datos, realizando consultas y modificaciones directas a la base de datos, siendo así, el manejo aparte del código, la base de datos en sí.
- Soporte Java Beans.- Esto incorpora con la versión Java EE 5 donde el IDE proporciona un modelo de Beans en el Navegador con sus propiedades.
- Generador JSF CRUD (Crear, Obtener, Borrar y Actualizar)[22].- Este nuevo complemento, ayuda mucho a la programación de las transferencias básicas contra la base de datos, utilizando un lenguaje propio de java como es el JSF el se realiza a partir de clases de entidades soportando todo tipo de entidades y claves en la base de datos.
- Soporte RESTful Web Service.- Se puede realizar compilaciones de servidores JSR311-compliantRESTful Web utilizando Jersey[23].Se puede realizar pruebas haciendo clic derecho sobre el proyecto REST y darle a REST Test Client[24].

[21] MVC.- Es un patrón en el cual se separan los componentes de una aplicación en tres capas: datos, interfaz y lógica. http//www.tumaestroweb.com/curiosidades/que-es-mvc/
[22] CRUD.- Hace referencia a las cuatro operaciones básicas que se realiza sobre la información del modelo de datos. http://www.lobroweb.es/symfony_formularios/cap4/el_generador_crud.html
[23] Jersey.- API que permite trabajar con servicios Web basados en REST. http://cafelojano.wordpress.com/2010/01/24/un-cliente-jersey.
[24] http://netbeans.org/community/releases/61/index_es.html

2.2.1.2. MySQL 5.0

Es un sistema de administración de bases de datos para base de datos relacionable, multihilo y multiusuario. Así MySQL no es más que una aplicación que permite gestionar archivos llamados de bases de datos desarrollada como software libre, siendo una de las mejores bases de datos y más populares del mundo.

Aunque MySQL es software libre, MySQL también tiene una distribución comercial, la cual tiene derecho a un soporte técnico pero que no difiera de la versión que no es licenciada. Desarrollado en código ANSI[25], la mayor parte del código de la base de datos no es open-source, está patrocinado por una empresa privada que posee el copyright del código, esto puede diferenciar de otros proyectos que son de total open-source.

Características destacadas de la versión 5.0:

- Soporte multiplataforma.
- Procedimientos almacenados.
- Vistas actualizables.
- Modo Strict.
- Soporte X/Open XA[26] de transacciones distribuidas; transacción en dos fases como parte de esto, utilizando el motor InnoDB[27] de Oracle.
- Query caching.
- Sub-SELECTs
- Réplica con un maestro por esclavo, varios esclavos por maestro, sin soporte automático para múltiples maestros por esclavo.

[25] ANSI.- Se utilizan para dar formato a la salida de un terminal de texto y se basan en un estándar ANSI. http://es.wikipedia.org/wiki/codigo_escape_ANSI
[26] X/Open XA.- Es un estándar abierto para coordinar cambios en varios recursos, mientras se asegura la integridad de ellos. http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/ad/c0006984.htm
[27] InnoDB.- Es un motor transaccional de base de datos que permite el uso de commit y rollback, y recuperar fallos.http://dev.mysql.com/doc/refman/5.0/es/innodb-overview.htmln

- Usa tablas en disco b-tree para búsquedas rápidas con comprensión de índice.
- Seguridad: Ofrece un sistema de contraseña y privilegios seguros mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor.
- Soporta gran cantidad de datos hasta 50 millones de registros.
- Se permiten hasta 64 índices por tablas, cada índice puede consistir desde 1 hasta 16 columnas o partes de columnas. El máximo ancho de límite son 1000 bytes[28].

2.2.1.3. RESTful

REST (Representational State Transfer) es una arquitectura de software para sistemas hipermedias distribuidos tales como la Web. Son principios para el diseño de arquitecturas en la red que resumen como los recursos son definidos y decepcionados.

Aunque REST no es un estándar, está basado en estándares:

- HTTP[29]
- URL
- Representaciones de los recursos: XML/HTML[30]/GIF/JPEG..
- TIPOS MIME: text/xml, text/HTML.

El estilo de arquitectura subyacente a la Web es el modelo REST. Los objetivos de este estilo de arquitectura se listan a continuación:

[28] http://es.wikipedia.org/wiki/MySQL
[29] HTTP.- Hypertext Transfer Protocol, es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol
[30] HTML.- HypertextMarkupLenguage, es un lenguaje predominante en la elaboración de paginas web basado en el uso de etiquetas. http://es.wikipedia.org/wiki/HTML

- Escalabilidad de la interacción con los componentes. Esto es posible gracias a que la web ha crecido exponencialmente, se refleja en la variedad de clientes que pueden acceder a través de la Web: estaciones de trabajo, sistemas industriales, dispositivos móviles.
- Generalidad de interfaces. A diferencia de SOAP, REST está sobre el protocolo HTTP, esto significa que cualquier cliente puede acceder sin ninguna configuración adicional.
- Funcionamiento independiente. Esta es otra ventaja al estar sobre el protocolo HTTP, debido a que durante el transcurso de los años, van surgiendo nuevas tecnologías, pues sería un tanto complicado que servidores viejos se entiendan con servidores actuales, gracias al protocolo HTTP, esto puede ser posible.
- Compatibilidad con componentes intermedios. Los componentes intermedios se refiere a: proxys, caches, firewall y Gateway, es por esto que la compatibilidad con estos componentes permite reducir la latencia de interacción, reforzar la seguridad y encapsular otros sistemas.

REST logra satisfacer estos objetivos aplicando cuatro restricciones:

- Identificación de recursos y manipulación de ellos a través de representaciones. Esto se consigue mediante las URL[31] que se generan, puesto a que por medio de ellas se manipula los recursos,

[31] URL.- UniformResourceLocator, es una dirección que permite acceder a un archivo o recurso a nivel de la Web. http://www.alegsa.com.ar/Dic/url.php

pero hasta cierto punto, dependiendo de que se pueda realizar con la URL.

- Mensajes auto descriptivos. REST dicta que los mensajes http deberían ser tan detallado como sea posible, para que los componentes intermedios puedan entenderlos y así ejecutar métodos de acuerdo a los mensajes.
- Hipermedia como un mecanismo del estado de la aplicación. El navegador se encarga de saber navegar recurso por recurso y recoge información que le es de necesidad y si es necesario cambia el estado de la aplicación o de ciertos recursos, todo esto se realiza del lado del cliente, del lado del servidor, el servidor es de poco interés seguir la pista a todas las sesiones de clientes conectados a él, pero él sabe en qué estado se encuentran los recursos[32].

2.2.1.3.1. REST utiliza los métodos HTTP.

Esta es una de las características básicas de REST, puesto que la mayor parte de su funcionalidad lo hace a través de http, por ejemplo, HTTP GET es un método por el que se obtiene datos, el cual ejecuta una consulta esperando que el servidor Web la realice y devuelva un conjunto de recursos. Esto sucede con todos los métodos, convirtiendo una asociación uno-a-uno entre las operaciones de crear, leer, actualizar, borrar y los métodos HTTP. Se detalla a continuación la asociación:

- Se usa POST para crear un recurso en el servidor.
- Se usa GET para obtener un recurso.

[32] http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf

- Se usa PUT para cambiar el estado de un recurso o actualizarlo.
- Se usa DELETE para eliminar un recurso.

Cabe recalcar, que el uso indebido de estos recursos, puede perjudicar en cierta forma a la base de datos, por ejemplo, se menciona que el método GET se utiliza para realizar una consulta y que devuelve datos, pues existen algunas APIs que dentro de este método, incluyen operaciones transaccionales como update, los cuales alteran los datos, esto es muy riesgoso porque si se realiza una consulta, se espera que devuelva recursos y no que se alteren. El desarrollador debe de interpretar bien cada uno de los conceptos. Para ejecutar alguna acción de este tipo, se puede utilizar el método POST, que por medio de algún lenguaje como el XML, se envía el dato que se quiere que cambie y no enviar datos en el método GET.

2.2.1.3.2. Servicios con estado vs. Sin estado.

La siguiente imagen, muestra un servicio con control de estado en el servidor web, el cual una aplicación realiza peticiones para la página siguiente en un conjunto de resultados multi-página, en donde el servidor almacena la última página que solicitó.

ESPACIO EN BLANCO INTENCIONAL

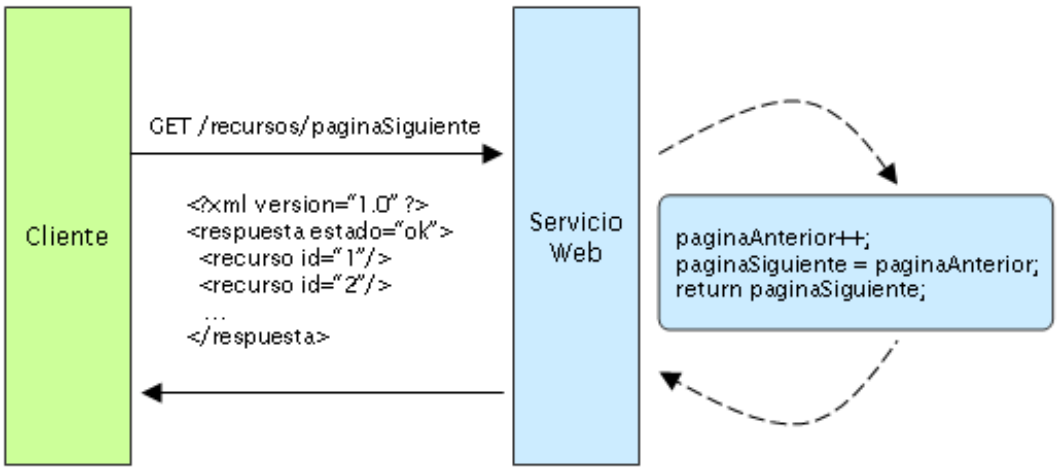


Figura 2.1: Servicio con Control de Estado[33]

Los servicios con estado tienden a volver complicado su manejo, ya que se necesita diseñarlo de una mejor manera, para distribuir la carga de procesamiento cuando entren en funcionamiento, de otra forma, se podría tener un error de java.io.NotSerializableException si se utilizaservlets/JSP y EJB.

La sobre carga de procesamiento produce este tipo de errores, debido a que se tiene que controlar cada una de las HttpSession[34] que se genere por cada uno de los clientes que se tengan conectados y controlar en que página se encuentra, aparte de que almacena en cache cada página que va solicitando.

Lo mejor que se debe realizar, sin mucha controversia es llevar el control del lado del cliente, la siguiente imagen, muestra cómo funcionaría el servicio sin control de estado.

[33] http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.html
[34] HttpSession.- Genera sesiones y relaciona cada pedido Http que haga el cliente con la sesión adecuada. http://docs.oracle.com/javaee/5/api/javax/servlet/http/HttpSession.html

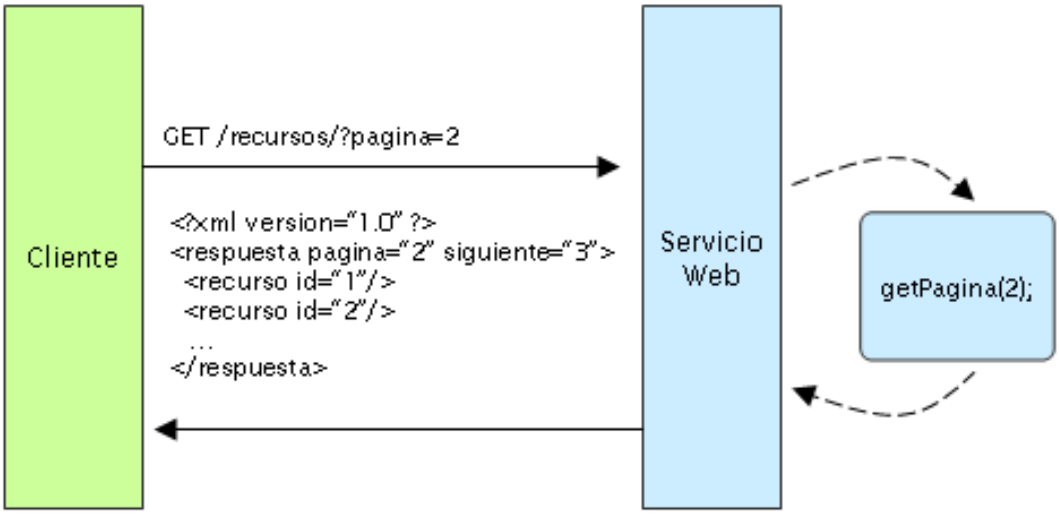


Figura 2.2: Servicio sin Control de Estado[35]

En donde el servidor es el que tiene el método getPagina(n) y el cliente es el que tiene el control de cual página es la que necesita procesar.

2.2.1.3.3. Responsabilidad del servidor.

Proporcionar los enlaces necesarios a la aplicación navegadora, para que pueda desplazarse por todos los recursos del servidor de acuerdo a lo que se requiera. Si la aplicación navegadora, realiza una petición a un nodo padre o contenedor, este debe de tener relacionado nodos hijos para mantener la conexión.

Proporciona respuestas que indican si se deben realizar cache o no, de esta forma, si se realizan peticiones que ya se han realizado, si están en cache, simplemente se envía los datos y así se elimina la generación de recursos que ya están realizados. Para esto el

[35] http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.html

servidor utiliza los atributos Cache-Control y Las-Modified de la cabecera en la respuesta HTTP para indicarlo.

2.2.1.3.1. Responsabilidades del cliente de la aplicación.

Utiliza el atributo Cache-Control para saber si debe de almacenar en cache o no. El cliente también lee el atributo Last-Modified y envía la fecha en el atributo If-Modified-Since del encabezado para preguntarle al servidor si el recurso cambió desde entonces y a esto se le llama GET Condicional.

Envía peticiones completas que pueden ser serviciadas en forma independiente a otras peticiones. Esto implica que el cliente hace uso completo de los encabezados HTTP tal como está especificado por la interfaz del servicio web, y envía las representaciones del recurso en el cuerpo de la petición[36].

2.2.1.4. Glassfish

Es servidor de aplicaciones que implementa la plataforma JavaEE5, por lo que soporta las últimas versiones de tecnología como: JSP, JSF, Servlets[37], EJBs, Java API para Servicios Web (JAX-WS[38]), Arquitectura JAVA para enlaces XML (JAXB) metadatos de Servicios Web para la Plataforma Java 1.0. y muchas otras tecnologías el cual se está convirtiendo en una opción muy atractiva para una gran cantidad de empresas y de muchos desarrolladores en el mundo. Eso se debe porque maneja una licencia de código abierto donde el desarrollador contribuye con la comunidad actualizando el código. Esto significa que el código mejora a un ritmo continuo más rápido que el código

[36] http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.html
[37] Servlets.- Son programas que se ejecutan en el lado del servidor y que están fabricadas a través del lenguaje JAVA. http://manuales.dgsca.unam.mx/webdina/servlets.htm
[38] JAX-WS.- Es un componente que sirve para construir frameworks para desarrollo basado en Eclipse y también para proveer de ejemplos de como utilizar el framework . http://wiki.eclipse.org/JAXWS

propietario, esto hace que las mejoras se enfoquen en lo que realmente quiere el usuario con un costo bajo.

Desarrollado por Sun Microsystems[39], hoy en la actualidad es su propietario Oracle Corporation[40], el cual tiene licencia dual a través de la licencia CDDL[41] y la GNU GPL[42].

A continuación se describe una de las características más importantes de GlassFish:

- **Seguridad en almacenamiento.** Los tipos de almacenamiento utilizados se los conoce como claves y certificados, y se describen de la siguiente manera: JKS (JAVA KEY STORE), y el NSS (NETWORK SECURITY SERVICES), esto engloba a un conjunto de bibliotecas diseñadas para permitir el desarrollo multiplataforma de seguridad habilitado para el cliente y las aplicaciones de servidor.
- **Inicio rápido.** Esta opción sirve para activar el servidor de aplicaciones más rápida. Para un arranque del servicio se lleva a cabo a través de una implementación en JAVA NIO la cual es parte de on-demand services framework.
- JVM. JAVA VIRTUAL MACHINE que utiliza de cierta forma Glassfish para configurar algunos parámetros.
- **Mecanismo de replicación de sesiones.** Opción para la replicación de sesiones.

[39]SunMicrosystem.- Fue una empresa que se dedicaba a vender ordenadores, componentes informáticos y servicios informáticos. http://es.wikipedia.org/wiki/Sun_Microsystems
[40]Oracle Corporation.- Es el proveedor líder mundial de gestión de información y la segunda mayor compañía de software independiente.
[41] CDDL.- Common Development and Distribution License, es una licencia open source y Free Software producida por Sun Microsystem. http://www.maestrosdelweb.com/editorial/licencias-libres-de-software-ii/j
[42] GNU GPL.- General PublicLicence, es una licencia CopyLeft fuerte, determina que en caso de hacer una distribución pública del software, este y sus versiones modificadas deben estar bajo la GNU GPL. http://www.maestrosdelweb.com/editorial/licencias-libres-de-software-ii/

Glassfish ofrece un amplio conjunto de motores de servicio y elementos de unión, debido a que el Open ESB es JBI-based, puede agregar servicio de motor JBI-compliant y componentes que ofrece Glassfish.

La siguiente figura muestra como está distribuido los servicios en un servidor de aplicaciones GlassFish con JBI runtime environment a través de JAX-WS y llamada a componentes de Sun JAVA EE Engine.

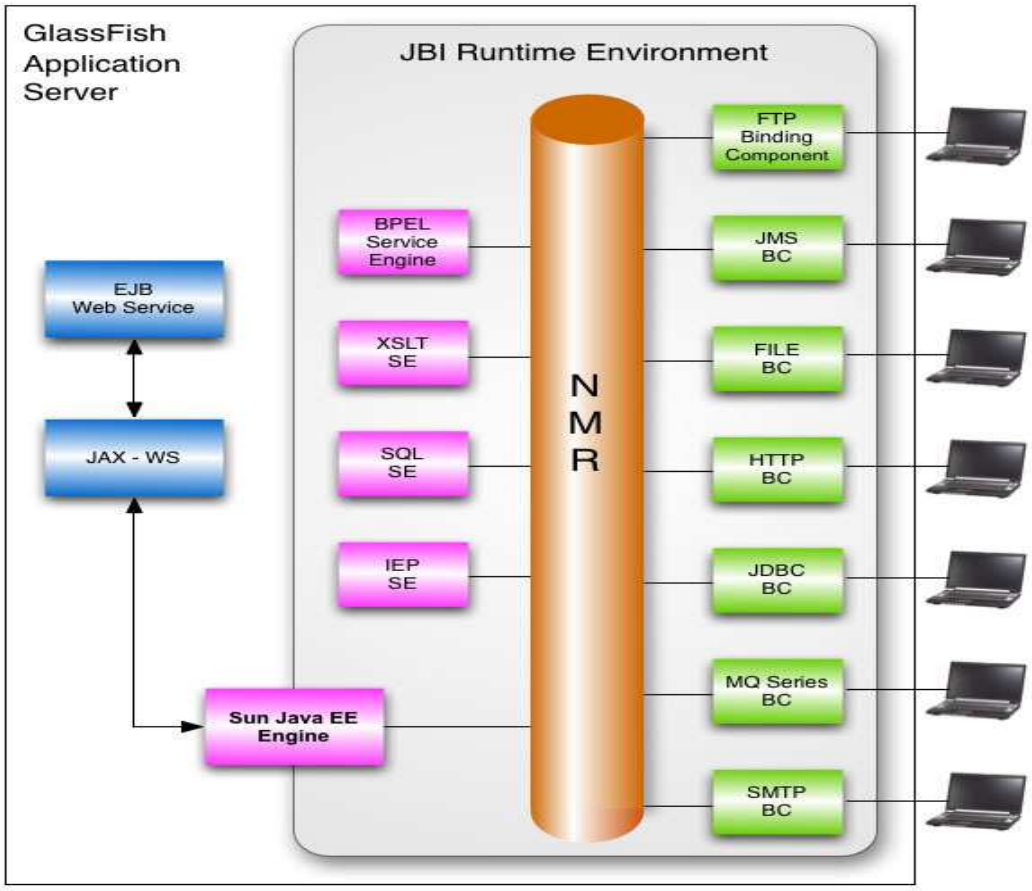


Figura 2.3: Distribución de Servicio[43]

[43] http://java.sun.com/developer/technicalArticles/glassfish/GFv2OpenforBusiness/index.html?cid=923034

2.2.1.5. Primefaces

PrimeFaces es una librería de componentes visuales open source para JavaServerFaces que cuenta con un conjunto de componentes ricos que facilitan la creación de las aplicaciones web, desarrollada y mantenida por Prime Technology, una compañía Turca de IT especializada en consultoría ágil, JSF, Java EE y Outsourcing. El proyecto es liderado por CagatayCivici, un miembro de "JSF ExpertGroup".

Las principales características que se pueda nombrar son las siguientes:

- Soporte nativo de Ajax[44], incluyendo Push/Comet.
- Kit para crear aplicaciones web para móviles.
- Es compatible con otras librerías de componentes, como JBossRichFaces.
- Uso de java script no intrusivo (no aparece en línea dentro de los elementos, sino dentro de un bloque <script>).
- Es un proyecto Open Source, activo y bastante estable entre versiones.

Algunos inconvenientes que podrían surgir:

- Para utilizar el soporte de Ajax se tiene que indicar explícitamente, por medio de atributos específicos de cada componente.
- No se puede utilizar el soporte de Ajax de JSF 2 (mediante <f:ajax>) con los componentes de Primefaces.

[44] Ajax.- Asynchronous JavaScript and XML, es un conjunto de tecnologías combinadas que permiten la actualización de la pagina por partes. http://www.masadelante.com/faqs/ajax

En la tabla 2.1 se muestra las diferencias entre otras librerías de componentes visuales en las que se puede desarrollar la capa cliente.

Tabla 2.1: Comparación Entre Diferentes Librerías[45]

Característica	ICEfaces	RichFaces	Primefaces
Soporte Ajax	Implementación de forma nativa mediante partialSobmit	Se implementa con una forma no muy transparente con Ajax4JSF	Se implementa de forma transparente utilizando algunos atributos específicos.
Librerías en las que se basan	Usa prototypejs, para ajax esta rescrito y efecto visuales script.aculo.us	Usa prototypejs y script.aculo.us aunque también jquery.	Utiliza jQuery y jQuery UI para efectos visuales.
Personalización de la interfaz de usuario	Incorpora el concepto de skin y distribuye 3 temas.	Incorpora el concepto de skins y distribuye 12 temas, aunque se puede encontrar más en el	Incorpota el concepto de skins, utilizando ThemeRoller, y dispone de 26 temas prediseñados.

[45]http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=introduccionPrimefaces

		repositorio de SNAPSHOTS	
Número de componentes	Tiene 79 componentes en la versión básica y 32 componentes más en la versión empresarial.	Tiene 212 componentes OpenSource entre los propios de RichFaces y los de Ajax4JSF.	Tiene 90 componentes OpenSource, algunos muy avanzados como el HTMLEditor. Además dispone de un kit para crear intefaces web para teléfonos móviles.
Licencia	MPL 1.1, que cubre la LGPL V 2.1. Si bien disponen de una versión empresarial con licencia comercial.	LGPL V 2.1. en su totalidad.	Apache License V2

Relevancia	Ha sustituido a Woodstock como librería de componentes de referencia de Sun para el desarrollo de aplicaciones RIA. Se distribuye, por defecto, con NetBeans.	Es la librería de componentes visuales de Jboss, se integar, por defecto con JbossSeam, aunque éste también soporta ICEfaces.	Ha sido una de las primeras librerías capaces de integrarse con JSF2 y viene pisando fuerte debido a la diversidad y calidad de sus componentes. Puede utilizarse junto a Richfaces, pero no es compatible con ICEfaces.

2.2.1.6. EJBs

Los Enterprise JavaBeans es un API que forma parte del estándar de construcción de aplicaciones empresariales J2EE de Oracle Corporation, los cuales cumplen la función de intermediar entre la capa web y diversos sistemas empresariales donde se encapsula la lógica de negocio de una forma integrada, no quedando dispersa su representación. Los EJBs están especialmente pensados para integrar la lógica de la empresa que se encuentra en sistemas distribuidos, de tal forma que el desarrollador no tenga que preocuparse por la programación a nivel de sistema sino que se centre en la representación de entidades y

reglas de negocio. Su especificación detalla cómo los servidores de aplicaciones proveen objetos desde el lado del servidor que son precisamente los EJB:

- Comunicación remota utilizando CORBA[46]
- Transacciones
- Control de la concurrencia
- Eventos utilizando JMS (Java messaging service)
- Servicios de acceso por nombres y de directorio
- Seguridad
- Servicio de transacciones con otros servidores
- Persistencia
- Ubicación de componentes en un servidor de aplicaciones.

