Sistema de gestión de información para el proceso de pago por resultado en la división Norte de la empresa COPEXTEL

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores

Alberto Martínez Oquendo

Juan Carlos Casadesús Rades

Tutores

M.Sc Angel Alberto Vazquez Sánchez

Ing. Lizandra Hernández Hernández

Consultante

Ing. Lester Oquendo Céspedes

¨Año 57 de la Revolución¨

La Habana, junio del 2015

Declaración de autoría

Declaramos ser los únicos autores de la presente tesis “Sistema de apoyo al proceso de pago por resultado en la empresa COPEXTEL” y concedemos a la Universidad de las Ciencias Informáticas y a la empresa COPEXTEL los derechos para el uso de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

_____ _____

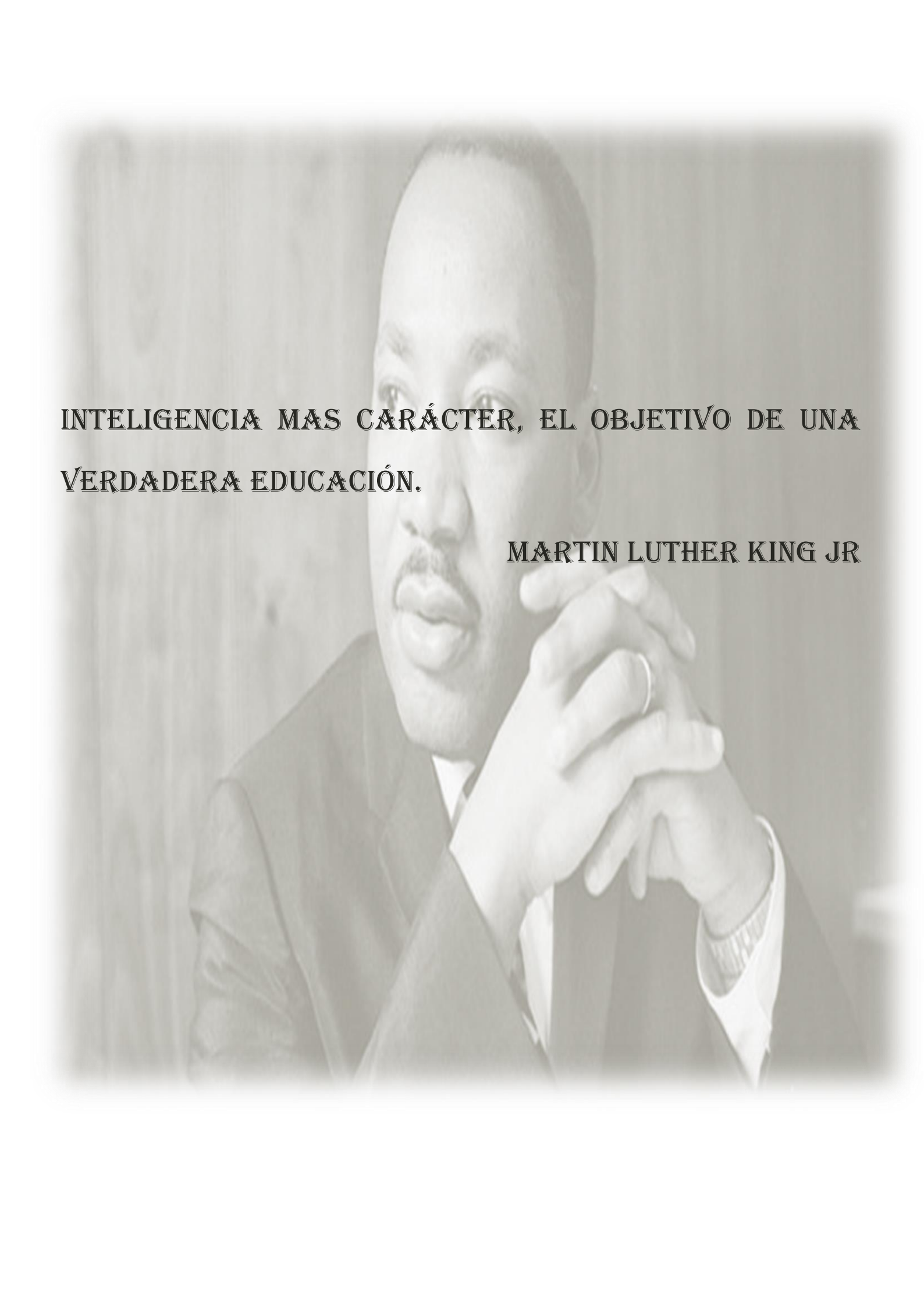
Firma del autor
Alberto Martínez Oquendo

Firma del autor
Juan Carlos Casadesús Rades

_____ _____

Firma del tutor
MCs. Angel Alberto Vazquez Sánchez

Firma del tutor
Ing. Lizandra Hernández Hernández



INTELIGENCIA MAS CARÁCTER, EL OBJETIVO DE UNA VERDADERA EDUCACIÓN.

MARTIN LUTHER KING JR

De Alberto

A mis padres, por haberme educado y apoyado en todos mis momentos, haciéndome la persona que soy hoy.

De Juan Carlos

A mi querida abuela donde quiera que esté, a mis padres y mi hermana por ser mi razón de vivir.

De Alberto

Ante todo quiero agradecer a mi mamá, que siempre me ha dado su cariño y su amor, a mi papá, que aún en tiempos difíciles confió en mí y me dio su apoyo para poder seguir mi carrera, a ambos por haberme guiado durante todo este tiempo, por haber confiado en mí, por su apoyo incondicional en todos los momentos que he atravesado en mi vida y sobre todo por ser luz y guía de mi futuro.

A mi abuelita Julia, que aunque ya no está físicamente, siempre me brindó su cariño y apoyo. Siempre la llevaré en mi corazón.

A mi hermana, que aún estudia, espero que le sirva de ejemplo para impulsar su carrera universitaria.

A todas mis tías, tíos y primos, que fueron de gran ayuda para mí todos estos años.

A mi abuela Clara, que siempre se preocupa por mí.

A mis amigos, esas personas que tengo definida como la familia que uno escoge, gracias por siempre estar ahí, por compartir conmigo todos esos momentos, los buenos y los malos, gracias por todo.

A Rosalía, por haberme ayudado en todos estos años de mi universidad, gracias.

A mi compañero de tesis, Juan Carlos, por ser un amigo incondicional.

A todas las personas que conocí en esta casa de altos estudios, a los que llegaron al final del camino y a los que ya no están, agradezco a las personas que de una forma u otra contribuyeron a mi formación como mejor persona.

De Juan Carlos

A mi abuela Delia, por su cariño excepcional y dedicación entera no solo a mí, sino a la familia completa.

A mi madre por ser mi luz en el camino de la vida, a mi papá por el ejemplo y la virtud, a ambos por tanto esfuerzo, sacrificio y amor.

A mi querida hermana por ser la bujía de la familia, y por ser tan especial.

A mi amada tía Mayelín, por ser madre, tía, hermana y de todo un poco.

A toda mi familia que bien abundante es, pero en especial a mis tios Pedro, y Osmar, por tanto cariño.

A mis hermanos del barrio, aunque ya andamos por caminos separados, nuestras mentes y corazones siguen juntos.

A mis amigos de la UCI, han sido 5 años especiales para mi compartiendo todo con ustedes.

A mi país por darme la posibilidad de materializar un sueño como este.

A la UCI, y a la facultad 4, por contribuir a mi formación como profesional.

A mi compañero de tesis, por tanto esfuerzo y por ser un buen amigo.

Resumen

En las empresas cubanas se han aplicado múltiples sistemas de pago a partir de los resultados alcanzados en la producción o los servicios, transformándose en sistemas de pago por resultados a los trabajadores. En el año 2014, el Ministerio de Trabajo y Seguridad Social, emite la Resolución 17, que establece nuevos términos de pago en cuanto al desempeño de los trabajadores. La División Norte de la empresa COPEXTEL, en aras de aplicar la resolución a cada unidad organizativa, realizó un análisis para establecer este sistema de pago a partir de indicadores de eficiencia, determinando varias dificultades en el proceso de pago. El propósito de la presente investigación se enfoca en realizar un sistema informático para la gestión de la información, el control y la evaluación de los trabajadores de dicha empresa. Para la implementación del sistema se utilizó el lenguaje PHP en su versión 5.5.9, donde se integraron los *framework*s JQuery 1.10.1, Bootstrap 2.3.1 y Symfony 2.3.1. Se utilizó como entorno de desarrollo NetBeans 7.4, como herramienta de modelado Visual Paradigm para UML 8.0 y la metodología XP para el proceso de desarrollo de software. Finalmente, se logró un sistema de gestión de información capaz de llevar a cabo el proceso de pago por resultado en la división Norte, a partir de los indicadores establecidos por la empresa COPEXTEL.

Palabras clave: pago por resultado, sistema de gestión de información, sistema de pago.

Índice

Índice de Tablas

Índice de figuras

Introducción

En el mundo, las Tecnologías de la Información y las Comunicaciones (TICs) se han convertido en parte importante de la cotidianidad. Las empresas y entidades utilizan las TICs como un nuevo canal de difusión de los productos y servicios que ofrecen en su entorno organizacional. El empleo de las TICs facilita la gestión de los recursos humanos y materiales, desde los inventarios de un producto hasta la forma de pago que efectúan las entidades.

Desde la década del 90, en Cuba se han aplicado múltiples sistemas de pago y de estimulación a partir de los resultados alcanzados por las empresas, apoyados en resoluciones ministeriales. Estas resoluciones son emitidas por el Ministerio de Trabajo y Seguridad Social (MTSC), organismo que propone, controla y dirige la política del Estado y el Gobierno en materia de trabajo, protección y seguridad social. Las formas de pago establecidas hasta hoy han logrado potenciar el factor motivación en los trabajadores, incrementando la productividad del trabajo en las empresas.

En el año 2014, a partir de la disposición del país de establecer un sistema de pago con el objetivo de aumentar factores como la productividad y la rentabilidad en las empresas, el MTSC emite la Resolución 17. La misma establece nuevos términos de pago en cuanto al desempeño de los trabajadores así como la productividad de la empresa en general. Con esta resolución se pretende impulsar a todas las empresas del país a adoptar el pago por resultado e incrementar el nivel de ingresos de los trabajadores en correspondencia con su aporte individual y colectivo, con vista al mejoramiento y perfeccionamiento empresarial.

La empresa COPEXTEL S.A, brinda servicios técnicos especializados a equipos de computación, gastronomía hotelera y electrodomésticos. En la actualidad se distingue por ser un proveedor de soluciones integrales, que ofrece productos y servicios ingenieros en una variada gama de esferas o sectores de producción nacional. La infraestructura organizacional de la división Norte de la empresa está compuesta por una gerencia general, a la que se subordinan las direcciones de economía y finanzas y la técnica comercial, esta última a su vez cuenta con varias unidades organizativas: clima, electrodomésticos y ofimática (**Anexo I**). En aras de aplicar la Resolución 17 a cada unidad organizativa, se realizó un análisis para establecer este sistema de pago a partir de indicadores de eficiencia aplicados a su subsistema organizacional, al identificar y adaptar los mismos a la división Norte en el marco de la legislación emitida.

Actualmente en la División Norte de la empresa la información está descentralizada, para establecer el pago dentro de la división, el gerente consulta el Hércules y el Arcas, sistemas informáticos rectores de la empresa COPEXTEL. El primer sistema mencionado, contiene toda la información referente a las hojas de servicios técnicos, reporta la productividad de los trabajadores de acuerdo con los resultados obtenidos. El segundo sistema, es de alcance nacional, contiene toda la información referente a la productividad de las divisiones y de los trabajadores de las sucursales de COPEXTEL en el país, por ejemplo el plan de ventas totales acumuladas, el cumplimiento del plan de ventas

totales acumuladas, gasto de salario real acumulado, entre otros. Esto constituye una limitante por el tiempo que se pierde en la búsqueda y exportación de los datos existentes en los sistemas antes mencionados, para poder establecer el pago. Como consecuencia el trabajo se vuelve lento, lo que provoca un mayor consumo de tiempo y riesgo de cometer errores. Además, los trabajadores, no cuentan con un espacio para llevar el registro del cumplimiento de las actividades establecidas en su plan de trabajo y que tributan a su evaluación mensual.

Por otra parte, el proceso de pago por resultado se realiza de forma manual, con el apoyo de herramientas ofimáticas tales como Microsoft Office con los programas Excel, Outlook. Además, se torna compleja la posibilidad de realizar cambios en la evaluación de los trabajadores y que sean del conocimiento inmediato de todos los involucrados.

A partir de los argumentos antes expuestos se define como **problema a resolver:** ¿Cómo contribuir a la gestión de la información del proceso de pago por resultado en la división Norte de la empresa COPEXTEL?

Para solucionar el problema científico se trazó como **objetivo general:** desarrollar un sistema para la gestión de la información del proceso de pago por resultado que se realiza en la división Norte de la empresa COPEXTEL.

La presente investigación enfocó su **objeto de estudio** en sistemas informáticos dentro del proceso de pago por resultado enmarcado en el **campo de acción**: proceso de pago por resultado a los trabajadores de la División Norte en la empresa COPEXTEL.

Para dar solución al objetivo general se definieron los siguientes **objetivos específicos**:

- ✓ Realizar un estudio del estado del arte referente a sistemas de apoyo al pago en empresas en el mundo, en Cuba y dentro de la empresa COPEXTEL.
- ✓ Seleccionar la metodología, herramientas y tecnologías a utilizar para el desarrollo del sistema.
- ✓ Implementar el sistema de acuerdo con la estructura de diseño definida.
- ✓ Efectuar pruebas para garantizar la funcionalidad del sistema.

Por todo lo antes expuesto se plantea la siguiente **hipótesis**: con el desarrollo de la solución propuesta se disminuirá el tiempo del proceso de pago por resultado en la división Norte de la empresa COPEXTEL.

Para dar cumplimiento al objetivo general se definieron las siguientes **tareas de investigación**:

- ✓ Caracterización de la resolución del pago por resultado.
- ✓ Análisis del reglamento de la empresa COPEXTEL.
- ✓ Realización de un estudio del estado del arte del proceso de pago por resultado en Cuba.
- ✓ Selección de la metodología, herramientas y tecnologías a utilizar.
- ✓ Descripción de los artefactos generados en el proceso de desarrollo de la solución.
- ✓ Implementación de la herramienta.
- ✓ Validación de la aplicación mediante las pruebas.

Como **posible resultado** se espera obtener un sistema informático que contribuya a la gestión del proceso de pago por resultado que se lleva a cabo dentro de la división Norte perteneciente a la empresa COPEXTEL, y que aporte a los trabajadores información acerca de cómo se efectúa el pago dentro de la empresa, así como el historial de pago de los mismos de acuerdo a sus resultados.

Métodos Científicos de Investigación

Los métodos de investigación científica constituyen el modo de abordar la realidad, de estudiar los fenómenos de la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir la esencia de los mismos y sus relaciones. Se definen como la estructura del proceso de Investigación Científica para enriquecer la ciencia. (1)

Los métodos teóricos que se emplean en la presente investigación son:

Histórico-lógico: se utiliza para realizar un análisis de la documentación relacionada con el proceso de gestión y control del pago por resultado en la división Norte de la empresa COPEXTEL.

Analítico-Sintético: se estudian las diferentes documentaciones y bibliografías especializadas del tema de la resolución de pago por resultado, además de los documentos que contribuyen al entendimiento del proceso de pago y el reglamento de la empresa, con el propósito de extraer los elementos fundamentales relacionados con el objeto de estudio.

Encuesta: para verificar la aceptación de los usuarios que utilizarán el sistema a desarrollar.

Como métodos empíricos utilizados para cumplir con las tareas se emplean:

Observación: se utiliza para observar el funcionamiento del proceso de pago dentro de la división Norte de la empresa COPEXTEL, así como las situaciones problemáticas relacionadas con el mismo y la gestión de la información que se genera en este proceso. **Anexo II**

Entrevista: se aplicó para obtener información, opiniones y criterios de expertos, especialistas y directivos de la división. También, se establecieron reuniones con los administradores de los sistemas de gestión de resultados de la empresa COPEXTEL y se tuvieron en cuenta las sugerencias de los usuarios finales con respecto a las posibilidades reales sobre la asimilación de un nuevo sistema para el pago a los trabajadores. **Anexo III**

La estructura de la presente investigación está conformada como se plantea a continuación:

CAPITULO 1:"Fundamentación teórica"

En este capítulo se exponen los conceptos asociados al dominio del problema, se realiza un estudio de las soluciones similares y se selecciona la metodología, herramientas y tecnologías para el desarrollo del sistema.

CAPITULO 2: "Descripción y análisis de la propuesta de solución"

En este capítulo se describe la solución propuesta mediante los artefactos ingenieriles generados de acuerdo a la metodología seleccionada, se determinan los requisitos funcionales y no funcionales y se detalla la arquitectura propuesta. Además, se realiza un análisis de la implementación de la aplicación.

CAPITULO 3: “Implementación y pruebas”

Este capítulo contiene todo lo relacionado con la implementación del sistema y el proceso de pruebas. Se implementan todas las funcionalidades definidas con anterioridad y se detallan las pruebas realizadas al sistema una vez desarrollado, con el objetivo de asegurar la calidad y eficiencia de la solución.

Capítulo 1 Fundamentación teórica

Introducción

En este capítulo se abordan conceptos asociados a los sistemas de pago por desempeño y lo relacionado con estos para lograr un mejor entendimiento de la propuesta de solución. Se describen las soluciones similares existentes en el mundo y en Cuba que faciliten funcionalidades que se desean optimizar, y nuevas que se puedan implementar. Se seleccionan la metodología y herramientas de desarrollo para la implementación de la propuesta de solución.

1.1 Conceptos asociados al dominio del problema

Sistema de gestión de Información

Davis y Olson (1985) en (2) conceptualizan los Sistemas de Gestión de la Información (SGI) como: Sistema integrado y automatizado para proveer la información que sostenga las funciones de operatividad, gestión y toma de decisiones en una organización.

Según plantea la Universidad de California, un SGI es la aplicación de las tecnologías de la información para apoyar las principales funciones y actividades de una empresa, ya sea del sector privado o institución del sector público. (3)

La investigadora Lic. Lourdes Aja Quiroga, define en (4) la gestión de información como un aspecto esencial de la infraestructura para la gestión del conocimiento, esta aporta información con el objetivo de encontrar soluciones a los problemas que enfrentan las organizaciones. Permite analizar el impacto de los resultados alcanzados y manejar el comportamiento de los individuos ante la información. A partir de la bibliografía consultada y los conceptos referenciados anteriormente, los autores de la presente investigación emiten el siguiente concepto:

Un SGI gestiona los recursos de información tanto internos como externos de las organizaciones. Su finalidad es generar servicios que respondan a las necesidades de los usuarios. Estos aprovechan al máximo los recursos de la información en función de la mejora continua de las empresas, y de la toma de decisiones a todos los niveles de la misma. Analiza las repercusiones de los resultados alcanzados.