ESPACIO EN BLANCO INTENCIONAL

[46] CORBA.- Common Object Request Broker Architecture, es un estándar que permite escribir componentes de software en múltiples computadoras para trabajar juntos. http://www.alegsa.com.ar/Dic/corba.php

Del lado servidor se puede distinguir la siguiente arquitectura:

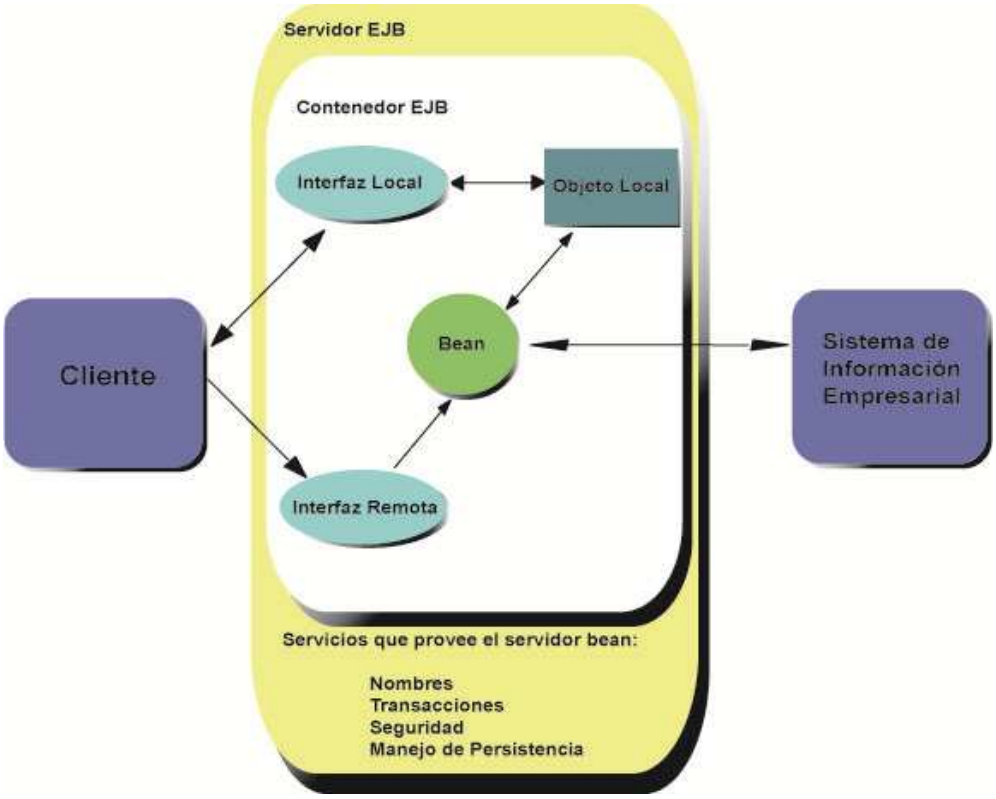


Figura 2.4: Arquitectura Ejb[47]

El servidor es el recipiente y gestor de los contenedores de EJB. Los contenedores con el contexto donde se ejecutan los Beans.

Los contenedores sirven como interfaz entre el Bean y el servidor, y tienen como reglas las siguientes:

- Pueden generar y desplegar los Beans.
- Un servidor puede tener varios contenedores pero cada contenedor puede tener a un solo Bean.

[47] Imagen obtenida del siguiente sitio Web:
http://www.proactiva-calidad.com/java/ejb/introduccion.html

- Para los desarrolladores, solo necesitan saber su funcionamiento principal, no necesitan saber, como opera internamente.

Tipos de EJB.

- Beans de sesión. Se crean por cada cliente conectado para realizar ciertas funciones necesarias para cada cliente por ejemplo, mostrar las transacciones consolidadas por cada cliente conectado.
- Beans de entidad. Se utilizan para acceder a la base de datos y mantienen la integridad de los datos de persistencia, es decir, mantienen los datos integrados entre los datos de la base de datos y los datos que se crean en memoria.
- Beans orientados al mensaje. Son invocados asincrónicamente y utiliza el java MessageService.

Elementos del EJB.

Para desarrollar un EJB se necesita los siguientes elementos:

- Un interfaz local, que ayuda a la creación, acceso y destrucción de las instancias de los Beans y están implementadas en el contenedor.
- A través de la interfaz remota se accede a los métodos de negocio que han sido implementados en el Bean.
- La clase Bean define la lógica de negocio a la que se accede a través del interfaz remoto.

2.2.2. Herramientas de Diseño.

2.2.2.1. Star UML 5.0

StarUML es una herramienta para el modelamiento de software basado en los estándares UML (Unified Modeling Language) y MDA (Model Driven Arquitecture)[48] corriendo sobre la plataforma Win32, de licenciamiento libre aunque en un principio era de uso comercial, de esta forma se puede tener una herramienta de modelamiento de software que reemplaza a las de uso comercial tradicionales como son RationalRose, Together, entre otras.

Esta herramienta posee algunas características como se detallan a continuación:

- Cumple con el diseño UML mediante el uso de:
  - Diagrama de caso de uso
  - Diagrama de clase
  - Diagrama de secuencia
  - Diagrama de colaboración
  - Diagrama de estados
  - Diagrama de componentes
  - Diagrama de despliegue
  - Diagrama de composición estructural
- Crear elementos que no necesariamente sean del estándar UML.
- La capacidad de generar código desde los diagramas y viceversa, actualmente diseñado para los lenguajes c++, c# y java.
- Generar documentos como Word, Excel y PowerPoint sobre los diagramas.

[48] MDA.- Concibe la construcción de modelos de software, distintos niveles de abstracción, como artefacto principal en el desarrollo de software. http://users.dsic.upv.es/~einsfran/mda/

- Patrones GoF(GangofFour)[49], EJB (Enterprise JavaBeans) y personalizados.
- Plantillas de proyectos.
- Posibilidad de crear plugins para el programa.[50]

2.3. JEE

Java Platform, Enterprise Edition o Java EE, es una plataforma que sirve para ejecutar aplicaciones en el lenguaje de programación java de tipo n Capas distribuidas con componentes de software modulares ejecutándose en un servidor de aplicaciones.

2.3.1. Razones para la creación de la plataforma JEE

A continuación se detalla algunas de las razones para la creación de la plataforma JEE:

- Programación eficiente. Cuando se utiliza la plataforma JEE, se tiene un estándar para realizar el desarrollo por Capas y una distribución de tecnologías a lo largo de las capas, por ejemplo, en la capa de cliente, se tiene paginas applets[51], aplicaciones JAVA, en la capa web, se tiene servlets, paginas JSP, etc. Esto hace diferencia cuando se usa HTML, JavaScript, CGI[52], las cuales no tienen un modelo único de desarrollo.
- Extensibilidad frente a la demandad del negocio. La extensión del negocio de acuerdo al número de usuarios que se tenga, se incorpora

[49] Es el nombre con el que se conoce comúnmente a los autores de libro DesignPattern. http://aspalliance.com/827_Gang_of_Four_GOF_Design_Patterns
[50] http://cnx.org/content/m15092/latest/
[51] Applet.- Componente de software escrito en lenguaje de programación Java que se ejecuta bajo una aplicación más grande que lo contiene como el navegador web. http://www.alegsa.com.ar/Dic/applet.php
[52] CGI.- Common Gateway Interfaces, tecnología de la WWW, que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en el servidor Web. http://www.alegsa.com.ar/Dic/cgi.php

nuevos recursos y deben de usar tecnología que pueda funcionar en múltiples servidores y que puedan tener un balanceo de carga.

- Integración. Los equipos de ingeniería deben de precisar estándares que permitan integración entre diversas capas de software.

A continuación se detalla alguna de las capas de la arquitectura JEE que se complementan en función del tipo de servicio y contenedores involucrados:

- Capa de cliente, también conocida como capa de Presentación o de Aplicación. Existen componentes de Java (applets o aplicaciones) y no-Java (Html, JavaScript, etc.).
- Capa Web. Es la capa intermedia donde se encuentra servlets y las JSP que ayuda a conectar a la capa cliente con el resto de capas.
- Capa Enterprise JavaBeans. Permite que múltiples aplicaciones tengan acceso de forma concurrente a datos y lógica de negocio.

ESPACIO EN BLANCO INTENCIONAL

La gráfica siguiente, muestra a mayor detalle cómo están distribuida cada una de las capas antes mencionadas, donde está incorporado cada uno de los posibles elementos que se puede implementar:

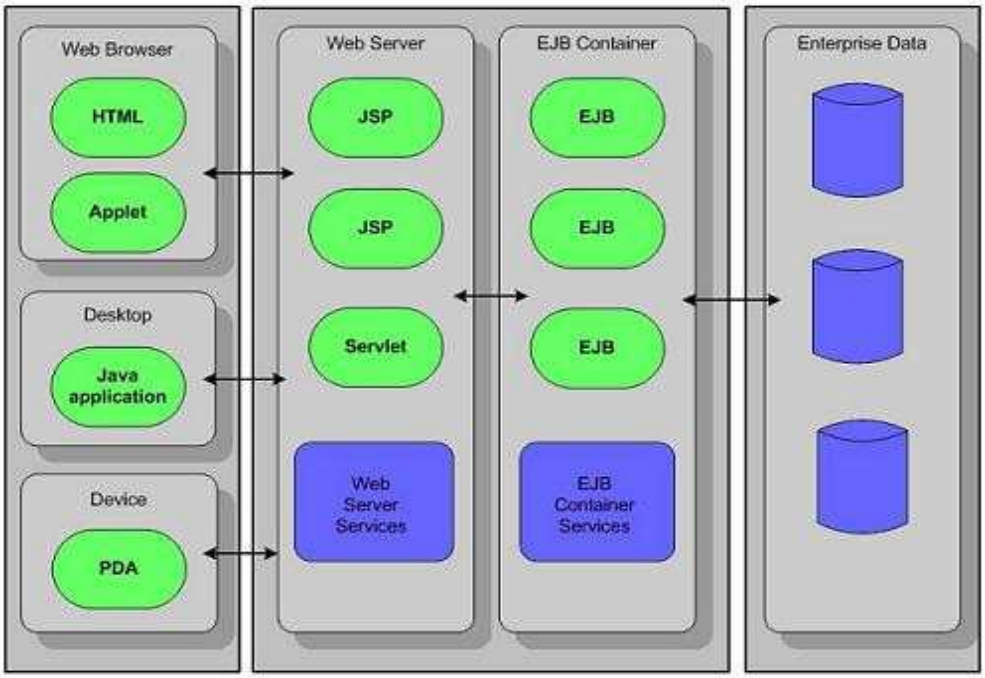


Figura 2.5: Distribución de Capas[53]

2.3.2. Diferencias entre Java EE y SE

La tecnología Java es un lenguaje de programación y también es una plataforma. El lenguaje de programación Java es de un alto nivel de programación orientada a objetos que tiene una sintaxis y estilo particular. Hay varias plataformas las cuales muchos programadores no entienden las diferencias que existen entre sí.

Las plataformas de lenguaje de programación Java. Existen cuatro plataformas del lenguaje de programación Java:

- Java Plataform, Standard Edition (Java SE)
- Java Plataform, Enterprise Edition (Java EE)
- Java Plataform, Micro Edition (Java ME)
- Java FX

[53]http://www.proactiva-calidad.com/java/arquitectura/index.html

Todas las plataformas Java consisten en un Java Virtual Machine (VM) y un Application Programming Interfaces (API). El Java Virtual Machine es un programa, para una particular plataforma de hardware y software en donde se ejecuta la tecnología Java. Cada plataforma provee una máquina virtual y un API, y estos permiten a las aplicaciones escribir para esa plataforma ejecutar en algunos sistemas compatibles con todas sus ventajas que da el lenguaje de programación Java como son independencia de plataforma, poderoso, estabilidad, fácil para el desarrollo y seguridad.

Java SE

El API de Java SE provee la funcionalidad del núcleo del lenguaje de programación Java, esto define todo desde básicos types y objetos del lenguaje hasta altos niveles de clases las cuales están usadas para la red, seguridad, acceso a base de datos, interfaz gráfica de usuario y parseo de XML.

Java EE

La plataforma Java EE está construida en lo alto de la plataforma Java SE. La plataforma Java EE provee de un API y un entorno para desarrollo y ejecución de larga escala, multi-tiered[54], escalable, y aplicación de redes seguras.

2.4. Metodología XP

2.4.1. ¿Por qué utilizar esta metodología en este proyecto?

Esta metodología es muy idónea debido a que posee las herramientas necesarias para que el proyecto se base en la realización del sistema y no en demasiada documentación que en algunas metodologías aplican y que se llega a invertir mucho

[54]Multi-tiered.- Es un término referente a aplicaciones distribuidas en muchas capas. http://www.alegsa.com.ar/Dic/ap0*-55licacion%20distribuida.php

tiempo. También se ajusta muy fácilmente debido a que se cuenta con la participación del usuario administrador el sistema, quien facilito todo lo necesario para que el sistema se ajuste en su totalidad a lo que el usuario necesita.

Otro gran aporte son las reuniones diarias al inicio de las labores, ya que con esta ayuda se puede ajustar sin ningún problema cada una de las variantes que se pueden dar en el transcurso del desarrollo del sistema.

La ayuda de las tarjetas CRC junto con las historias de usuario, ayuda en gran medida al desarrollador a entender como de una forma abstracta lo que el usuario quiso decir.

XP se aplica a proyectos pequeños donde el usuario es la principal herramienta para que se cumpla lo ágil que es la metodología y este proyecto cumple en gran parte estas principales condiciones.

2.4.2. Fundamentos (Mejores prácticas)

La Programación Extrema (XP) es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado la cual se diferencia de las otras metodologías tradicionales en que pone más énfasis en la adaptabilidad que en la previsibilidad, esto se refiere a que durante la marcha se puede ver afectado los requisitos y estos pueden ser modificados para que el proyecto le sirva de mejor forma a el usuario.

De los muchos beneficios que se puede destacar con esta metodología, se puede considerar que la Programación Extrema es una de las metodologías que adopta las mejores prácticas de otras metodologías de desarrollo de acuerdo a lo que se quiere llevar a cabo con el proyecto y aplicarlo de manera dinámica durante el Ciclo de Vida del software.

Mientras otras metodologías basadas en la perfecta sincronización entre documentación diseño y código dividiendo estas tareas entre diversos grupos de programación y contando además con la necesidad de volver a aplicar los pasos anteriores estimando a que el presupuesto no se acabe para continuar el proyecto, la XP proporciona también una serie de reglas que ayuda a solucionar estos inconvenientes en el Ciclo de Vida de un proyecto de desarrollo de software. A continuación se detalle alguno de los aspectos destacados de esta sección:

Planificación:

Para realizar un buen trabajo, hay que planificarse de una mejor forma, esto en la XP se traduce de la siguiente forma:

- La planificación en XP, se comienza por las "user-stories" o historias de usuarios con el propósito de analizar estos documentos.
- Los programadores no pueden tomar decisiones y funcionalidades del sistema que no consten en el diseño, debido a que si se las aplica, puede que no se las incluya en el diseño y nunca conste en el mismo, es como que si nunca hubiera existido en el sistema, se debe de controlar esa parte.

Sistema metafórico:

Cuando se comienza a desarrollar el software, siempre existe el problema de cómo nombrar los métodos, funciones, variables, relaciones etc. de un modo coherente y que todos los componentes del equipo puedan entenderlo.

Lo que se debe hacer es que los identificadores deberán ponerse en concordancia con ciertas reglas preestablecidas con anterioridad por el grupo, con esto se consigue, no perder tiempo valioso de programación en pensar un nombre para un método o similar y, que el equipo tenga la confianza y rapidez en colocar un nombre a los elementos que va a utilizar.

40 horas por semana:

Este punto, es de suma importancia puesto a como Jefes de Proyecto, no se puede decir que porque se trabaja más tiempo, se va a acabar más rápido, no es verdad, porque largas jornadas de trabajo, provocan más errores debido a la concentración y a la motivación del equipo.

La programación extrema se basa en 40 horas por semana. De tal forma se evita que los programadores e fatiguen y que las horas de trabajo sean horas altamente productivas. Estas horas semanales, pueden tener varias formas:

- Las 8 horas diarias pueden repartirse en 5 días de forma que 4 horas por la mañana y otras 4 horas por la tarde. El fin de semana descanso merecido.
- 7 horas cada día durante 5 días. 5 horas mañana y 3 horas por la tarde. El sábado se trabajan 5 horas y el domingo se descansa.

Versiones pequeñas:

Lo ideal en todo proyecto es que el cliente o usuario tuviese una imagen clara y concisa de lo que realmente quiere. El cliente sabe que es lo que quiere, pero

desgraciadamente no tiene la capacidad de transmitir esa idea en un lenguaje técnico que el desarrollador pueda entender.

La idea principal de versiones pequeñas es crear una versión de la aplicación pequeña, con funcionalidades básicas para que el usuario pueda verla y pueda ir introduciendo cambios que le pueda dar forma a la idea final de que es lo que quiere.

Estas versiones pequeñas, tiene varias ventajas:

- El cliente puede palpar como está evolucionando el software introduciendo cambios continuos, lo cual permite que el cliente no se sienta engañado y se sienta líder del proyecto.
- La vuelta atrás siempre será más fácil corregir versiones pequeñas, que una gran versión que contiene un gran número de modificaciones.
- Evita que el código se descontrole rápidamente y facilita la claridad y la sencillez tanto del diseño como del código del programa.

Propiedad colectiva del código:

El código que el equipo genere para llevar a cabo un proyecto es propiedad de cada uno de los componentes del equipo.

De esta forma, cualquier componente del equipo, podrá modificar un módulo o porción de código si lo cree conveniente.

Dentro del equipo, no habrá, derechos de autor ni ninguna restricción para poder modificar el código de otro compañero.

Todo el código que se genera dentro del equipo, el autor pondrá a disposición al equipo para de ser el caso, se posible su modificación.

Todos estos atributos direccionados a la colectividad del código, contribuyen a que el desarrollo sea de más rapidez sin pasar por el dueño del código, si no simplemente se requiere un cambio y el que desea ese cambio, el mismo modifica.

2.4.3. Programación en pareja.

La programación del código se realiza en pareja, es una manera de trabajar según la cual dos desarrolladores se responsabilizan de una parte del desarrollo, en el mismo puesto de trabajo y usando un mismo computador, de forma que los dos están pendientes del trabajo que toca realizar y los dos enfrentaran cada problema que se vayan, independiente de alguna necesidad de que el programador lo requiera, se lo realiza todo el tiempo. Los beneficios son los siguientes:

- Revisión inmediata del código mientras se lo construye.
- Corrección de errores en el código de inmediato.
- La otra persona será la encargada de guiar al que programa porque conoce el diseño que tiene que seguir y así no se perderá en el diseño el programador principal.
- Se alcanza objetivos más grandes que el encargarle el mismo trabajo a una sola persona.
- El diseño desarrollado por este equipo, es más fácil y más reutilizable.

- Permite el entrenamiento de programadores novatos y el aprendizaje de ellos hacia los expertos.
- Las personas se sienten a gusto porque tienen una compañía que siempre está pendiente del trabajo que tiene que realizar.
- Las parejas producen un programa de mejor calidad en un tiempo corto.
- La programación en pareja, hace que se produzca la eliminación de procesos erróneos redundantes, y eso es un aporte muy importante en la mejora continua del producto.

2.4.4. Fases de la metodología XP.

Como metodología, la XP, presenta algunos puntos en el desarrollo incremental.

Para ver estos puntos, se presenta varias fases:

Codificar: Esto hace referencia que cada vez que se codifique, se debe de tener al final del día algo que funcione y que sea útil para el cliente. Se puede concluir que todo el software se produce mediante pequeñas versiones terminadas incrementales de producción corta.

Probar: Se debe de asegurar que desde el momento que se tiene las historias de usuario, se debe de probar toda esta información para poder comprender que eso realmente funciona y de paso tener una visión más objetiva del problema.

Escuchar: Tanto para diseñar como para desarrollar, se tiene que saber exactamente qué es lo que vas a realizar, para ello debes de saber escuchar al cliente, al jefe de proyecto y a todo el mundo en general.

Diseñar: El diseño debe de ser muy claro y conciso, por lo que se debe de diseñar lo que realmente se va hacer no lo que se quiere que se haga, diseñando cosas que se van a utilizar no cosas que son para el futuro porque es una pérdida de tiempo.[55]

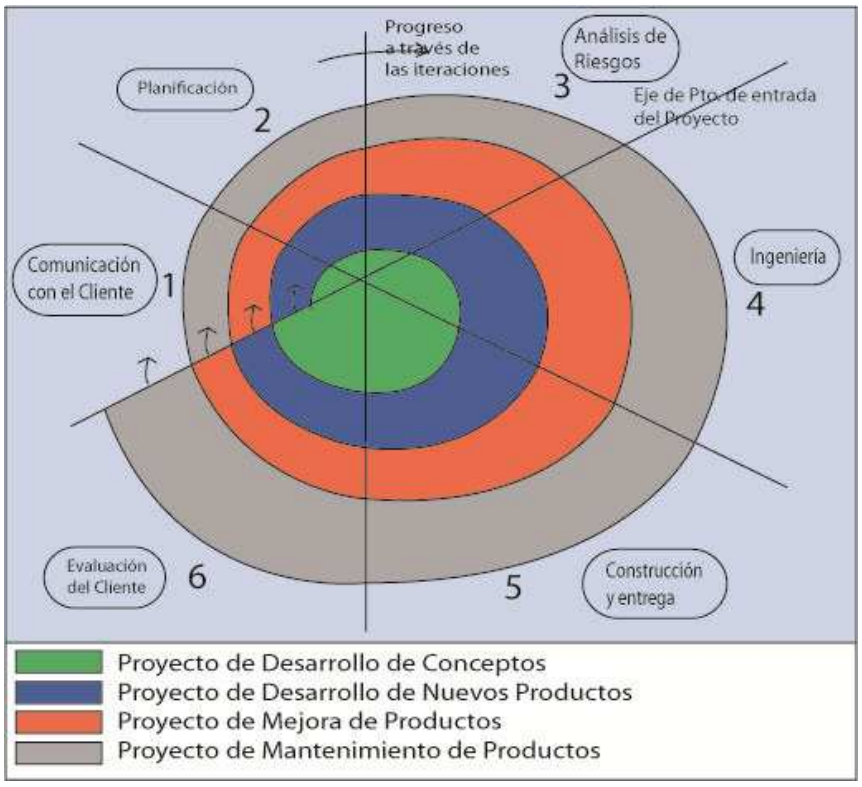


Figura 2.6: Entregables de Xp

2.4.5. Ciclo de vida del software en XP.

El ciclo de vida en XP, consisten de seis fases:

- Exploración.- Se analiza las historias de usuario a grandes rasgos para la primera entrega del producto. Por otro lado, el equipo de desarrollo se familiariza con las herramientas que se va a utilizar. Esta fase toma entre pocas semanas a pocos meses, dependiendo el que tan es familiar la tecnología que se va a utilizar con los desarrolladores.

[55] http://www.esp.uem.es/jccortizo/xp.pdf

- Planificación de la Entrega (Release).- Aquí el cliente clasifica por su prioridad cada una de las historias de usuario y se analiza con el equipo de desarrollo cuanto esfuerzo toma hacer cada una de ellas.
- Iteraciones.- En esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. En la primera iteración se intenta establecer una arquitectura donde en ella se procederá a implementar las demás iteraciones. Las demás iteraciones, se realizaran de acuerdo a la prioridad que el cliente le ah dado a las historias de usuario.
- Producción.- Esta fase requiere de pruebas antes de implementar donde el cliente, para que esté listo en el funcionamiento. Después que se prueba, se procede a la implementación y es donde pueden surgir cambios donde se los procederá a cambiar otra fase como por ejemplo la de mantenimiento.
- Mantenimiento.- El proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para esto se requiere de tareas de soporte donde el cliente.
- Muerte del Proyecto.- Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema[56].

2.4.6. Estándares de codificación.

La estandarización ayuda a que el código sea más rápido de implementar, debido a que se sabe como nombrar los elementos que se van a colocar y se sabe que paso se va a seguir durante el desarrollo gracias a que se tiene reglas, esto hace referencia a lo que se

[56] http://oness.sourceforge.net/proyecto/html/ch05s02.html

habló en sistema metafórico. También es entendible para todo el equipo de desarrollo y de esta forma se tiene un desarrollo más ordenado.

2.4.7. Refactoring.

Se refiere a un grupo de técnicas de codificación usadas para mejorar código defectuoso escrito por otros programadores o de los propios creadores. Se ha de confiar en las pruebas a las que se comenta el código para mantener la confianza de que el "Refactoring" no ha perjudicado algo más y los cambios producidos han sido demasiado pequeños.

Durante el ciclo de vida del desarrollo, se puede ver en la necesidad de crear nuevos métodos o que se cambian métodos que ya existen, esto se discutirá entonces la necesidad de refactorizar por parte del equipo, que confirmará la validez o no de dicho proceso.

Se trabajará en la nueva estructura, y cuando esté listo se le comunicará al equipo de programación. Una vez decidida su aprobación, el equipo de desarrollo procederá a la construcción del mismo. Concluida la nueva funcionalidad del objeto, se procederá a las pruebas respectivas, luego se procede a incluir este objeto a otros objetos los cuales requieren de la funcionalidad del nuevo objeto.

Se debe señalar que la modificación de código ya existente, es algo de sumo cuidado, porque de hacerlo mal, se puede dañar muchas cosas y puede resultar algo catastrófico en el desarrollo de software.

Modelos de Refactorización:

- Encadenación de los constructores. Si se tiene varios constructores que contienen código duplicado, el objetivo es unir los constructores para obtener menos código duplicado.
- Encapsulamiento de subclases con métodos de fábrica. El objetivo de hacer esta refactorización es ocultar los detalles de implementación para aquellas personas que no necesitan conocerla.
- Creación de clases. En ciertas ocasiones, no se puede encontrar con clases que tienen demasiados métodos, lo cual hace que la funcionalidad de esa clase se vea distorsionar, por lo que hay que crear clases auxiliares para así ayudar al entendimiento del código.
- Reemplace cálculos condicionales con "Strategy". Existen métodos que tienen muchas condiciones y esto distorsiona el entendimiento del mismo, lo mejor es reemplazar estas condiciones por métodos que tengan estos cálculos.
- Reemplace Construcciones Primitivas con Compuestas".
- Encapsule Compuestos con Constructores.
- Extraiga la Lógica del caso-uso hacia "Decorators". Las clases o métodos tienen casos especiales que tratan la lógica pero se debe conservar la lógica exportando los casos especiales hacia "Decorators".
- Reemplace las Notificaciones de código por "Observadores".
- Transforme Acumulaciones a un Parámetro Colectivo.
- Métodos Compuesto.

2.4.8. Pruebas.

El Test before Programming en XP[57] se basa en elaborar un completo conjunto de pruebas para un módulo del programa incluso antes de que el módulo este acabado con la finalidad de trabajar siempre en un entorno seguro, libre de errores. Solo realizando estas pruebas sobre el código y obteniendo resultados esperados se tendrá la certeza de que el código es correcto.

Se puede distinguir varias situaciones basadas en lo visto anteriormente:

- Código correcto # Pruebas correctas. Es lo ideal que se quiere encontrar, es de suponer que si el código está correcto, va a pasar las pruebas con mucha satisfacción.
- Código correcto # Conjunto de pruebas incompleto. Esto es común cuando solo se hace pruebas pequeñas y que no sedan ningún tipo de error.
- Código correcto # Conjunto de pruebas incorrecto. Es un caso grave debido a que el código que se escribe se le realizapruebas que no tienen que ver con el caso.
- Código incorrecto # Conjunto de Pruebas correcto. Es el caso más común debido a que se hizo pruebas correctas, devolvió muchos errores y en consecuencia a eso, se procedió a corregir los mismos.

2.4.9. Integración de código.

El idea de la metodología XP es la de tener siempre versiones simples y manejables del sistema.

[57] eXtremeProgramming.- Es una metodología de desarrollo de ingeniería de software dentro del grupo de metodologías ágiles. http://www.extremeprogramming.org/

Los cambios no se deben de realizar después o cuando el programador pueda, esto retrasa y se torna dificultoso en el momento que se lo realiza tarde, se debe de realizarlo en forma rápida, de acuerdo a como vayan surgiendo, aprovechando el entendimiento del código en ese momento.

Los cambios también deben de ser continuos y sin dejarse acumular e integrándolos de un golpe a todos. La integración continua favorece no solo la legibilidad del código sino además la prevención de errores y si se produce su localización rápida y sencilla.

La forma de llevar a cabo la integración continua es introducir todos los cambios producidos a lo largo del día a final de la jornada.

CAPÍTULO 3

3. ANÁLISIS Y DISEÑO DEL SISTEMA

3.1. Planeación.

3.1.1. Historias de usuarios entregadas por la empresa

En esta sección se especifica los requisitos que se levantó previamente mediante lo que se conoce en XP como historias de usuarios, las cuales fueron redactadas por el desarrollador del sistema para que el usuario se concentre en el requerimiento y este pueda ser bien detallado, concluyendo con la revisión y aprobación del usuario final de cada una de las historias de usuario. A continuación se mostrará todas las historias de usuarios entregadas para la realización de este proyecto.

MÓDULO SISTEMA

Tabla 3.1: Historia de Usuario 1

Historia de usuario	
Número: 1	Usuario: Administrador
Nombre historia: Registrar usuarios nuevos	
Prioridad en negocio: Alta	Riesgo en desarrollo: Baja
Puntos estimados: 2	Iteraciones estimadas: 1
Descripción: • El sistema podrá registrar usuarios o crear usuario por medio de una pantalla de mantenimiento de usuario	

Observaciones:

Tabla 3.2: Tarea 1.1

Tarea	
Número: 1.1	Número de Historia: 1
Nombre tarea: Crear interfaz de formulario para el registro	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario ingrese los datos requerido del usuario nuevo del sistema acompañado de una interfaz amigable.	

Tabla 3.3: Tarea 1.2

Tarea	
Número: 1.2	Número de Historia: 1
Nombre tarea: Crear base de datos para usuarios	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para guardar nuevos usuarios	

Tabla 3.4: Historia de Usuario 2

Historia de usuario	
Número: 2	Usuario: Administrador
Nombre historia: Permisos de acceso a usuarios	
Prioridad en negocio: Alta	Riesgo en desarrollo: Baja
Puntos estimados: 2	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de filtrar a los usuarios el nivel de acceso a cada una de las pantallas del sistema.	
Observaciones:	

Tabla 3.5: Tarea 2.1

Tarea	
Número: 2.1	Número de Historia: 2
Nombre tarea: Crear formulario de mantenimiento de permisos	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario ingrese los datos requeridos de los permisos de cada usuario correspondientes a cada pantalla.	

Tabla 3.6: Tarea 2.2

Tarea	
Número: 2.2	Número de Historia: 2
Nombre tarea: Crear base de datos para permisos de usuarios	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	

Descripción: • Se diseña una base de datos para registrar los permisos de pantalla correspondientes a cada usuario

Tabla 3.7: Historia de Usuario 3

Historia de usuario	
Número: 3	Usuario: Administrador
Nombre historia: Registro del Menú del sistema	
Prioridad en negocio: Alta	Riesgo en desarrollo: Baja
Puntos estimados: 2	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de poder modificar y crear el menú del sistema.	
Observaciones:	

Tabla 3.8: Tarea 3.1

Tarea	
Número: 3.1	Número de Historia: 3
Nombre tarea: Crear formulario de mantenimiento de Menú	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	

Descripción: • Se realiza el desarrollo de un formulario para que el usuario ingrese los datos requeridos para el menú del sistema.

Tabla 3.9: Tarea 3.2

Tarea	
Número: 3.2	Número de Historia: 3
Nombre tarea: Crear base de datos para Menú	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar el menú del sistema.	

Tabla 3.10: Historia de Usuario 4

Historia de usuario	
Número: 4	Usuario: Administrador
Nombre historia: Registro de empresas que maneja el sistema	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 3	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar empresas y llevar	

cada una de ellas de forma separadas para el mejor control de l información.
Observaciones: Tomar en cuenta que no se debe de confundir la información entre empresas para su óptimo funcionamiento.

Tabla 3.11: Tarea 4.1

Tarea	
Número: 4.1	Número de Historia: 4
Nombre tarea: Crear formulario de mantenimiento de empresas	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario ingrese los datos requeridos de cada una de las empresas que maneja el sistema.	

Tabla 3.12: Tarea 4.2

Tarea	
Número: 4.2	Número de Historia: 4
Nombre tarea: Crear base de datos para empresas del sistema	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1

Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar las empresas que maneja el sistema.	

Tabla 3.13: Tarea 4.3

Tarea	
Número: 4.3	Número de Historia: 4
Nombre tarea: Diseñar el registro de datos por empresas	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos global para que cada registro del sistema sea indexado por empresa en todas las tablas del sistema.	

MÓDULO NÓMINA

Tabla 3.14: Historia de Usuario 5

Historia de usuario	
Número: 5	Usuario: Administrador

Nombre historia: Registro de personas empleadas en cada empresa	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 2	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar a las personas empleadas en cada empresa.	
Observaciones:	

Tabla 3.15: Tarea 5.1

Tarea	
Número: 5.1	Número de Historia: 5
Nombre tarea: Crear formulario de mantenimiento de personas empleadas de cada empresa	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda registrar a cada persona empleada en el cada una de las empresas del sistema	

Tabla 3.16: Tarea 5.2

Tarea

Número: 5.2	Número de Historia: 5
Nombre tarea: Crear base de datos para empleados de cada empresa del sistema	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar las personas empleadas de cada una de las empresas.	

Tabla 3.17: Historia de Usuario 6

Historia de usuario	
Número: 6	Usuario: Administrador
Nombre historia: Registro de tipo de rubro	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 2	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar los diferentes tipos de rubros que maneja la nómina de cada empresa.	
Observaciones:	

Tabla 3.18: Tarea 6.1

Tarea

Número: 6.1	Número de Historia: 6
Nombre tarea: Crear formulario de mantenimiento de tipo de rubro	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda registrar a cada tipo de rubro funcional para la nómina.	

Tabla 3.19: Tarea 6.2

Tarea	
Número: 6.2	Número de Historia: 6
Nombre tarea: Crear base de datos para tipos de rubros	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar los tipos de rubro que se maneja para la nómina.	

Tabla 3.20: Historia de Usuario 7

Historia de usuario

Número: 7	Usuario: Administrador
Nombre historia: Registro de Rubros	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 3	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar los rubros para el procesamiento de los roles de pagos de la nómina.	
Observaciones: Tener en cuenta que en esta historia es muy indispensable para el buen cálculo del rol de pagos.	

Tabla 3.21: Tarea 7.1

Tarea	
Número: 7.1	Número de Historia: 7
Nombre tarea: Crear formulario de mantenimiento de rubros	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda registrar a cada rubro de la nómina.	

Tabla 3.22: Tarea 7.2

Tarea	
Número: 7.2	Número de Historia: 7
Nombre tarea: Diseño de ubicación de fórmulas de cada rubro	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el diseño de ubicación de fórmulas que ayudarán al funcionamiento del cálculo de los rubros de cada rol.	

Tabla 3.23: Tarea 7.3

Tarea	
Número: 7.3	Número de Historia: 7
Nombre tarea: Crear base de datos para rubros	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar los rubros que maneja la nómina.	

Tabla 3.24: Historia de Usuario 8

Historia de usuario

Número: 8	Usuario: Administrador
Nombre historia: Administración de Roles de Pago	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 4	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de administrar roles de pago de cada una de las empresas. • La información procesada, también podrá ser vista a través de reportes	
Observaciones: Tener en cuenta que en esta historia es muy indispensable ya que es la pantalla principal del módulo de Nómina.	

Tabla 3.25: Tarea 8.1

Tarea	
Número: 8.1	Número de Historia: 8
Nombre tarea: Crear formulario Administración Roles de Pago	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda crear cada nómina, así como también procesar con los datos de rubros previamente cargados por cada uno de los empleados para roles de pago.	

Tabla 3.26: Tarea 8.2

Tarea	
Número: 8.2	Número de Historia: 8
Nombre tarea: Crear base de datos para Administración Roles de Pago	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar el proceso de roles de pago.	

Tabla 3.27: Tarea 8.3

Tarea	
Número: 8.3	Número de Historia: 8
Nombre tarea: Crear reporte Rol individual	
Tipo de tarea: Diseño de reporte	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se diseña el reporte Rol Individual como salida de la información procesada por la interfaz antes realizada.	

Tabla 3.28: Tarea 8.4

Tarea	
Número: 8.4	Número de Historia: 8
Nombre tarea: Crear reporte Rol General	
Tipo de tarea: Diseño de reporte	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se diseña el reporte Rol General como salida de la información procesada por la interfaz antes realizada.	

MÓDULO DE INVENTARIO

Tabla 3.29: Historia de Usuario 9

Historia de usuario	
Número: 9	Usuario: Administrador
Nombre historia: Registro de bodegas	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 2	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar las bodegas para el inventario de las empresas.	
Observaciones:	

Tabla 3.30: Tarea 9.1

Tarea	
Número: 9.1	Número de Historia: 9
Nombre tarea: Crear formulario de mantenimiento de bodega	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda registrar las bodegas utilizadas en inventarios.	

Tabla 3.31: Tarea 9.2

Tarea	
Número: 9.2	Número de Historia: 9
Nombre tarea: Crear base de datos para bodegas	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar de las bodegas del inventario.	

Tabla 3.32: Historia de Usuario 10

Historia de usuario	
Número: 10	Usuario: Administrador
Nombre historia: Registro de productos	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 2	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar los productos para el inventario de las empresas.	
Observaciones: El producto debe de tener una bodega donde almacenar	

Tabla 3.33: Tarea 10.1

Tarea	
Número: 10.1	Número de Historia: 10
Nombre tarea: Crear formulario de mantenimiento de productos	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda registrar los productos del inventario, tomando en cuenta que podrá elegir la bodega donde esta almacenado.	

Tabla 3.34: Tarea 10.2

Tarea	
Número: 10.2	Número de Historia: 10
Nombre tarea: Crear base de datos para productos	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar de los productos del inventario con clave foránea de bodega.	

Tabla 3.35: Historia de Usuario 11

Historia de usuario	
Número: 11	Usuario: Administrador
Nombre historia: Registro de Movimientos de productos	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 3	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar los movimientos por productos para determinar las entradas y salidas con el cálculo de existencia en bodega.	
Observaciones: Cada producto será registrado con el fin de saber el stock.	

Tabla 3.36: Tarea 11.1

Tarea	
Número: 11.1	Número de Historia: 11
Nombre tarea: Crear formulario de movimientos de productos	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda registrar los movimientos por producto para el cálculo de stock.	

Tabla 3.37: Tarea 11.2

Tarea	
Número: 11.2	Número de Historia: 11
Nombre tarea: Crear base de datos para registro de movimiento	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar los movimientos de cada uno de los productos.	

Tabla 3.38: Tarea 11.3

Tarea	
Número: 11.3	Número de Historia: 11
Nombre tarea: Crear reporte Inventario de Productos	
Tipo de tarea: Diseño de reporte	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se diseña el reporte Inventario de Productos como salida de la información procesada por la interfaz antes realizada.	

MÓDULO DE ACTIVOS FIJOS

Tabla 3.39: Historia de Usuario 12

Historia de usuario	
Número: 12	Usuario: Administrador
Nombre historia: Registro de Tipos de Activos Fijos	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 2	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar los tipos de activos fijos para identificar a que activo fijo corresponde o su clasificación con respecto al porcentaje de depreciación.	
Observaciones:	

Tabla 3.40: Tarea 12.1

Tarea	
Número: 12.1	Número de Historia: 12
Nombre tarea: Crear formulario de mantenimiento de activos fijos	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda dar mantenimiento a los diferentes tipos de activos fijos.	

Tabla 3.41: Tarea 12.2

Tarea	
Número: 12.2	Número de Historia: 12
Nombre tarea: Crear base de datos para registro tipos de activos fijos	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar de tipos de activos fijos.	

Tabla 3.42: Historia de Usuario 13

Historia de usuario	
Número: 13	Usuario: Administrador
Nombre historia: Registro de Activos Fijos	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 3	Iteraciones estimadas: 1
Descripción: • El sistema tendrá la capacidad de registrar los activos fijos para el cálculo de depreciación.	
Observaciones:	

Tabla 3.43: Tarea 13.1

Tarea	
Número: 13.1	Número de Historia: 13
Nombre tarea: Crear formulario de mantenimiento de activos fijos	
Tipo de tarea: Desarrollo	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se realiza el desarrollo de un formulario para que el usuario pueda dar mantenimiento a los diferentes activos fijos.	

Tabla 3.44: Tarea 13.2

Tarea	
Número: 13.2	Número de Historia: 13
Nombre tarea: Crear base de datos para registro de activos fijos	
Tipo de tarea: Diseño de base de datos	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador de la base de datos	
Descripción: • Se diseña una base de datos para registrar de activos fijos.	

Tabla 3.45: Tarea 13.3

Tarea	
Número: 13.3	Número de Historia: 13
Nombre tarea: Crear reporte Activos Fijos	
Tipo de tarea: Diseño de reporte	Puntos estimados: 1
Fecha inicio:	Fecha fin:
Responsable: Diseñador del sistema	
Descripción: • Se diseña el reporte Activos Fijos como salida de la información procesada por la interfaz antes realizada.	

3.1.2. Detalle de las historias de usuario.

MÓDULO SISTEMA

Historia de usuario 1: Registrar usuarios nuevos.

Tarea1: Crear interfaz de formulario para el registro

Tarea2: Crear base de datos para usuarios

Historia de usuario2: Permisos de acceso a usuarios

Tarea1: Crear formulario de mantenimiento de permisos

Tarea2: Crear base de datos para permisos de usuarios

Historia de usuario3: Registro del Menú del sistema

Tarea1: Crear formulario de mantenimiento de Menú

Tarea2: Crear base de datos para Menú

Historia de usuario4: Registro de empresas que maneja el sistema

Tarea1: Crear formulario de mantenimiento de empresas

Tarea2: Crear base de datos para empresas del sistema

Tarea3: Diseñar el registro de datos por empresas

MÓDULO NÓMINA

Historia de usuario 5: Registro de personas empleadas en cada empresa

Tarea1: Crear formulario de mantenimiento de personas empleadas de cada empresa

Tarea2: base de datos para empleados de cada empresa del sistema

Historia de usuario 6: Registro de tipo de rubro

Tarea1: Crear formulario de mantenimiento de tipo de rubro

Tarea2: Crear base de datos para tipos de rubros

Historia de usuario 7: Registro de Rubros

Tarea1: Crear formulario de mantenimiento de rubros

Tarea2: Diseño de ubicación de fórmulas de cada rubro

Tarea3: Crear base de datos para rubros

Historia de usuario 8: Administración de Roles de Pago

Tarea1: Crear formulario Administración Roles de Pago

Tarea2: Crear base de datos para Administración Roles

Tarea3: Crear reporte Rol Individual

Tarea4: Crear reporte Rol General

MÓDULO INVENTARIO

Historia de usuario 9: Registro de bodegas

Tarea1: Crear formulario de mantenimiento de bodega

Tarea2: Crear base de datos para bodegas

Historia de usuario 10: Registro de productos

Tarea1: Crear formulario de mantenimiento de productos

Tarea2: Crear base de datos para productos

Historia de usuario 11: Registro de Movimientos de productos

Tarea1: Crear formulario de movimientos de productos

Tarea2: Crear base de datos para registro de movimiento

Tarea3: Crear Reporte Inventario de Productos

MÓDULO ACTIVOS FIJOS

Historia de usuario 12: Registro de Tipos de Activos Fijos

Tarea1: Crear formulario de mantenimiento de activos fijos

Tarea2: Crear base de datos para registro tipos de activos

Historia de usuario 13: Registro de Activos Fijos

Tarea1: Crear formulario de mantenimiento de activos fijos

Tarea2: Crear base de datos para registro de activos fijos

Tarea3: Crear reporte de Activos Fijos

3.1.3. Iteraciones y mini versiones

El proyecto está dividido en cuatro iteraciones, cada una correspondiente a cada módulo del sistema y el orden estará definido como: iteración 1 módulo sistema, iteración 2 módulo nómina, iteración 3 módulo inventario, iteración 4 módulo activos fijos. Este orden tiene que ver con la necesidad del cliente y la necesidad del funcionamiento del sistema ya que sin comenzar con el módulo de sistema, no se podrá configurar ciertos parámetros del sistema y debido al giro de negocio de la empresa, el módulo de nómina es uno de los más importantes para el cliente.

La planeación de cada iteración va a tener un tiempo de alrededor de dos semanas, lo cual permite una buena concordancia con la entrega de toda las iteraciones sin quedar ninguna historia de usuario que postergar según la metodología XP.

En caso de que exista alguna eventualidad, dependería del cliente por el tiempo que también dedica a su trabajo más no del tiempo del personal a cargo del desarrollo del proyecto.

En el transcurso del proyecto, se realizará algunas reuniones con el administrador del sistema, quien es el experto en todo lo que necesita la empresa. Las primeras reuniones serán para definir las funciones del sistema a grandes rasgos, saber hasta qué punto se va a llegar y que es lo que debe perseguir el sistema. Las demás reuniones serán más específicas donde se llevará a cabo en las oficinas de ASESTRATÉGICO CÍA. LTDA. También habrá reuniones donde se expresará sobre la experticia de los usuarios para tener una idea de qué tipo de interfaz se puede implementar en el sistema.

XP recomienda que el orden de las historias definidas en las iteraciones sean determinadas por el cliente, sin embargo, para definir el orden se tomará en cuenta la disponibilidad del hardware, la disponibilidad de los usuarios y la dificultad de la implementación mezclado con la prioridad que cada una representa.

Para acortar el tiempo de entrega se tuvo en cuenta los aspectos explicados a continuación:

- XP recomienda que el tiempo de desarrollo durante una semana sea de 40 horas semanales distribuidas de lunes a viernes sin embargo, se considero que de acuerdo a que los usuarios responsables de cada área, cambiar este aspecto a trabajar de lunes a domingo, balanceando con más horas de lunes a viernes. Esto fue dinámico puesto a que no se tenía un horario específico para cada día y se trabaja en función del tiempo del usuario.
- Las reuniones se harían a partir de las 5 de la tarde; esto fue considerado por el cliente, debido a que en esa hora no existía mucho flujo de trabajo.
- Las reuniones con el cliente para entrega y definición de nuevas historias de usuario, se las realizaba los días domingos, al finalizar la jornada semanal.

La siguiente tabla, muestra las fechas de cada miniversión de acuerdo a la iteración e historia de usuario.

Tabla 3.46: Iteraciones Vs Historias de Usuario

Iteraciones	**Nº**	**Historias**	**Inicio**	**Fin**	**Observaciones**
Primera	1	Registrar usuarios nuevos.	02/07/12	15/07/12	

	2	Permisos de acceso a usuarios	05/08/12	08/08/12	
	3	Registro de Menú del sistema	08/08/12	11/08/12	
	4	Registro de empresas que maneja el sistema	11/08/12	13/08/12	
Segunda	5	Registro de personas empleadas en cada empresa	13/08/12	15/08/12	
	6	Registro de tipo de rubro	15/08/12	18/08/12	
	7	Registro de Rubros	18/08/12	24/08/12	
	8	Administración de Roles de Pago	01/09/12	08/09/12	
Tercera	9	Registro de bodegas	09/09/12	14/09/12	
	10	Registro de productos	15/09/12	25/09/12	
	11	Registro de Movimientos de productos	01/10/12	09/10/12	
Cuarta	12	Registro de Tipos de Activos Fijos	10/10/12	28/10/12	
	13	Registro de Activos Fijos	01/11/12	18/11/12	

3.1.4. Velocidad del Proyecto

El número de historias de usuario realizadas por iteración fue la medida correcta para la planeación de entregas de versiones y la duración de las iteraciones, ya que se contaba con puntos asignados a las historias de usuario de acuerdo a la dificultad de la implementación de la historia.

Durante la estimación de los puntos por cada historia de usuario y el orden de entrega en las iteraciones se tuvo en cuenta la puntuación del riesgo en el desarrollo y la prioridad en el negocio.

El riesgo causado por cambios en requerimientos se lo calificó de la siguiente manera:

- BAJO.- Corresponde a cambios que modifican a los puntos en un máximo de 8 horas.
- MEDIO.- Corresponde a cambios de modificación a los puntos en un tiempo mínimo de 8 horas y máximo de 16 horas.
- ALTO.- Corresponde a cambios o procesos mayores a 16 horas.

Para el orden de entrega, se tiene otra clasificación de acuerdo a la prioridad de las historias de usuarios:

- ALTA.- Corresponde a las funciones diarias del sistema.
- MEDIA.- Corresponde a los procesos administrativos periódicos del sistema.
- BAJA.- Corresponde a procesos que no afectan o que no tienen gran impacto en las prioridades de alta ni de media.

La tabla quedaría de la siguiente forma de acuerdo a las historias de usuario y a la clasificación de riesgo y prioridad antes mencionada:

Tabla 3.47: Iteraciones Vs Tiempo

	Iteración 1	**Iteración 2**	**Iteración 3**	**Iteración 4**
Horas	46	41	42	30
Semanas	2	2	2	1
Horas Semanales	23	20,5	21	30
Historias de usuario(Velocidad del proyecto)	5	6	4	6

3.1.5. Ajustes

Si después de la entrega de cada miniversiones, surgen pequeños ajustes, estos serán cubiertos de inmediato para poder continuar con la realización del proyecto previo a una reunión del equipo para verificar que sea factible o no aquellos ajustes.

3.1.6. Estimación de esfuerzo

Para este proyecto se ha planificado cuatro iteraciones en las cuales se entrega diferentes versiones del programa de acuerdo a lo entregado por el cliente.