Sistema de pago: conjunto de instrumentos, procesos y canales de transferencia de fondos entre los distintos individuos, necesario para el desarrollo de la actividad económica. Los sistemas de pago deben permitir la realización segura y rápida de todo tipo de transacciones. (5)

Pago por desempeño: herramienta para lograr mayores incrementos en la productividad, eficiencia y calidad de las empresas. En muchas ocasiones se le entiende como un fin en sí mismo y no un medio. El sistema de pago por los resultados busca que el salario se incremente a partir de la producción obtenida por una entidad o un trabajador. (6)

Entre las ventajas del pago por desempeño se encuentran (7):

- ✓ **Establece pautas para los sistemas de pago:** responde a las preguntas relacionadas con el tipo de aumento de salario, las bonificaciones o los otros incentivos que puede recibir un empleado por haber superado las expectativas.
- ✓ **Consistencia:** los empleados reciben aumentos de salario, bonificaciones y beneficios adicionales por haber alcanzado los objetivos de desempeño. Para que un empleado pueda recibir el incremento base de salario, debe cumplir con las expectativas de rendimiento del empleador. Si las supera, tendrá mayores posibilidades de recibir incentivos monetarios adicionales. La consistencia es fundamental para los sistemas de este tipo.
- ✓ **Incentivo:** tanto los empleados como los empleadores se benefician de un sistema de pago por desempeño que ofrece incentivos de acuerdo con el rendimiento. Los empleados luchan por obtener mejores resultados, por lo que trabajan con dedicación.
- ✓ **Gestión del pago por resultado:** se define como el proceso que lleva a la remuneración de la estimulación a los trabajadores de acuerdo con el cumplimiento de sus tareas definidas en su plan de trabajo. (8)

Después de haber analizado los conceptos referentes al sistema de pago, se realiza un estudio del significado de las variables asociadas al pago por desempeño, las cuales se detallan a continuación:

Monto de salario máximo (MSM): suma que es posible pagar sin causar deterioro del Gasto de Salario por peso de Valor Agregado Bruto (GS/VAB), planificado a nivel de empresa y por cada unidad organizativa. Este incluye el salario por tiempo real trabajado de los trabajadores abarcados. (9)

Monto de salario por tiempo real trabajado (STRT): suma de los salarios correspondientes a cada grupo de la escala, más todos los pagos adicionales aprobados para la empresa que constituyan salario al que se le deducen las ausencias, interrupciones laborales e infracciones de la jornada laboral de cada trabajador en el período. (9)

Monto de Salario Base de Cálculo (SBC): sumatoria de los STRT de los trabajadores abarcados al que se le aplican los porcientos de penalización por la Evaluación del Desempeño. Cuando el salario a distribuir es igual o menor que el STRT, este último pasa a ser el SBC a los efectos de la distribución del salario. (9)

Monto de salario a otorgar por encima del SBC: diferencia entre el MSM y el SBC. Se otorga el derecho a obtener el total de este monto cuando se cumplen los indicadores directivos y los indicadores específicos. (9)

Salario a distribuir: está conformado por el SBC y el sueldo formado por encima de este, al que se le deduce el importe correspondiente a las vacaciones. (9)

Salario mínimo(SM): corresponde al Grupo I de la escala salarial vigente en el país cuyo monto es de 225 pesos. (9)

Indicadores directivos: parámetros de evaluación aprobados en el plan de la economía que tributan al resultado final de la empresa. (9)

Indicadores específicos: parámetros propios de cada unidad organizativa. (9)

Indicadores individuales: parámetros que caracterizan el aporte individual de cada trabajador al resultado del área o grupo de trabajo. (9)

Partiendo de todos los conceptos estudiados anteriormente, se realizó un estudio de las soluciones similares a la propuesta de solución.

1.2 Sistemas similares

Con el propósito de mantener un control de los indicadores que deben cumplir en las empresas los trabajadores respecto a sus planes de trabajo. Surge la necesidad de incrementar el desarrollo de sistemas informáticos capaces de cumplir con este objetivo, para de esta forma tener un punto de partida al otorgar las evaluaciones correspondientes. A continuación se realiza el análisis de sistemas similares, a partir de los siguientes parámetros:

- ✓ Funcionalidades que realizan.
- ✓ Código abierto.
- ✓ Aplicación en la actualidad.
- ✓ Beneficios y desventajas.
- ✓ Tecnologías que utilizan.

Internacional

Sistema de Gestión de Pagos

El Sistema de Gestión de Pagos (SGP) permite de manera fácil y segura la gestión del pago de anticipos, honorarios, reembolsos, etc., sin la intervención de un tercero. Pertenece a la empresa *Pendulum Associates* residente en México y brinda varios beneficios a las empresas (10):

- ✓ **Reducción de errores fiscales y de importes:** el sistema se encuentra diseñado para orientar en estos rubros.
- ✓ **Reducción de tiempos en el pago de facturas o recibos de honorarios:** se evitará que los documentos sean traspapelados o se pierdan.
- ✓ **Conocer el estado o estatus que tiene cada una de las solicitudes:** se puede ingresar al sistema las veces que el usuario lo requiera y así revisar el avance en el proceso de pago.

Entre sus principales funcionalidades se encuentran:

- ✓ Honorarios e incentivos pre autorizados para cobro.
- ✓ Solicitud de anticipos.
- ✓ Solicitud de reembolso.
- ✓ Listar solicitudes de cobro.
- ✓ Consultar reporteador legal.

En los términos y condiciones de uso de este sistema se establece, "…el diseño, contenido, funciones y en general todo lo relativo al diseño y configuración de este sistema, así como cualquier anexo o documento que se imprima desde el mismo, son propiedad intelectual y material exclusiva de Pendulum, S. de R.L de C.V., y de las empresas cuyas carteras administre…". (10)

El sitio oficial de la empresa antes mencionada, no brinda información acerca del diseño e implementación de los sistemas de gestión que esta posee. Motivo por el cual en la presente investigación, no se pudo tener en cuenta características del software como: lenguaje de programación, *framework* de desarrollo, herramientas y tecnologías utilizadas.

Nacionales

Sistema para la gestión de pagos en la Empresa Gráfica ¨Juan Marinello ¨de Guantánamo

Este sistema permite la gestión los modelos de sistemas de pago en el Departamento de Recursos Humanos de la Empresa Gráfica "Juan Marinello" de Guantánamo. Presenta cambios en la forma de realizar los diferentes procedimientos a ejecutar, minimiza el tiempo para decidir y utilizar la información que responde a sus necesidades y exigencias. Tiene implementadas varias funcionalidades, entre las que se encuentran (8):

- ✓ Gestionar datos de los sistemas de pagos.
- ✓ Gestionar trabajador.
- ✓ Gestionar sistema de pagos.
- ✓ Generar reporte de pago adicional.
- ✓ Generar reporte de sistemas de pagos.

Sistemas de pago por resultados de la Empresa Avícola Santa Clara (PAYSAX)

El sistema **PAYSAX** es una herramienta automatizada desarrollada por la empresa Desoft de Villa Clara, para la gestión del pago por resultado en la empresa avícola de Santa Clara. Ha sido diseñado mediante el uso de las tecnologías vigentes de desarrollo en software libre de forma tal que constituye una herramienta multi-usuario Cliente-Servidor. Cuenta con dos módulos fundamentales: Ponedora y Reproductor, y tiene como objetivos (11):

- ✓ Gestionar los ciclos productivos según los propósitos para su posterior pago por resultado.
- ✓ Gestionar la producción y asistencia de los trabajadores.
- ✓ Generar los reportes de pago por resultados e indicadores del Ciclo Productivo de las naves.

Entre sus principales funcionalidades destacan:

- ✓ Gestionar registro de asistencia.
- ✓ Gestionar indicador productivo.
- ✓ Gestionar evaluación de producción.

Para el diseño de esta aplicación, se utilizaron tecnologías Web, con Programación Orientada a Objetos (POO), se utiliza el estándar de codificación del lenguaje PHP, Zend Framework como *framework* de desarrollo bajo los patrones de diseño: Modelo–Vista–Controlador (MVC) y Data Access Object en español Objeto de Acceso a Datos (DAO).Además, emplea Propel para el mapeo y consulta de las tablas de la base de datos(BD), MySQL como gestor de bases de datos y la biblioteca ExtJS en su versión 3.3.0. (11) A continuación se muestra una tabla de los sistemas similares, a través de parámetros que establecen puntos de comparación entre estos.

Tabla 1: Comparación de los sistemas similares a la solución propuesta

Parámetros	SGP	PAYSAX	SGP Guantánamo
Ajuste a la problemática de la presente investigación	no	no	no
Privativo	sí	sí	sí
Utilización de tecnologías libres	no	sí	sí
Código abierto	no	no	no
Aplicación en la actualidad	sí	no	no

El análisis de las características de los sistemas estudiados permitió obtener funcionalidades útiles para la propuesta de solución. Ejemplo es el **Sistema para la gestión de pagos en la empresa Juan Marinello de Guantánamo,** donde las funcionalidades Gestionar trabajador, Gestionar sistema de pagos, Generar reportes y dentro de este último: Reporte de pago adicional y Reporte sistemas de pagos, generaron el interés por parte de los autores de la presente investigación para ser incorporados en la solución propuesta por su similitud con algunos de los requisitos del sistema a implementar. El resto, no aportó elementos que puedan adaptarse o reutilizarse en la presente propuesta de solución, debido a que la lógica de su negocio no se corresponde con las pautas que plantea la empresa COPEXTEL para efectuar el pago, basándose en el marco de la resolución 17 del MTSC. Además son privativos, por lo que se dificulta la reutilización de alguno de sus módulos.

1.3 Metodología, Herramientas y Tecnologías

1.3.1 Metodología de desarrollo

Las metodologías de desarrollo de software, son un conjunto de procedimientos y técnicas que ofrecen un enfoque estructurado para el proceso de desarrollo que incluye modelos de sistemas, notaciones, reglas, sugerencias de diseño y guías de procesos. Estas indican las actividades a realizar para lograr el producto informático deseado. Además, muestran qué personas deben participar en el desarrollo de las actividades y qué rol desempeñarán. También describen la información que se debe producir como resultado de una actividad y la que se necesita para comenzarla. (12)

En la actualidad se han erigido dos corrientes en cuanto a metodologías de desarrollo respecta, las llamadas metodologías ágiles y las pesadas. La diferencia fundamental entre ambas es que mientras las robustas llegan a su objetivo mediante el orden y la documentación excesiva, las ligeras intentan mejorar la calidad del software a través de la comunicación directa e inmediata de las personas que intervienen en el proceso.

Existen varias metodologías de desarrollo de software, entre las que se encuentran: Proceso Unificado de Desarrollo de Software (RUP), SCRUM y Programación Extrema (XP), estas constituyen metodologías punteras en el desarrollo de software, por lo cual serán estudiadas en el presente epígrafe.

Los siguientes parámetros fueron definidos por los autores de la presente investigación para ayudar a la selección de la metodología a utilizar:

- ✓ Experiencia del equipo de desarrollo con la metodología a utilizar.
- ✓ Necesidad de una constante retroalimentación entre el cliente y el equipo de desarrollo.
- ✓ Idoneidad para equipos de desarrollo pequeños.
- ✓ Basta documentación para comprender todo el proceso de desarrollo.
- ✓ Apropiada para guiar el proceso de desarrollo de un proyecto pequeño.
- ✓ Flexibilidad ante los cambios.

Proceso Unificado de Desarrollo (RUP)

RUP es una de las metodologías de desarrollo de software más utilizada en el mundo para el análisis, implementación y documentación de sistemas orientados a objetos. Durante todo el ciclo de desarrollo se administran los requisitos, con el objetivo de mantener la trazabilidad de los requerimientos funcionales, siendo así el proceso de desarrollo guiado por casos de uso. RUP permite la reutilización de estos requerimientos para otros sistemas o funcionalidades del propio software. Es centrado en la arquitectura, por lo que es fundamental identificar los casos de uso de mayor prioridad e importancia. (13)

Este tipo de metodología se caracteriza fundamentalmente en realizar un mayor énfasis en la planificación y control del proyecto que se desee realizar, además de especificar de forma precisa los requisitos y el modelado de los flujos de trabajo que se planteen. Sin embargo, presenta algunas desventajas en torno al marco del proyecto que se desea realizar tales como:

- ✓ Presenta una gran multitud de artefactos, lo cual consume mucho tiempo.
- ✓ Se cuenta con poco personal para el desarrollo, debido a que solo dos personas están a cargo del desarrollo del sistema.
- ✓ No presenta flexibilidad ante el cambio constante, ya que de aparecer un nuevo requisito o de ocurrir un cambio en alguno de ellos hay que comenzar una nueva iteración para dar cumplimiento a la funcionalidad.

SCRUM

SCRUM es un marco de trabajo en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente en equipo y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección se originó a partir de un estudio de la manera de trabajar de equipos altamente productivos. (14)

Más que una metodología de desarrollo software, es una forma de autogestión de los equipos de programadores; los mismos deciden cómo hacer sus tareas y cuánto van a tardar en ello. SCRUM ayuda a que todos trabajen juntos en la misma dirección y con un objetivo claro. Permite seguir el avance de las tareas a realizar, de forma que los jefes puedan ver cómo progresa el trabajo. (15)

En esta metodología se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto.

Programación extrema o eXtreme Programming (XP)

Es la más destacada de los procesos ágiles de desarrollo de software. Los programadores que utilizan XP consideran que los cambios en los requisitos durante el desarrollo del software son un aspecto natural e inevitable. Algunas de las ventajas que presenta XP es que permite la retroalimentación, la presión está a lo largo de todo el proceso y no en una entrega final, permite definir en cada iteración cuales son los objetivos de la siguiente y es apropiada para entornos volátiles. Por ello, está especialmente indicada para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Tabla 2: Comparación de metodologías

Parámetros para la selección de la metodología	XP	SCRUM	RUP
Experiencia del equipo de desarrollo con la metodología a utilizar	alta	baja	alta
Necesidad de una constante retroalimentación entre el cliente y el equipo de desarrollo	sí	sí	no
Idoneidad para equipos de desarrollo pequeños	sí	sí	no
Propicia para agilizar el proceso de desarrollo de software	sí	no	sí
Capacidad adaptativa al cambio	sí	sí	no

A partir de lo expuesto anteriormente, se selecciona la metodología XP debido a que entre sus pilares fundamentales engloba las siguientes características (16):

- ✓ Grupos pequeños y trabajando en el mismo sitio.
- ✓ Existe una comunicación efectiva entre el equipo de desarrollo y el cliente.
- ✓ La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software.
- ✓ Capacidad adaptativa al cambio.
- ✓ Los diseños del software son sencillos y libres de complejidad.
- ✓ Los cambios se implementan rápidamente tal y como fueron sugeridos, convirtiéndose en parte sustantiva del proceso.
- ✓ La estructura de roles es adaptable al equipo de desarrollo.
- ✓ Pruebas continuas.

1.3.2 Herramientas CASE

Las Herramientas CASE (*Computer Aided Software Engineering*, en español Ingeniería de Software Asistida por Computadora) se define como un conjunto de programas y ayudas que dan asistencia automatizada a los analistas, ingenieros de software y desarrolladores, durante todo el ciclo de vida de desarrollo de un software.

Para realizar un nuevo software se requiere que las tareas sean organizadas y completadas de forma correcta y eficiente. Las herramientas CASE fueron desarrolladas para automatizar esos procesos y facilitar las tareas de coordinación de los eventos que necesitan ser mejorados en el ciclo de desarrollo de software. (17)

Objetivos de las herramientas CASE (17)

- ✓ Mejorar la productividad en el desarrollo y mantenimiento del software.
- ✓ Aumentar la calidad del software.
- ✓ Mejorar el tiempo, costo de desarrollo y mantenimiento de los sistemas informáticos.
- ✓ Mejorar la planificación de un proyecto.
- ✓ Aumentar la biblioteca de conocimiento informático de una empresa y ayudar en la búsqueda de soluciones para los requisitos.
- ✓ Ayudar en la reutilización del software, portabilidad y estandarización de la documentación.
- ✓ Facilitar la gestión global en todas las fases de desarrollo de software con una misma herramienta.
- ✓ Facilitar el uso de las distintas metodologías propias de la ingeniería del software.

Para la selección de las herramientas y tecnologías a utilizar en el desarrollo de la propuesta de solución, partiendo del poco tiempo que existe para la implementación del sistema a desarrollar y la complejidad del mismo, se establecieron los siguientes parámetros:

- ✓ Experiencia del equipo de desarrollo.
- ✓ Apropiada para realizar proyectos pequeños y ligeros.
- ✓ Soporte al ciclo de vida del desarrollo de software.
- ✓ Curva de aprendizaje.
- ✓ Tecnología libre.
- ✓ Plataformas que las soportan.
- ✓ Extensa documentación para el trabajo con las mismas.

Umbrello

Umbrello es una herramienta CASE totalmente concebida como software libre y de código abierto. Libreada bajo licencia GPL[1], se utiliza para la diagramación de UML[2], soporta diferentes diagramas, entre ellos:

- ✓ Diagrama de Clase.
- ✓ Diagrama de Secuencia.

[1] La Licencia Pública General de GNU, en inglés General Public License (GNU GPL): es la licencia más usada en el mundo del software, garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir y modificar el software.

[2] UML (Unified Modeling Language) Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

- ✓ Diagrama de Colaboración.
- ✓ Diagrama de Caso de Uso.
- ✓ Diagrama de Estado.
- ✓ Diagrama de Actividad.
- ✓ Diagrama Componente.
- ✓ Diagrama de Desarrollo.

Umbrello puede importar código fuente de proyectos existentes para ayudar a construir un modelo de sistema, soporta solo código fuente C++. Permite crear un diagrama y generar el código en varios lenguajes de programación ,debido a que cuenta con soporte de generación de código para ActionScript, Ada, C++, CORBA IDL, Java, JavaScript, PHP, Perl, Python, SQL y XMLSchema. (18)

Dia Project

Dia Project es un programa de creación de diagramas basado en GTK +, es multiplataforma y liberado bajo la licencia GPL. Está basado en el programa comercial de Windows 'Visio'. Actualmente tiene objetos especiales para ayudar a realizar diagramas entidad relación, diagramas UML, diagramas de flujo, diagramas de red, entre otros.

También es posible añadir soporte para nuevas formas escribiendo archivos XML simples. Puede cargar y guardar diagramas en un formato XML personalizado, exportar diagramas a un número de formatos, incluyendo EPS, SVG, xfig, WMF y PNG, e imprimir diagramas incluyendo los de múltiples páginas. (19)

Visual Paradigm para UML

Entre las herramientas CASE más usadas actualmente se encuentra Visual Paradigm. Es una herramienta UML profesional considerada en el mercado mundial un estándar en el desarrollo del software. Visual Paradigm es multiplataforma y proporciona excelentes facilidades de interoperabilidad con otras aplicaciones y soporta el ciclo de vida completo del desarrollo de software. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad a un menor costo. Tiene disponibilidad de múltiples versiones para cada necesidad y disponibilidad de integrarse en los principales IDEs. Apoya los estándares más recientes de las notaciones de Java y de UML. Además, Incorpora el soporte para trabajo en equipo, que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros. (20)

Tabla 3: Comparación de las herramientas CASE

Parámetros para la selección de la herramienta CASE	Umbrello	Dia Project	Visual Paradigm
Experiencia del equipo de desarrollo con la herramienta a utilizar	baja	baja	alta
Apropiada para realizar proyectos pequeños y ligeros	sí	sí	sí
Soporte al ciclo de vida del desarrollo de software	sí	sí	no

Curva de aprendizaje	baja	baja	alta
Tecnología libre	sí	sí	sí
Multiplataforma	no	sí	sí
Documentación para el trabajo con las mismas	media	baja	alta

Fundamentación de la selección

Se escogió como herramienta CASE Visual Paradigm en su versión 8.0, la misma presenta varias ventajas, tales como el rápido manejo de grandes y complicadas estructuras de un proyecto en una forma muy eficiente. Además es un software con una licencia libre, multiplataforma y presenta una gran variedad de documentación para su uso.

1.3.3 Aplicaciones Web

Para elaborar la propuesta de solución se propone el desarrollo de una aplicación web, esto permitirá que se pueda utilizar desde cualquier área de la división Norte de la empresa COPEXTEL con solo una computadora conectada a la red con un navegador web instalado**.**

Uno de los servicios más usados de Internet es WWW (World Wide Web), está definido como un lenguaje que presenta textos, imágenes y sonidos. Además, incluye enlaces hacia otras páginas web o servicios brindados por Internet. A la evolución de este servicio se le conoce como Aplicación Web, que es una interfaz diseñada para cubrir las necesidades de un negocio y gestionar su información. (21)

Las aplicaciones web son populares debido a lo práctico del navegador web como cliente ligero, a la independencia del sistema operativo, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales. Existen aplicaciones como los webmails, wikis, weblogs, tiendas en línea entre otras aplicaciones que son ejemplos bien conocidos de aplicaciones web. (21)

Tim Bernes-Lee se le considera el padre de la WWW. A él se le deben los tres elementos que fueron clave en el nacimiento de la Web:

HTML (HyperText Markup Language), como lenguaje para la creación de los contenidos de la Web.