La cantidad y orden en las iteraciones de las historias de usuario son las siguientes:

Módulo Sistema

Tabla 3.48: Estimación de Esfuerzo Módulo Sistema

Historias de Usuario	**Pts.**

Registro de usuarios nuevos	2
Permisos de acceso a usuarios	2
Registro del menú del sistema	2
Registro de empresa que maneja el sistema	3

Módulo Nómina

Tabla 3.49: Estimación de Esfuerzo Módulo Nómina

Historias de Usuario	**Pts.**
Registro de personas empleadas en cada empresa	2
Registro de tipo de rubro	2
Registro de rubro	3
Administración de Roles de Pago	4

Módulo Inventario

Tabla 3.50: Estimación de Esfuerzo Módulo Inventario

Historias de Usuario	Pts.
Registro de Bodega	2
Registro de Producto	2
Registro de Movimientos de Productos	3

Módulo Activos Fijos

Tabla 3.51: Estimación de Esfuerzo Módulo Activos Fijos

Historias de Usuario	Pts.
Registro de Tipos de Activos Fijos	2
Registro de Activos Fijos	3

3.2. Diseño.

3.2.1. Diseño simple de miniversiones

Según XP el tiempo invertido en el proyecto con respecto a diagramas, es solo el necesario, dando paso al diseño de diagramas del sistema totalmente sencillos, es por eso que se implementa las Tarjetas CRC, algunos diagramas como Arquitectura del Sistema, Diagrama Lógico de Base de Datos y Diagrama Físico de Base de Datos.

3.2.1.1. Descripción de Escenarios y Tarjetas CRC.

Escenarios del sistema.

Módulo Sistema

Tabla 3.52: Escenario Ingresar al Sistema

Título: Ingresar al Sistema	Objetivo: Permitir que el usuario pueda acceder al sistema como usuario
Utiliza a:	

Extiende a:
Descripción: Para ingresar al sistema, se debe de loguear con su nombre de usuario y contraseña, elegir la empresa en la que se va a trabajar.
Precondiciones: Usuario registrado
Poscondiciones:
Actores:
Episodios: 1.- El usuario ingresa al sistema. 2.- El sistema muestra la página principal del sistema. 3.- El usuario da clic en la imagen de logueo. 4.- Se despliega la pantalla de ingreso de usuario y contraseña. 5.- El usuario ingresa el usuario, contraseña y elije la empresa en la que va a trabajar. 6.- El sistema verifica que los datos sean correctos. 7.- El sistema responde ingresando a la página con menú cargando a todas las pantallas que tiene acceso el usuario ingresado

Tabla 3.53: Escenario Mantenimiento Usuario

Título: Mantenimiento de usuarios	Objetivo: Permitir que el usuario pueda crear usuarios
Utiliza a:	
Extiende a:	
Descripción: Para ingresar un nuevo usuario se debe de llenar los datos necesarios del usuario y se necesita que el usuario debe de tener permiso a la pantalla.	
Precondiciones: Permiso a la pantalla por parte del usuario	
Poscondiciones: Llenar todos los campos necesarios	
Actores:	
Episodios: 1.- El usuario ingresa al menú principal ingresando al módulo sistema y luego a usuarios. 2.- Se despliega la pantalla de usuario. 3.- El usuario ingresa todos los datos necesarios del sistema. 4.- El usuario pulsa el botón guardar para salvar los datos.	

Tabla 3.54: Escenario Mantenimiento de Acceso para los Usuarios

Título: Mantenimiento de accesos para los	Objetivo: Permitir que el usuario pueda

usuarios	dar permiso a cada una de las opciones del sistema.
Utiliza a:	
Extiende a:	
Descripción: Para que el sistema cargue las opciones que tiene el sistema según el usuario, se debe de dar permiso al usuario.	
Precondiciones: Usuario registrado, Menú creado	
Poscondiciones:	
Actores:	
Episodios: 1.-El usuario ingresa a la pantalla mediante el módulo sistema y haciendo clic en permisos. 2.- Se despliega la pantalla de permisos con su información. 3.- El usuario hace clic en insertar. 4.- Se despliega la pantalla de inserción de permisos según el usuario. 5.- El usuario ingresa los datos necesarios eligiendo el usuario y la pantalla que quiere que obtenga. 6.- El usuario pulsa el botón guardar para salvar los datos.	

Tabla 3.55: Escenario Mantenimiento de Menú

Título: Mantenimiento de Menú	Objetivo: Permitir que el usuario pueda modificar y crear el menú desplegable del sistema.
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar cada uno de los menús del sistema, para mayor comodidad de identificación de cada uno de ellos.	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en sistema y luego en menú. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en insertar o el ícono de editar. 4.- Se habilita la opción de modificar o insertar del sistema. 5.- El usuario pulsa el botón guardar para salvar los datos ingresados o modificados.	

Tabla 3.56: Escenario Mantenimiento de Empresas

Título: Mantenimiento de Empresas	Objetivo: Permitir que el usuario pueda modificar y crear empresas clientes al sistema
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar cada una de las empresas clientes del sistema.	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa al menú haciendo clic en el menú sistema y luego en empresas. 2.- Se despliega la pantalla de mantenimiento de empresas. 3.- El usuario pulsa en el botón ingresar o en el icono de modificar. 4.- Se ingresa todos los datos pertinentes. 5.- El usuario pulsa en guarda.	

Módulo Nómina.

Tabla 3.57: Escenario Mantenimiento de Personas Empleadas en la Empresa

Título: Mantenimiento de personas empleadas en la empresa	Objetivo: Permitir que el usuario pueda modificar y crear las personas empleadas en la empresa.
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar o crear cada uno de las personas empleadas por las empresa.	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en nómina y luego en persona. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en insertar o el ícono de editar. 4.- Se habilita la opción de modificar o insertar del sistema. 5.- El usuario pulsa el botón guardar para salvar los datos ingresados o modificados.	

Tabla 3.58: Escenario Mantenimiento de Tipo De Rubro

Título: Mantenimiento de tipo de rubro	Objetivo: Permitir que el usuario pueda modificar y crear el tipo de rubro de la nómina
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar el tipo de rubro, para mayor funcionalidad	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en nómina y luego en tipo de rubro. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en insertar o el ícono de editar. 4.- Se habilita la opción de modificar o insertar del sistema. 5.- El usuario pulsa el botón guardar para salvar los datos ingresados o modificados.	

Tabla 3.59: Escenario Mantenimiento de Rubro

Título: Mantenimiento de rubro	Objetivo: Permitir que el usuario pueda modificar y crear rubros de la nómina
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar el tipo de rubro, para mayor funcionalidad	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en nómina y luego en rubro. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en insertar o el ícono de editar. 4.- Se habilita la opción de modificar o insertar del sistema. 5.- El usuario pulsa el botón guardar para salvar los datos ingresados o modificados.	

Tabla 3.60: Escenario Roles de Pago

Título: Administración de Roles de Pago	Objetivo: Permitir que el usuario pueda

	realizar los roles de pago.
Utiliza a:	
Extiende a:	
Descripción: El usuario puede crear la nómina de cada periodo cargando los empleados y sus rubros para después procesar para el cálculo de cada uno de ellos.	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en nómina y luego en roles de pago. 2.- Se despliega la pantalla de menú con la información antes ingresada. 3.- El usuario da clic en Nuevo. 4.- El usuario ingresa las fechas correspondientes a la Nómina. 5.- El usuario da clic en guardar. 6.- Luego pulsa el botón cargar y procede a visualizar los empleados y sus rubros. 7.- El usuario da clic en procesar para actualizar cada uno de los valores de los rubros por empleado. 8.- El usuario da clic en Rol individual y se despliega el rol individual del empleado que esta seleccionado en ese momento. 9.- El usuario da clic en Rol general y se despliega el rol general de la nómina que esta seleccionado en ese momento.	

Módulo Inventario

Tabla 3.61: Escenario Mantenimiento de Bodega

Título: Mantenimiento de bodega	Objetivo: Permitir que el usuario pueda modificar y crear bodegas para los productos
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar o crear las bodegas, para mayor funcionalidad	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en inventario y luego en bodega. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en insertar o el ícono de editar. 4.- Se habilita la opción de modificar o insertar del sistema. 5.- El usuario pulsa el botón guardar para salvar los datos ingresados o modificados.	

Tabla 3.62: Escenario Mantenimiento de Productos

Título: Mantenimiento de productos	Objetivo: Permitir que el usuario pueda modificar y crear los productos
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar los productos, para mayor funcionalidad.	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en Inventario y luego en producto. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en insertar o el ícono de editar. 4.- Se habilita la opción de modificar o insertar del sistema. 5.- El usuario pulsa el botón guardar para salvar los datos ingresados o modificados.	

Tabla 3.63: Escenario Movimiento de Productos

Título: Registro de Movimientos de productos	Objetivo: Permitir que el usuario pueda registrar cada uno de los movimientos por producto.
Utiliza a:	
Extiende a:	
Descripción: El usuario registrar los movimientos por producto.	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en Inventario y luego en movimiento. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en nuevo producto. 4.- El usuario ingresa la información relacionada al producto. 5.- El usuario da clic en el botón guardar. 6.- El usuario da clic en nuevo detalle. 7.- El usuario ingresa la información ligada al registro del movimiento del producto ya sea de ingreso o egreso de mercadería. 8.- El usuario da clic en guardar. 9.- El usuario da clic en recalcular existencia para saber cuánta mercadería hay de ese	

producto. 10.- El usuario da clic en Inventario y se despliega el inventario de todos los productos.

ESPACIO EN BLANCO INTENCIONAL

Módulo Activos Fijos

Tabla 3.64: Escenario Mantenimiento de Tipos De Activos Fijos

Título: Mantenimiento de tipos de activos fijos	Objetivo: Permitir que el usuario pueda modificar y crear los tipos de activos fijos
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar los tipos de activos fijos, para mayor funcionalidad.	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en Activos Fijos y luego en Tipos de Activos Fijos. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en insertar o el ícono de editar. 4.- Se habilita la opción de modificar o insertar del sistema. 5.- El usuario pulsa el botón guardar para salvar los datos ingresados o modificados.	

Tabla 3.65: Escenario Mantenimiento de Artículos de Activos Fijos

Título: Artículos de Activos Fijos	Objetivo: Permitir que el usuario pueda modificar y crear los artículos de activos fijos
Utiliza a:	
Extiende a:	
Descripción: El usuario puede modificar los artículos de activos fijos, para mayor funcionalidad.	
Precondiciones: Permiso a esta opción	
Poscondiciones:	
Actores:	
Episodios: 1.- El usuario ingresa a esta opción haciendo clic en Activos Fijos y luego en Artículos de Activos Fijos. 2.- Se despliega la pantalla de menús con la información antes ingresada. 3.- El usuario da clic en insertar o el ícono de editar. 4.- Se habilita la opción de modificar o insertar del sistema. 5.- El usuario pulsa el botón guardar para salvar los datos ingresados o modificados. 6.- El usuario da clic en Activos Fijos y se despliega el reporte con el cálculo de la depreciación de los artículos de activos fijos.	

Tarjetas CRC.

Tarjeta CRC ManejoUsuario

Tabla 3.66: Tarjeta Crc Manejousuario

Nombre de la clase: ManejoUsuario	
Responsabilidad Cargar las cuentas de usuario Ingresa una cuenta de usuario Actualizar la cuenta de usuario	**Colaboración** GET_USUARIO , Listar a los usuarios POST_USUARIO, Actualizar e Ingresar nuevo usuario

Tarjeta CRC ManejoPermisos

Tabla 3.67: Tarjeta Crc Manejopermisos

Nombre de la clase: ManejoPermisos	
Responsabilidad	**Colaboración**
Cargar los Permisos	GET_PERMISOS , Listar a los Permisos
	GET_MENU, Listar menú
Seleccione el menú	GET_USUARIO, Listar usuario
Seleccione el usuario	POST_PERMISOS, Actualizar e Ingresar
Ingreso de permisos	nuevo permisos
Actualizar los permisos	

Tarjeta CRC ManejoMenu

Tabla 3.68: Tarjeta Crc Manejomenú

Nombre de la clase: ManejoMenu	
Responsabilidad	**Colaboración**
Cargar el Menú	GET_MENU , Listar el menú
Cargar el Menú	GET_MENU , Listar el menú padre
Ingreso de menú	POST_MENU, Actualizar e Ingresar
Actualizar menú	nuevo menú.

Tarjeta CRC ManejoEmpresa

Tabla 3.69: Tarjeta Crc Manejoempresa

Nombre de la clase: ManejoEmpresa	
Responsabilidad Cargar el Empresa Ingreso de menú Actualizar menú	**Colaboración** GET_EMPRESA , Listar el menú POST_EMPRESA, Actualizar e Ingresar nueva empresa.

Tarjeta CRC ManejoPersona

Tabla 3.70: Tarjeta Crc Manejopersona

Nombre de la clase: ManejoPersona	
Responsabilidad Cargar persona Ingreso de persona Actualizar menú	**Colaboración** GET_PERSONA , Listar las personas. POST_PERSONA, Actualizar e Ingresar nueva persona.

Tarjeta CRC ManejoTipoRubro

Tabla 3.71: Tarjeta Crc Manejotiporubro

Nombre de la clase: ManejoTipoRubro	
Responsabilidad	**Colaboración**
Cargar tipo rubro	GET_TIPORUBRO, Listar los tipos de rubro.
Ingreso de tipo rubro	POST_TIPORUBRO, Actualizar e Ingresar los tipos de rubro.
Actualizar tipo rubro	

Tarjeta CRC ManejoRubro

Tabla 3.72: Tarjeta Crc Manejorubro

Nombre de la clase: ManejoTipoRubro	
Responsabilidad	**Colaboración**
Cargar tipo rubro	GET_TIPORUBRO, Listar los tipos de rubro.
Cargar rubro	GET_RUBRO, Listar los rubros.
Ingreso de tipo rubro	POST_RUBRO, Actualizar e Ingresar los tipos de rubro.
Actualizar tipo rubro	

ESPACIO EN BLANCO INTENCIONAL

Tarjeta CRC ManejoRolPagos

Tabla 3.73: Tarjeta Crc Manejorolpago

Nombre de la clase: ManejoRolPagos	

Responsabilidad	Colaboración
Cargar nomina	GET_NOMINAS, Listar las nóminas creadas.
Cargar empleado	GET_EMPLEADOSXNOMINA, Listar los empleados por nómina.
Cargar rubros	GET_RUBROSXEMPLEADO, Listar los rubros por cada uno de los empleados.
Ingresar nomina	POST_NOMINA, Ingresar las nóminas.
	POST_EMPLEADONOM, Ingresar los empleados de la nómina.
Ingresar empleados	POST_RUBROEMP, Ingresar los rubros por cada empleado.
Ingresar rubros	

Tarjeta CRC ManejoBodega

Tabla 3.74: Tarjeta Crc Manejobodega

Nombre de la clase: ManejoBodega	
Responsabilidad	**Colaboración**
Cargar bodega	GET_BODEGA, Listar las bodegas.
Ingreso de bodega	POST_BODEGA, Actualizar e Ingresar
Actualizar bodega	las bodegas.

Tarjeta CRC ManejoProducto

Tabla 3.75: Tarjeta Crc Manejoproducto

Nombre de la clase: ManejoProducto	
Responsabilidad	**Colaboración**
Cargar producto	GET_PRODUCTO, Listar los producto.
	GET_BODEGA, Listar las bodegas.
Cargar bodega	POST_PRODUCTO, Actualizar e
Ingreso de producto	Ingresar los productos.
Actualizar producto	

Tarjeta CRC ManejoMovimiento

Tabla 3.76: Tarjeta Crc Manejomovimiento

Nombre de la clase: ManejoMovimiento	
Responsabilidad	**Colaboración**
Insertar producto	POST_NUPROMOVI, Inserta un nuevo producto del movimiento.
Insertar detalle	POST_DETALLMOVI, Inserta un nuevo detalle de movimiento.
Actualiza stock	POST_RECALEXIS, Actualiza el stock existente de cada producto.
Actualiza detalle	POST_EDIT, Actualiza el detalle que se ha modificado.
Cargar producto	GET_PRODUCTO, Listar los producto.
Cargar detalle	GET_DETALLE, Lista los detalles de movimiento.

Tarjeta CRC ManejoTipoActivoFijo

Tabla 3.77: Tarjeta Crc Manejotipoactivo

Nombre de la clase: ManejoTipoActivoFijo	
Responsabilidad Cargar Tipo Activo Ingreso de tipo de activo Actualizar tipo de activo	**Colaboración** GET_TIPOACTIVO, Listar las tipo activo. POST_TIPOACTIVO, Actualizar e Ingresar el tipo de activo.

Tarjeta CRC ManejoArticuloActi

Tabla 3.78: Tarjeta Crc Manejoproducto

Nombre de la clase: ManejoArticuloActi	
Responsabilidad Cargar artículo Cargar tipo de artículo Ingreso de producto Actualizar producto	**Colaboración** GET_ARTICULO, Listar los producto. GET_TIPOARTICULO, Listar los tipos de artículos. POST_ARTICULO, Actualizar e Ingresar los artículos.

3.3. Estándares del sistema

3.3.1. Nomenclatura para objetos visuales

La utilización de nomenclaturas con estándares, permite reconocer desde cualquier ámbito a cada objeto que se crean, en la siguiente tabla se muestra el prefijo, tipo de objeto y un ejemplo de cómo nombrar cada uno de los objetos visuales:

Tabla 3.79: Nomenclatura Para Objetos Visuales

Prefijo	**Tipo de objeto**	**Ejemplo**
Ani	AnimationBotton	AniMailBox
Bed	PenBedit	BedFirstName
Cbo	Combo box, Drop box	CboEnglish
Chk	Checkbox	ChkReadOnly
Clp	Picture clip	ClpToolBar
Cmd	Commandbutton	cmdOk
Dat	DataControl	DatBiblio
Dlg	CommanDialogControl	DlgFileOpen
Frm	Form	frmProducto
Gra	Graph	graRevenue
Img	Image	imgIcon
Lbl	Label	LblName
Lst	Listbox	lstMothers
Mnu	Menu	MnuPrincipal
Opt	OptionButton	OptCargas
Pic	Picture	PicVGA
Tmr	Timer	tmrAlarm
Vsb	Vertical Scroll bar	vsbSSS

Txt	TextBox	txtLastName

3.3.2. Estándares de Diseño Web

Los estándares web se enfocan en la construcción de sitios en los cuales exista la presentación de la información que el creador del sitio quiera mostrar al usuario final y que el sitio web pueda tener la velocidad de descarga necesaria para que el usuario pueda visualizar la página con rapidez.

En este caso, se utiliza una librería de componentes llamada Primefaces, que dentro de su código implementa estos estándares para que el desarrollador no se preocupe por cumplirlos y pueda invocar cada uno de los objetos visuales, sin ningún tipo de problemas de esta índole.

En el aspecto visual mantiene una línea muy estricta, difícil de cambiar, a la hora de querer modificar la parte visual de cada uno de sus componentes, debido a la simplicidad de código a la hora de invocarlos.

De acuerdo a los requerimientos del cliente, PrimeFaces, cuenta con muchos componentes visual totalmente listos para usar y fáciles de incorporar al código, sin mencionar que tiene una galería de Themes que ayuda a enfatizar el entorno visual. Todas estas ventajas son únicamente para brindar al usuario final una buena interfaz muy amigable.

En lo que a estándar web se refiere, se puede mencionar a los CSS que ayudan a mejor la parte visual pero ellos internamente ya están embebidos por PrimeFaces.

3.3.3. Estándares de programación sobre java

Detrás de JSF, en los Baking Bean se tiene código escrito en java como es habitual, donde se puede mencionar algunos estándares a seguir en la construcción del proyecto.

A continuación, se muestra la lista de algunos estándares implementados:

- Todo paquete inicia con mayúscula.
- Toda clase inicia con mayúscula.
- Todo método inicia con minúscula.
- Toda variable inicia con minúscula.
- Sea, clase, método o variable, se escribe con notación camel independiente de cuál sea la forma en que inicie.
- Todo el código debe estar correctamente identado.
    - Se abre una llave en la misma línea donde se escribió alguna definición que va a tomar forma con la llave.
    - Se cierra la llave en la siguiente línea.
    - Debe estar en la misma columna de donde está la definición.
    - No se escribe nada más en la línea donde se cierra, salvo el caso de que sea un ifelse, donde en el else se cierra el if , se escribe else y se abre la llave.
- Solo se declaran variables al inicio de una clase, método o ciclo.
- Se debe comprimir las versiones que se van generando con el siguiente formato.

- ddMesaaaa-servidor.zip.

- El nombre de las tablas es en singular y minúscula con el siguiente formato:
  - gru_nomtabla.
  - Donde gru, es referente a las tres primeras letras del nombre del grupo o módulo y el nombre de la tabla.
- Las entidades que manejan las tablas, estarán compuestas de la misma forma que las tablas con la diferencia de que están escritas con notación camel y sin guión bajo comenzando con mayúscula.
- Todos los atributos de una clase están encapsulados. Son privados y su acceso se da por medio de los métodos setAtributo() y/o getAtributo.

3.4. Diagramas de la base de datos

3.4.1. Diagrama Físico

Ver Anexo A.

3.4.2. Diagrama Lógico

Ver Anexo B

ESPACIO EN BLANCO INTENCIONAL

3.4.3. Diccionario de datos.

Tabla 3.80: Nomenclatura Para Objetos Visuales

Tabla	Campo de la columna	Tipo de dato	Comentario
gen_persona	gper_codigo	varchar(12)	Código de la persona
gen_persona	gper_nombre	varchar(100)	Nombre de la persona
gen_persona	gper_direccion	varchar(300)	Dirección de la persona
gen_persona	gper_telefono_1	varchar(15)	Teléfono 1 de la persona
gen_persona	gper_telefono_2	varchar(15)	Teléfono 2 de la persona
gen_persona	gper_movil	varchar(15)	Teléfono móvil de la persona
gen_persona	gper_identificacion	varchar(15)	Identificación de la persona
gen_persona	susu_codigo_ua	varchar(12)	Código de usuario de ingreso de nueva persona
gen_persona	gper_fecha_ua	date	Fecha de actualización del registro
gen_persona	susu_codigo_um	varchar(12)	Código de usuario de modificación
gen_persona	gper_fecha_um	date	Fecha de modificación del registro
gen_persona	gper_mo_oc	varchar(10)	Mostrar u Ocultar
gen_persona	gper_img_grand	longblob	Imagen grande de la persona
gen_persona	gper_img_peq	longblob	Imagen pequeña de la persona
gen_persona	gper_img_recar	varchar(3)	Imagen recarga
gen_persona	gper_apellido	varchar(100)	Apellido de la persona
gen_persona	gper_cargo	varchar(100)	Cargo que ocupa la persona
gen_persona	gper_sueldo	decimal(10,2)	Sueldo que tiene la persona
gen_persona	gper_fech_ingre	date	Fecha de ingreso de la persona
gen_persona	gper_referenci	varchar(200)	Referencia de la persona
gen_persona	gper_fech_nacimi	date	Fecha de nacimiento de la persona
gen_persona	gper_carg_famili	int(10)	Número de cargas familiares
inv_bodega	ibod_cod	varchar(12)	Código de bodega
inv_bodega	ibod_nomb	varchar(100)	Nombre de bodega
inv_bodega	ibod_desc	varchar(200)	Descripción de bodega

inv_bodega	susu_cod_ua	varchar(12)	Código de usuario quien agrego el registro
inv_bodega	ibod_fech_a	date	Fecha de añadidura de bodega
inv_bodega	susu_cod_um	varchar(12)	Código de usuario quien modifico el registro
inv_bodega	ibod_fech_m	date	Fecha de modificación de registro
inv_bodega	ibod_mo_oc	varchar(10)	Mostrar u Ocultar
inv_producto	ipro_cod	varchar(12)	Código de bodega
inv_producto	ipro_cod_us	varchar(30)	Código interno que el usuario ingresa
inv_producto	ibod_cod	varchar(12)	Código de bodega correspondiente al producto
inv_producto	ipro_desc	varchar(100)	Descripción adicional del producto
inv_producto	ipro_p_prim	double(20,4)	Precio primario del producto
inv_producto	ipro_u_com	double(20,4)	Unidad de compra del producto
inv_producto	ipro_iva	varchar(5)	Iva del producto ingresado
inv_producto	ipro_p_iva	double(20,4)	Porcentaje del Iva del producto
inv_producto	ipro_mo_oc	varchar(10)	Mostrar u Ocultar el registro
inv_producto	susu_cod_ua	varchar(12)	Usuario quien agrego el producto
inv_producto	ipro_fech_a	date	Fecha de ingreso del registro
inv_producto	susu_cod_um	varchar(12)	Código de usuario quien modifico el registro
inv_producto	ipro_fech_m	date	Fecha de modificación del registro
nom_rubros	nrub_cod	varchar(12)	Código de rubro
nom_rubros	nrub_concep	varchar(100)	Nombre o concepto del rubro
nom_rubros	nrub_formu	varchar(300)	Fórmula del rubro para el cálculo de la nómina
nom_rubros	ntir_cod_tip	varchar(12)	Código del tipo de rubro
nom_rubros	nrub_cod_pro	varchar(100)	Código pro
nom_rubros	susu_cod_ua	varchar(12)	Código de usuario quien agrego el registro
nom_rubros	nrub_fech_ua	date	Fecha de ingreso de registro
nom_rubros	susu_cod_um	varchar(12)	Código de usuario quien modifico el registro
nom_rubros	nrub_fech_um	date	Fecha de modificación de registro

nom_rubros	nrub_mo_oc	varchar(50)	Mostrar u Ocultar registro
nom_tip_ru	ntir_cod	varchar(12)	Código de tipo de rubro
nom_tip_ru	ntir_nomb	varchar(100)	Nombre del tipo de rubro
nom_tip_ru	ntir_desc	varchar(200)	Descripción del tipo de rubro
nom_tip_ru	susu_cod_ua	varchar(12)	Código de usuario quien agregó el registro
nom_tip_ru	ntir_fech_ua	date	Fecha de ingreso de registro
nom_tip_ru	susu_cod_um	varchar(12)	Código de usuario quien modificó el registro
nom_tip_ru	ntir_fech_um	date	Fecha de modificación de registro
nom_tip_ru	ntir_mo_oc	varchar(50)	Mostrar u Ocultar
nom_tip_ru	ntir_signo	varchar(5)	Signo del tipo del rubro
sis_empresa	sem_cod	varchar(12)	Código de empresa
sis_empresa	sem_nomb	varchar(100)	Nombre de empresa
sis_empresa	sem_desc	varchar(200)	Descripción adicional para la empresa
sis_empresa	sem_direcc	varchar(150)	Dirección de la empresa
sis_empresa	sem_telf	varchar(50)	Teléfono de la empresa
sis_empresa	sem_represen	varchar(100)	Representante legal de la empresa
sis_empresa	sem_cod_inter	varchar(5)	Código interno de la empresa
sis_empresa	gper_contact	varchar(12)	Código de la persona de contacto en la empresa
sis_empresa	susu_cod_ua	varchar(12)	Código de usuario quien agregó el registro
sis_empresa	sem_fech_a	date	Fecha de ingreso del registro
sis_empresa	susu_cod_um	varchar(12)	Código del usuario quien modificó el registro
sis_empresa	sem_fech_um	date	Fecha de modificación del registro
sis_empresa	sem_mo_oc	varchar(10)	Mostrar u Ocultar
sis_menu	smen_cod	varchar(12)	Código del menú
sis_menu	smen_nomb	varchar(100)	Nombre del menú
sis_menu	smen_desc	varchar(200)	Descripción del menú
sis_menu	smen_cod_men	varchar(12)	Código del padre menú
\sis_menu	susu_cod_ua	varchar(12)	Código de usuario quien agregó el registro
sis_menu	smen_fech_a	date	Fecha en que se agregó el registro
sis_menu	susu_cod_um	varchar(12)	Usuario quien modificó el registro
sis_menu	smen_fech_um	date	Fecha en que se modificó el registro
sis_menu	smen_accion	varchar(100)	Código de programación que se va a ejecutar cuando de clic
sis_menu	smen_mo_oc	varchar(10)	Mostrar u Ocultar
sis_permisos	sper_cod	varchar(12)	Código permisos

sis_permisos	susu_cod_per	varchar(12)	Código de usuario a la que se le asigna el permiso
sis_permisos	smen_cod	varchar(12)	Código de menú para los permisos
sis_permisos	sper_permi	varchar(15)	Permite o deniega
sis_permisos	susu_cod_ua	varchar(12)	Código de usuario quien agregó el registro
sis_permisos	sper_fech_ua	date	Fecha en que se agregó el registro
sis_permisos	susu_cod_um	varchar(12)	Código de usuario quien modificó el registro
sis_permisos	sper_fech_um	date	Fecha donde se modificó el registro
sis_permisos	sper_mo_oc	varchar(10)	Mostrar u Ocultar
sis_usuario	susu_codigo	varchar(12)	Código del usuario
sis_usuario	gper_codigo	varchar(12)	Código de la persona asignada a este usuario
sis_usuario	susu_nusuario	varchar(30)	Nombre de usuario
sis_usuario	susu_contrasenia	varchar(30)	Contraseña
sis_usuario	susu_codigo_ua	varchar(12)	Código de usuario
sis_usuario	susu_fecha_a	date	Fecha en que se agregó el registro
sis_usuario	susu_codigo_um	varchar(12)	Código en el que se modificó el usuario
sis_usuario	susu_fecha_um	date	Fecha en que se modificó el usuario
sis_usuario	susu_mo_oc	varchar(10)	Mostrar u Ocultar

3.5. Arquitectura Implementada

La figura 3.1 muestra los elementos empleados en el transcurso del sistema, desde una vista macro como son el servidor y base de datos hasta cosas pequeñas como representaciones básicas y métodos más importantes.

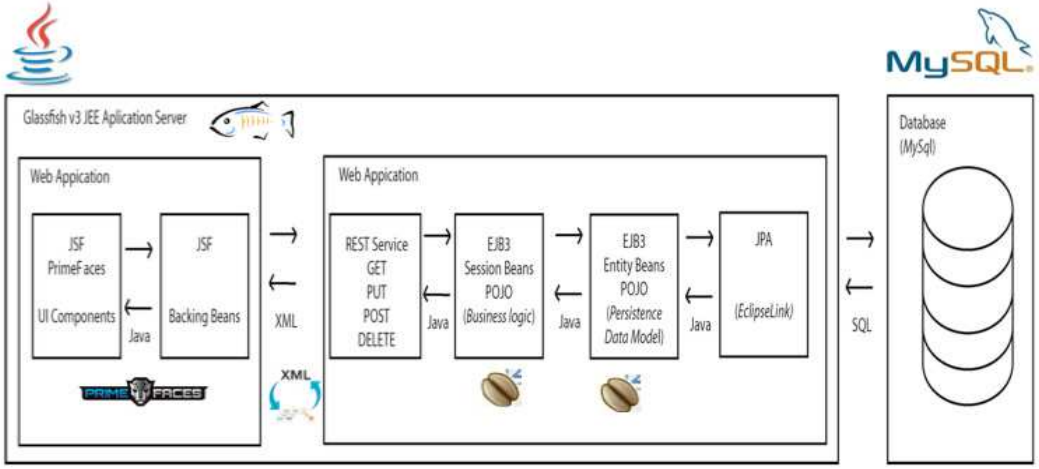


Figura 3.1: Arquitectura del sistema

CAPÍTULO 4

4. CONSTRUCIÓN DEL SISTEMA.

4.1. Etapas en la construcción de los módulos.

4.1.1. Presencia del cliente.

Según la metodología XP, el cliente debe de estar en el sitio de trabajo para la construcción óptima del sistema y aclarar dudas cara a cara que el desarrollador tenga, pero para este proyecto, fue complicado aplicar esta etapa en la construcción del sistema, ya que el cliente tenía que realizar su trabajo habitual de todos los días y el desarrollador no podía dejar su sitio de trabajo para ir donde el cliente y realizar preguntas, se debería pagar las horas de trabajo al cliente para que esta etapa se pueda cumplir de una forma estricta, cosa que no se pudo realizar esa estrategia; en lugar de aquello, se suplanto la idea de que tanto el programador como el cliente estén cara a cara por colocar un teléfono en el sitio de trabajo del cliente y del programador y así poder comunicarse por teléfono cuando el desarrollador necesitaba aclarar alguna duda que tuviera.

Es significativo aclarar que, es de suma importancia que el cliente se encuentre con el programador, para que pueda aclarar dudas sobre el desarrollo de la aplicación y es esto que esta metodología es fuerte en este aspecto, pero se pudo sustituir con la comunicación vía telefónica y con reuniones pre programadas con el cliente, donde se aprovecharán todas las dudas que tenga y de esta forma no perder mucho tiempo al cliente.

4.1.2. Programación en pareja.

Según la metodología XP, toda la producción de código debe de ser en parejas, sentadas en frente de un solo monitor para tener un diseño de mejor calidad y un código

más organizado, teniendo en cuenta que los problemas que se presentan se pueden solucionar más fácilmente y rápidamente.

Debido a que el desarrollo de este proyecto, es individual, este objetivo fue muy difícil de aplicar; lo que se tuvo que implementar, fue que el cliente en algunos casos, ayudó a colaborar con el programador a realizar la programación en pareja en sus tiempos libres. Esto se pudo llevar a cabo, porque el cliente tenía un poco de conocimiento de programación y sistemas, sin mencionar que su carrera profesional, no tiene nada que ver con eso, pero con la gran experiencia que contaba, se pudo suplir este inconveniente.

En la primera iteración, se realizó el apoyo del cliente como segundo programador en el desarrollo, pero debido a que no contaba con mucho tiempo y con lo que se realizó un balance del tiempo de desarrollo cuando se contaba con esa ayuda extra; se concluyó que no era tan indispensable de su ayuda, pues esto fue sustituido con la ayuda externa de programadores que tienen experiencia sobre la tecnología implementada en este proyecto, citando a cada uno de ellos por medio de sistemas de mensajerías electrónica para dialogar de temas como problemas suscitados y su resolución.

Desde la segunda iteración se vio el cambio en el tiempo de desarrollo, cubriendo con esta estrategia, los principales objetivos, como son, rapidez, fácil resolución de problemas con un diseño de mejor calidad y un código más organizado.

4.1.3. Integración secuencial.

Para aplicar este etapa de XP, se debe de saber que antes de aplicar nuevo código al proyecto, se debe garantizar que la última versión halla pasado todas las pruebas, solo se

debe hacer una integración a la vez y se debe de tener claro cual es la última versión funcional.

Desde el primer momento, se llevó un registro de los respaldos del último código funcional, el cual su nombre estaba compuesto de la fecha y que tipo de servidor maneja para así saber que es lo que contiene, si es la parte de datos junto a la base de datos o la parte de presentación del sistema.

Estos respaldos eran almacenados en dos partes, un respaldo externo en un host y un respaldo local, haciendo que la información está libre de cualquier eventualidad con respecto al equipo en que se está manejando.

De esta forma y cumpliendo con la explicación primera, se garantizó la fluidez en la integración de nuevo código, haciendo que si existe algún error en la versión que se está trabajando, se pueda volver a la versión anterior sin ningún problema.

4.1.4. Integraciones frecuentes.

Los principios fundamentales de la metodología XP con respecto a este tema, son que se debe hacer integraciones en cada momento como un máximo de tiempo de un día; entre más se tarde en encontrar un problema, resultará más costoso resolverlo y la integración frecuente evita problema como el de trabajar sobre una clase obsoleta.

Durante el transcurso de este proyecto, no fue aplicado este tema con frecuencia, ya que el equipo de desarrollo se conformaba por una persona y el fue el único encargado de

modificar cada versión y de ser necesario, el podía cambiar otras clases sin que otra parte se dallen en su desarrollo.

4.1.5. Aplicación de estándares.

Otro de las más importantes etapas es la estandarización del código y en este proyecto se aplicó desde el principio. Esta estandarización del código, fue desarrollada antes de comenzar a codificar, dedicando el tiempo necesario y cubriendo todos los casos posibles que podían suscitarse en el desarrollo de esta aplicación.

Si bien el fin de estandarizar el código es de poder dar un punto colectivo y en el caso de este proyecto no se tiene la parte colectiva, ya que se tiene a un desarrollador; el poder desarrollar el sistema de esta forma, se ven el resultado a futuro, cuando se quiera desarrollar otras funciones y de forma muy acelerada y con más personas, así se realice con la misma persona que desarrolló el proyecto, simplemente es para que se pueda acordar de lo que en algún momento desarrollo, porque tiene un orden preestablecido.

Este no es un principio que solo sigue esta metodología, si no muchas otras, las cuales hacen que sea un principio muy global en la vida del desarrollador promedio en nuestro entorno.

4.2. Desarrollo de Iteraciones.

4.2.1. Primera iteración (Módulo Sistema).

Historia de Usuario 1: Registrar usuarios nuevos.

Tarea 1.1: Crear interfaz de formulario para el registro

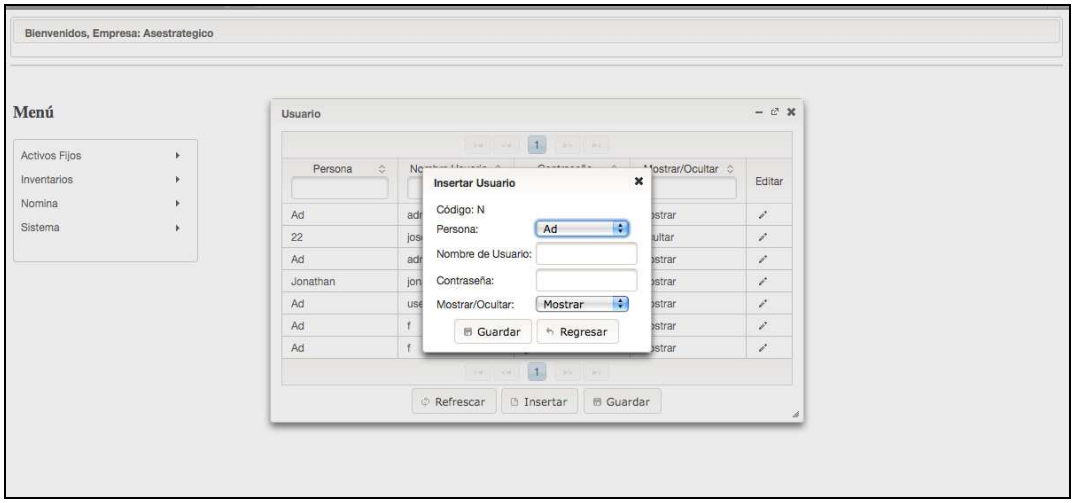


Figura4.1: Formulario para el Registro

Tarea 1.2: Crear base de datos para usuarios.

sis_usuario			
susu_codigo	<pi>	Variable characters (12)	<M>
susu_nusuario		Variable characters (30)	
susu_contrasenia		Variable characters (30)	
susu_codigo_ua		Variable characters (12)	
susu_fecha_a		Date	
susu_codigo_um		Variable characters (12)	
susu_fecha_um		Date	
susu_mo_oc		Variable characters (10)	<M>
PK_sis_usuario <pi>			

Figura4.2: Tabla sis_usuario

Historia de Usuario 2: Permisos de acceso a usuarios.

Tarea 2.1: Crear formulario de mantenimiento de permisos

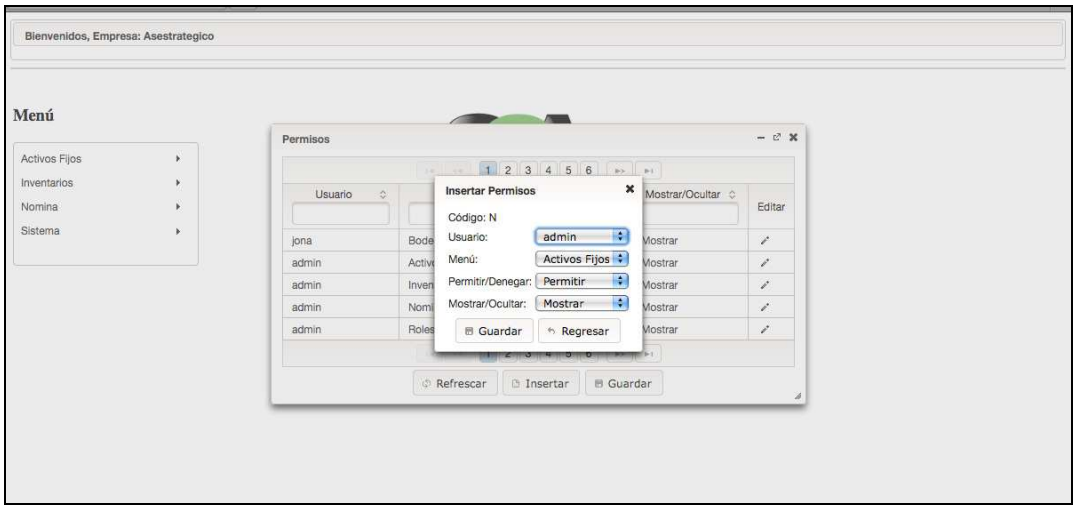


Figura 4.3: Formulario de mantenimiento de permiso

Tarea 2.2: Crear base de datos para permisos de usuarios.

sis_permisos			
sper_cod	<pi>	Variable characters (12)	<M>
sper_permi		Variable characters (15)	<M>
sper_fech_ua		Date	<M>
sper_fech_um		Date	<M>
sper_mo_oc		Variable characters (10)	<M>
PK_sis_permisos	<pi>		

Figura 4.4: Tabla sis_permiso

Historia de Usuario 3: Registro del menú del sistema.

Tarea 3.1: Crear formulario de mantenimiento de menú.

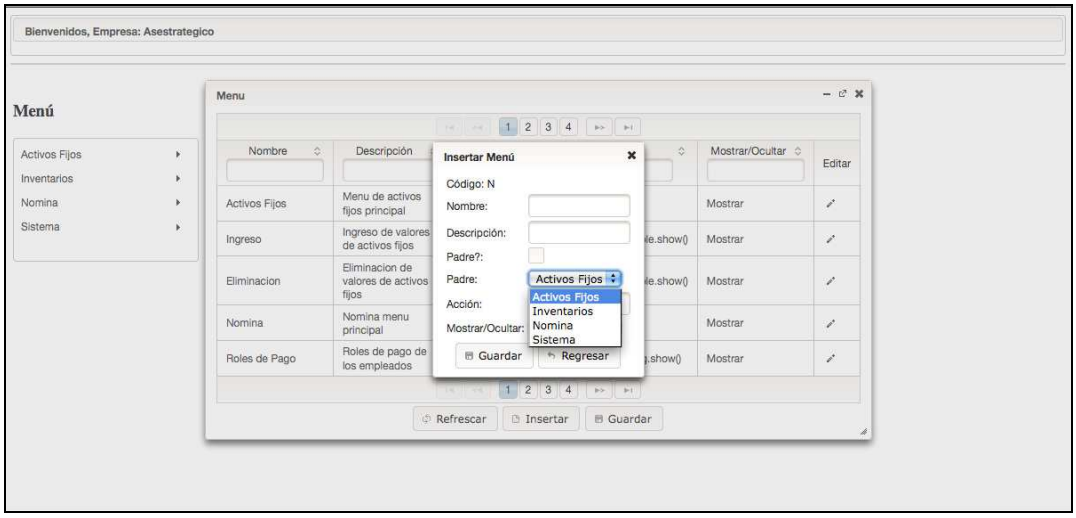


Figura 4.5: Formulario de mantenimiento de menú

Tarea 3.2: Crear base de datos para menú.

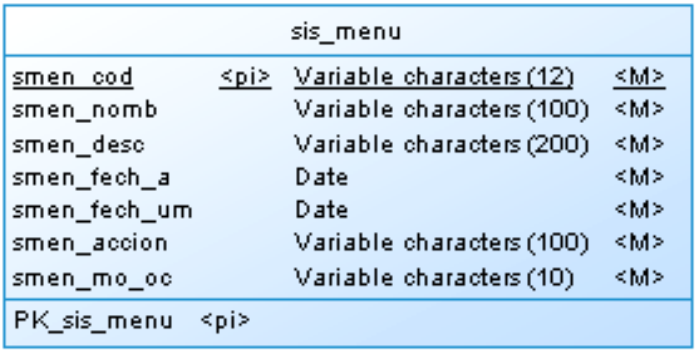


Figura4.6: Tabla sis_menu

Historia de Usuario 4: Registro de empresas que maneja el sistema.

Tarea 4.1: Crear formulario de mantenimiento de empresas.

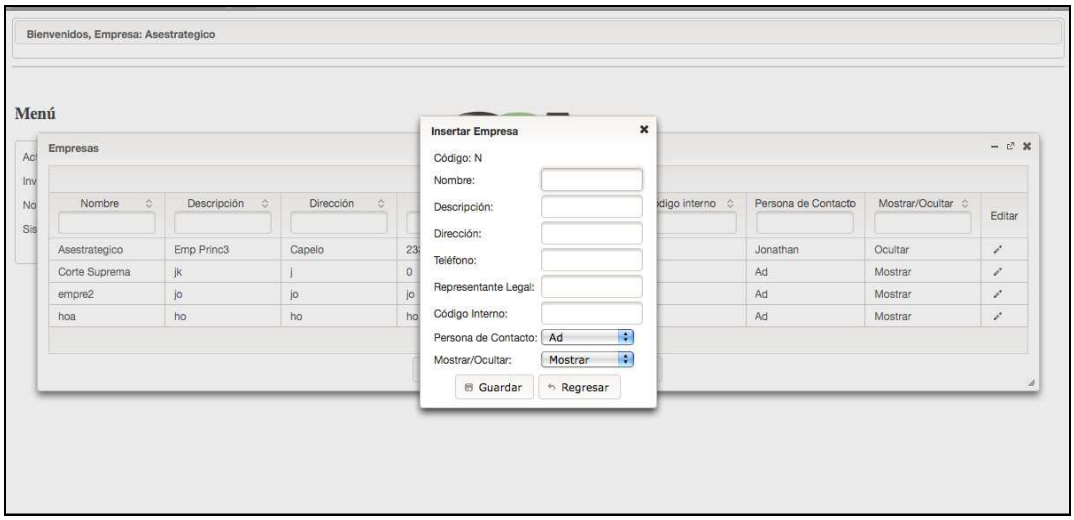


Figura4.7: Formulario de mantenimiento de empresas

Tarea 4.2: Crear formulario de mantenimiento de menú.

sis_empresa			
sem_cod	<pi>	Variable characters (12)	<M>
sem_nomb		Variable characters (100)	<M>
sem_desc		Variable characters (200)	<M>
sem_direcc		Variable characters (150)	<M>
sem_telf		Variable characters (50)	<M>
sem_represen		Variable characters (100)	<M>
sem_cod_inter		Variable characters (5)	<M>
sem_fech_a		Date	<M>
sem_fech_um		Date	<M>
sem_mo_oc		Variable characters (10)	<M>
PK_sis_empresa	<pi>		

Figura4.8: Tabla sis_empresa

4.2.2. Segunda iteración (Módulo Nómina).

Historia de Usuario 5: Registro de personas empleadas en cada empresa.

Tarea 5.1: Crear formulario de mantenimiento de personas empleadas en las empresas.

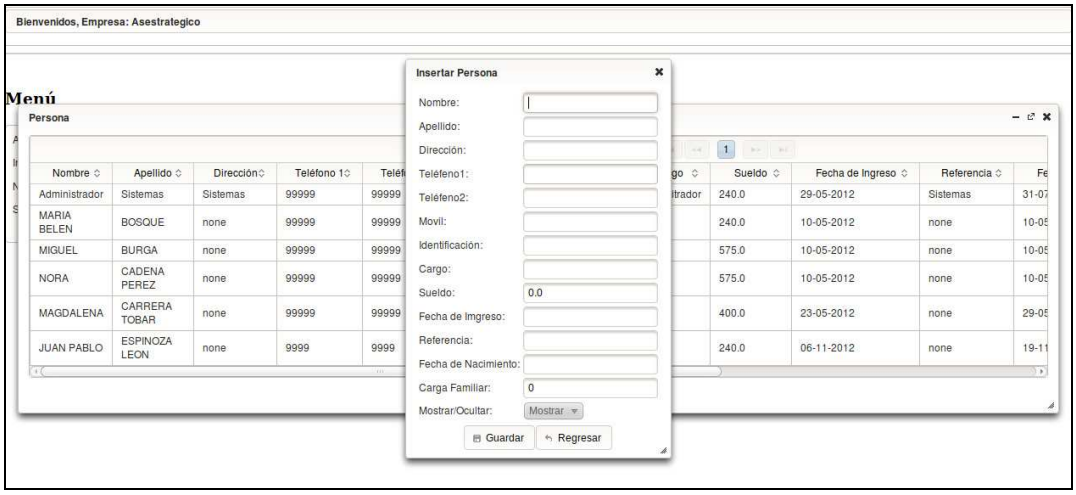


Figura4.9: Formulario de mantenimientos de personas empleadas en las empresas

Tarea 5.2: Crear base de datos para empleados de cada empresa.

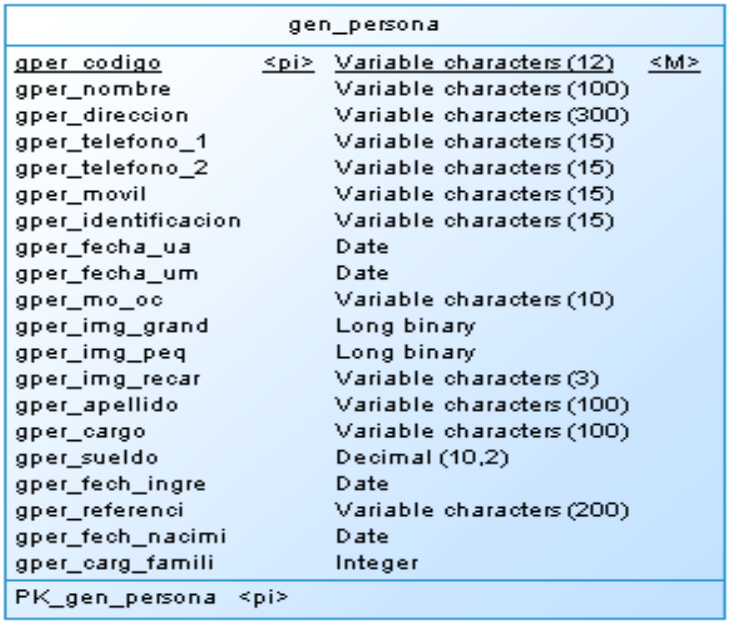


Figura4.10: Tabla gen_persona

Historia de Usuario 6: Registro de tipo de rubro.