HTTP (HyperText Transfer Protocol), como protocolo de comunicación entre los ordenadores de la Web.

URL (Uniform Resource Locators), como medio de localización de los distintos recursos en internet.

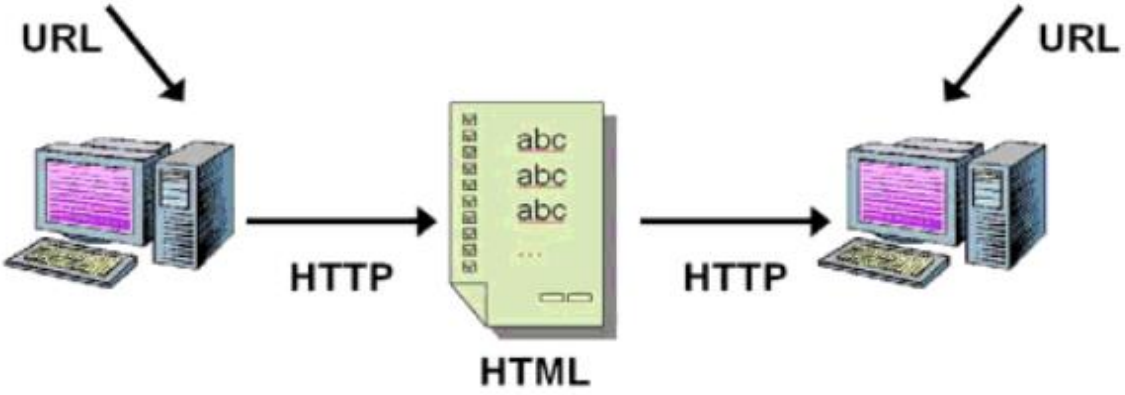


Fig. 1 Pilares de la web

Las aplicaciones Web permiten la generación automática de contenido y la creación de páginas personalizadas según el perfil de usuario. Además, facilitan la interacción con los sistemas de gestión de información de una empresa. Se encuadran dentro de la arquitectura cliente/servidor: el ordenador solicita servicios (el cliente) y otro está a la espera de recibir solicitudes y las responde (el servidor). (21)

Fundamentación de su selección

Las siguientes ventajas de estas aplicaciones avalan su selección:

- ✓ **Ahorro de tiempo**: se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.
- ✓ **No hay problemas de compatibilidad**: basta tener un navegador actualizado para poder utilizarlas.
- ✓ **Consumo de recursos bajo**: dado que gran parte de la aplicación no se encuentra en los ordenadores, muchas de las tareas que realiza el sistema no consumen recursos porque se realizan desde otro ordenador.
- ✓ **Multiplataforma**: se pueden usar desde cualquier sistema operativo porque sólo es necesario tener un navegador.
- ✓ **Portables**: es independiente del ordenador porque se accede a través de una página web, sólo es necesario disponer de acceso a Internet.
- ✓ **La disponibilidad suele ser alta** porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.
- ✓ **Los virus no dañan** los datos porque éstos están guardados en el servidor de la aplicación.

1.3.4 Entorno Integrado de Desarrollo

Los Entorno Integrado de Desarrollo (IDE) son programas informáticos compuestos por un conjunto de herramientas de programación. Son entornos que han sido empaquetados como un programa de aplicación, contienen un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI).

- ✓ **Editor de código fuente:** sirve para editar el código de aplicaciones informáticas.

- ✓ **Un compilador:** es un traductor de código fuente, lo convierte a un lenguaje que sea legible para las máquinas.
- ✓ **Un depurador:** aplicación que tiene como función probar y eliminar posibles errores de un programa en desarrollo.
- ✓ **Constructor de interfaz gráfica:** herramienta que sirve para crear y diseñar las interfaces con las cuales habrá interacción entre la aplicación y el usuario.

A continuación se hace un estudio de algunos IDE**: Eclipse, PHP Storm y el NetBeans,** haciendo alusión a varios parámetros que avalarán su selección:

Multiplataforma.

Código abierto.

Curva de aprendizaje.

Rendimiento.

Eclipse

Es un programa informático compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. La plataforma Eclipse está diseñada para la construcción de IDEs que puedan ser utilizados para la construcción de aplicaciones web, aplicaciones Java de todo tipo, programas C++, y *Enterprise JavaBeans* (EJBs). Gran parte de su funcionalidad es muy genérica, permite que los componentes puedan utilizar nuevos tipos de contenido, para realizar tareas con contenidos existentes. Proporciona un núcleo de componentes de construcción básico. Eclipse fue liberado originalmente bajo la *Common Public License*, pero después fue nuevamente licenciado bajo la *Eclipse Public License*. La *Free Software Foundation* ha dicho que ambas son licencias de software libre, pero son incompatibles con la licencia pública general de GNU (GNU GPL). (38) (39)

PHP Storm

Php Storm es un editor de código inteligente que proporciona completado de código, configuración de proyectos, refactorización, edición avanzada de JavaScript, edición HTML/CSS, posee herramientas de depuración, permite la comprobación de errores durante el desarrollo, etc. Este IDE permite hacer depuraciones de manera fácil y entendible para los usuarios, sobre todo con la validación de la configuración del depurador. (40)

NetBeans

NetBeans es un IDE de código abierto y una plataforma de aplicaciones que le permite a los desarrolladores crear rápidamente páginas Web, está escrito en Java. Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, así como la creación de aplicaciones compatibles con teléfonos móviles.

Está desarrollado en Java y es de código abierto, fácil y portable en plataformas como Unix, Linux y Windows. Además, posee otras características las cuales se enuncian a continuación:

- ✓ Presenta un rendimiento óptimo en tiempo de ejecución y optimización de recursos.
- ✓ NetBeans IDE proporciona diferentes vistas de los datos, de múltiples ventanas del proyecto a herramientas útiles para la creación de las aplicaciones y la gestión de manera eficiente, lo que le permite profundizar en los datos de forma rápida y sencilla.
- ✓ Puede ejecutarse en una gran variedad de sistemas operativos.

Para un mejor entendimiento del estudio de los IDEs, se confeccionó la siguiente tabla a partir de los parámetros establecidos anteriormente:

Tabla 4: Comparación de los IDE

Parámetros	Eclipse	PHP Storm	NetBeans
Multiplataforma	sí	sí	sí
Código abierto	sí	sí	sí
Curva de aprendizaje	baja	media	alta
Rendimiento	alto	alto	alto

La versión 7.3.1 constituye un mejor soporte para las tecnologías Java más recientes, une edición de código de forma rápida e inteligente, una administración de proyectos más eficiente, además de proporcionar un desarrollo rápido de la interfaz de usuario. (41)

1.3.5 Lenguajes de programación

Los lenguajes de programación son definidos como una expresión formal diseñada para expresar procesos. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Un lenguaje de programación actúa como un traductor entre el usuario y el equipo. (22)

1.3.5.1 Lenguajes del lado del cliente

Los lenguajes del lado del cliente son aquellos que son interpretados y ejecutados en el navegador. Son totalmente independientes del servidor y permiten que la página pueda ser albergada en cualquier sitio. Con la programación del lado del cliente se pueden validar algunos de los datos en la máquina cliente antes de enviarlos al servidor. Esto proporciona a los usuarios informes de error inmediatos, mientras siguen en esa página de formulario y sin necesidad de volver atrás tras recibir un mensaje de error. Puede resultar necesario acceder a una base de datos para validar determinados valores, mientras que no suele disponer de un acceso directo a la misma en la máquina del cliente, aunque ese acceso es factible. (23)

HTML5[3]

[3] HTML, siglas de HyperText Markup Language, hace referencia al lenguaje de marcado para la elaboración de páginas web.

Lenguaje de marcación diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas Web. Gracias a Internet y a los navegadores del tipo Internet Explorer, Opera, Firefox o Netscape, el HTML se ha convertido en uno de los formatos más populares que existen para la construcción de documentos y también de los más fáciles de aprender. (24)

1.3.5.2 Lenguaje del lado del servidor

Los lenguajes del lado del servidor son aquellos que se ejecutan en el servidor web, justo antes de que se envíe la página a través de Internet al cliente. Son independientes del navegador por lo que no necesitan de plugins especiales para visualizar correctamente cualquier página. (29)

PHP[4] 5.5.9

Para realizar la implementación del sistema se utilizará el lenguaje PHP, se trata de un tipo de lenguaje de programación de código del lado del servidor, diseñado fundamentalmente para el desarrollo web de contenido dinámico.

Entre los aspectos positivos de PHP a destacar se encuentra que su código fuente es invisible al navegador y al cliente, ya que es el propio servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Es también multiplataforma, sencillo y ofrece una gran variedad de funciones para la explicación de bases de datos sin complicaciones. (25) (26)

La versión de PHP que se utilizó es la 5.5.9, porque posee entre sus características: que es un lenguaje multiplataforma, consume pocos recursos, es fácil de aprender y posee un alto rendimiento.

1.3.6 Frameworks de desarrollo

Los *frameworks* son definidos como un software o conjunto de bibliotecas, que están diseñados para dar soporte al desarrollo de sitios y en general a la construcción de cualquier aplicación web. Tratan de facilitar aquellas actividades comunes realizadas durante el desarrollo de la aplicación, por ejemplo: acceso a la base de datos, uso de plantillas, manejo de sesiones, separación de aspectos de programación; además de promover la reutilización de código. (27)

Algunas de las facilidades que proporciona utilizar un *framework* son:

- ✓ **Seguridad:** permite identificar los usuarios de la aplicación, y restringir el acceso a funciones basadas en algún criterio definido o mediante listas de control de acceso.
- ✓ **Acceso a base de datos:** permite acceder a varios manejadores de bases de datos sin realizar cambios en el código.
- ✓ **Utilización de plantillas:** aplica temas y el uso de variables para aquellas partes dinámicas donde se insertan los datos extraídos de una BD, se puede reducir dramáticamente el número de páginas en un sitio.

1.3.6.1 Frameworks del lado del cliente

JQuery

[4] Hypertext Preprocessor

JQuery es un framework de JavaScript rápido y conciso que simplifica el recorrido de un documento HTML, de manejo de eventos, animaciones e interacciones Ajax para el desarrollo web rápido. Estas características posibilitan que los contenidos se puedan mostrar y se interactúe con los mismos de forma dinámica. Ofrece una infraestructura con la que el usuario posee mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Permite la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de Ajax, etc., tiene licencia para uso en cualquier tipo de plataforma, personal o comercial (28).Implementa una serie de clases (de POO) que permite programar sin preocupaciones del navegador con el que está utilizando el usuario, porque funcionan de forma exacta en las plataformas más habituales. Para el desarrollo del sistema se utilizará JQuery en su versión 1.10.2.

Bootstrap

Bootstrap es un *framework* CSS [5] creado y diseñado por Twitter para facilitar la creación de sitios web. Es un tipo de técnica de remuestreo de datos que ayuda a resolver problemas relacionados con la estimación de los intervalos de confianza o la prueba de significación estadística. Resulta más accesible y simple de comprender. Para la implementación del sistema se escogió Bootstrap en su versión 2.3.1

Entre sus ventajas se encuentran (29):

- ✓ **Mantenimiento y actualización**: gran parte del trabajo interno se realiza por sus desarrolladores.
- ✓ **Ofrece paquetes de elementos web personalizables**: con Bootstrap se puede diseñar una web jugando con elementos compuestos por diferentes combinaciones de HTML, CSS y *Javascript*, de manera que las piezas siempre encajan.
- ✓ **Utiliza componentes vitales para los desarrolladores**: como HTML5, CSS3, jQuery o GitHub, entre otros.
- ✓ **Sus plantillas son de sencilla adaptación *responsive***: se desarrolló para facilitar el proceso de adaptación web a todo tipo de dispositivos.
- ✓ **Se integra con librerías JavaScript.**
- ✓ **Es una herramienta de uso ágil y sencillo**: facilita enormemente el diseño de interfaces y además incluye por defecto una plantilla bastante optimizada.
- ✓ **Contiene tutoriales:** este framework facilita mucha documentación para resolver dudas tanto a principiantes como a desarrolladores expertos.

1.3.6.2 Framework del lado del servidor

Para la selección del framework que se utilizará en la presente propuesta de solución se definieron los siguientes parámetros:

[5] CSS (Hojas de Estilo en Cascada), se define como un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura.

- ✓ Multiplataforma.
- ✓ Código abierto.
- ✓ Escalabilidad.
- ✓ Vasta documentación.
- ✓ Soporte.
- ✓ Curva de aprendizaje.
- ✓ Comunidad.

A continuación se describen algunos *frameworks* PHP que contribuirán a la selección del marco de trabajo a utilizar en la presente propuesta de solución:

Phalcon PHP

Phalcon es un nuevo framework para PHP, proporciona una herramienta avanzada para desarrollar sitios y aplicaciones web sin importar el rendimiento. Es un *framework full-stack* de código abierto para PHP 5 escrito como extensión en C, optimizado para alto rendimiento liberado bajo la licencia BSD[6]. Todas sus funcionalidades están expuesta como clases PHP listas para usar, por lo que no es necesario aprender C. También débilmente acoplado permitiendo utilizar clases como componentes de acuerdo a lo que la aplicación requiera (30).Posee otras características, entre ellas:

- ✓ Los componentes están libremente acoplados, otorga la libertad de usar todo el framework, o solo las partes que se necesiten.
- ✓ Presenta optimizaciones de bajo nivel que ayudan a reducir la sobrecarga requerida para correr aplicaciones Modelo Vista Controlador (MVC).
- ✓ Las operaciones con base de datos se efectúan con la máxima eficiencia al usar un ORM para PHP escrito en C.
- ✓ Phalcon accede directamente a las estructuras internas de PHP optimizando además cada ejecución.

A pesar del poco conocimiento que posee el equipo de desarrollo acerca de este framework, posee otras desventajas. El mismo, al ser una extensión de php los proyectos serán tan pequeños como el código de fuente generado en los mismos. Esta desventaja radica en que hay que “instalar” el framework en el servidor web, por lo que constituye una desventaja a la hora de desplegar la aplicación, pues muchos servidores brindan el servicio de *hosting* en modo compartido (*shared hosting*), impidiendo acceder a la consola de comandos y configurar PHP.

Yii PHP

Yii es un *framework* PHP liberado bajo la nueva licencia BSD, basado en componentes de alto rendimiento para desarrollar aplicaciones Web de gran escala. Implementa el patrón MVC. Permite la máxima reutilización en la programación web y acelera el proceso de desarrollo, puede ser utilizado

[6] BSD traducido al inglés como Berkeley Software Distribution: licencia que permite utilizar el marco de trabajo de forma gratuita para desarrollar cualquier aplicación web de código abierto o software privativo.

para todo tipo de aplicaciones web. Está equipado con soluciones de cacheo sofisticadas, es adecuado para desarrollar aplicaciones de gran tráfico como portales, foros, sistemas de administración de contenidos (CMS), sistemas de comercio electrónico, etc. Yii está basado en la Programación Orientada a Objeto (POO). Su curva de aprendizaje es intermedia. Está liberado bajo la licencia BSD. (31)

Symfony

Symfony es uno de los framework de PHP 5 más usado en la actualidad. Es de código abierto, fue publicado por la agencia en el año 2005, bajo la licencia MIT[7], y hoy se encuentra entre los *frameworks* líderes de PHP. Este separa la lógica del negocio, del servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Automatiza las tareas más comunes que permiten al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. (32)

Está desarrollado completamente con PHP 5 y es compatible con la mayoría de los gestores de bases de datos como: MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. En la presente investigación, se selecciona Symfony como *framework* de desarrollo.

Fundamentación de la selección

Para un mejor entendimiento del estudio de los *frameworks* antes mencionados se confeccionó la siguiente tabla comparativa, a partir de los parámetros definidos anteriormente:

Tabla 5: Comparación de frameworks PHP

Parámetros	Phalcon	Yii	Symfony
Multiplataforma	sí	sí	sí
Código abierto	sí	sí	sí
Escalabilidad	alta	alta	alta
Vasta documentación	no	sí	sí
Soporte	no	hasta diciembre 2015	hasta mayo 2016
Curva de aprendizaje	baja	media	alta
Comunidad	pequeña	mediana	grande

Entre los aspectos que propiciaron la selección de Symfony en su versión 2.3.1 para el desarrollo del presente sistema informático, se encuentran las características que se enuncian a continuación (32):

[7] MIT, Massachusetts Institute of Technology licencia que permite reutilizar el software así licenciado tanto para ser software libre como para ser software no libre, permitiendo no liberar los cambios realizados al programa original. También permite licenciar dichos cambios con licencia BSD, GPL, u otra cualquiera que sea compatible (es decir, que cumpla las cláusulas de distribución).

- ✓ Su desarrollo es activo, incluye constantes mejoras al combinar la flexibilidad con la rapidez de ejecución.
- ✓ Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
- ✓ Posee una vasta documentación asociada al trabajo con el mismo.
- ✓ Reutilización activa de usuarios: la comunidad de usuarios crece cada día y ofrece un gran soporte de forma gratuita.
- ✓ Flexibilidad: tanto en el diseño global del *framework* como en su sistema de configuración y en los plugins.
- ✓ Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ La curva de aprendizaje es reducida.

1.3.7 Técnicas para mapear objetos relacionales ORM

Los ORM (*Object-Relational mapping*) o lo que es lo mismo, mapeo de objeto relacional son una técnica de programación para convertir datos entre el lenguaje de programación orientado a objetos utilizado y el sistema de base de datos relacional. En la práctica esto crea una base de datos virtual orientada a objetos sobre la BD relacional y el uso de las características propias de la orientación a objetos esencialmente la herencia y el polimorfismo. Facilita las labores básicas de cualquier acceso a datos, el CRUD (*Create*, *Read*, *Update* y *Delete*) al realizar todas estas labores a través de un lenguaje de alto nivel orientado a objetos. Poseen varias ventajas, entre ellas:

- ✓ **Facilidad y velocidad de uso:** la mayoría de las herramientas actuales permiten la creación del modelo por medio del esquema de la base de datos, leyendo el esquema, crea el modelo adecuado.
- ✓ **Abstracción de la base de datos:** al utilizar un ORM se separa el programador del sistema de Base de datos que se utilice.
- ✓ **Seguridad de la capa de acceso a datos contra ataques:** suelen implementar sistemas para evitar tipos de ataques como pueden ser las *SQL injections*.
- ✓ **Reutilización:** permite utilizar los métodos de un objeto de datos desde distintas zonas del sistema, incluso desde aplicaciones distintas.
- ✓ **Mantenimiento del código:** facilita el mantenimiento debido al correcto orden de la capa de datos, haciendo que sea mucho más sencillo.
- ✓ **Lenguaje propio para realizar las consultas:** cuentan con un lenguaje propio para las consultas, lo que permite a los usuarios dejar de utilizar las sentencias SQL y emplear el lenguaje propio de cada herramienta.

El framework Symfony2 viene con el ORM Doctrine incluido e integrado, este fue escrito en PHP y posee un bajo nivel de configuración para empezar un proyecto. Además, genera clases a partir de una BD existente y durante el desarrollo, se puede especificar relaciones y añadir funcionalidades extra a las clases autogeneradas. Otra característica importante de Doctrine, es la posibilidad de escribir consultas utilizando un dialecto de SQL denominado **DQL** (*Doctrine Query Language*) que está inspirado en *Hibernate* (*Java*).Además, posee otras propiedades, entre ellas (33):

- ✓ Brinda soporte para datos jerárquicos.
- ✓ Ofrece soporte para *hooks* (métodos que pueden validar o modificar las escrituras y lecturas de la base de datos) y eventos para manejar la lógica de negocio relacionada.
- ✓ Herencia.
- ✓ Diversos comportamientos del modelo (conjuntos anidados, internacionalización, log, índice de búsqueda).
- ✓ Una función "compilar" que combina varios archivos PHP del framework en uno solo para evitar el descenso de rendimiento que provoca incluir varios archivos PHP.