Tarea 6.1: Crear formulario de mantenimiento de tipo de rubro.

Figura4.11: Formulario de mantenimiento de tipo de rubro

Tarea 6.2: Crear base de datos de tipo de rubro.

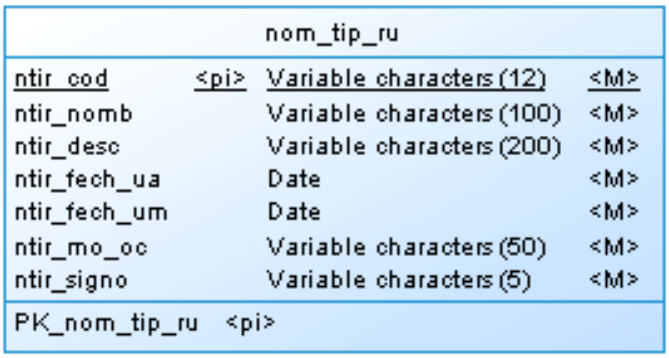


Figura4.12: Tabla nom_tip_ru

Historia de Usuario 7: Registro de rubros.

Tarea 7.1: Crear formulario de mantenimiento de rubros.

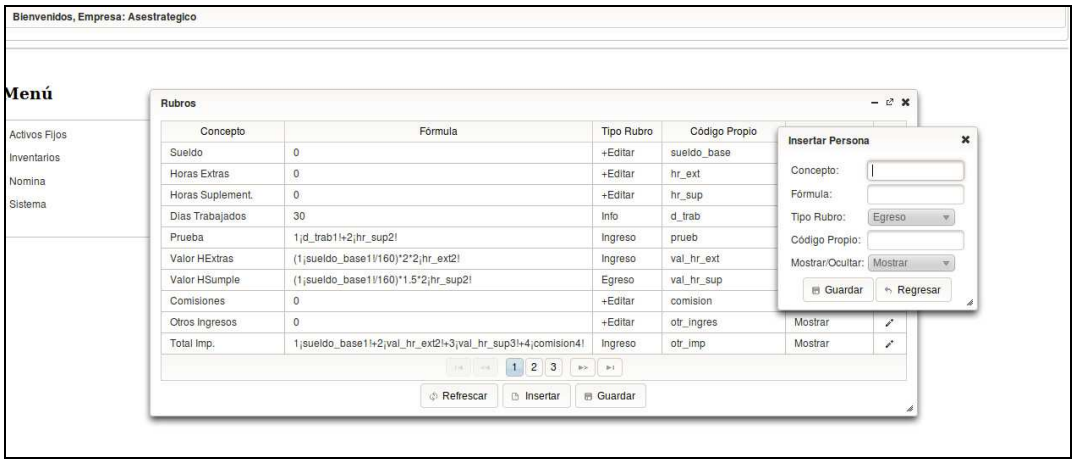


Figura4.13: Formulario de mantenimiento de rubro

Tarea 7.2: Crear base de datos para rubros.

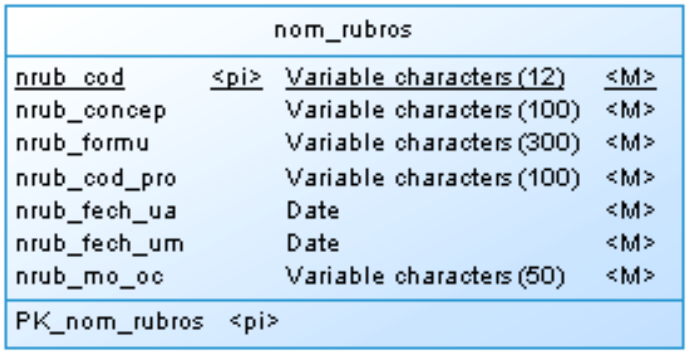


Figura4.14: Tabla nom_rubros

Historia de Usuario 8: Administración de Roles de Pago.

Tarea 8.1: Crear formulario para Administración Roles de Pago.

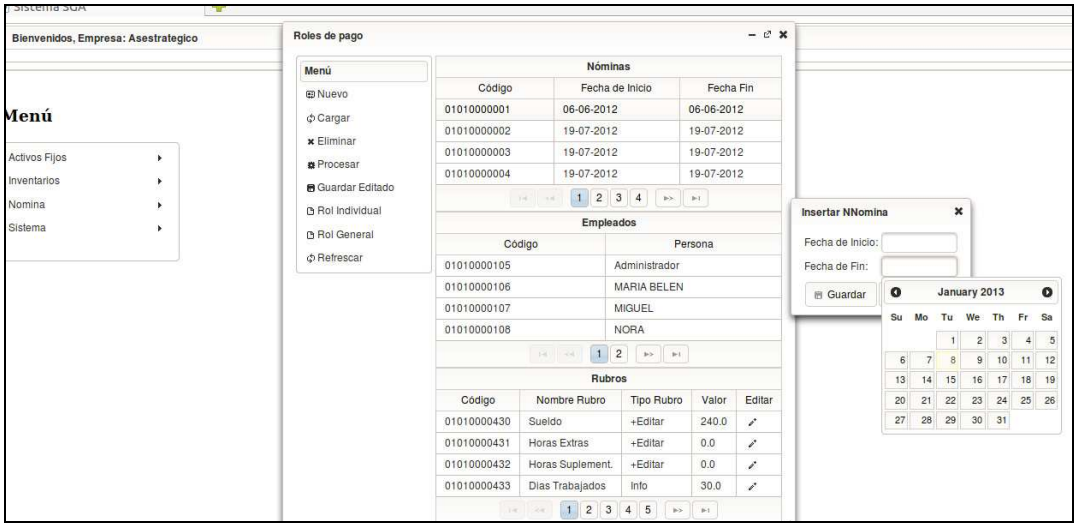


Figura4.15: Formulario de administración de roles de pago

Tarea 8.2: Crear base de datos para Administración Roles

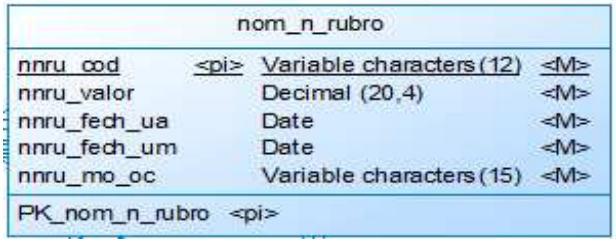


Figura4.16: Tabla nom_n_rubro

Figura4.17: Tabla nom_n_nomina

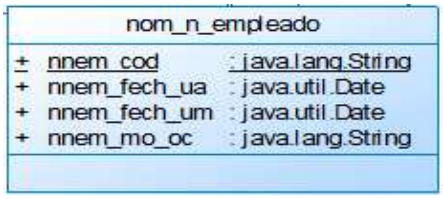


Figura4.18: Tabla nom_n_empleado

Tarea 8.3: Crear Reporte Rol Individual

Figura 4.19: Reporte Rol Individual

Tarea 8.4: Crear Reporte Rol General

Figura4.20: Reporte Rol General

ESPACIO EN BLANCO INTENCIONAL

4.2.3. Tercera iteración (Módulo Inventario).

Historia de Usuario 9: Registro de bodega.

Tarea 9.1: Crear formulario de mantenimiento de bodega.

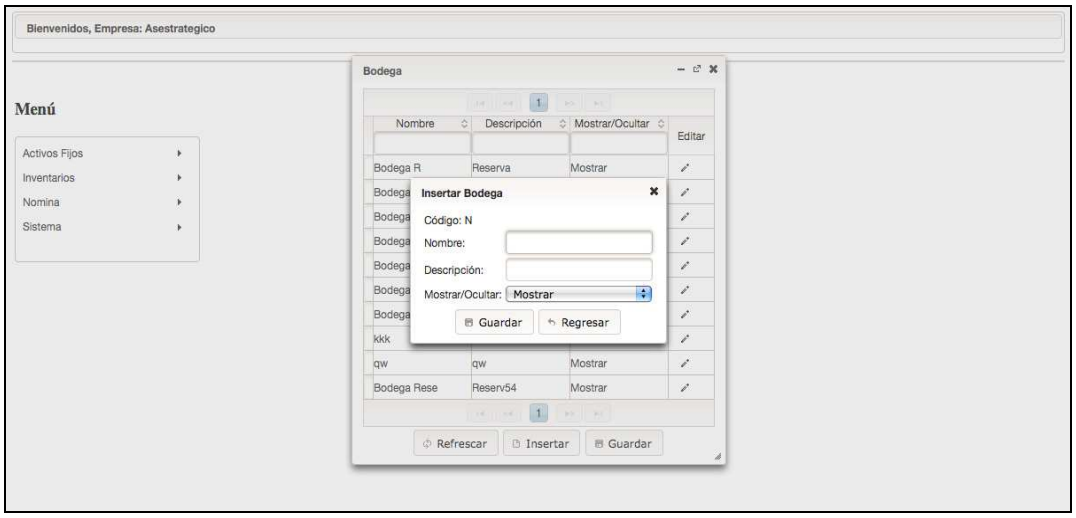


Figura4.21: Formulario de mantenimiento de bodega

Tarea 9.2: Crear base de datos para bodega.

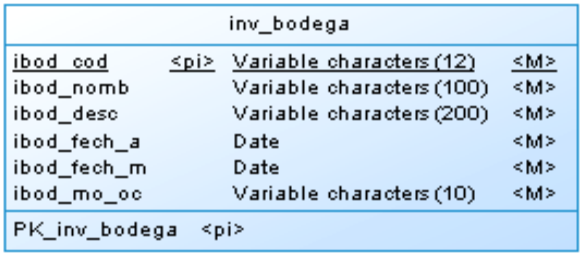


Figura4.22: Tabla inv_bodega

Historia de Usuario 10: Registro de producto.

Tarea 10.1: Crear formulario de mantenimiento de producto.

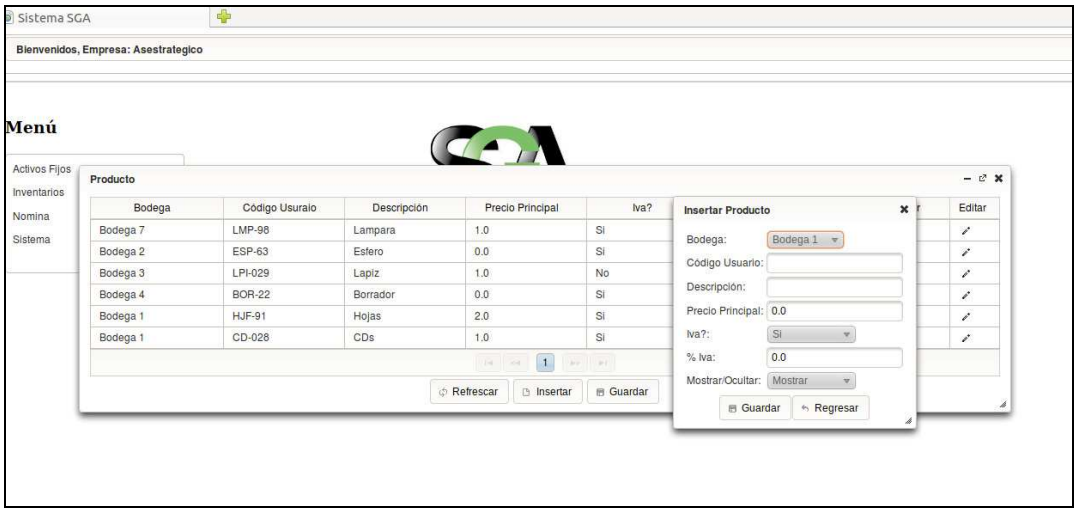


Figura4.23: Formulario de mantenimiento de producto

Tarea 10.2: Crear base de datos para producto.

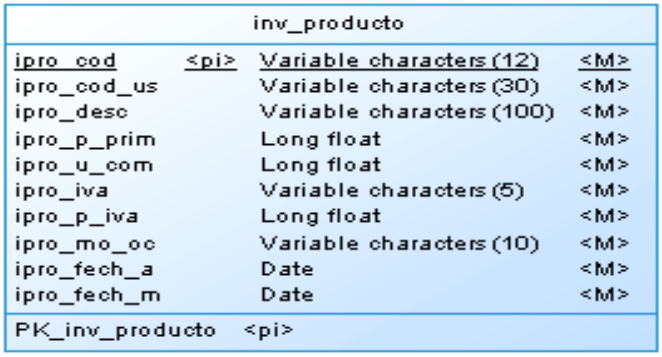


Figura 4.24: Tabla inv_producto

Historia de Usuario 11: Registro de Movimientos de productos

Tarea 11.1: Crear formulario de movimientos de productos

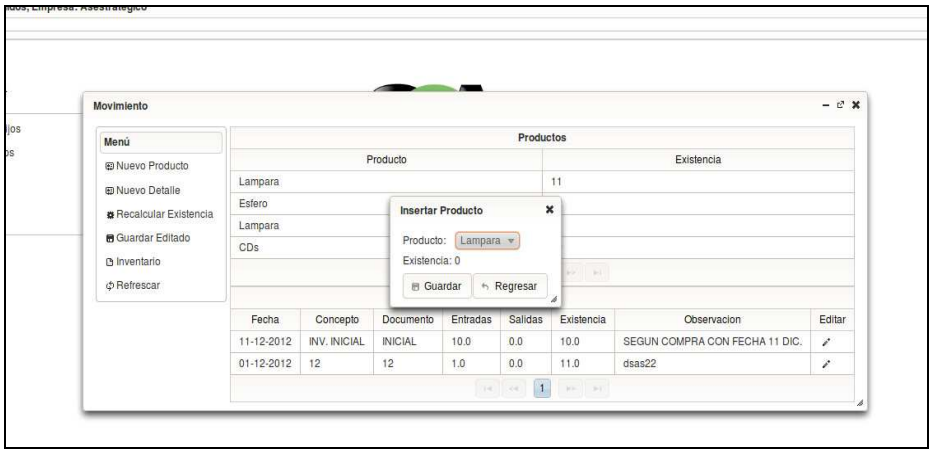


Figura4.25: Formulario de movimiento de producto

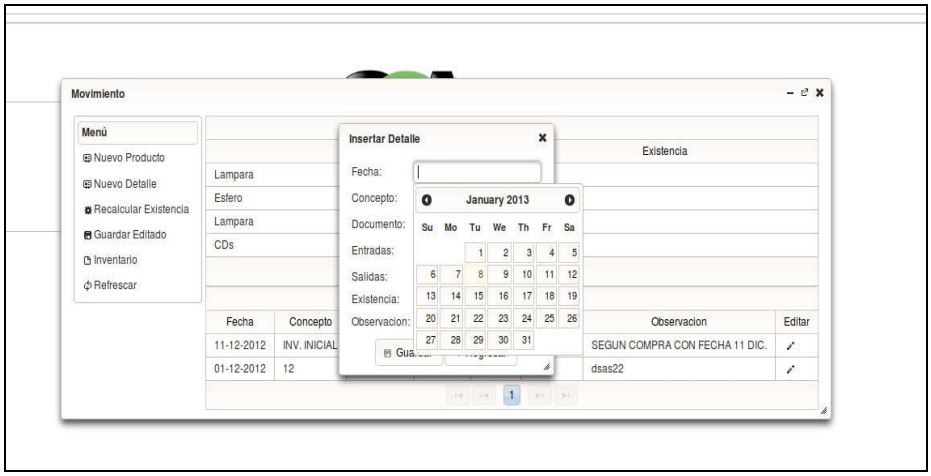


Figura4.26: Formulario de detalle del producto

Tarea 11.2: Crear base de datos para registro de movimiento

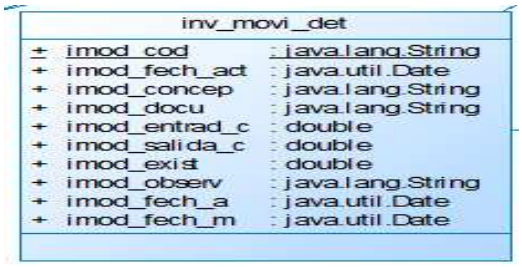


Figura4.27: Tabla inv_movi_det

Figura 4.28: Tabla inv_movi_det

ESPACIO EN BLANCO INTENCIONAL

Tarea 11.3: Crear Reporte Inventario de Productos

Figura 4.29: Reporte Inventario de Productos

4.2.4. Cuarta iteración (Módulo Activos Fijos).

Historia de Usuario 12: Registro de Tipos de Activos Fijos

Tarea 12.1: Crear formulario de mantenimiento de activos fijos

Figura 4.30: Formulario de Mantenimiento de Activos Fijos

Tarea 12.2: Crear base de datos para registro de movimiento

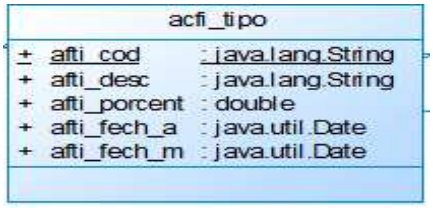


Figura 4.31: Tabla acfi_tipo

Historia de Usuario 13: Registro de Activos Fijos

Tarea 13.1: Crear formulario de mantenimiento de activos fijos

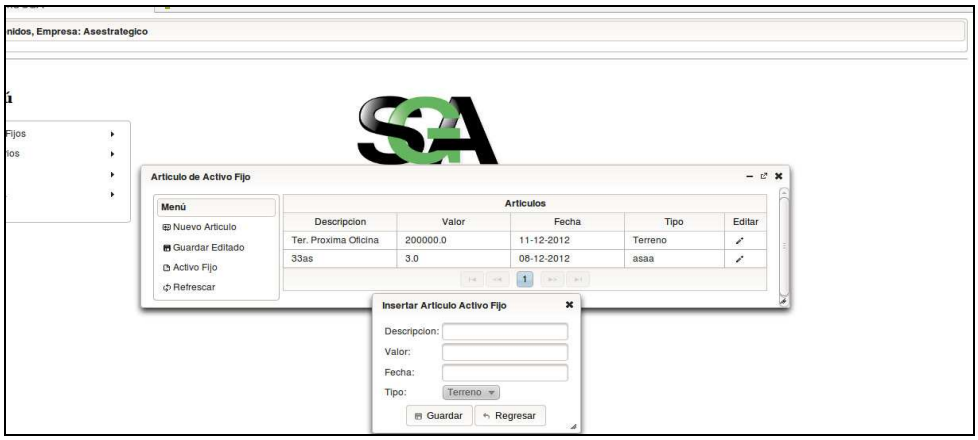


Figura 4.32: Formula de mantenimientos de Activos Fijos

Tarea 13.2: Crear base de datos para registro de activos fijos

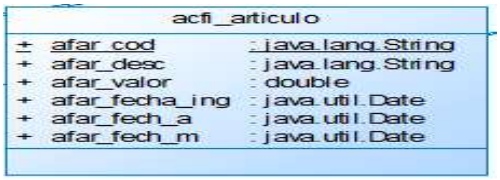


Figura 4.33: Tabla acti_articular

Tarea 13.3: Crear Reporte de Activos Fijos

Figura 4.34: Reporte de Activos Fijos

CAPÍTULO 5

5. PRUEBAS.

5.1. Módulo sistema

5.1.1. Historia de Usuario 1 – Registro de Usuarios Nuevos

5.1.1.1. Prueba de aceptación

a.- Identificar todos los posibles resultados observables de la historia.

- Interfaz para el llenado de datos individuales de los usuarios (Datos necesarios).
- Formulario para captura de datos de los usuarios.
- Interfaz de llenado de datos en el registro de usuarios.

b.- Identificar los resultados que terminan el registro de usuario y los que permiten continuar en el proceso.

- La historia termina cuando termina de llenar los datos de usuario.
- Dependiendo de la persona que se asigne al usuario, tendrá diferentes características del empleado.

c.- Identificar los caminos de ejecución disponibles.

- Almacena esta información en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que se introduce sobre los datos personales de los usuarios que se registran en el sistema

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.1.1.2. Prueba Unitaria

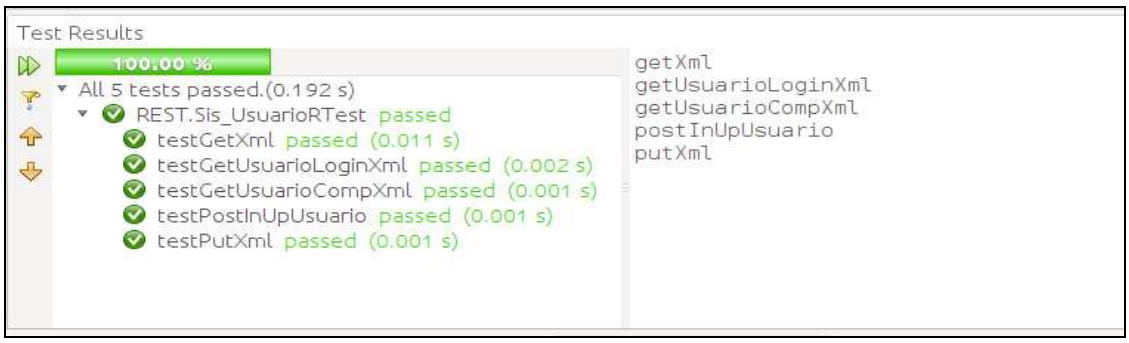


Figura 5.1: Prueba unitaria de registro de usuarios

5.1.2. **Historia de Usuario 2:** Permisos de acceso a usuarios.

5.1.2.1. Prueba de aceptación

a.- Identificar todos los posibles resultados observables del acceso a usuarios.

- Pantalla para mostrar la información de permisos de usuario. (Usuario, Menú, Permiso/Denegar).
- Interfaz para ingresar los permisos de acceso de los usuarios al sistema.
- Interfaz para editar los permisos de acceso de los usuarios al sistema

b.- Identificar los resultados que terminan el acceso del usuario y los que permiten continuar con el proceso.

- La historia termina cuando acaba de dar permiso al usuario.
- La historia continua cuando se quiere volver a dar permiso al usuario del sistema.

c.- Identificar los caminos de ejecución disponibles.

- Almacena y actualiza esta información en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información de permisos de usuario con usuario que se han registrado en el sistema para denegar/ otorgar permiso.

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.1.2.2. Prueba Unitaria

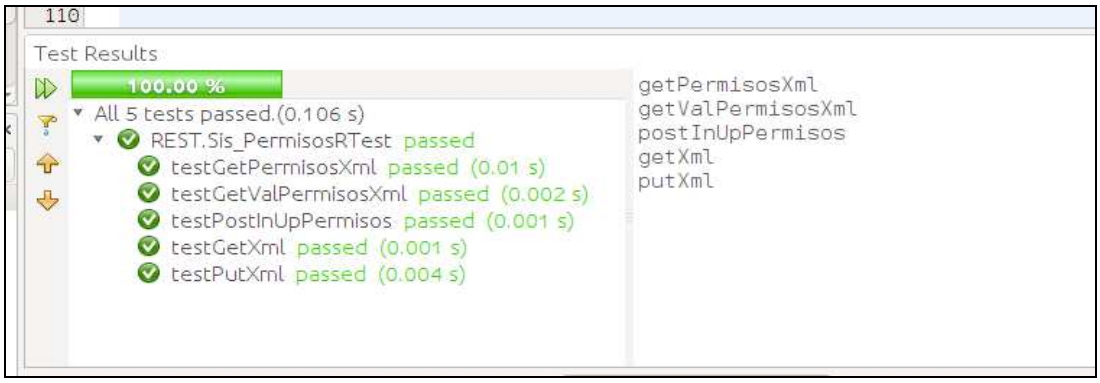


Figura 5.2: Prueba unitaria de permisos de usuario

5.1.3. Historia de Usuario 3: Registro del menú del sistema

5.1.3.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del registro del menú del sistema.

- Pantalla para mostrar la información del registro de menú del sistema (Código: N, Nombre, Descripción, Padre?, Padre, Acción, Mostrar/Ocultar).
- Interfaz para ingresar los registros de menú del sistema.
- Interfaz para editar los registros del menú del sistema.

b.- Identificar los resultados que terminan el registro de un menú y los que permiten continuar en el proceso.

- La historia termina cuando se realiza el registro de un menú.
- La historia continua cuando se quiere volver a crear otro menú.

c.- Identificar los caminos de ejecución disponibles.

- Se guarda el menú que se creó para luego utilizarlo.
- Almacena y actualiza esta información en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el menú como son (Código: N, Nombre, Descripción, Padre?, Padre, Acción, Mostrar/Ocultar).

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.1.3.2. Prueba Unitaria

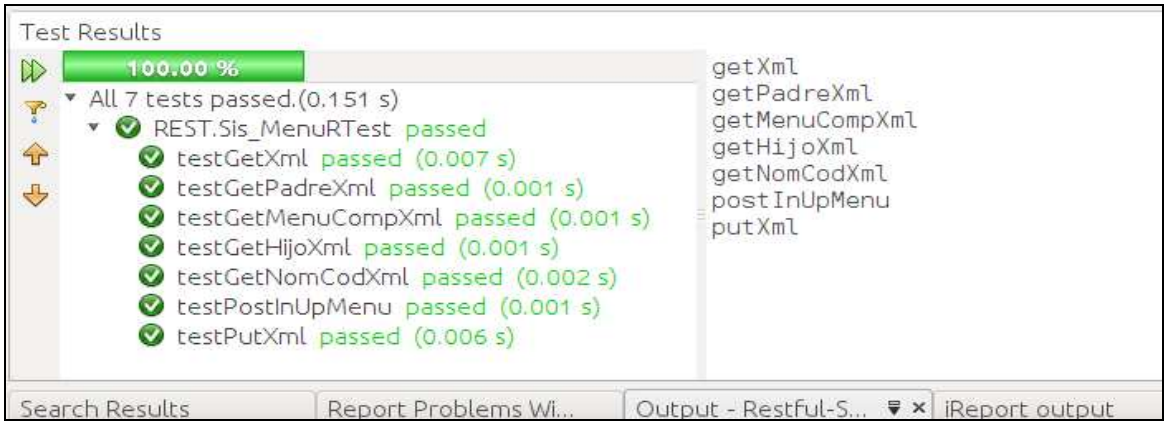


Figura 5.3: Prueba unitaria sobre los menú

5.1.4. Historia de Usuario 4: Registro de empresas que maneja el sistema.

5.1.4.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del registro de una empresa.

- Pantalla para mostrar la información necesaria para el registro de una empresa.
- Interfaz para ingresar los registros de una nueva empresa en caso que no exista en la base de datos.
- Interfaz para editar los registros de una empresa de la manera más conveniente y almacenarla en el sistema.

b.- Identificar los resultados que terminan el registro de una empresa y los que permiten continuar en el proceso.

- La historia termina cuando se realiza el registro de la empresa y la almacenas.
- La historia continua cuando se quiere volver a editar una empresa ya almacenada o registrar una empresa nueva.

c.- Identificar los caminos de ejecución disponibles.

- Se guardara la empresa registrada para luego utilizarla.
- Almacena y actualiza esta información en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registro de cada empresa como son (Código: N, Nombre, Descripción, Dirección, Teléfono, Representantes Legal, Código Interno, Personas de Contacto, Mostrar/Ocultar).

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.1.4.2. Prueba Unitaria

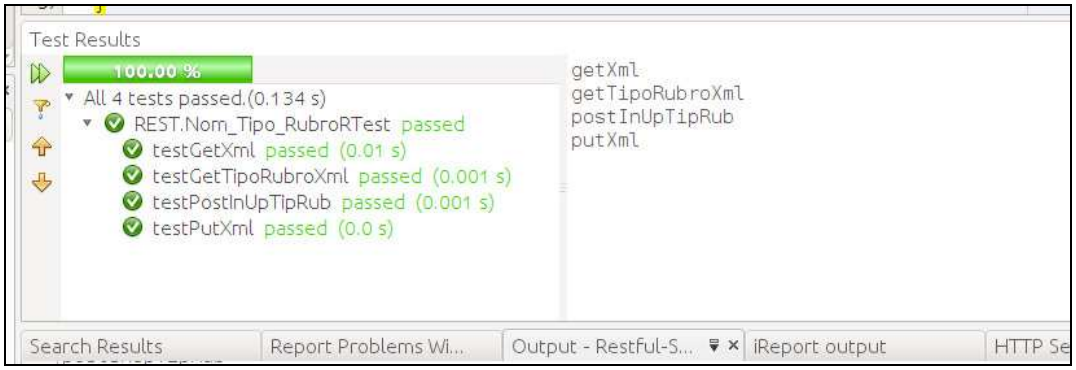


Figura 5.4: Prueba unitaria sobre tipo de rubro

5.2. Módulo Nómina

5.2.1. Historia de Usuario 5: Registro de personas empleadas en cada empresa.

5.2.1.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del registro de personas empleadas en cada empresa.

- Pantalla para mostrar la información necesaria para el registro de personas empleadas en cada empresa.
- Interfaz para ingresar el registro de personas empleadas en cada empresa.
- Interfaz para editar el registro de personas empleadas en cada empresa de la manera más conveniente y almacenarla en el sistema.

b.- Identificar los resultados que terminan el registro de personas empleadas en cada empresa y los que permiten continuar en el proceso.

- La historia termina cuando se realiza el registro de cada persona empleada en cada empresa y la guarda en el sistema.

- La historia continua cuando se quiere volver a editar una persona ya ingresada o cuando se quiere registrar un empleado nuevo.

c.- Identificar los caminos de ejecución disponibles.

- Almacena y actualiza la información del empleado de cada empresa en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registro de cada empleado de la empresa.

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.2.1.2. Prueba Unitaria

Figura 5.5: Prueba unitaria de personas

5.2.2. Historia de Usuario 6: Registro de tipo de rubro.

5.2.2.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del registro de tipo de rubro.

- Pantalla para mostrar la información para el registro de tipo de rubro.

- Interfaz para ingresar el registro del tipo de rubro.
- Interfaz para editar el registro de tipo de rubro.

b.- Identificar los resultados que terminan el registro de tipo de rubro y los que permiten continuar en el proceso.

- La historia termina cuando se realiza el registro de cada tipo de rubro y la guarda en el sistema.
- La historia continua cuando se quiere volver a crear otro tipo de rubro.

c.- Identificar los caminos de ejecución disponibles.

- Almacena y actualiza la información de cada tipo de rubro en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registro de tipo rubro identificando si es ingreso o egreso el rubro para el empleado de la empresa.

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.2.2.2. Prueba Unitaria

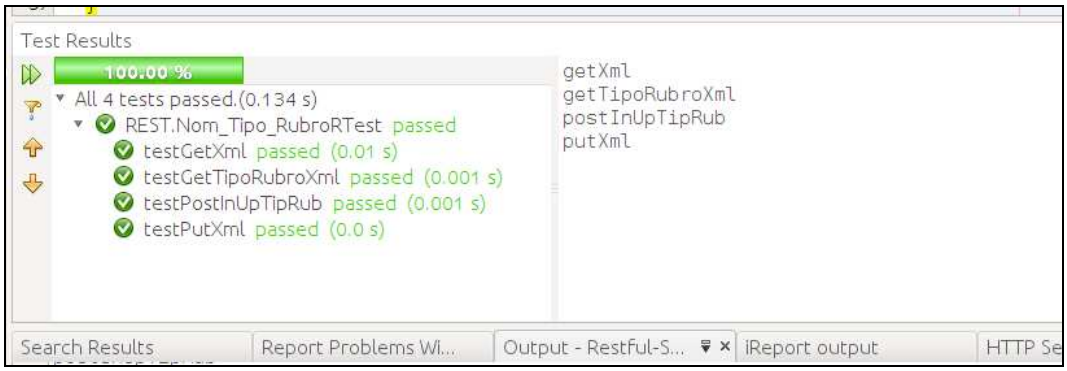


Figura 5.6: Pruebas unitarias de tipo de rubro

5.2.3. Historia de Usuario 7: Registro de rubros

5.2.3.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del registro de rubros.

- Pantalla para mostrar la información para el registro de rubros.
- Interfaz para ingresar el registro de rubros.
- Interfaz para editar el registro de rubro.

b.- Identificar los resultados que terminan el registro de rubros y los que permiten continuar en el proceso.

- La historia termina cuando se acaba de llenar el registro de rubros.
- La historia continua cuando se quiere realizar un nuevo registro de rubros en el sistema.

c.- Identificar los caminos de ejecución disponibles.

- Almacena y actualiza esta información en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registra rubros como son (Concepto, formula, Tipo rubro, Código propio, Mostrar/Ocultar)

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.2.3.2. Prueba Unitaria

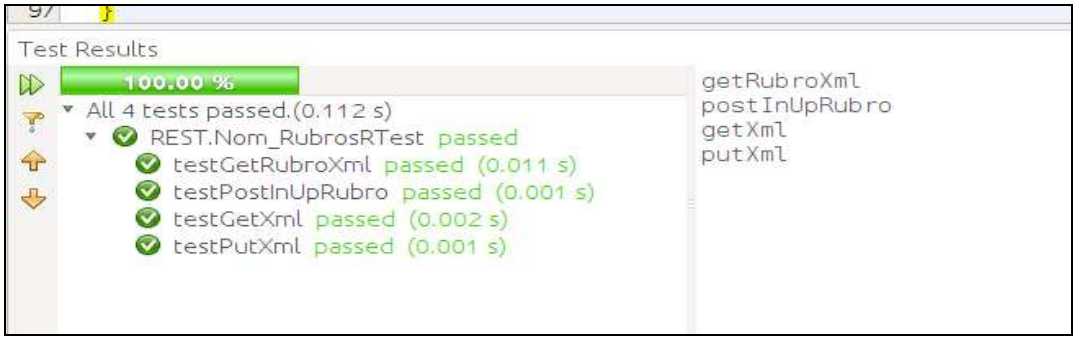


Figura 5.7: Pruebas unitarias de registro de rubro.

5.2.4. Historia de Usuario 8: Administración de Roles de Pago.

a.- Identificar todos los posibles resultados observables de la administración de roles de pago.

- Pantalla para mostrar la información para la administración de roles de Pago.
- Interfaz para la administración de roles de pagos de los empleados de la empresa en el sistemas.
- Interfaz para editar el rol de pago de los empleados de la empresa.

b.- Identificar los resultados que terminan la administración de roles de pagos y los que permiten continuar en el proceso.

- La historia termina cuando se acaba la administración de roles de pagos de cada uno de los empleados.
- La historia continua cuando se quiere realizar una nueva administración de los roles de pagos de los empleados de la empresa.

c.- Identificar los caminos de ejecución disponibles.

- Almacena y actualiza esta información en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información necesaria para poder realizar la administración de los roles de pago de los empleados de la empresa.

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.2.4.1. Prueba Unitaria

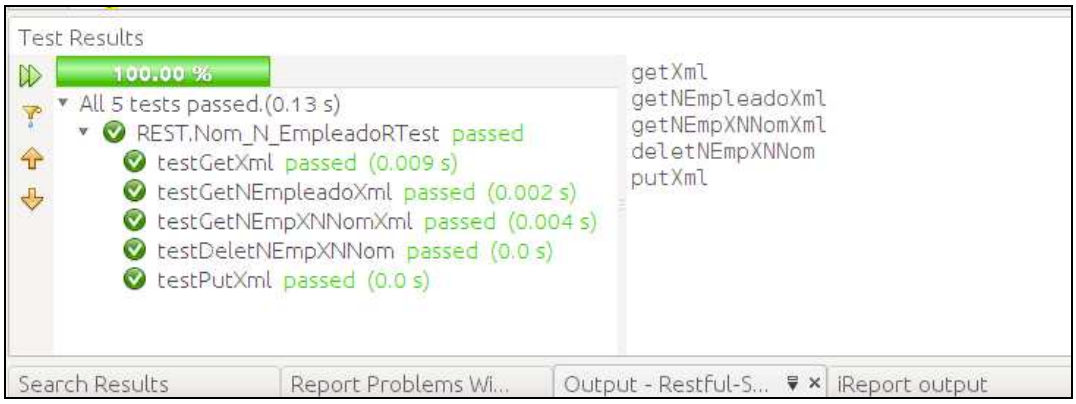


Figura 5.8: Prueba unitaria de empleados de la nómina

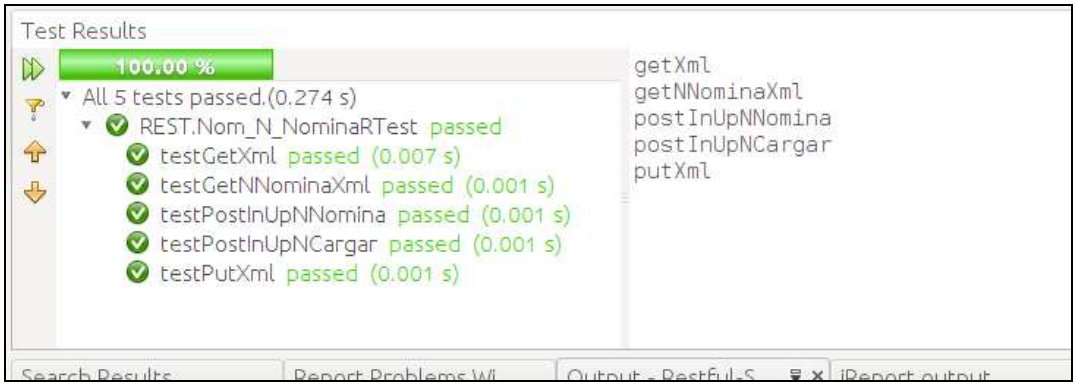


Figura 5.9: Prueba unitaria de la información de cada nómina

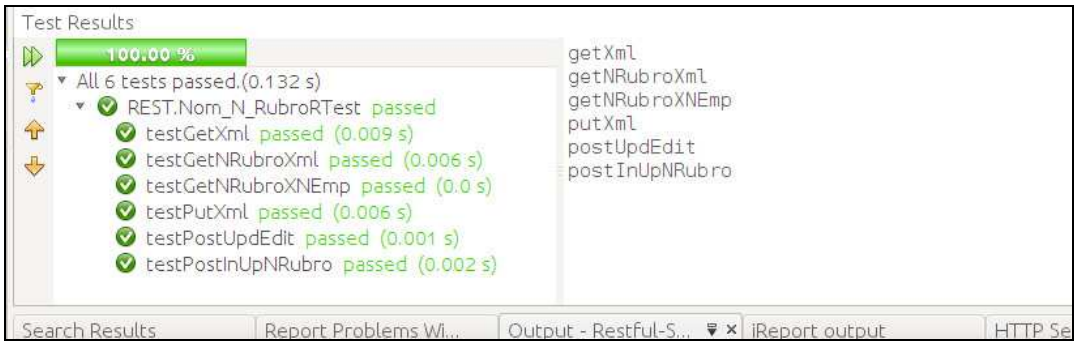


Figura 5.10: Prueba unitaria de los rubros pertenecientes a cada nómina

5.3. Módulo Inventario

5.3.1. Historia de Usuario 9: Registro de bodega.

5.3.1.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del registro de bodega.

- Pantalla para mostrar la información para el registro de bodega.
- Interfaz para ingreso del registro de una nueva bodega.
- Interfaz para editar el registro de bodega.

b.- Identificar los resultados que terminan el registro de bodega y los que permiten continuar en el proceso.

- La historia termina cuando se acaba de llenar el registro de bodega.
- La historia continua cuando se quiere realizar un nuevo registro de bodega en el sistema.

c.- Identificar los caminos de ejecución disponibles.

- Especificación de bodega por productos.
- Almacena y actualiza esta información en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registra bodega como son (Código: N, Nombre, Descripción, Mostrar/Ocultar)

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.3.1.2. Prueba Unitaria

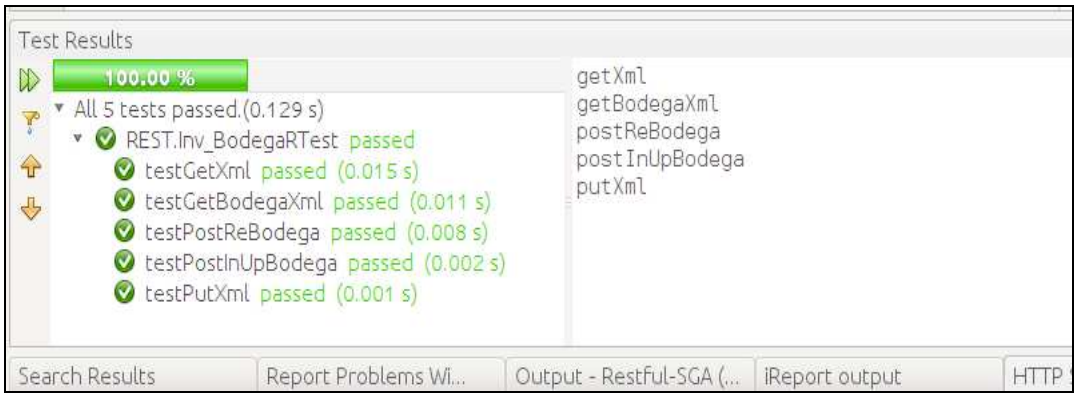


Figura 5.11: Prueba unitaria de registro de bodega

5.3.2. Historia de Usuario 10: Registro de producto.

5.3.2.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del registro de producto.

- Pantalla para mostrar la información para el registro de producto
- Interfaz para ingresar el registro de producto en bodega.
- Interfaz para editar el registro de producto en bodega.

b.- Identificar los resultados que terminan el registro de producto y los que permiten continuar en el proceso.

- La historia termina cuando se realiza el registro de producto en bodega y la almacena en el sistema.
- La historia continua cuando se quiere volver a ingresar el registro de un nuevo producto en bodega.

c.- Identificar los caminos de ejecución disponibles.

- Almacena y actualiza la información de los productos existentes en bodega en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registro de producto especificando (Código: N, Bodega, Código Usuario, Descripción, Precio Principal, Utilidad Compra, Iva)

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.3.2.2. Prueba Unitaria

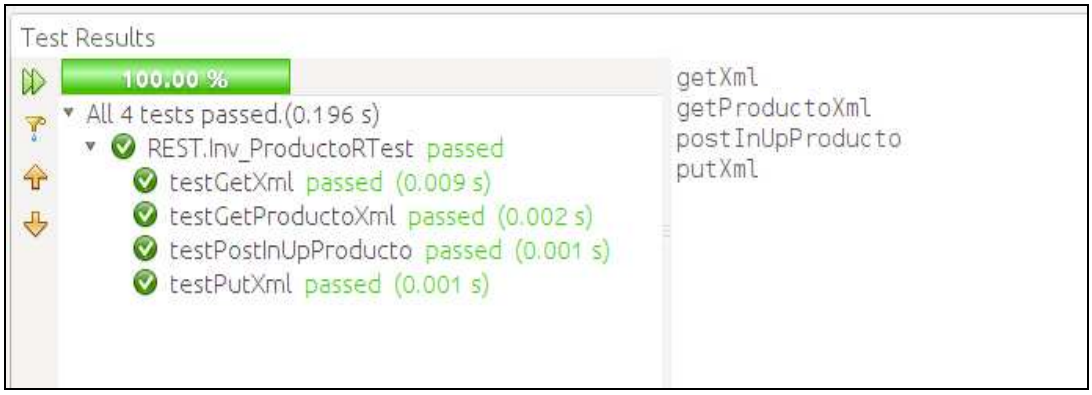


Figura 5.12: Prueba unitaria de registro de producto

5.3.3. **Historia de Usuario 11:** Registro de Movimientos de productos

5.3.3.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del Registro de Movimientos de productos

- Pantalla para mostrar la información para el Registro de Movimientos de productos.
- Interfaz para ingreso del Registro de Movimientos de productos existentes en bodega.
- Interfaz para editar el Registro de Movimientos de productos en bodega.

b.- Identificar los resultados que terminan el Registro de Movimientos de productos y los que permiten continuar en el proceso.

- La historia termina cuando se acaba de llenar el del Registro de Movimientos de productos.
- La historia continua cuando se quiere realizar un nuevo registro de Movimientos de productos.
- c.- Identificar los caminos de ejecución disponibles.
- Especificación de cuantos productos existen en cada tipo de bodega.
- Almacena y actualiza esta información en la base de datos del sistema.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registro de Movimientos de productos.

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.3.3.2. Prueba Unitaria

Figura 5.13: Prueba unitaria de cabecera de movimiento

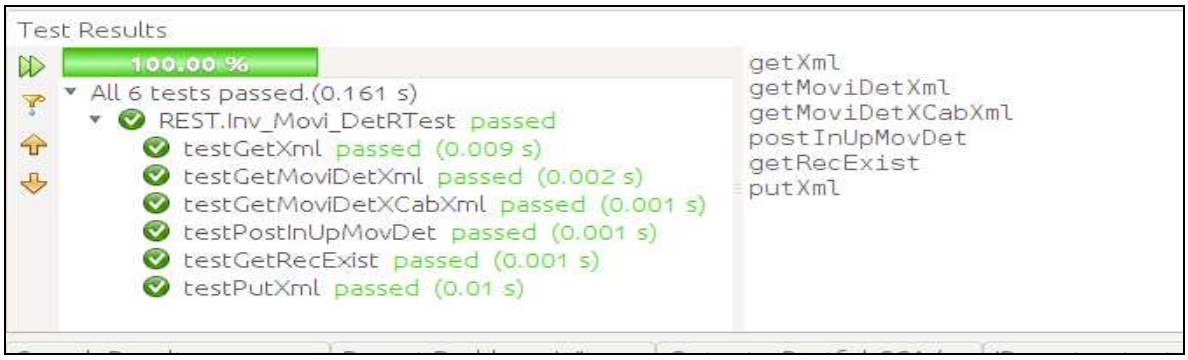


Figura 5.14: Prueba unitaria de detalle de movimiento

5.4. Módulo Activos Fijos

5.4.1. **Historia de Usuario 12:** Registro de Tipos de Activos Fijos

5.4.1.1. Pruebas de aceptación

a.- Identificar todos los posibles resultados observables del registro del tipo de Activo Fijo.

- Pantalla para mostrar la información para el registro del tipo de activos fijos. . (Descripción, Porcentaje).
- Interfaz para ingresar el registro del tipo de activos fijos.
- Interfaz para editar el registro del tipo de activos fijos.

b.- Identificar los resultados que terminan el registro del tipo de activos fijos y los que permiten continuar en el proceso.

- La historia termina cuando se realiza el registro del tipo de activos fijos y lo almacena en el sistema.
- La historia continua cuando se quiere volver a crear otro registro de tipo de activos fijos.

c.- Identificar los caminos de ejecución disponibles.

- Almacena y actualiza la información de cada registro de tipo de activo fijo en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registro de tipo de activo fijo.

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.4.1.2. Prueba Unitaria

Figura 5.15: Prueba unitaria de tipos de activos fijos

5.4.2. Historia de Usuario 13: Registro de Activos Fijos

5.4.2.1. Pruebas de aceptación

a.-Identificar todos los posibles resultados observables del registro del tipo de Activo Fijo.

- Pantalla para mostrar la información para el registro de activos fijos.
- Interfaz para ingresar el registro de activos fijos.
- Interfaz para editar el registro de activos fijos.

b.- Identificar los resultados que terminan el registro de activos fijos y los que permiten continuar en el proceso.

- La historia termina cuando se realiza el registro de activos fijos y lo almacena en el sistema.
- La historia continua cuando se quiere volver a crear otro registro de activos fijos.

c.- Identificar los caminos de ejecución disponibles.

- Almacena y actualiza la información de cada registro de activo fijo en la base de datos.

d.- Asignar un conjunto de valores válidos y valores de entorno a cada camino de ejecución para obtener resultado esperado:

- El conjunto de valores validos está dado por el camino de datos compuesto por la información que contiene el registro de activo fijo. (Descripción, Valor, Fecha, Tipo).

e.- Eliminación de caminos redundantes.

- No existen caminos redundantes.

5.4.2.2. Prueba Unitaria

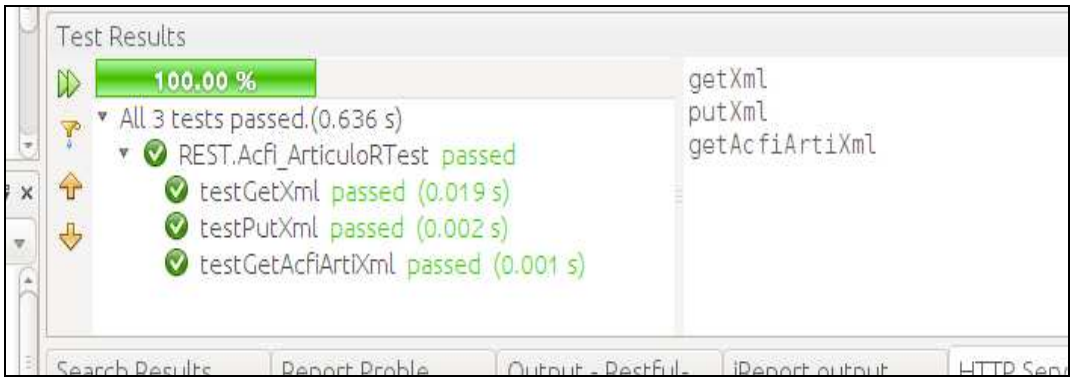


Figura 5.16: Prueba unitaria de Activos Fijos

CAPÍTULO 6

6. CONCLUSIONES Y RECOMENDACIONES.

6.1. CONCLUSIONES

A través de la presente tesis se ha logrado automatizar los procesos de gestión administrativa de Activos Fijos, Inventarios y Nómina con la generación de sus respectivos reportes; ahorrando recursos como: tiempo, esfuerzo y dinero dentro de un software confiable, estable y de alto rendimiento con la tecnología que en la actualidad se encuentra al alcance.

De acuerdo con el análisis de las diferentes historias de usuario, se puedo planificar y realizar las miniversiones del sistema según la metodología XP, sin embargo este proyecto fue de un corto alcance, lo que genera un gran limitante al momento de realizar nuevas funciones sin mencionar también que las herramientas de desarrollo permiten esto. Restful da un aporte muy esencial y significativo en la construcción del aplicativo, convirtiéndose en el eje central de todo el desarrollo. Si se analiza la arquitectura del sistema, se tiene el mapeo de objetos de acuerdo a la base de datos representado en los POJOs siguiendo lo especificado por EJB3; y por otro lado se tiene la capa de usuario donde se construyó con la librería de componentes Primefaces. Ambos lados se comunican a través de XML.