Algunas de las ventajas que posee son (34):

- ✓ Posee una base de código de alta calidad.
- ✓ Objeto de mapeo y de consulta características extremadamente flexibles y potentes.
- ✓ Apoyo a un alto nivel y la programación de base de datos de bajo nivel para todos sus casos de uso.
- ✓ Gran Comunidad e integraciones con muchos marcos de trabajo (Symfony, Zend Framework, CodeIgniter, FLOW3, etc.).

1.3.8 Servidor web

Un servidor web es un programa que se ejecuta continuamente en una computadora, manteniéndose a la espera de peticiones de ejecución que le hará un cliente o un usuario de Internet. Se encarga de contestar a estas peticiones de forma adecuada, y entrega como resultado una página web o información de todo tipo de acuerdo a los comandos solicitados.

Apache es uno de los servidores web de código abierto y gratuito, por lo que se puede decir que corre sobre cualquier plataforma. Una de sus características es que muestra mensajes de error altamente configurables, posee bases de datos de autenticación, es adaptable, robusto y estable, en la actualidad se le considera el servidor web más utilizado en el mundo. (35)

Apache cuenta con varios módulos multiprocesos para cada uno de los sistemas operativos sobre los que se ejecuta, optimizando el rendimiento y la rapidez del código. A continuación se muestran las tres categorías de estos subsistemas:

Módulos Base: contienen las funciones básicas de Apache.

Módulos Multiproceso: son los responsables de la unión con los puertos de la máquina, aceptando las peticiones.

Módulos Adicionales: cualquier otro subsistema que le añada una funcionalidad al servidor.

En la presente propuesta de solución se utilizará Apache 2.4.12 como servidor web para el desarrollo del sistema, debido a que la empresa COPEXTEL establece el uso de este en su infraestructura tecnológica, además de las ventajas enunciadas a continuación:

Ventajas de Apache (35)**:**

- ✓ Es personalizable.
- ✓ Corren en gran número de sistemas operativos.
- ✓ Actualmente existen diversos módulos para este tipo de servidor, que pueden ser usados cuando sea necesario.
- ✓ Permite la implementación de los últimos y más nuevos protocolos.
- ✓ El servidor puede ser administrado mediante línea de comandos, lo que hace la administración remota muy conveniente.

Las actualizaciones de Apache, se basan en erradicar las fallas que se encuentran en las versiones en uso. La versión 2.4.12 está basada en la corrección de errores y en la seguridad de este servidor. Además incluye nuevas características (35):

- ✓ Proxy FGI y *websockets* mejorados.
- ✓ Capacidad de proxy a través de controlador.
- ✓ Mayor control sobre el alcance de *RewriteRules*.
- ✓ Unix Domain Socket (UDS) apoyo a *backends mod_proxy*.
- ✓ Soporte para ampliar el tamaño de memoria compartida para *mod_socache_shmcb* y mejoras de *mod_lua* y *mod_ssl*.
- ✓ Grupos de apoyo y referencias hacia atrás con nombre dentro de las directivas *LocationMatch*, *DirectoryMatch*, *FilesMatch* y *ProxyMatch*.

1.3.9 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) es un software cuyo objetivo es proporcionar una interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Las principales funciones que debe cumplir un SGBD se relacionan con la creación y mantenimiento de la BD, el control de accesos, la manipulación de datos de acuerdo con las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, evitar redundancias e inconsistencias y mantener la integridad. En la empresa COPEXTEL se utiliza el PostgreSQL. (36)

Características de los SGBD (36):

- ✓ **Abstracción de la información**: ahorran a los usuarios detalles acerca del almacenamiento físico de los datos.
- ✓ **Independencia**: consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven en ella.
- ✓ **Consistencia**: consiste en la actualización simultánea de los datos repetidos en la BD.

- ✓ **Seguridad**: deben garantizar que la información almacenada en la base de datos se encuentre segura frente a usuarios malintencionados que intenten leer, modificar o eliminar la información privilegiada; frente a ataques que deseen manipular o destruir la información.
- ✓ **Integridad**: se trata de adoptar todas las medidas necesarias para garantizar la validez de los datos almacenados.

En la presente investigación, se utilizará PostgreSQL en su versión 9.1, que es un SGBD objeto-relacional, distribuido bajo licencia BSD, es de código abierto, por lo que el código del programa está disponible a cualquier persona. Incluye características de la POO, como la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional.

A continuación se enumeran las principales características de este gestor de bases de datos (36):

- ✓ Implementa el estándar (lenguaje) SQL92/SQL99.
- ✓ Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP) y cadenas de bits y permite la creación de tipos propios.
- ✓ Incorpora una estructura de datos arreglo (*Array*).
- ✓ Permite la declaración de funciones propias, así como la definición de disparadores.
- ✓ Soporta el uso de índices, reglas y vistas.
- ✓ Incluye herencia entre tablas (aunque no entre objetos, ya que no existen).
- ✓ Permite la gestión de diferentes usuarios y los permisos asignados a cada uno de ellos.
- ✓ Cuenta con un amplio conjunto de enlaces con lenguajes de programación, entre ellos C, C++, Java, Python, PHP y otros.
- ✓ El progreso continuo del gestor de datos de código abierto PostgreSQL brinda a los consumidores la opción de instalar una base de datos no privativa.

PostgreSQL se convierte en una gran alternativa al momento de decidirse por un sistema de bases de datos. Es el SGBD que utiliza la empresa COPEXTEL para todas sus divisiones y presenta varias ventajas que se exponen a continuación (37):

- ✓ Gran flexibilidad ya que funcionar la mayoría de los sistemas Unix.
- ✓ Tiene características que permiten extender fácilmente el sistema.
- ✓ PostgreSQL puede ser integrada al ambiente Windows permitiendo de esta manera a los desarrolladores, generar nuevas aplicaciones o mantener las ya existentes.
- ✓ Permite desarrollar o migrar aplicaciones desde Access, Visual Basic, FoxPro, Visual Fox Pro, C/C++ Visual C/C++, Delphi, para que utilicen a PostgreSQL como servidor de BD.

1.3.10 Herramientas para las pruebas de software

LoadUI

LoadUI es una herramienta que permite crear, configurar y redistribuir sus pruebas de carga de forma interactiva y en tiempo real. Soporta todos los protocolos estándar de REST, SOAP / WSDL, AMF, JDBC, POX a HTTP (S) y HTML. En un solo entorno de prueba, LoadUI ofrece cobertura de la prueba

completa, incluyendo pruebas de carga Web, automatizada, de estrés, entre otras. Una de sus desventajas es que es bajo licencia de pago. Entre sus características principales se encuentran (42):

- ✓ Estrategias de actuación múltiples.
- ✓ Información en tiempo real acerca de los resultados de la prueba.
- ✓ Reutilización de las pruebas funcionales existentes.
- ✓ Prueba de carga simultánea que se utiliza para escenarios complejos.
- ✓ Utiliza *ServiceV Pro* para superar los límites de velocidad de los tiempos de respuesta en las pruebas.

FunkLoad

FunkLoad es un probador web funcional y de carga, escrito en Python, y se utiliza para realizar varias funciones y tipos de prueba, entre ellas (43):

- ✓ Prueba de funcionamiento de los proyectos web y pruebas de regresión.
- ✓ Las pruebas de rendimiento al cargar la aplicación web.
- ✓ El monitoreo de los servidores que le ayuda a localizar los cuellos de botella, dando un informe detallado de la medición del desempeño.
- ✓ Herramienta de prueba de carga para exponer los errores que no se muestran en la prueba superficial, como prueba el volumen o la prueba de la longevidad.
- ✓ Pruebas de estrés herramienta para abrumar a los recursos de la aplicación web y probar la capacidad de recuperación de aplicaciones.
- ✓ Escribir agentes web mediante secuencias de comandos de cualquier tarea repetitiva web.

Apache JMeter

Para las pruebas de carga y estrés que se realizarán en la presente investigación se empleará la herramienta Apache JMeter. Esta es una aplicación de escritorio de código abierto desarrollada 100% con Java, liberada bajo la licencia Apache 2.0. Diseñada para realizar pruebas funcionales de comportamiento y para medir el rendimiento. Se utiliza para simular una carga pesada en el servidor, en la red o a un objeto para poner a prueba su resistencia o para analizar el rendimiento global en diferentes tipos de carga. Originalmente se diseñó para probar aplicaciones Web, pero ha ampliado su perfil a otras funciones de prueba (44). En la presente propuesta de solución se utilizará la versión 2.3.1 de esta herramienta.

Conclusiones parciales

- ✓ El estudio realizado, apoyado en los métodos de investigación científica especificados anteriormente, permitió elaborar el marco teórico conceptual que sustenta la investigación.
- ✓ El análisis de las soluciones similares permitió obtener funcionalidades útiles para el desarrollo del sistema propuesto.
- ✓ Se utilizará como framework de desarrollo Symfony 2.3.1 y el IDE NetBeans en su versión 7.4.Se seleccionó el gestor de base de datos PostgreSQL 9.1 y servidor Web Apache 2.4.7.

Para las pruebas unitarias al software se seleccionó la biblioteca PHPUnit y la herramienta Apache JMeter en su versión 2.3.1 para las pruebas de carga y estrés. Los lenguajes de programación que se eligieron son: del lado del servidor PHP 5.5.9 y del lado del cliente HTML5, JQuery 1.10.1 como framework JavaScript, y Bootstrap 2.3.1 como *framework* CSS. Se utilizará la metodología XP como base para el desarrollo del sistema.

Capítulo 2: Descripción y análisis de la solución propuesta

Introducción

El presente capítulo tiene como objetivo describir las características del sistema a desarrollar, así como los roles que intervienen en la aplicación. Se muestran los principios, técnicas y prácticas que constituyen una guía para el desarrollo de la presente investigación. De igual forma se generan los artefactos ingenieriles correspondientes a la metodología seleccionada.

2.1 Objetivos del sistema

Un objetivo es la definición de una meta o propósito a alcanzar, de acuerdo al ámbito donde sea utilizado o formulado cuenta con cierto nivel de complejidad. (45)

Para poder llevar a cabo el desarrollo de un sistema informático, se hace necesario la definición de los objetivos del sistema. Para la presente propuesta de solución se define:

- ✓ Contribuir a la gestión y almacenamiento de la evaluación de desempeño de los trabajadores de la división Norte de la empresa COPEXTEL.
- ✓ Mostrar la información referente a la evaluación de los trabajadores.
- ✓ Mostrar el salario de los trabajadores dada una evaluación.
- ✓ Dar seguimiento a la evaluación de los trabajadores por parte de los jefes de cada área.
- ✓ Obtener un sistema para la evaluación del desempeño de los trabajadores que permita contribuir al proceso de pago por resultado en la división Norte de la empresa COPEXTEL.

2.2 Usuarios del sistema

Los usuarios del sistema son personas que se conectan a la aplicación para hacer uso de los servicios que esta proporciona, obteniendo una respuesta a partir de las peticiones que hacen y los procesos que se ejecutan en la misma. En la presente investigación se definen los siguientes usuarios:

Tabla 6: Usuarios del sistema

Usuario	Descripción
Gerente general	Es la persona autorizada para la gestión del sistema, encargado de actualizar, modificar, eliminar e insertar toda la información. Dispone de posibilidades ilimitadas para ejecutar todas las funciones administrativas del sistema. En la presente propuesta, este sería el Gerente de la división
Jefe de área	Es el usuario con permisos para gestionar las evaluaciones de los trabajadores de un área determinada. En la presente propuesta se refiere a los Jefes de cada unidad organizativa.

Trabajador	Tiene permiso para acceder a la evaluación individual, y ver otros reportes generales realizados por la empresa.

2.3 Sistemas de control de acceso

Uno de los aspectos fundamentales para el desarrollo de la presente investigación es definir el sistema de control de acceso de los usuarios a la aplicación, encargado de conceder los permisos para acceder a los datos o recursos que los usuarios requieren para realizar sus tareas. Para determinar si un usuario tiene permitido el acceso a los recursos, han sido propuestos diferentes modelos para la gestión del control de acceso, entre los que se encuentran: el control de acceso discrecional (DAC), el control de acceso obligatorio (MAC) y el control de acceso basado en roles (RBAC). (46)

El control de acceso discrecional restringe el acceso a objetos basándose en la identidad de los sujetos o los grupos a los que pertenecen éstos (46). Los modelos DAC no son adecuados para representar flujos de información, no poseen restricciones para controlar el acceso a los recursos de información de forma efectiva. Por otra parte los modelos MAC implementan el control de acceso basándose en permisos fijos otorgados por una autoridad central. Su principal desventaja es que no proporciona soluciones factibles dado que les falta suficiente flexibilidad. Los modelos DAC y MAC son inadecuados para cubrir las necesidades de la mayor parte de las organizaciones. En tanto el modelo RBAC tiene como objetivo prevenir que los usuarios tengan libre acceso a la información de la organización. (46)

En el desarrollo de la presente investigación se propone utilizar el modelo RBAC, debido a que se hace necesario restringir el acceso a un conjunto de funcionalidades mediante la definición de los roles, a continuación se exponen algunas especificaciones del mismo:

Este modelo regula el acceso de los usuarios a la información en base a las actividades y funciones que ejecutan en el sistema, representando de forma natural la estructura de las organizaciones. Los modelos RBAC simplifican la gestión de autorizaciones incluyendo mecanismos para asignar controles de acceso, los usuario asumen los permisos asociados con los roles a los que está asignado. En RBAC se define un rol como como un conjunto de derechos de acceso asociados a una posición dentro de una organización, o con una actividad de trabajo. Este modelo permite la construcción jerárquica de estas políticas de acceso, por herencia o especialización, por ello tiene el potencial de reducir la complejidad y el coste de la administración de seguridad en entornos heterogéneos. (46)

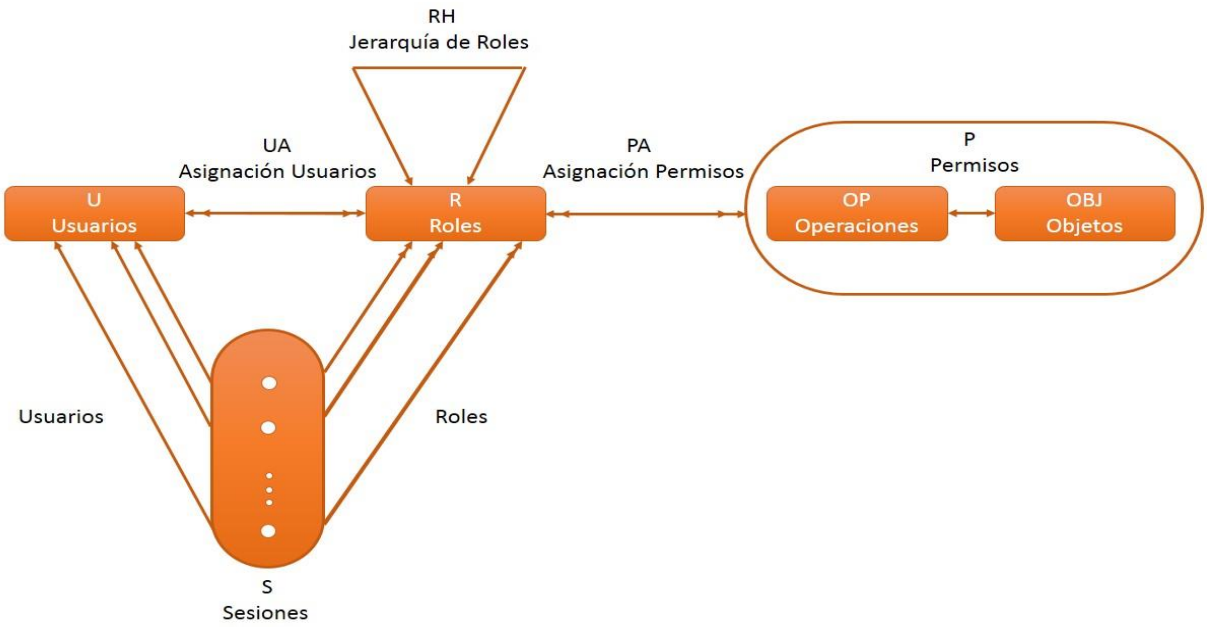


Fig. 2 Modelo RBAC

2.4 Requerimientos de software

Los requerimientos de software constituyen una necesidad documentada sobre el contenido, la forma o funcionalidades de un producto, considerando las especificidades de los clientes. Son declaraciones que identifican atributos, características, capacidades, cualidades que necesita cumplir un sistema o software para que tenga valor y utilidad para el usuario. (47)

2.4.1 Requisitos funcionales

Según Sommerville (2005) en (48) "Los requerimientos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones específicas. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que le sistema no debe hacer". A continuación se describen los requisitos y las restricciones que fueron identificados para darle solución a los objetivos propuestos, así como la complejidad de desarrollo de los requisitos funcionales de la presente propuesta de solución.

Tabla 7: Funcionalidades del sistema

No.	Descripción	Prioridad
1	Gestionar trabajador	Media
2	Gestionar área	Media
3	Gestionar cargo	Media
4	Autenticar usuario	Media
5	Gestionar indicador de eficiencia	Alta
6	Gestionar indicador específico	Alta

7	Gestionar evaluación de desempeño	Media
8	Mostrar evaluaciones	Media
9	Gestionar valores de entrada	Media
10	Gestionar método de pago	Alta
11	Efectuar pago	Alta
12	Mostrar salario histórico de los trabajadores	Media
13	Mostrar monto a pagar histórico en la división	Media
14	Crear reporte del cumplimiento de la empresa	Media
15	Mostrar datos estadísticos en forma de gráfico	Alta
16	Exportar a pdf	Media
17	Exportar a excel	Media

2.4.2 Requisitos no funcionales

Los requisitos no funcionales son aquellos que actúan para obligar la solución a cumplir con determinados parámetros, se conocen a veces como apremios o requisitos de calidad. Pueden ser clasificados más a fondo según si son requisitos de funcionamiento, requisitos de capacidad de mantenimiento, requisitos de seguridad, requisitos de confiabilidad, etc. Estos, describen aspectos del sistema que son visibles por el usuario que no incluyen una relación directa con el comportamiento funcional del sistema, estos no describen ninguna información a guardar. (48)

A continuación se muestran los requisitos funcionales definidos para el sistema:

Usabilidad

Ambiente

RnF 1. Requerimientos de Software

Navegadores para acceder: Internet Explorer 7.0 o superior, Mozilla Firefox 3.5 o superior, y Google Chrome.

RnF 2. Requerimientos de Hardware

El ordenador donde se instalará el sistema debe tener los siguientes requerimientos mínimos:

1 GB de memoria RAM

40 GB de disco duro

Soporte

RnF 3. El sistema debe responder correctamente ante cambios o mejoras en sus funcionalidades.

Restricciones de diseño

RnF 4.

Se utilizarán los lenguajes de programación:

- ✓ HTML5
- ✓ PHP 5.5.9

Los frameworks que se emplearán en el desarrollo son:

- ✓ JQuery 1.10.1
- ✓ Boostrap 2.3.1
- ✓ Symfony 2.3.1

RnF 5. Restricciones de arquitectura. Se utilizará el patrón arquitectónico MVC que establece el framework de desarrollo Symfony.

Requisitos para la documentación de usuarios en línea y ayuda del sistema

RnF 6. El sistema debe estar acompañado de un documento que refleje la descripción de sus funcionalidades.