En la construcción del sistema se tienen ejes fundamentales y atractivos por ambas partes; la primera parte está enfocada en el desarrollo, mencionando que la arquitectura que se empleó, permite realizar cambios de una forma en la que el desarrollador conoce donde va cada función, método o algoritmo y así poder reutilizar el código ya escrito; la segunda parte es la del usuario, gracias a la implementación de la librería de componentes

Primefaces en la interface, muestra una solución muy visible, interactiva y fácil de manejar.

6.2. RECOMENDACIONES

Utilizar herramientas de desarrollo reconocidas con versiones estables, que poseen suficiente documentación de apoyo, ya que si se utilizan versiones actuales suelen carecer de documentación y de estabilidad al momento de utilizarlas en el desarrollo del sistema.

Realizar interfaces de usuario amigables, siguiendo un estándar de diseño y utilizando la experiencia de usuario hace que este no sea muy difícil de manejar y de comprender para el usuario final.

Utilizar herramientas de software libre para evitar los altos costos de licenciamiento, además se tiene las mismas prestaciones que el software propietario y en algunos casos resulta más conveniente.

Implementar la integración de este sistema con otros aplicativos de la empresa que cuentan con módulos complementarios y que se encuentran de igual forma con los requerimientos que necesita el usuario dentro de un software estable y actual.

Implementar una interfaz móvil (ya que se cuenta con las capas de negocio donde se interactúa a través de RESTFul y XML con la capa de presentación) donde estén funciones básicas como reportes y una que otra funcionalidad necesaria para el usuario, la cual permita disponibilidad de la aplicación en cualquier momento a través del internet.

BIBLIOGRAFIA

[1] Leonardo de Seta, LS "Introducción a los servicios Web RESTfull" Noviembre 2008 http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.html

[2] Oracle Corporation, OR "Java EE Technical Documentation" Enero 2013 http://download.oracle.com/javaee.

[3] Michellen Chen, MC "Extreme-Programming" Agosto 2008 http://www.slideshare.net/joaquin_win/extreme-programming-456979

[4] Wikipedia, WP, "PrimeFaces" Enero 2013 http://es.wikipedia.org/wiki/PrimeFaces

[5] Wikipedia, WP, "Front-end y back-end JSF2" Enero 2013 http://es.wikipedia.org/wiki/Front-end_y_back-end_JSF2

[6] Wikipedia, WP, "JavaServer Faces" Enero 2013 http://es.wikipedia.org/wiki/JavaServer_Faces

[7] Wikipedia, WP, "AJAX" Enero 2013 http://es.wikipedia.org/wiki/AJAX

[8] MovilWare, MW, "Sistemas de Control de Activos y Código de Barras" Enero 2010 http://www.mobileware-mx.com/controlactivosfijos.asp

[9] Jose Trecet, JT, "La Nómina" Enero 2012 http://www.finanzzas.com/la-nomina

[10] Wikipedia, WP, "Sistema de administración de recursos humanos" Enero 2013 http://es.wikipedia.org/wiki/Sistema_de_administración_de_recursos_humanos

[11] Wikipedia, WP, " Inventario" Enero 2013 http://es.wikipedia.org/wiki/Inventario

[12] LampHispano, LH, “Sistema de Inventario” Enero 2010
http://www.desarrollandoweb.com/demos/sistema-de-inventario-3.php

[13] Emagister, MG, “Kardex y métodos de valoración de inventarios” Enero 2009
http://www.mailxmail.com/curso-estados-financieros-ejercicios-herramientas auxiliares/kardex-metodos valoracion-inventarios-1

[14] Oracle Corporation, OC, “Java” Febrero 2011
http://www.java.com/es/faq/whatis_java.xml

[15] Luoman, LO, “Aprender Java(Programación Orienta a objetos)”, Enero 2010
http://aprender-java.blogspot.com/ide.html

[16] Toni Recio, TR, “API”, Diciembre 2011
http://desarrolloweb.com.ar/API

[17] Microsoft, MS, “SOA”, Octubre 2010
http://download.microsoft.com/0701-Real_World_SOA.pdf

[18] Masadelante, MA, “XML”, Septiembre 2010
http://www.masadelante.com/flags/xml

[19] Joaquin Gracia, JG, “UML: Diagramas UML ¿Qué es UML? Análisis y Diseño. Ingeniería del Software.” Mayo 2005
http://www.ingenierosoftware.com/analisisydiseno/uml.php

[20] Tumaestroweb, TE, “¿Qué es MVC?”, Junio 2009
http//www.tumaestroweb.com/curiosidades/que-es-mvc/

[21] Lobroweb, LW, “CRUD”, Julio 2009
http://www.lobroweb.es/symfony_formularios/cap4/el_generador_crud.html

[22] Jorge Lopez, Jorgaf, “Programa en Java”, Enero 2010
http://cafelojano.wordpress.com/2010/01/24/un-cliente-jersey

[23] Netbeans, NB, “Información NetBeans IDE 6.1”, Enero 2012
http://netbeans.org/community/releases/61/index_es.html

[24] Wikipedia, WP, “Código escape ASNI” Enero 2013
http://es.wikipedia.org/wiki/codigo_escape_ANSI

[25] Publib, PB, “DB2 Universal Database”, Abril 2009
http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp?topic=/com.ibm.db2.udb.doc/ad/c0006984.htm

[26] MySQL, MS, “MySQL”, Febrero 2009
http://dev.mysql.com/doc/refman/5.0/es/innodb-overview.htmln

[27] Wikipedia, WP, “MySQL” Enero 2013
http://es.wikipedia.org/wiki/MySQL

[28] Wikipedia, WP, “Hypertext transfer Protocol” Enero 2013
http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol

[29] Wikipedia, WP, “HTML” Enero 2013
http://es.wikipedia.org/wiki/HTML

[30] Alegsa, AS, “Definición de URL(dirección)¿Qué es URL?”, Julio 2010
http://www.alegsa.com.ar/Dic/url.php

[31] Willian Brogden, WB, “ Web Services”, Agosto 2011
http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf

[32] Leonardo De Seta, DS, “Introducción a los servicios Web RESTfull” Noviembre 2008
http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.html

[33] Sun Microsystem, SM, HttpSession (Java EE5 SDK), 2007
http://docs.oracle.com/javaee/5/api/javax/servlet/http/HttpSession.html

[34] DGSCA, DS, “Servlets”, Abril 2012
http://manuales.dgsca.unam.mx/webdina/servlets.htm

[35] Wikipedia, WP, “JAXWS” Enero 2013
http://wiki.eclipse.org/JAXWS

[36] Wikipedia, WP, “Sun Microsystems” Enero 2013
http://es.wikipedia.org/wiki/Sun_Microsystems

[37] Roger L. Costello, RC, “Licencia Libre de Software” Julio 2009
http://www.maestrosdelweb.com/editorial/licencias-libres-de-software-ii/j

[38] Franco Iacomella, FI, “Licencias Libres de Software II Maestros del Web”, Abril 2007
http://www.maestrosdelweb.com/editorial/licencias-libres-de-software

[39] Oracle Corporation, OC, “Oracle Technology Network for Java Developers”, Enero 2011
http://java.sun.com/developer/technicalArticles/glassfish/GFv2OpenforBusiness/index.html?cid=923034

[40] Masadelante, MA, “¿Qué es Ajax? Definición de Ajax”, Enero 2010
http://www.masadelante.com/faqs/ajax

[41] Enrrique Viñe, EV, “Introducción a PrimeFaces” Junio 2010
http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=introduccionPrimefaces

[42] Elegsa, EL, “Definición de CORBA”, Agosto 2011
http://www.alegsa.com.ar/Dic/corba.php

[43] Proactiva, PC, “EJB”, Marzo 2010
http://www.proactiva-calidad.com/java/ejb/introduccion.html

[44] Emilio Insfrá, EI, “MDA”, Mayo 2008

http://users.dsic.upv.es/~einsfran/mda/

[45] Stephen Wong, SW, “StarUML Tutorial”, Septiembre 2007

http://cnx.org/content/m15092/latest/

[46] Elegsa, EL, “Definición de Applet”, Octubre 2008

http://www.alegsa.com.ar/Dic/applet.php

[47] Elegsa, EL, “Definición de CGI”, Octubre 2008

http://www.alegsa.com.ar/Dic/cgi.php

[48] Proactiva, PC, “Arquitectura Java”, Diciembre 2009

http://www.proactiva-calidad.com/java/arquitectura/index.html

[49] Elegsa, EL, “Definición de Aplicaciones Distribuidas”, Octubre 2008

http://www.alegsa.com.ar/Dic/aplicacion%20distribuida.php

[50] Universidad de Madrid, UM “XP” Enero 2010:

http://www.esp.uem.es/jccortizo/xp.pdf

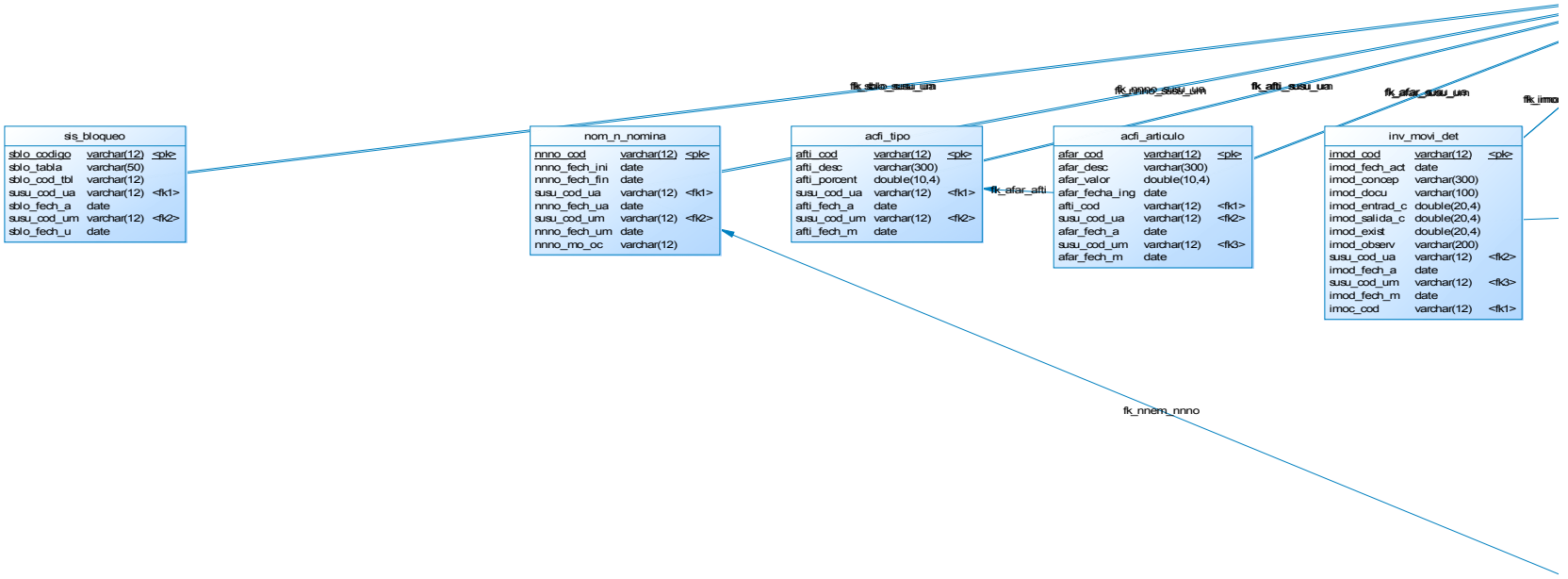
[51] Carlos Sanchez, CS “Ciclo de vida de un proyecto XP” Enero 2005:

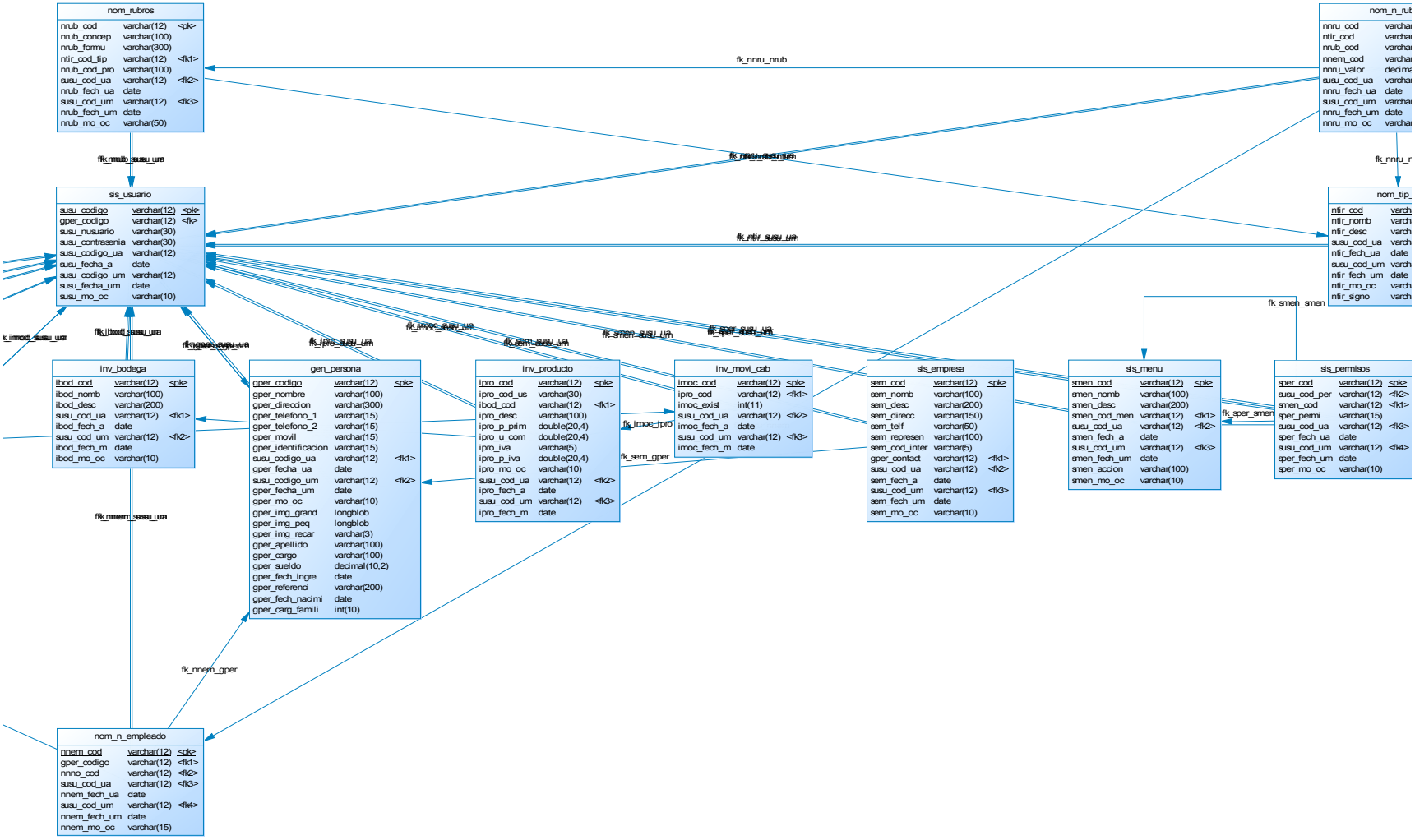
http://oness.sourceforge.net/proyecto/html/ch05s02.html

[52] Don Wells, DW “Extreme Programming Introduction” Septiembre 2009:

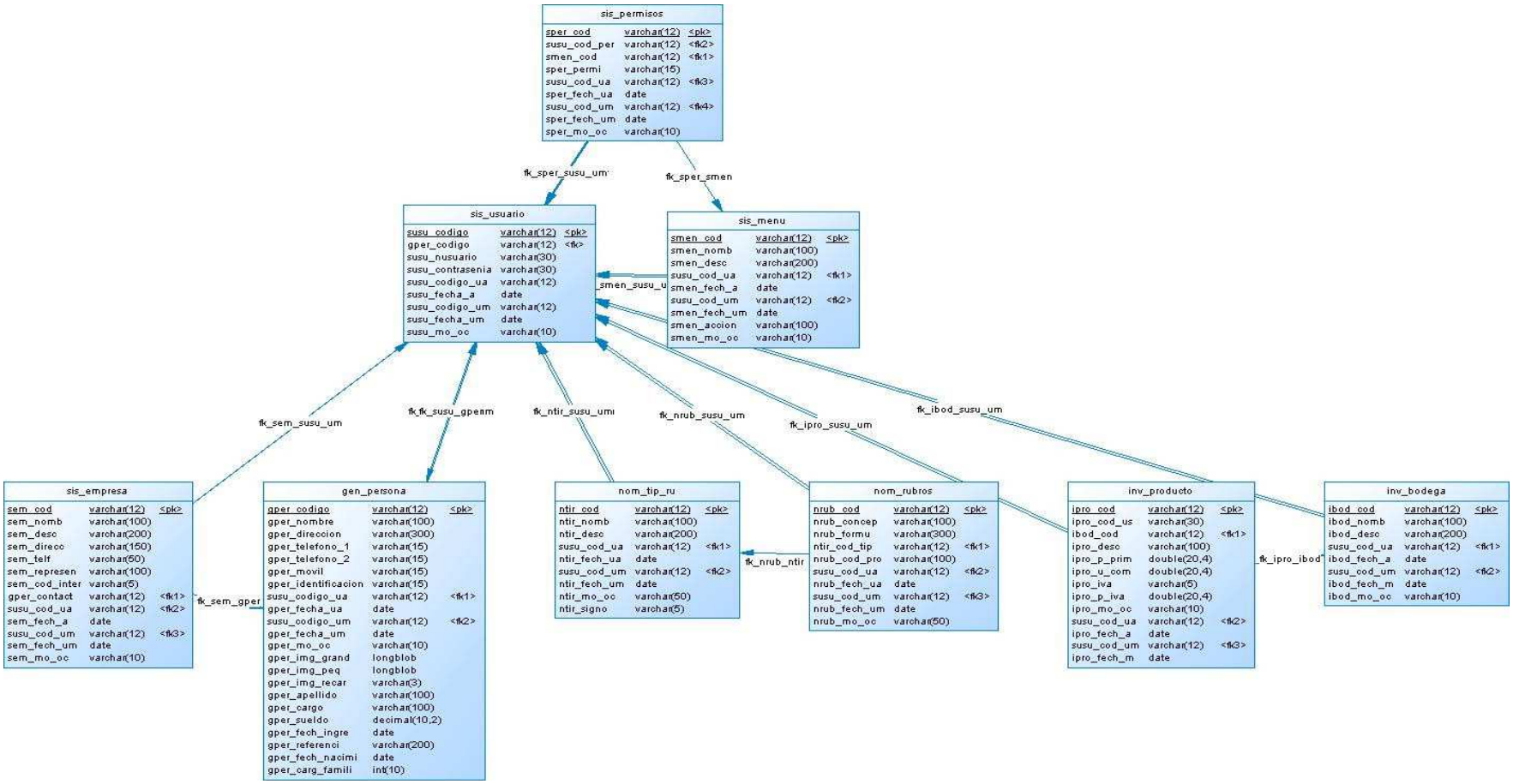
http://www.extremeprogramming.org/

Anexo A– Diagrama Físico




nom_rubros
nrub_cod varchar(12) <pk>
nrub_concep varchar(100)
nrub_formu varchar(300)
ntir_cod_tip varchar(12) <fk1>
nrub_cod_pro varchar(100)
susu_cod_ua varchar(12) <fk2>
nrub_fech_ua date
susu_cod_um varchar(12) <fk3>
nrub_fech_um date
nrub_mo_oc varchar(50)
fk_nnru_nrub
nom_n_rut
nnru_cod varchar
ntir_cod varchar
nrub_cod varchar
nnem_cod varchar
nnru_valor decima
susu_cod_ua varchar
nnru_fech_ua date
susu_cod_um varchar
nnru_fech_um date
nnru_mo_oc varchar
sis_usuario
susu_codigo varchar(12) <pk>
gper_codigo varchar(12) <fk>
susu_nusuario varchar(30)
susu_contrasenia varchar(30)
susu_codigo_ua varchar(12)
susu_fecha_a date
susu_codigo_um varchar(12)
susu_fecha_um date
susu_mo_oc varchar(10)
nom_tip
ntir_cod varch
ntir_nomb varch
ntir_desc varch
susu_cod_ua varch
ntir_fech_ua date
susu_cod_um varch
ntir_fech_um date
ntir_mo_oc varch
ntir_signo varch
fk_smen_smen
inv_bodega
ibod_cod varchar(12) <pk>
ibod_nomb varchar(100)
ibod_desc varchar(200)
susu_cod_ua varchar(12) <fk1>
ibod_fech_a date
susu_cod_um varchar(12) <fk2>
ibod_fech_m date
ibod_mo_oc varchar(10)
gen_persona
gper_codigo varchar(12) <pk>
gper_nombre varchar(100)
gper_direccion varchar(300)
gper_telefono_1 varchar(15)
gper_telefono_2 varchar(15)
gper_movil varchar(15)
gper_identificacion varchar(15)
susu_codigo_ua varchar(12) <fk1>
gper_fecha_ua date
susu_codigo_um varchar(12) <fk2>
gper_fecha_um date
gper_mo_oc varchar(10)
gper_img_grand longblob
gper_img_peq longblob
gper_img_recar varchar(3)
gper_apellido varchar(100)
gper_cargo varchar(100)
gper_sueldo decimal(10,2)
gper_fech_ingre date
gper_referenci varchar(200)
gper_fech_nacimi date
gper_carg_famili int(10)
inv_producto
ipro_cod varchar(12) <pk>
ipro_cod_us varchar(30)
ibod_cod varchar(12) <fk1>
ipro_desc varchar(100)
ipro_p_prim double(20,4)
ipro_u_com double(20,4)
ipro_iva varchar(5)
ipro_p_iva double(20,4)
ipro_mo_oc varchar(10)
susu_cod_ua varchar(12) <fk2>
ipro_fech_a date
susu_cod_um varchar(12) <fk3>
ipro_fech_m date
inv_movi_cab
imoc_cod varchar(12) <pk>
ipro_cod varchar(12) <fk1>
imoc_exist int(11)
susu_cod_ua varchar(12) <fk2>
imoc_fech_a date
susu_cod_um varchar(12) <fk3>
imoc_fech_m date
fk_imoc_ipro
fk_sem_gper
sis_empresa
sem_cod varchar(12) <pk>
sem_nomb varchar(100)
sem_desc varchar(200)
sem_direcc varchar(150)
sem_telf varchar(50)
sem_represen varchar(100)
sem_cod_inter varchar(5)
gper_contact varchar(12) <fk1>
susu_cod_ua varchar(12) <fk2>
sem_fech_a date
susu_cod_um varchar(12) <fk3>
sem_fech_um date
sem_mo_oc varchar(10)
sis_menu
smen_cod varchar(12) <pk>
smen_nomb varchar(100)
smen_desc varchar(200)
smen_cod_men varchar(12) <fk1>
susu_cod_ua varchar(12) <fk2>
smen_fech_a date
susu_cod_um varchar(12) <fk3>
smen_fech_um date
smen_accion varchar(100)
smen_mo_oc varchar(10)
sis_permisos
sper_cod varchar(12) <pk>
susu_cod_per varchar(12) <fk2>
smen_cod varchar(12) <fk1>
sper_permi varchar(15)
susu_cod_ua varchar(12) <fk3>
sper_fech_ua date
susu_cod_um varchar(12) <fk4>
sper_fech_um date
sper_mo_oc varchar(10)
fk_sper_smen
fk_nnem_gper
nom_n_empleado
nnem_cod varchar(12) <pk>
gper_codigo varchar(12) <fk1>
nnno_cod varchar(12) <fk2>
susu_cod_ua varchar(12) <fk3>
nnem_fech_ua date
susu_cod_um varchar(12) <fk4>
nnem_fech_um date
nnem_mo_oc varchar(15)

Anexo B – Diagrama Lógico



BIOGRAFIA

Jonathan Patricio Cárdenas Ruperti nació en Esmeraldas el 15 de Mayo del 1988 realizando sus estudios primarios en la Escuela Fiscomicional Mixta "**Cristo Rey**" de Esmeraldas, estudios secundarios en el Colegio Fiscomicional Mixto "**Sagrado Corazón** " de Esmeraldas obteniendo el título de Técnico en Comercio y Administración Especialización informática en Enero del 2006 y sus estudios superiores en Quito en la Escuela Politécnica del Ejercito "**ESPE**" obteniendo el título de Suficiencia en el Idioma Inglés en el 2010 y actualmente en culminación de su proyecto de tesis.

HOJA DE LEGALIZACIÓN DE FIRMAS

ELABORADO POR

_____

Jonathan Cárdenas Ruperti

COORDINADOR DE LA CARRERA

_____

Ing. Mauricio Campaña

Lugar y fecha: Sangolquí, 30 de Enero de 2013