RnF 7. El sistema debe estar acompañado de un documento que refleje el software necesario para instalar la aplicación, así como los pasos para su puesta en funcionamiento.

Interfaz

RnF 8. El sistema tiene que ofrecer una interfaz con un diseño accesible para los usuarios.

Requisitos de licencia

RnF 9. Las librerías y tecnologías utilizadas para la confección del sistema deben estar bajo la licencia:

GPL (*General Public License*)

BSD (*Berkeley Software Distribution*)

MIT (*Massachusetts Institute of Technology*)

Apache 2.0

Requisitos de portabilidad

RnF 10. El sistema no tendrá ninguna dependencia con los siguientes sistemas operativos:

Windows (XP, Vista, Seven, 8)

Linux (*Unix)

2.5 Historias de usuario

Las Historias de usuario (HU) constituyen el primer paso de cualquier proyecto que utilice XP como metodología de desarrollo. Son definidas con el cliente en la fase de exploración, constan de 3 o 4 líneas escritas por el cliente en un lenguaje no técnico sin hacer mucho hincapié en los detalles; en las mismas no se hace no se debe hacer alusión a posibles algoritmos para su implementación, ni de diseños de base de datos adecuados. Son utilizadas para estimar tiempos de desarrollo de la parte de la aplicación que describen. En la fase de pruebas, se emplean para verificar si el programa cumple con lo que especifica la HU. Cuando llega la hora de implementar una HU, el cliente y los desarrolladores se reúnen para concretar y detallar lo que tiene que hacer la misma. El tiempo de desarrollo ideal para una HU es entre 1 y 3 semanas. (49) (50)

Teniendo en cuenta la escala de prioridad en el negocio las HU se clasifican en:

- ✓ **Alta:** se le otorga a las HU que constituyen funcionalidades de vital importancia en el desarrollo del proyecto.
- ✓ **Media:** se le otorga a las HU que para el cliente constituyen funcionalidades a tener en cuenta sin que tengan una afectación directa sobre el proyecto que se está desarrollando.
- ✓ **Baja:** se le otorga a las HU que constituyen funcionalidades que sirven de ayuda al control de la estructura y que no tienen nada que ver con el proyecto en desarrollo.

Teniendo en cuenta la escala de riesgo en el desarrollo las HU se clasifican en:

- ✓ **Alta:** se otorga cuando para la implementación de la HU se avizora la posible existencia de errores que lleven a la inoperatividad del código.
- ✓ **Media**: se otorga cuando pueden aparecer errores en la implementación de la HU que puedan retrasar la entrega de la versión.
- ✓ **Baja:** se otorga cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto.

Para definir las HU se utilizó la siguiente plantilla para establecer una correcta definición de las mismas.

Tabla 8: Plantilla definida para las HU

<table>
<tr><th colspan="3">Historia de Usuario</th></tr>
<tr><td colspan="2">Número: posee el número asignado a la HU</td><td>Nombre Historia de Usuario: atributo que contiene el nombre de la HU.</td></tr>
<tr><td colspan="3">Usuario: el usuario del sistema que utiliza o es protagonista de la HU.</td></tr>
<tr><td colspan="2">Prioridad de Negocio: evidencia el nivel de prioridad de la HU para el cliente.</td><td>Riesgo de desarrollo: evidencia el nivel de riesgo de desarrollo en caso de no realizarse la HU. (Alto/Medio/Bajo).</td></tr>
<tr><td colspan="2">Puntos estimados: permite estimar la duración de la implementación. Cuando el valor es uno equivale a una semana de trabajo (5 días trabajando 40 horas, es decir, 8 horas diarias). Por lo que cuando el valor de dicho atributo es de 0.5 equivale a 2 días y medio de trabajo, lo que se traduce en 20 horas.</td><td>Iteración asignada: precisa la iteración en la que se desarrollará la historia de usuario.</td></tr>
<tr><td colspan="3">Descripción: recoge una breve descripción de lo que realiza la HU.</td></tr>
<tr><td colspan="3">Observación: brinda información extra que se estime agregar para hacer más comprensible la HU.</td></tr>
</table>

A continuación se muestran 5 HU, las restantes se encuentran en el **Anexo IV**.

Tabla 9: HU Gestionar trabajador

Historia de Usuario	
Número:1	**Nombre Historia de Usuario:** Gestionar trabajador
Usuario: Gerente general	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados:1	**Iteración asignada**: 1
Descripción: el gerente general tiene la posibilidad de gestionar la información de los trabajadores.	
Observaciones: la funcionalidad gestionar incluye adicionar, modificar y eliminar los datos de los trabajadores definidos en el sistema.	

Tabla 10: HU Gestionar área

Historia de Usuario	
Número: 2	**Nombre Historia de Usuario:** Gestionar área
Usuario: Gerente general	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: el gerente general tiene la posibilidad de gestionar las áreas pertenecientes a la división Norte de la empresa COPEXTEL.	
Observaciones: la funcionalidad gestionar incluye adicionar, modificar y eliminar áreas.	

Tabla 11: HU Gestionar cargo

Historia de Usuario	
Número: 3	**Nombre Historia de Usuario:** Gestionar cargo
Usuario: Gerente general	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: el gerente general tiene la posibilidad de gestionar los cargos asociados a los trabajadores que trabajan en la división Norte de la empresa COPEXTEL.	
Observaciones: la funcionalidad gestionar incluye adicionar, modificar y eliminar cargos.	

Tabla 12: HU Autenticar usuario

Historia de Usuario	
Número: 4	**Nombre Historia de Usuario:** Autenticar usuario
Usuario: Gerente general, Jefe de área, Trabajador	

Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: los usuarios pueden acceder al sistema a través de un identificador y una contraseña asignada a por el gerente general en el sistema cuando se crea un trabajador.	
Observaciones:	

Tabla 13: HU Gestionar indicador de eficiencia

Historia de Usuario	
Número: 5	**Nombre Historia de Usuario:** Gestionar indicador de eficiencia
Usuario: Gerente general	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 1	**Iteración asignada**: 1
Descripción: el gerente general tiene la posibilidad de gestionar los indicadores de eficiencia a medir para cada uno de los trabajadores.	
Observaciones: la funcionalidad gestionar incluye adicionar, modificar y eliminar los indicadores.	

Tabla 14: HU Gestionar indicador específico

Historia de Usuario	
Número: 6	**Nombre Historia de Usuario:** Gestionar indicador específico
Usuario: Gerente general	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: el gerente general tiene la posibilidad de gestionar los indicadores específicos de cada uno de los trabajadores.	
Observaciones: la funcionalidad gestionar incluye adicionar, modificar y eliminar los indicadores específicos.	

2.6 Estimación de esfuerzos por historias de usuario

Las estimaciones de esfuerzo pertenecientes a la implementación de las HU la establecen los programadores utilizando como unidad de medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos. (51)

Tabla 15: Estimación de esfuerzos por HU

No.	Descripción	Estimación (Semanas)
1	Gestionar trabajador	1
2	Gestionar área	0.5
3	Gestionar cargo	0.5

4	Autenticar usuario	0.5
5	Gestionar indicador de eficiencia	0.5
6	Gestionar indicador específico	0.5
7	Gestionar evaluación de desempeño	0.5
8	Mostrar evaluaciones	0.5
9	Gestionar valores de entrada	0.5
10	Gestionar método de pago	0.5
11	Efectuar pago	1
12	Mostrar salario histórico de los trabajadores	0.5
13	Mostrar monto a pagar histórico en la división	0.5
14	Crear reporte del cumplimiento de la empresa	0.5
15	Mostrar datos estadísticos en forma de gráfico	1
16	Exportar a pdf	0.5
17	Exportar a excel	0.5

2.7 Plan de Iteraciones

La creación de un sistema, según la metodología seleccionada se divide en iteraciones, cada una con una duración ideal de 1 a 3 semanas. En el plan de iteraciones, las HU seleccionadas para cada entrega son desarrolladas y probadas en un ciclo de iteración, de acuerdo al orden preestablecido. Al finalizar las iteraciones la aplicación contará con todas las funcionalidades implementadas para darle cumplimiento a los objetivos propuestos. (16)

Iteración 1: en esta iteración se entregarán las funcionalidades que tienen prioridad alta para el cliente, correspondiendo a las HU:

- ✓ HU1: Gestionar trabajador.
- ✓ HU2: Gestionar área.
- ✓ HU3: Gestionar cargo.
- ✓ HU4: Autenticar usuario.
- ✓ HU5: Gestionar indicador de eficiencia.

Iteración 2: en esta iteración se realizarán las funcionalidades que son de prioridad media para el cliente, siendo estas:

- ✓ HU6: Gestionar indicador específico.
- ✓ HU7: Gestionar evaluación de desempeño.
- ✓ HU8. Mostrar evaluaciones.
- ✓ HU9. Gestionar valores de entrada.

Iteración 3: en la tercera iteración se recogerán otras funcionalidades que son de prioridad media.

- ✓ HU10.Gestionar método de pago.
- ✓ HU11: Efectuar pago.

- ✓ HU12: Mostrar salario histórico de los trabajadores.
- ✓ HU13: Mostrar monto a pagar histórico en la división.

Iteración 4: en esta iteración se recogerán las restantes funcionalidades de prioridad media.

- ✓ HU14: Crear reporte del cumplimiento de la empresa.
- ✓ HU15: Mostrar datos estadísticos en forma de gráfico.
- ✓ HU16: Exportar a pdf.
- ✓ HU17: Exportar a Excel.

2.8 Plan de duración de las iteraciones

El plan de duración de las iteraciones se realiza luego de tener el estimado en días que demora implementar cada HU. Se tendrá en cuenta además la prioridad que el cliente le asigna a cada historia de usuario así como el nivel de complejidad que estas poseen.

Tabla 16: Plan de duración de las iteraciones

Iteración	Orden de las HU	Duración total (Semanas)
1	Gestionar trabajador	3 semanas
	Gestionar área	
	Gestionar cargo	
	Autenticar usuario	
	Gestionar indicador de eficiencia	
2	Gestionar indicador específico	3 semanas
	Gestionar evaluación de desempeño	
	Mostrar evaluación	
	Gestionar valores de entrada	
3	Gestionar método de pago	3 semanas
	Efectuar pago	
	Mostrar salario histórico de los trabajadores	
	Mostrar monto a pagar histórico en la división	
4	Crear reporte del cumplimiento de la empresa	3 semanas
	Mostrar datos estadísticos en forma de gráfico	
	Exportar a pdf	
	Exportar a Excel	
	Crear reporte del cumplimiento de la empresa	
	Consultar datos estadísticos en forma de gráfico	

2.9 Plan de entrega

En el plan de entregas se realiza un cronograma donde el cliente establece la prioridad de cada HU, cuáles serán agrupadas para conformar una entrega, y el orden de las mismas. Este entregable será el resultado de una reunión entre todos los actores del proyecto. Se realiza en base a las estimaciones de tiempos de desarrollo realizadas por los desarrolladores. (50)

Tabla 17: Plan de entrega

Historia de Usuario	1 iteración	2 iteración	3 iteración	4 iteración
Gestionar trabajador	V 1.0	Finalizado	Finalizado	Finalizado
Gestionar área	V 1.0	Finalizado	Finalizado	Finalizado
Gestionar cargo	V 1.0	Finalizado	Finalizado	Finalizado
Autenticar usuario	V 1.0	Finalizado	Finalizado	Finalizado
Gestionar Indicador de eficiencia	V1.0	Finalizado	Finalizado	Finalizado
Gestionar Indicador específico	-	V 1.1	Finalizado	Finalizado
Gestionar evaluación de desempeño	-	V 1.1	Finalizado	Finalizado
Mostrar evaluaciones	-	V 1.1	Finalizado	Finalizado
Gestionar valores de entrada	-	V1.1	Finalizado	Finalizado
Gestionar método de pago	-	-	V2.0	Finalizado
Efectuar pago	-	-	V2.0	Finalizado
Mostrar salario histórico a los trabajadores	-	-	V2.0	Finalizado
Mostrar monto a pagar histórico en la división	-	-	V2.0	Finalizado
Crear reporte del cumplimiento de la empresa	-	-	-	Finalizado
Mostrar datos estadísticos en forma de gráfico	-	-	-	Finalizado
Exportar a pdf	-	-	-	Finalizado
Exportar a Excel	-	-	-	Finalizado

2.10 Prototipo de Interfaz de usuario no funcional

Un prototipo no funcional de interfaz de usuario (IU) es un modelo que representa el comportamiento del sistema. Es utilizado con el fin de que el cliente entienda ciertos aspectos del sistema. También para que pueda refinar sus necesidades y así poder transmitirlas al equipo de desarrollo. Para mostrar una vista preliminar del sistema a desarrollar se crearon prototipos de IU no funcionales apoyados en la herramienta Visual Paradigm en su versión 8.0, permitiendo al cliente, un mayor entendimiento de las características del sistema. A continuación se muestran los prototipos de IU no funcional de varias vistas del sistema.

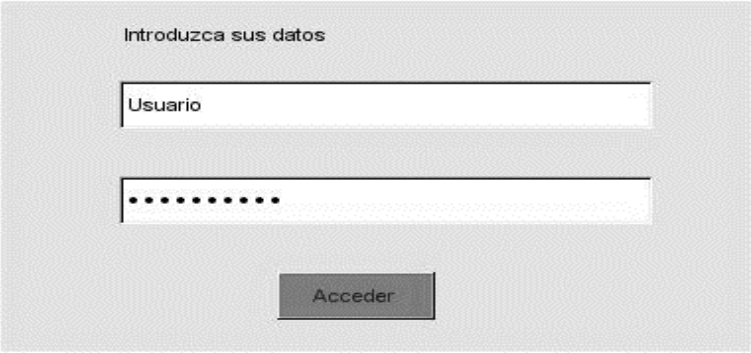


Fig. 3 Prototipo IU Autenticar usuario

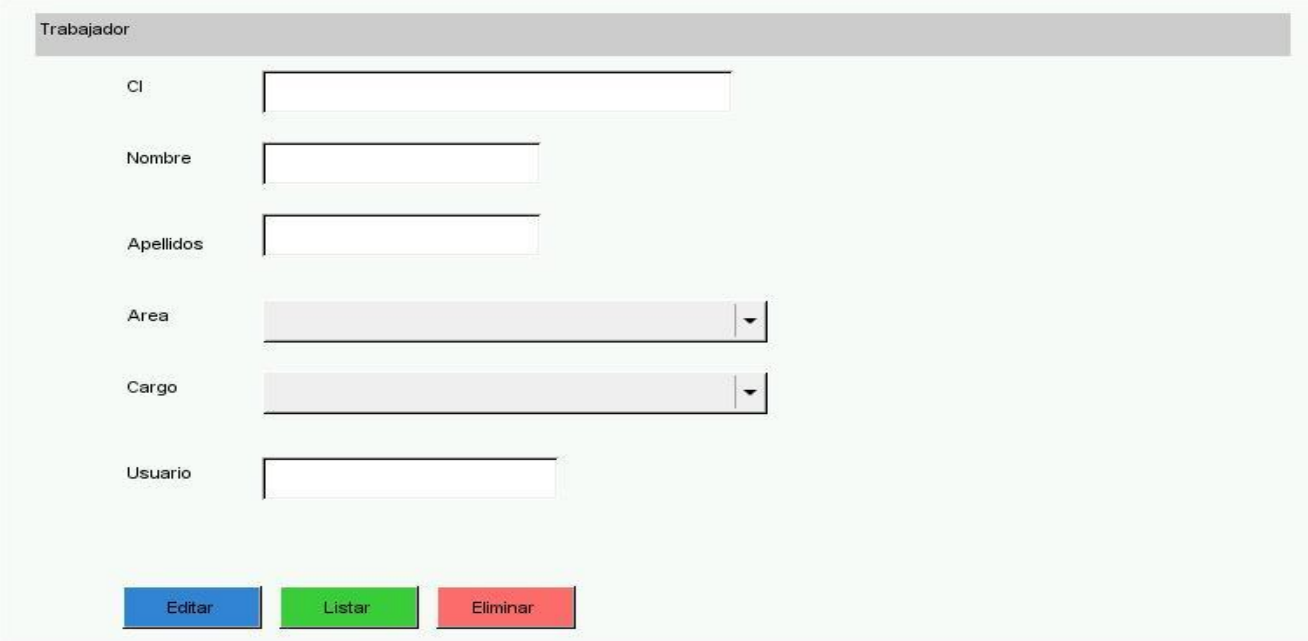


Fig. 4 Prototipo IU Editar trabajador

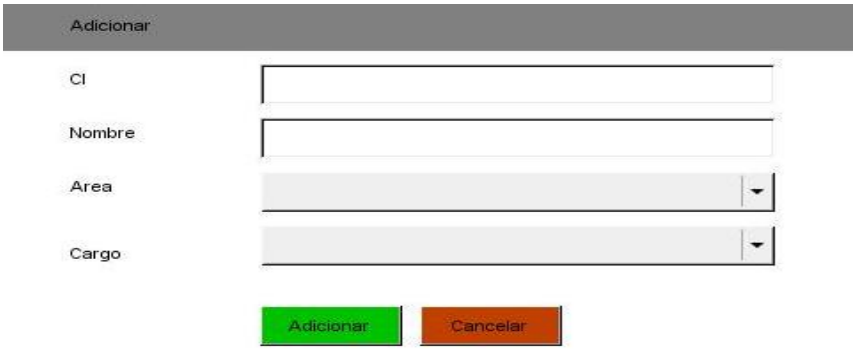


Fig. 5 Prototipo IU Adicionar trabajador

Fig. 6 Prototipo IU Efectuar pago

2.11 Tarjetas Clases Responsabilidad Colaborador (CRC)

En la metodología XP, no se requiere de la representación del sistema mediante diagramas de clases utilizando notación UML, en su lugar se usan las tarjetas CRC (Clase-Responsabilidad-Colaboración). Una tarjeta CRC no es más que una ficha de papel o cartón que representa una entidad del sistema. La metodología establece 3 dimensiones para la elaboración de las tarjetas CRC, las cuales identifican el rol de un objeto en análisis y/o diseño: nombre de la clase, responsabilidades y colaboraciones. (52)

- ✓ **Una clase:** representa una colección de objetos similares.
- ✓ **Una responsabilidad**: es aquello que la clase conoce o realiza. Una clase puede cambiar el valor de lo que sabe pero no puede cambiar el valor de lo que saben otras clases. Algunas veces una clase tiene una responsabilidad que cumplir pero no tiene toda la información para hacerlo. Esto hace que deba interactuar con otras clases para obtener su colaboración.
- ✓ **Las colaboraciones** adoptan una de dos formas: pueden ser un pedido de información o una petición de que se realice una operación.

Las principales **características** de las tarjetas CRC son (50):

- ✓ Identificación de clases y asociaciones que participan en el diseño del sistema.
- ✓ Obtención de las responsabilidades que debe cumplir cada clase.

- ✓ Establecimiento de cómo una clase colabora con otras clases para cumplir con sus responsabilidades.

A continuación se presentan algunas de las tarjetas CRC efectuadas en la presente investigación. El resto se encuentran en el **Anexo V**.

Tabla 18: Tarjeta CRC Evaluación

Nombre: Evaluación	
Responsabilidad	**Colaboraciones**
✓ Crear una evaluación de un trabajador ✓ Editar parámetros evaluados	Trabajador Metodologia ValoresEntrada

Tabla 19: Tarjeta CRC Indicador específico

Nombre: Indicador específco	
Responsabilidad	**Colaboraciones**
✓ Crear un indicador para evaluar a un trabajador ✓ Insertar un indicador en la lista de indicadores específicos ✓ Editar un indicador específico ✓ Eliminar un indicador específico	IndicadorEficiencia

Tabla 20: Tarjeta CRC EvaluaciónDesempeño

Nombre: Evaluación de desempeño	
Responsabilidad	**Colaboraciones**
✓ Crear evaluación de desempeño ✓ Editar evaluación de desempeño	Trabajador IndicadorEspecífico IndicadorEficiencia Evaluacion

2.12 Modelo de datos

El modelo de datos no es más que una colección de herramientas conceptuales que se emplean para especificar datos, las relaciones entre ellos, su semántica asociada y las restricciones, permiten describir y manipular los datos de un cierto universo que deseamos almacenar en la base de datos (45). A continuación se presenta el modelo de datos asociado a la presente propuesta de solución.

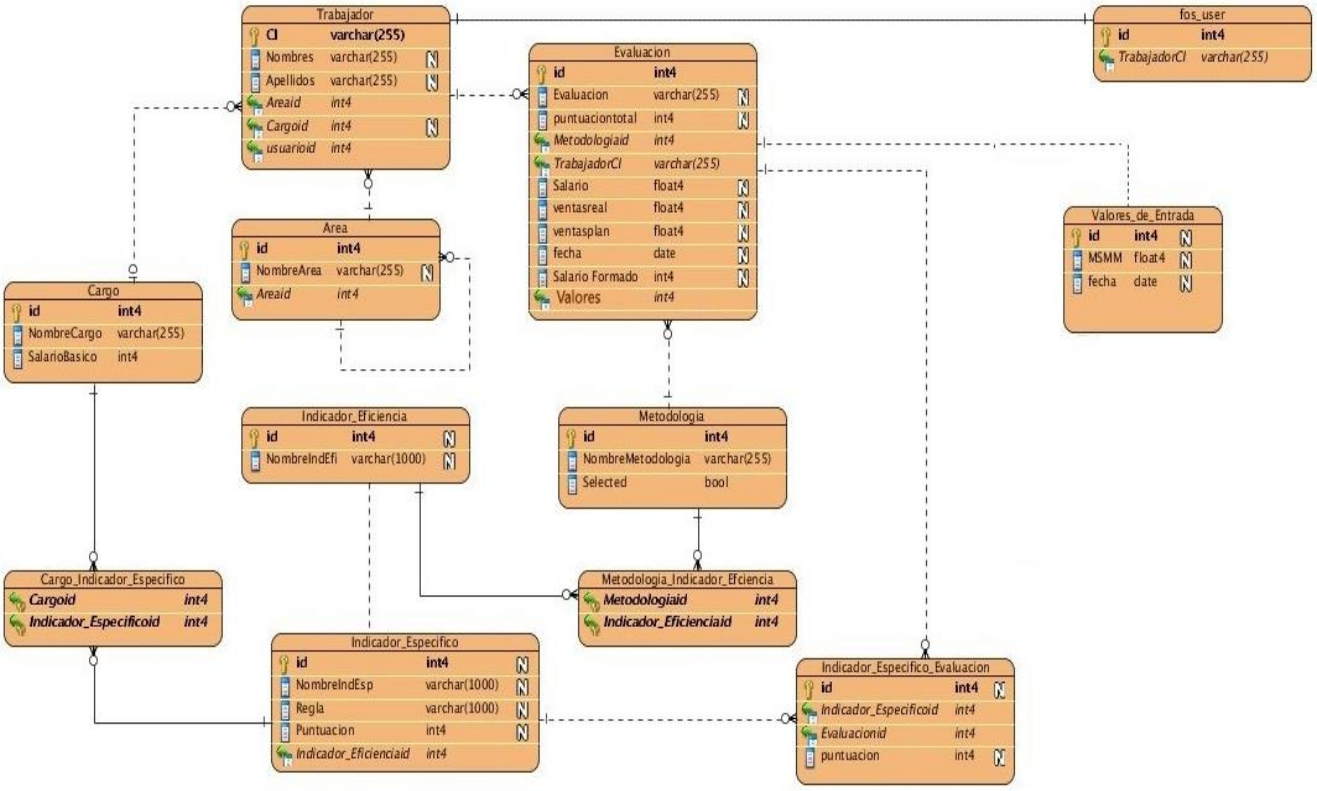


Figura 1: Modelo de datos

Conclusiones parciales

- ✓ Se identificaron 3 tipos de usuarios para el acceso al sistema, lo que permitirá una mejor gestión de la seguridad y la disponibilidad de los datos.
- ✓ Los artefactos ingenieriles elaborados permitieron realizar una descripción detallada de la propuesta de solución que posteriormente servirá de guía durante la implementación.
- ✓ Se detallaron un total de 17 HU, permitiendo al equipo de desarrollo un mejor entendimiento de las funcionalidades para la implementación de la aplicación.
- ✓ Se estableció la prioridad de cada una de las HU para el negocio, ayudando a establecer el orden de su implementación así como el número de iteraciones en que serán desarrolladas.
- ✓ Se construyó el plan de entregas de la propuesta de solución con el propósito de ir realizando entregas periódicas de los módulos del sistema, dicho plan terminará en 12 semanas.
- ✓ Se elaboraron las tarjetas CRC, lo que permitió encontrar las clases del dominio del sistema, sus responsabilidades y cómo colaboran con otras para realizar tareas.
- ✓ La confección de los prototipos de interfaz de usuario no funcional brindó un mejor acercamiento al diseño de la presente propuesta de solución.

Capítulo 3: Implementación y pruebas

Introducción

El presente capítulo muestra los diferentes artefactos resultantes en la fase de iteraciones y de producción (50). La fase de iteraciones es la fase principal en el desarrollo de XP, donde las funcionalidades del sistema a implementar son desarrolladas, generando al final cada una un entregable funcional que implementa las HU asignadas a la iteración. En la fase de producción no se realizan más módulos funcionales, pero pueden ser necesarias tareas de ajustes, en inglés *fine tunning*.

Además se hace un análisis de las pruebas a realizar con el fin de garantizar un correcto funcionamiento de la aplicación y la calidad del producto. A continuación se explica cómo fueron desarrolladas las funcionalidades del sistema y las pruebas que desarrollaron para detectar las no conformidades y erradicarlas.

3.1 Arquitectura de software

Eugenia Bahit en (53) emite el siguiente concepto: La Arquitectura de Software es la forma en la que se organizan, interactúan y se relacionan entre sí los componentes de un sistema. Aplica normas y principios de diseño y calidad que fortalecen y fomentan la usabilidad, a la vez que dejan preparado el sistema para su propia evolución. En la presente propuesta de solución se utiliza el marco de trabajo Symfony, el cual implementa el patrón Modelo Vista Controlador (MVC).

3.1.1 Patrón Modelo Vista controlador

El MVC es un patrón de arquitectura de software encargado de separar la lógica de negocio de la interfaz del usuario y es el más utilizado en aplicaciones Web. Facilita la funcionalidad, estabilidad y escalabilidad del sistema, de forma simple y sencilla, a la vez que permite “no mezclar lenguajes de programación en el mismo código”. (53)

El MVC divide las aplicaciones en tres niveles de abstracción (53):

Modelo: representa la lógica de negocios. Es el encargado de la abstracción de los datos, permitiendo que la vista y las operaciones internas de la aplicación sean independientes del tipo de gestor de bases de datos utilizado.

Vista: es la encargada de mostrar la información al usuario de forma gráfica.

Controlador: es el intermediario entre la vista y el modelo. Controla las peticiones del usuario solicitando los datos al modelo y entregándolos a la vista.

3.2 Patrones de diseño

Un patrón es una pareja de problema/solución con un nombre y que es aplicable a otros contextos, con una sugerencia sobre la manera de usarlo en situaciones nuevas. Estos codifican principios y sugerencias relacionados con la asignación de responsabilidades. (54)

3.2.1 Patrones GRASP

Los patrones GRASP (*General Responsability Assignment Software Patterns*) describen la asignación de responsabilidades a objetos, expresados en forma de patrones. (54)

En el desarrollo del sistema se utilizaron los siguientes patrones GRASP:

- ✓ **Experto:** según Craig, este patrón se encarga de asignar una responsabilidad al experto en información, es decir a la clase que cuenta con la información necesaria para cumplir la responsabilidad. La utilización de este en la propuesta de solución se hace evidente mediante el empleo del ORM Doctrine. Symfony2 realiza una abstracción de las tablas de la base de datos mediante clases php. Al mismo tiempo les asocia un repositorio, generalmente bajo el nombre de *entity_nameRepository*, el cual es otra clase php encargada de manipular los datos.
- ✓ **Alta cohesión:** es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme (54). En la aplicación se evidencia el uso de este patrón en varias clases, a continuación se describe un ejemplo: la entidad ***TrabajadorController*** contiene el método ***createAction()***, encargado de crear una instancia de ***FormBuilder***. Esta clase se ocupa de capturar los datos de las peticiones y asociárselos a la entidad que se especifique a través del método ***bind()***. De esta forma se garantiza que ***TrabajadorController***, se limite solo a gestionar la información que le concierne**.**
- ✓ **Bajo acoplamiento:** el acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases. Mientras menos componentes dependan de otro, más reusable y flexible se vuelve el sistema (54). A continuación se expone un ejemplo que evidencia el uso de este patrón para la case ***Evaluacion***: la gestión de esta entidad se divide según las capas de MVC. La interfaz de esta se maneja en un fichero llamado ***index.html.twig***. Las acciones, y peticiones para la misma se manejan desde la clase ***EvaluacionController***. Mientras las operaciones sobre la base de datos, las realiza la clase ***EvaluacionRepository***. De esta forma se garantiza que la modificación de cualquiera de ellas no afecte el funcionamiento de las demás.
- ✓ **Controlador:** este patrón se evidencia cuando las clases controladoras convierten un evento realizado por un usuario en una llamada a un método del modelo específico, convirtiéndose de esta forma en controladoras del flujo de eventos generados en el sistema. El uso de este se evidencia en la clase ***PdfController***. La misma se encarga de capturar las peticiones de exportar a pdf a partir de la petición realizada. Esta identifica el tipo de reporte y genera un flujo de operaciones que permiten obtener los datos deseados de la base de datos, y exportarlos en fichero pdf.

3.2.2 Patrones GOF

En el libro *Design Patterns: Elements of Reusable Object-Oriented Software" (Gamma et al, 1994),* se presenta un conjunto de 23 patrones de diseño identificados a partir del estudio y la experiencia del grupo GOF *(The Gang of Four)*, quienes se dedicaron a analizar los problemas recurrentes en el desarrollo de software y realizaron una clasificación y agrupación a partir de dos criterios, su propósito y alcance, las categorías definidas son: creacionales, estructurales y de comportamiento.

Estructurales

Los patrones estructurales se refieren a como las clases y los objetos son organizados para conformar estructuras más complejas. A continuación se muestran los patrones utilizados:

- ✓ **Decorador:** añade funcionalidad a una clase de forma dinámica y transparente. En la aplicación el archivo base.html.twig o plantilla global almacena el código HTML que es común a todas las páginas de la herramienta, de ella hereda el archivo *frontend.html.twig* que es el que contiene el diseño general de la herramienta para no tener que repetirlo en cada página.

Comportamiento

Los patrones de comportamiento son aquellos que se centran en los algoritmos y en la asignación de responsabilidades entre los objetos.

- ✓ **Estrategia:** un ejemplo de su utilización es cuando se listan los usuarios para realizar la evaluación anual; si el usuario autenticado tiene el rol Gerente general se mostrarán los trabajadores, jefes de área, por otra parte, si tiene el rol Jefe de área, entonces se mostrarán los trabajadores del área a la que dirige.

3.3 Tareas de ingeniería

La metodología XP dentro de la etapa de planificación de la iteración, plantea que el cliente elige el conjunto de historias de usuario de mayor valor para que estas sean implementadas en la iteración planeada. Luego, son divididas en tareas más específicas denominadas tareas de ingeniería (TI) o tareas de programación. Estas se plasman en tarjetas de papel donde se describe qué se debe realizar y son muy dinámicas y flexibles. También pueden ser cambiadas por otras más generales o más específicas, agregarse nuevas o modificarse según las necesidades existentes. Cada una de estas tareas podrá ser comprobada a través de los casos de prueba. Luego, cada programador elige las tareas que desea implementar, las analiza en mayor detalle y realiza una estimación de su tiempo de desarrollo. Finalmente el cliente ordena, según sus necesidades, las HU estimadas, dejando para iteraciones posteriores las que sobrepasan la capacidad productiva de la iteración. (16) (51)

Para la elaboración de las TI del presente trabajo, se utilizó la siguiente plantilla con los parámetros correspondientes:

Tabla 21: Plantilla general para las tareas de ingeniería

Tarea

Número de tarea: número consecutivo a partir del 1.	**Número de HU:** identifica la HU**.**
Nombre de la tarea: identifica la TI	
Tipo de tarea: Desarrollo/ Configuración/ Mejora/ Otra.	**Puntos estimados:** permite estimar la duración de la implementación, donde un 1 equivale a una semana (0.2 equivale a 1 día).
Fecha inicio: comienzo de la tarea**.**	**Fecha fin:** fin de la tarea**.**
Programador responsable: persona encargada de la realización de la TI.	
Descripción: explica en qué consiste la tarea de ingeniería.	

Para la primera iteración se definieron las siguientes TI:

HU.1 Gestionar trabajador

- ✓ TI.1 Configuración de la HU Gestionar trabajador.
- ✓ TI.2 Implementación de la funcionalidad Gestionar trabajador.

HU.2 Gestionar área

- ✓ TI.3 Configuración de la HU Gestionar área.
- ✓ TI.4 Implementación de la funcionalidad Gestionar área.

HU.3 Gestionar cargo

- ✓ TI.5 Configuración de la HU Gestionar cargo.
- ✓ TI.6 Implementación de la funcionalidad Gestionar cargo.

HU.4 Autenticar usuario

- ✓ TI.7 Implementación de la funcionalidad Autenticar usuario.

HU.5 Gestionar indicador de eficiencia

- ✓ TI.8 Configuración de la HU Gestionar indicador de eficiencia.
- ✓ TI.9 Implementación de la funcionalidad Gestionar indicador de eficiencia.

Para la segunda iteración se definieron las siguientes TI:

HU.6 Gestionar indicador específico

- ✓ TI.10 Configuración de la HU Gestionar indicador específico.
- ✓ TI.11 Implementación de la funcionalidad Gestionar indicador específico.

HU.7 Gestionar evaluación de desempeño

- ✓ TI.12 Configuración de la HU Gestionar evaluación de desempeño.
- ✓ TI.13 Implementación de la funcionalidad Gestionar evaluación de desempeño.

HU.8 Mostrar evaluaciones

- ✓ TI.14 Implementación de la funcionalidad Mostrar evaluaciones.

HU.9 Gestionar valores de entrada

- ✓ TI.15 Configuración de la HU Gestionar valores de entrada.
- ✓ TI.16 Implementación de la funcionalidad Gestionar valores de entrada.

Para la tercera iteración se definieron las siguientes TI:

HU.10 Gestionar método de pago

- ✓ TI.17 Configuración de la HU Gestionar método de pago.
- ✓ TI.18 Implementación de la funcionalidad Gestionar método de pago.

HU.11 Efectuar pago

- ✓ TI.19 Implementación de la funcionalidad Efectuar pago.

HU.12 Mostrar salario histórico a los trabajadores

- ✓ TI.20 Implementación de la funcionalidad Mostrar salario histórico de pago a los trabajadores.

HU.13 Mostrar monto a pagar histórico en la división

- ✓ TI.21 Implementación de la funcionalidad Mostrar monto a pagar histórico en la división.

Para la cuarta iteración se definieron las siguientes TI

HU.14 Crear reporte del cumplimiento de la empresa

- ✓ TI.22 Implementación de la funcionalidad Crear reporte del cumplimiento de la empresa.

HU.15 Mostrar datos estadísticos en forma de gráfico

- ✓ TI.23 Implementación de la funcionalidad Mostrar datos estadísticos en forma de gráfico.

HU.16 Exportar a pdf

- ✓ TI.24 Implementación de la funcionalidad Exportar a pdf.

HU.17 Exportar a Excel

- ✓ TI.25 Implementación de la funcionalidad Exportar a Excel.

A continuación se muestran algunas TI desarrolladas en la presente investigación, las restantes se encuentran en el **Anexo VI**.

Tabla 22: TI configuración de la HU Gestionar trabajador

Tarea	
Número de tarea : 1	**Número de HU :**1
Nombre de tarea: configuración de la HU Gestionar trabajador	
Tipo de tarea: configuración	**Puntos estimados:** 1 día
Fecha de inicio :10/3/2015	**Fecha fin :** 10/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se seleccionan los datos con los que se debe trabajar en esta funcionalidad.	

Tabla 23: TI implementación de la funcionalidad Gestionar trabajador

Tarea	
Número de tarea : 2	**Número de HU :** 1
Nombre de tarea: implementación de la funcionalidad Gestionar trabajador	

Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio :13/3/2015	**Fecha fin :**14/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite crear un trabajador con todos los datos asociados al mismo. De igual forma, brinda la posibilidad de adicionar un trabajador, modificarlo, mostrarlo y eliminarlo.	

Tabla 24: TI Configuración de la HU Gestionar Área

Tarea	
Número de tarea : 3	**Número de HU :**2
Nombre de tarea: Configuración de la HU Gestionar área	
Tipo de tarea: configuración	**Puntos estimados:** 1 día
Fecha de inicio :14/3/2015	**Fecha fin :** 15/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se seleccionan los datos con los que se debe trabajar en esta funcionalidad.	

Tabla 25: TI implementación de la funcionalidad Gestionar Área

Tarea	
Número de tarea : 4	**Número de HU :** 2
Nombre de tarea: implementación de la funcionalidad Gestionar Área	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio:15/1/2015	**Fecha fin :**16/1/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite crear un área de trabajo con todos los datos asociados a un área. De igual forma, brinda la posibilidad de adicionar un área, modificarla, mostrarla y eliminarla.	

Tabla 26: TI configuración de la HU Gestionar cargo

Tarea	
Número de tarea: 5	**Número de HU :**3
Nombre de tarea: Configuración de la HU Gestionar cargo	
Tipo de tarea: configuración	**Puntos estimados:** 1 día
Fecha de inicio:11/1/2015	**Fecha fin :** 11/1/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se seleccionan los datos con los que se debe trabajar en esta funcionalidad.	

Tabla 27: TI implementación de la funcionalidad Gestionar cargo

Tarea	
Número de tarea: 6	**Número de HU :** 3
Nombre de tarea: implementación de la funcionalidad Gestionar cargo	

Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio:13/1/2015	**Fecha fin :**14/1/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite crear un cargo para el trabajador con todos los datos asociados al mismo. De igual forma, brinda la posibilidad de adicionar un cargo, modificarlo, mostrarlo y eliminarlo.	

3.4 Estándares de codificación

Los estándares de codificación son reglas que se siguen para la escritura del código fuente. De tal manera que a los programadores se les facilite entender el código escrito por otros. Como propósito fundamental tienen que el código tenga un estilo consistente, independiente del autor, permitiendo que el sistema resulte fácil de entender y a la misma vez fácil de mantener (55).Forman parte de una de las buenas prácticas que propone la metodología XP, donde todos los programadores deben escribir y documentar el código de la misma manera.

Los estándares de codificación son un complemento a la programación por pares que deben permitir:

- ✓ Clarificar más que confundir.
- ✓ Promover la intención del código.
- ✓ Permitir que los programas se acerquen lo mejor posible al lenguaje natural.
- ✓ Incorporar las mejores prácticas de la codificación.

Para la implementación del sistema se siguió un estándar de codificación que se encuentra dentro de los elementos generales que conforman un estilo de código que es la notación *camel*. Este tipo de notación cuenta con dos variantes:

- ✓ *UpperCamelCase, CamelCase o PascalCase*: En esta variante la primera letra también es mayúscula.

En la presente propuesta de solución este estándar se evidencia en el nombre de todas las clases, estas comienzan con la primera letra en mayúscula y las demás en minúscula, por ejemplo: Usuario. En caso de ser un nombre compuesto se escribe el segundo nombre seguido del primero, también con la primera letra mayúscula y el resto en minúscula, ejemplo: ***TrabajadorController***, ***MetodologiaController***.

- ✓ *lowerCamelCase, camelCase o dromedaryCase*: La primera letra es minúscula.

En la presente propuesta de solución este estándar se evidencia en el nombre de los métodos, los cuales comienzan con la primera letra en minúscula. En caso de ser un nombre compuesto se escribe el segundo nombre seguido del primero con la primera letra mayúscula y el resto en minúscula, ejemplo: ***establecerPagoAction()***.

3.5 Seguridad del sistema

La seguridad de los sistemas es uno de los aspectos más importantes a la hora de enfrentar el desarrollo de un sistema informático. Es definida como el conjunto de métodos, herramientas y

técnicas destinadas para proteger la integridad, viabilidad y disponibilidad del sistema (56). En aras de garantizar una correcta protección de los datos de los trabajadores asociados a la división Norte de la empresa COPEXTEL, se definió un control de acceso a nivel de usuario y contraseña.

El *framework* escogido en la presente investigación para el desarrollo del sistema, contiene un *bundle* que posee varios métodos para tener una aplicación segura, *Symfony* contempla la seguridad como un proceso de dos etapas, en la primera el sistema de seguridad identifica quién es el usuario obligándolo a presentar algún tipo de identificación. Este proceso se conoce como autenticación.

Una vez que el sistema sabe quién desea autenticarse, el siguiente paso es determinar si el usuario puede tener acceso a un determinado recurso. Esta parte del proceso se denomina autorización, y significa que el sistema está comprobando para ver si tiene suficientes privilegios para realizar una determinada acción.

3.6 Pruebas

Las pruebas de software constituyen un aspecto fundamental para garantizar la calidad del software, representan una revisión final de las especificaciones del diseño y de la codificación. Las pruebas son una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados (50). XP enfatiza mucho los aspectos relacionados con las pruebas, indicando quien, cuando y como deben ser implementadas y ejecutadas (57). Esta metodología divide las pruebas al sistema en dos grupos, las pruebas de unitarias que son diseñadas por los programadores y se encargan de verificar el código y las pruebas de aceptación, que se crean de acuerdo a las HU en cada una de las iteraciones, en las mismas el cliente verifica que los resultados de estas pruebas sean correctos. En caso de fallar varias pruebas se debe indicar el orden de prioridad de resolución.

3.6.1 Pruebas unitarias

Las pruebas unitarias o pruebas de caja blanca requieren del conocimiento de la estructura interna del programa y son derivadas a partir de las especificaciones internas de diseño o el código. Este tipo de pruebas constituyen la piedra angular de la metodología XP según José Joscowickz. Estas pruebas deben ser definidas antes de realizar el código, y se deben guardar junto a este en caso de futuras correcciones y actualizaciones. Cuando se encuentren errores, deben ser corregidos inmediatamente, y se deben definir nuevas pruebas para verificar que el error haya sido resuelto. (50)

Mediante los métodos de prueba de la caja blanca, los desarrolladores pueden obtener casos de prueba que garanticen los siguientes aspectos:

- ✓ Se ejerciten por lo menos una vez todos los caminos independientes para cada módulo.
- ✓ Se ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsa.
- ✓ Ejecuten todos los bucles en sus límites y con sus límites operacionales.
- ✓ Se ejerciten las estructuras internas de datos para asegurar su validez.

El *framework* Symfony 2.3.1, utilizado en la presente investigación utiliza la biblioteca *PHPUnit* para realizar los *test* (pruebas), esta constituye una herramienta para realizar pruebas de caja blanca a las aplicaciones web desarrolladas en PHP. Requiere de un amplio conocimiento del lenguaje, pues son escritas a código. De esta forma los *test* unitarios de Symfony 2.3.1 combinan la potencia de *PHPUnit* con las utilidades y facilidades proporcionadas por dicho *framework*.

Esta herramienta permitió a los desarrolladores de la presente investigación probar y corregir cada método al momento de ser elaborado. La cantidad de pruebas realizadas por entidad fueron las siguientes: Trabajador (3), Evaluacion (3), Calculo (2), Area (1), Cargo (1), IndicadorEspecifico (1), Metodologia (1), y para la portada (2).Se realizaron un total de 14 pruebas con 43 aserciones resultado todas satisfactorias. A continuación se presentan imágenes del código de las pruebas efectuadas, así como la imagen del resultado de la ejecución de la misma.

```
46          /** @test */
47      public function direccionaVistaTrabajador()
48      {
49          $client = static::createClient();
50
51          //SUT
52          $client->request('GET', '/super/trabajador');
53
54          $this->assertEquals(200, $client->getResponse()->getStatusCode(),
55              'El listado de trabajadores se genera correctamente.'
56          );
57      }
```

Fig. 7 Código de la prueba unitaria para la vista listar trabajador

```php
class TrabajadorControllerTest extends WebTestCase
{
    public function testCompleteScenario()
    {
        // Creando una nuevo cliente en el navegador
        $client = static::createClient();
        // Creando una nueva entrada en la base de datos
        $crawler = $client->request('GET', '/trabajador/');
        $this->assertEquals(200, $client->getResponse()->getStatusCode(), "Unexpected HTTP status code for GET /trabajador/");
        $crawler = $client->click($crawler->selectLink('Create a new entry')->link());
        // Haciendo submit con el formulario adicionar
        $em = $this->getDoctrine()->getManager();
        $area = $em->getRepository('SistemaBundle:Area')->Find(0);
        $cargo = $em->getRepository('SistemaBundle:Cargo')->Find(5);
        $usuario = new Usuario();
        $usuario->setUsername('juan');$usuario->setPassword('1');
        $usuario->setEnabled();
        $form = $crawler->selectButton('Create')->form(array(
            'proyecto_pagobundle_trabajadortype[nombres]'  => 'Juan',
            'proyecto_pagobundle_trabajadortype[apellidos]'  => 'Perez',
            'proyecto_pagobundle_trabajadortype[ci]'  => '75030177231',
            'proyecto_pagobundle_trabajadortype[areaid]'  => $area,
            'proyecto_pagobundle_trabajadortype[cargoid]'  => $cargo,
            'proyecto_pagobundle_trabajadortype[usuarioid]'  => $usuario,
        ));
        $client->submit($form);$crawler = $client->followRedirect();
        // Checkeando los datos insertados en la vista show
        $this->assertGreaterThan(0, $crawler->filter('td:contains("Juan")')->count(), 'Missing element td:contains("Juan")');
```

Fig. 8 Código del método de la prueba unitaria del escenario Trabajador

```php
class DefaultControllerTest extends WebTestCase
{
    /** @test */
    public function laPortadaSeGeneraCorrectamente()
    {
        $client = static::createClient();

        //SUT
        $client->request('GET', '/usuario/portada');

        $this->assertEquals(200, $client->getResponse()->getStatusCode(),
            'La portada se genera correctamente.'
        );
    }
    /** @test */
    public function laPortadaBrindalaOpcionSalir()
    {
        $client = static::createClient();
        $crawler = $client->request('GET', '/usuario/portada');

        //SUT
        $numeroEnlacesSalir = $crawler->filter('html:contains("Salir")')->count();

        $this->assertEquals(1, $numeroEnlacesSalir,
            'La portada muestra un enlace o botón para salir'
        );
    }
```

Fig. 9 Código del método de la prueba unitaria realizada a la portada

```php
class IndicadorEspecificoControllerTest extends WebTestCase
{

    public function testCompleteScenario()
    {
        // Creando una nuevo cliente en el navegador
        $client = static::createClient();

        // Creando una nueva entrada en la base de datos
        $crawler = $client->request('GET', '/indicadorespecifico/');
        $this->assertEquals(200, $client->getResponse()->getStatusCode(), "Unexpected HTTP status code "
                                                                          . "for GET /indicadorespecifico/");
        $crawler = $client->click($crawler->selectLink('Create a new entry')->link());
        // Haciendo submit con el formulario adicionar
        $em = $this->getDoctrine()->getManager();
        $eficiencia = $em->getRepository('SistemaBundle:IndicadorEficiencia')->Find(1);
        $form = $crawler->selectButton('Create')->form(array(
            'proyecto_pagobundle_indicadorespecificotype[nomindesp]'  => 'Probar la aplicación',
            'proyecto_pagobundle_indicadorespecificotype[indefiid]'  => $eficiencia,
            'proyecto_pagobundle_indicadorespecificotype[regla]'  => 10,
        ));
        $client->submit($form);
        $crawler = $client->followRedirect();

        // Checkeando los datos insertados en la vista show
        $this->assertGreaterThan(0, $crawler->filter('td:contains("Probar la aplicación")')->count(), 'Missing element'
                                                                                  . ' td:contains("Probar la aplicación")'
```

Fig. 10 Código del método de la prueba unitaria del escenario indicador específico

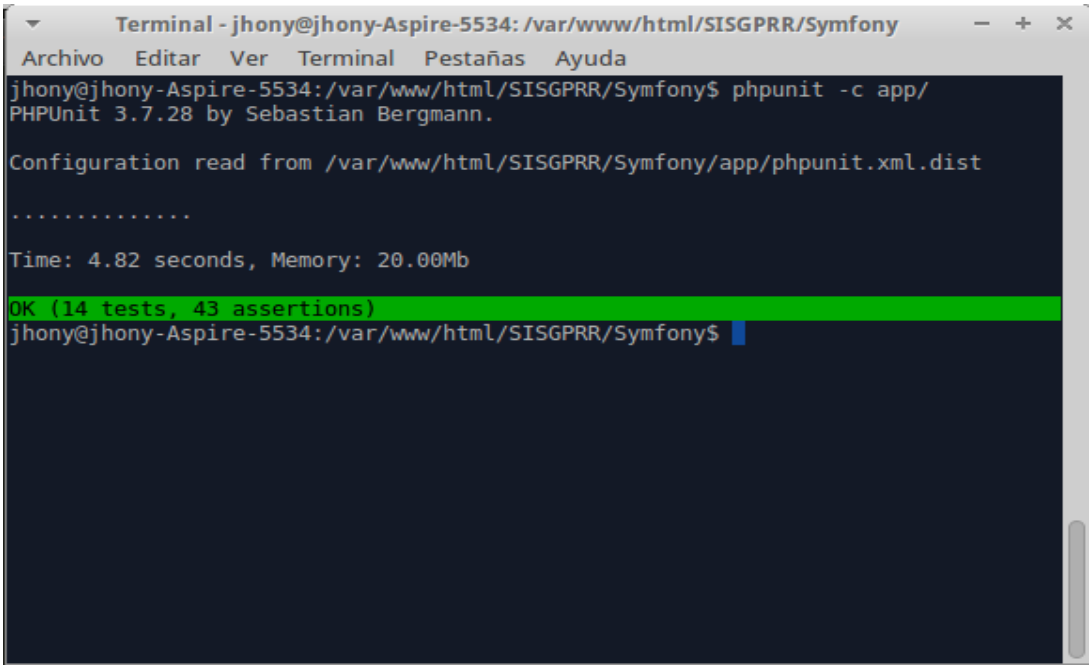


Fig. 11 Resultado de las pruebas unitarias aplicadas

3.6.2 Pruebas de carga y estrés

Este tipo de prueba permite analizar el comportamiento del sistema cuando es sometido a determinadas condiciones de trabajo. Las pruebas de carga brindan información sobre el comportamiento del sistema para un número determinado de peticiones. Mientras las de estrés permiten probar la cantidad máxima de peticiones que el sistema soporta. En la presente propuesta de solución se utiliza la herramienta Apache JMeter para efectuarlas. Esta aplicación puede utilizarse para simular una carga pesada en un servidor, en la red o a un objeto, para poner a prueba su resistencia o analizar el rendimiento en diferentes tipos de carga.

Para ejecutar estas pruebas, se simula el comportamiento de usuarios conectados al sistema desarrollado realizando varias transacciones, y se analizan los resultados para diferentes condiciones.

Las pruebas se llevarán a cabo haciendo uso de un ordenador con:

- ✓ Microprocesador AMD Athlon(tm) X2 Dual Core Processor L310 (2 CPUs), 1.2GHz.
- ✓ Memoria RAM 2GB.

Se considerarán:

- ✓ 10 hilos (simulación de 10 usuarios) y un período de subida de 1 segundo.
- ✓ 30 hilos (simulación de 30 usuarios) y un período de subida de 1 segundo.
- ✓ 50 hilos (simulación de 50 usuarios) y un período de subida de 1 segundo.

Se añadirá el elemento “Informe agregado” que brindará la siguiente información:

Label: El nombre de la muestra (conjunto de muestras).

 # Muestras: El número de muestras para cada URL.

Media: El tiempo medio transcurrido para un conjunto de resultados.

Mín: El mínimo tiempo transcurrido para las muestras de la URL dada.

Máx: El máximo tiempo transcurrido para las muestras de la URL dada.

% Error: Porcentaje de las peticiones con errores.

Rendimiento: Rendimiento medido en base a peticiones por segundo/minuto/hora.

Kb/sec: Rendimiento medido en Kilobytes por segundo.

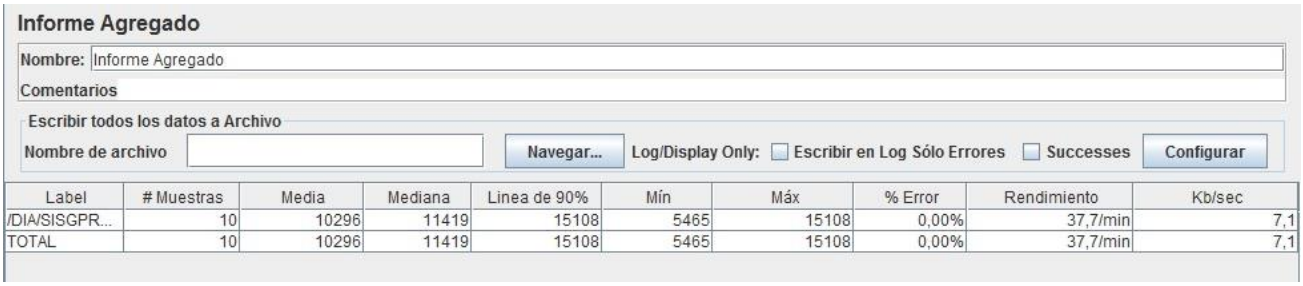
Media de Bytes: Tamaño medio de la respuesta de la muestra medido en bytes.

Tras haber introducido las variaciones indicadas, El elemento “Informe agregado” quedará como sigue:

Simulación de 10 usuarios con un período de subida de 1 segundo, en la que el servidor muestra un rendimiento de 37,7 peticiones por minuto.

Simulación de 30 usuarios con un período de subida de 1 segundo, en la que el servidor muestra un rendimiento de 39,3 peticiones por minuto. **Anexo VIII**

Simulación de 50 usuarios con un período de subida de 1 segundo, en la que el servidor muestra un rendimiento de 19,4 peticiones por minuto. **Anexo VIII**



Label	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
/DIA/SISGPR...	10	10296	11419	15108	5465	15108	0,00%	37,7/min	7,1
TOTAL	10	10296	11419	15108	5465	15108	0,00%	37,7/min	7,1

Fig. 12 Resultado de las pruebas de carga y estrés para 10 usuarios

3.6.3 Pruebas de aceptación

Roger Pressman en (58) define que las pruebas de aceptación o pruebas de caja negra, se centran en los requisitos funcionales del software. Estas permiten al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra no es una alternativa a las técnicas de prueba de caja blanca, se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores. A continuación algunos casos de prueba desarrollados en la presente investigación, los restantes CP se encuentran en el **Anexo VII**.

Tabla 28: CP HU1_P1

Caso de prueba de aceptación	
Código:HU1_P1	**Historia de usuario:**1
Nombre: Adicionar trabajador	
Descripción: prueba para la funcionalidad adicionar trabajador al sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como gerente general.	
Pasos de ejecución: Se despliega la opción Recursos humanos en el panel lateral y se selecciona el submenú trabajadores, se mostrará un listado con los todos los trabajadores, luego se acciona el botón Adicionar, se llenan los datos solicitados, si los datos solicitados no son correctos el sistema le notificará el error, en caso de serlos se añade al sistema.	
Resultado esperado: el sistema redireccionará a la vista mostrar trabajador con los datos insertados.	
Evaluación de la prueba: prueba satisfactoria	

Tabla 29: CP HU1_P2

Caso de prueba de aceptación	
Código: HU1_P2	**Historia de usuario:** 1
Nombre: Mostrar trabajadores	
Descripción: prueba para la funcionalidad que permite mostrar el listado de trabajadores de un área determinada.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	

Pasos de ejecución: se despliega el menú Recursos humanos en el panel lateral y se selecciona el submenú Trabajadores. El sistema mostrará un listado con todos los trabajadores.
Resultado esperado: el sistema debe mostrar el listado de trabajadores existentes.
Evaluación de la prueba: prueba satisfactoria.

Tabla 30: CP HU1_P3

Caso de prueba de aceptación	
Código:HU1_P3	**Historia de usuario:**1
Nombre: Modificar trabajador	
Descripción: prueba para la funcionalidad modificar trabajador en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: se despliega el menú Recursos humanos en el panel lateral y se selecciona el submenú Trabajadores. El sistema mostrará un listado con todos los trabajadores. Luego se acciona el botón Editar, y el sistema mostrará una formulario con los datos del trabajador que se desea editar, se modifican los datos, en caso de ser incorrectos el sistema notificará el error mediante un mensaje, en caso de no serlo se guardarán los datos actualizados en el sistema.	
Resultado esperado: el sistema redireccionará a la vista mostrar trabajador con los datos actualizados.	
Evaluación de la prueba: prueba satisfactoria.	

Tabla 31: CP HU1_P4

Caso de prueba de aceptación	
Código: HU1_P4	**Historia de usuario:** 1
Nombre: Eliminar un trabajador	
Descripción: prueba para la funcionalidad que permite eliminar un trabajador.	
Condiciones de ejecución: ✓ el usuario debe estar autenticado como gerente general. ✓ debe existir al menos un trabajador en un área específica.	

Pasos de ejecución: Se selecciona la opción Recursos humanos en el panel lateral y se selecciona el subíndice trabajadores, luego el botón Eliminar. Seguidamente se mostrará un mensaje para confirmar la eliminación, de ser afirmativo se elimina el trabajador del sistema
Resultado esperado: el trabajador fue eliminado satisfactoriamente.
Evaluación de la prueba: prueba satisfactoria.

3.6.4 Resultados de las pruebas

Las pruebas de aceptación realizadas al sistema se ejecutaron en cuatro iteraciones, una por cada entrega realizada, con el objetivo de verificar que las funcionalidades creadas en cada entrega del *software*. Algunas de estas fueron:

- ✓ Campos sin validar como el carnet de identidad de un trabajador.
- ✓ Error a la hora de realizar el cálculo del coeficiente de distribución salarial.
- ✓ Problemas con la autenticación, al crear un nuevo trabajador el usuario que se le genera se creaba deshabilitado en la base de datos y no permitía su acceso al sistema.
- ✓ Errores ortográficos.
- ✓ La opción exportar a pdf no funcionaba al exportar una evaluación de desempeño.

En las iteraciones se obtuvieron los siguientes resultados:

- ✓ **Primera iteración:** 17 No Conformidades (NC), de ellas 12 significativas y 5 no significativas. Todas fueron resueltas antes de la segunda iteración.
- ✓ **Segunda iteración**: 6 NC, de ellas 3 significativas y 3 no significativas. Estas también fueron solucionadas.
- ✓ **Tercera iteración**: 5 NC, de ellas 4 significativas y 1 no significativa. Estas fueron resueltas antes de la cuarta iteración.
- ✓ **Cuarta Iteración**: No se identificaron NC.

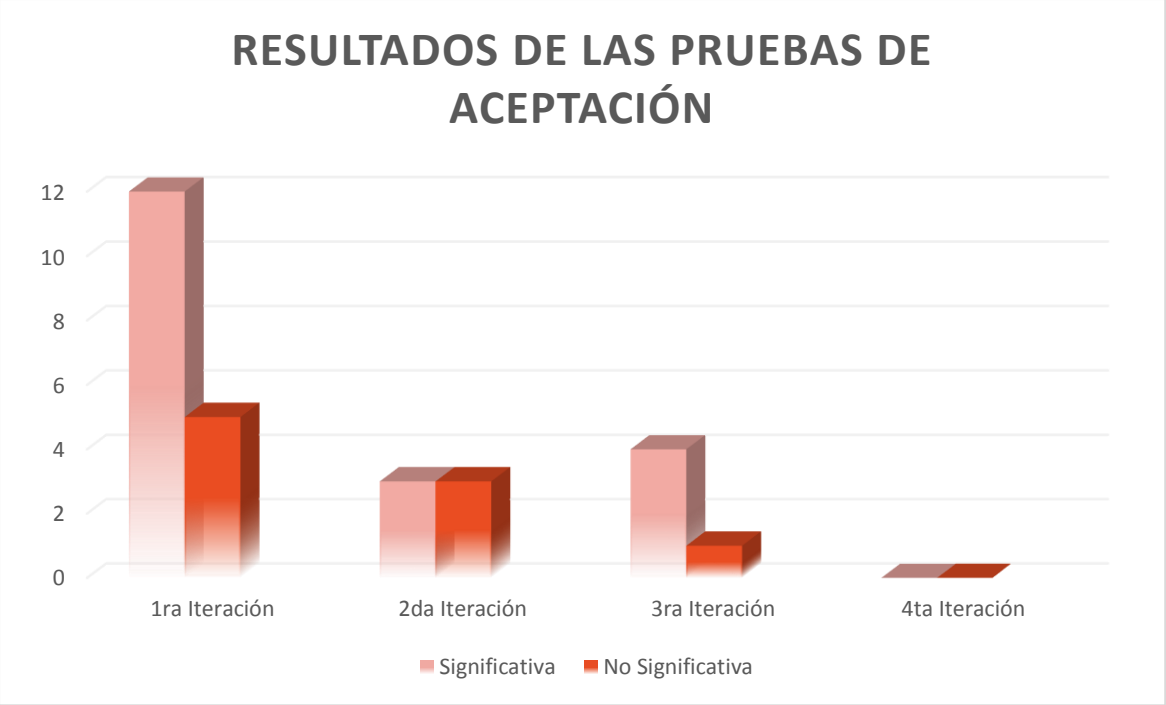


Fig. 13: Resultados de las pruebas de aceptación

3.7 Validación de la investigación

Para la validación de la presente investigación, fue necesario comprobar el cumplimiento de la hipótesis planteada. Esta define el tiempo como variable medible y la reducción de este como premisa fundamental para dar cumplimiento al problema a resolver identificado. La definición de la misma, está condicionada por la descentralización de la información y la duración con la que se realiza el proceso de pago en la división Norte de la empresa COPEXTEL. Este último, se desglosa en un grupo de tareas que constituyen una carga de tiempo para sus responsables.

3.7.1 Métodos para la validación

El método seleccionado para validar la investigación fue el descriptivo, el cual está relacionado con la etapa inicial de la investigación y usualmente involucra la recopilación de datos. El resultado principal es la información que describe lo que fue observado y experimentado. (59)

Para el empleo de este método se tomaron como herramientas de apoyo el uso de una tabla comparativa una encuesta **Anexo IX**. En la confección de la primera, se tuvieron en cuenta las tareas en las que se desglosa el proceso de pago en la división norte de la empresa COPEXTEL, utilizando para un mejor entendimiento de estas un diagrama de procesos **Anexo X**. Como parámetros de comparación se estableció, la realización del pago antes de la existencia del sistema y posterior a esta. Los tiempos de ejecución de cada tarea, se obtuvieron mediante el criterio de algunos de los trabajadores de las áreas gerencia general y gerencia de economía y finanzas, así como los resultados de la encuesta.

Tabla 32: Tabla comparativa del tiempo en el que se realiza el proceso de pago con el sistema de gestión y sin este.

Tarea	Sin Sistema de Gestión	Con Sistema de Gestión
1. Define la forma de pago según la legislación de la empresa.	1 hora	20 minutos
2. Consulta los datos de los trabajadores en el Hércules y los tabula.	3 horas	3 horas
3. Emite las evaluaciones de los trabajadores de su área según la forma de pago definida.	2 días	6 horas
4. Enviar evaluaciones tabuladas de cada trabajador al gerente general.	5 minutos	0
5. Recibe las evaluaciones de cada trabajador de la empresa enviada por los gerentes de áreas y tabula los resultados.	5 minutos	0
6. Consulta el Arcas y define las variables para efectuar el pago.	1 hora	20 minutos
7. Reunión con los gerentes de área para establecer el cierre del mes.	3 días	2 días
8. Se procede a efectuar el pago realizando los cálculos pertinentes.	5 horas	0
9. Se tabulan los resultados y genera la nómina.	1 día	4 horas
Duración total	7 días, 3 horas, 10 minutos	3 días, 5 horas ,40 minutos

Para la confección de la encuesta, se tuvieron en cuenta los siguientes indicadores:

Tabla 31: Indicadores para la encuesta

Indicadores	Descripción
Intuitividad del sistema	Permitirá comprobar que el sistema de gestión de información es lo suficientemente intuitivo como para ser utilizado por personas con conocimientos básicos de informática.
Necesidad de un sistema de gestión para el pago	Permitirá identificar los problemas que hoy influyen en la demora del proceso de pago y

	por consiguiente hacen necesario el desarrollo de un sistema de gestión.
Tiempo que demora en realizarse el proceso de pago	Permitirá ratificar el criterio de los trabajadores involucrados directamente en el pago, acerca del tiempo que demora el proceso.
Familiarización del usuario con el sistema	Permitirá afirmar que el proceso de adopción con el sistema no influirá en la demora de este.

3.7.2 Resultados de la validación

Fig. 14: Gráfica de la intuitividad del sistema a partir de los resultados de la encuesta

Fig. 15: Gráfica de la necesidad del sistema a partir de los resultados de la encuesta

Fig. 16: Gráfica del tiempo que se demora efectuar el pago a partir de los resultados de la encuesta

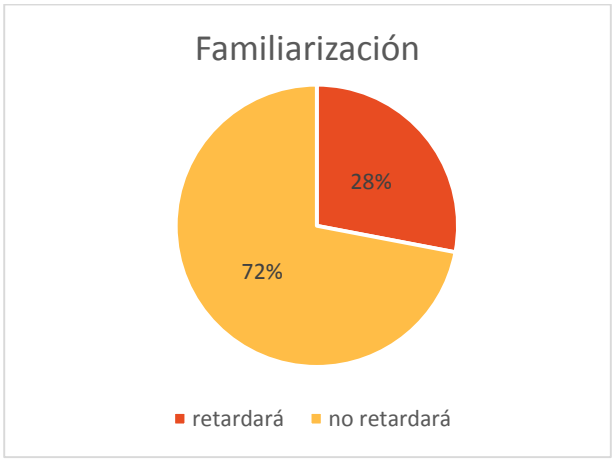


Fig. 17: Gráfica de la familiarización de los usuarios con el sistema a partir de la encuesta

Los tipos de muestras se dividen en dos grandes ramas: las muestras probabilísticas y las no probabilísticas. En estas últimas, la elección de los elementos no depende de la probabilidad, sino de causas relacionadas con las características del investigador o del que hace la muestra. (60)

Para la aplicación de la encuesta, se tomó una muestra no probabilística de 25 trabajadores de la división Norte de la empresa COPEXTEL. Se utilizó un muestreo intencionado, seleccionándose 10 trabajadores de la gerencia de economía y finanzas, 4 de la gerencia general, y el resto de los directivos de las diferentes áreas organizativas, debido a que estos son los que intervienen directamente en proceso de pago. Los resultados contribuyeron a corroborar el planteamiento de la hipótesis, así como la validación del requisito no funcional de usabilidad. Del total de encuestados, 22 consideran que la aplicación es intuitiva para un nivel de satisfacción de 92%, 23 afirman que es necesario un sistema para la gestión del pago para un 96%, todos coincidieron en que el tiempo que demora el proceso de pago es de más de 1 semana y 18 plantean que la familiarización con el sistema no influirá en el tiempo que se demora el pago para un 72%.

Igualmente, la tabla comparativa arrojó resultados que comprueban la eficacia de la solución propuesta. Antes de utilizar la aplicación web, para el realizar el proceso de pago se ejecutaban 9 de

tareas en tiempo estimado de 7 días, con el uso de esta se redujo el número de tareas a 5 pues el sistema gestiona 3 de ellas automáticamente para un tiempo estimado de 4 días. De esta forma se concluyó que la reducción del tiempo fue de 3 días laborables, teniendo en cuenta que un día laborable cuenta con 8 horas. Demostrándose así la validación de la hipótesis planteada.

Conclusiones parciales

- ✓ El diseño propuesto en el capítulo anterior permitió desarrollar las funcionalidades descritas a partir de las herramientas, lenguajes, metodologías y tecnologías seleccionadas.
- ✓ Los patrones de diseño que se utilizaron para la implementación del sistema, permitieron establecer un uso adecuado de buenas prácticas en la implementación de la propuesta de solución.
- ✓ Se especificaron las principales tareas de programación que dieron paso a la implementación del sistema, permitiendo detallar cómo los elementos del diseño se implementan y organizan.
- ✓ Se realizó un proceso de pruebas unitarias y de aceptación al software, lo que permitió detectar y corregir fallas en el funcionamiento del sistema.
- ✓ Se realizaron pruebas de carga y estrés al sistema, que ayudaron a comprobar el rendimiento del sistema cuando se somete a un gran número de peticiones.
- ✓ El uso de técnicas como el muestreo intencionado, la encuesta y la entrevista permitieron validar la presente propuesta de solución.

.

Conclusiones generales

Como resultado de la presente investigación se obtuvieron las siguientes conclusiones:

- ✓ El análisis de las soluciones similares permitió obtener funcionalidades útiles para el desarrollo del sistema propuesto, a partir de requisitos que mostraron similitud con la lógica del negocio.
- ✓ El estudio realizado durante la presente investigación permitió seleccionar correctamente las herramientas y tecnologías para el desarrollo del sistema, el cual se sustentó en las bases de la metodología XP.
- ✓ El diseño realizado permitió obtener un sistema que cumpliera con las funcionalidades definidas y que dan solución al objeto de estudio.
- ✓ El proceso de pruebas de aceptación demostró la correcta implementación de sus funcionalidades dando cumplimiento a las necesidades del cliente. En tanto las pruebas de carga y estrés permitieron comprobar la estabilidad del sistema.

Como resultado de la investigación, se logró desarrollar un sistema con el que se da cumplimiento a los objetivos propuestos en la presente investigación.

Recomendaciones

A partir del trabajo realizado y después de haber analizado los resultados obtenidos se recomienda:

- ✓ Añadir otras funcionalidades acorde a las nuevas necesidades que surjan en la empresa. Como por ejemplo un módulo que consuma servicios de los sitios Arcas y Hércules, y los integre al sistema, evitando la entrada de datos como el plan de ventas de cada trabajador, y de la división. Para lograr de esta forma hacer más eficiente el proceso de pago por resultados.
  .

Referencias bibliográficas

1. Zayas, Carlos Alvarez de. Metodología de Investigación Científica. Santiago de Cuba : Centro de Estudios de educación superior Manuel F Gran, 1995.

2. Davis G., Olsón. Management Information Systems: Conceptual foundations, Structure and Development. 2a ed. Nueva York: McGrawhill : s.n., 1985.

3. Technology Management. *https://tm.soe.ucsc.edu/undergraduates.* [En línea] Universidad de California. [Citado el: 31 de enero de 2015.] https://tm.soe.ucsc.edu/undergraduates.

4. Quiroga, Lic. Lourdes Aja. Gestión de información,gestión del conocimiento y gestión de la calidad en las organizaciones. [En línea] [Citado el: 31 de enero de 2015.] .] http://bvs.sld.cu/revistas/aci/vol10_5_02/aci04502.htm.

5. Eurosistema. *Sistemas de pago Banco de españa.* España : s.n., 2014.

6.http://www.eumed.net/cursecon/ecolat/cu/2010/acpr.htm. *http://www.eumed.net/cursecon/ecolat/cu/2010/acpr.htm.* [En línea] [Citado el: 15 de octubre de 2014.] http://www.eumed.net/cursecon/ecolat/cu/2010/acpr.htm.

7. pyme.lavoztx.com. *pyme.lavoztx.com.* [En línea] pyme.lavoztx.com.

8. Sosa San Millán. "Sistema para la gestión de pagos en la empresa gráfica "Juan Marinello" de Guantánamo". [En línea] 2012. [Citado el: 20 de Enero de 2015.] http://www.eumed.net/cursecon/ecolat/cu/2012/.

9. COPEXTEL. *Versión reglamento Sistema de Pago por Resultado.* 2014.

10. Pendulum Associates. Sistema de Gestión de Pago SGP. *Sistema de Gestión de Pago SGP.* [En línea] [Citado el: 12 de febrero de 2015.] http://www.pendulum.com.mx/.

11. Rojas, Ing.Roberto Crespo. Automatización de los sistemas de pago por resultados de la Empresa Avícola Santa Clara. [En línea] 2013. http://www.monografias.com.

12. Riola, Jose Carlos Carvajal. *METODOLOGÍAS ÁGILES:HERRAMIENTAS Y MODELO DE DESARROLLO PARA APLICACIONES.* 2008.

13. Gómez, Juan. *Fundamentos de la metodología RUP.* Septiembre-2007.

14. Palacio, Juan. *Flexibilidad con SCRUM.* 2007.

15. Proyectos Agiles. *Proyectos Agiles.* [En línea] [Citado el: 14 de enero de 2015.] http://www.proyectosagiles.org/que-es-scrum. http://www.proyectosagiles.org/que-es-scrum.

16. Beck, Kent. *Extreme Programming Explained.* 1999.

17. Elizabet De la Cruz Rodríguez, Daliamny Guzmán Hernández. *Propuesta de herramienta CASE para los proyectos del centro de Desarrollo de la Informática Industrial.* La Habana : CEDIN, 2010.

18. Umbrello UML Modeller. Umbrello. *Umbrello.* [En línea] [Citado el: 16 de enero de 2015.] https://docs.kde.org/stable/es/kdesdk/umbrello/.

19. Gnome.org. *Gnome.org.* [En línea] [Citado el: 16 de Febrero de 2015.] https://wiki.gnome.org/Apps/Dia/.

20. Visual Paradigm for UML. *Visual Paradigm for UML.* [En línea] [Citado el: 2 de 2 de 2015.] http://www.visual-paradigm.com/product/vpuml/..http://www.visual-paradigm.com/product/vpuml/..

21. Mora, Sergio Luján. *Programación de aplicaciones web:historia,principios básicos y clientes web.*

22. Microsoft. Microsoft. *Microsoft.* [En línea] [Citado el: 20 de Febrero de 2015.] Un lenguaje de programación actúa como un traductor entre el usuario y el equipo. En lugar de aprender el lenguaje nativo del equipo (conocido como lenguaje máquina), se puede utilizar un lenguaje de programación para dar instrucciones al equipo de un mod.

23. Programación Web. *Programación Web.* [En línea] Instituto Technológico de Matehuala. [Citado el: 23 de Febrero de 2015.] https://programacionwebisc.wordpress.com/2-2-lenguajes-de-programacion-del-lado-del-cliente/.

24. Muciano Chuck, Kennedy Bill. *HTML La Guía Completa.* México : Litografica ingramex, 1999. ISBN 970-10-2141-X.

25. González Martins, Miguel Angel. *Los lenguajes de programación actuales.* s.l. : Universidad de Alcalá de Henares, 2013.

26. PHP. *PHP.* [En línea] [Citado el: 14 de enero de 2015.] http://www.php.net. http://www.php.net.

27. Vázquez, Ing. Iván Mendoza. SEDICI. *SEDICI.* [En línea] 2011. [Citado el: 12 de marzo de 2015.] Tesis de maestría. http://hdl.handle.net/10915/4192.

28. Alvarez, Miguel Angel. *Manual de jQuery.* Manual.

29. Alba, Teresa. Diseño Creativo. *Diseño Creativo.* [En línea] 13 de Agosto de 2014. [Citado el: 12 de mayo de 2015.] http://diseñocreativo.com/10-razones-para-desarrollar-una-web-con-bootstrap/.

30. Phalcon PHP. *Phalcon PHP.* [En línea] [Citado el: 18 de febrero de 2015.] http://docs.phalconphp.com/en/latest/index.html.

31. Yii PHP Framework. *Yii PHP Framework.* [En línea] [Citado el: 18 de Febrero de 2015.] http://www.yiiframework.com/doc/guide/1.1/es/quickstart.what-is-yii.

32. SensioLabsNetwork. Symfony. *Symfony.* [En línea] [Citado el: 19 de Febrero de 2015.] http://symfony.com/es/about.

33. Programación. *Programación.* [En línea] [Citado el: 27 de mayo de 2015.] http://programacion.net/articulo/conceptos_basicos_de_orm_object_relational_mapping_349 .

34. Doctrine. Doctrine. *Doctrine.* [En línea] Doctrine Team. [Citado el: 28 de mayo de 2015.] http://www.doctrine-project.org/.

35. Apache. *Apache.* [En línea] [Citado el: 15 de enero de 2015.] http://httpd.apache.org/docs/2.0/new_features_2_0.html. http://httpd.apache.org/docs/2.0/new_features_2_0.html.

36. Pecos, Daniel. [En línea] [Citado el: 5 de 2 de 2015.] http://www.danielpecos.com/docs/mysql_postgres/x57.html. http://www.danielpecos.com/docs/mysql_postgres/x57.html.

37. Denzer, Patricio. *PostgresSQL.* 2002.

38. Garrido, Jesús Manuel Montero. *Plataforma Eclipse.Introducción técnica.*

39. Eclipse. Eclipse. *Eclipse.* [En línea] [Citado el: 20 de Febrero de 2015.] http://eclipse.org.

40. Mukund Chaudhary, Ankur Kumar. *PhpStorm Cookbook.* Primera. s.l. : Packt Publishing Ltd., 2014. pág. 238. ISBN 978-1-78217-387-8.

41. Up to dawn. *Up to dawn.* [En línea] [Citado el: 15 de enero de 2015.] http://netbeans-ide.uptodown.com/.. http://netbeans-ide.uptodown.com/..

42. LoadUI. LoadUI. *LoadUI.* [En línea] [Citado el: 28 de mayo de 2015.] http://www.loadui.org/.

43. GreenSQA. GreenSQA. *GreenSQA.* [En línea] [Citado el: 28 de mayo de 2015.] http://www.greensqa.com/herramientas.html.

44. Apache Software Foundation . Apache JMeter. *Apache JMeter.* [En línea] Apache. [Citado el: 28 de Mayo de 2015.] http://jmeter.apache.org/.

45. Definicion. *http://definicion.mx/objetivo/#ixzz3WjrAgxDB.* [En línea] 25 de marzo de 2015. http://definicion.mx/objetivo/#ixzz3WjrAgxDB.

46. Hidalgo, Sergio Pozo. *Técnicas Automáticas para la Diagnosis de Consistencia y Conformidad en políticas de control de acceso.* Departamento de lenguajes y Sistemas Informáticos de la Universidad de Sevilla.

47. Analisis y desarrollo de sistemas de informacion. ***Analisis y desarrollo de sistemas de informacion.*** **[En línea] 15 de abril de 2015. [Citado el: 12 de abril de 2015.] http://desasof2004.blogspot.com/2009/06/definicion-de-requerimientos.html.**

48. Sommerville, Ian. ***Ingeniería de Software,Séptima Edición.*** **Madrid : Pearson Education,S.A, 2005.**

49. Programación Extrema. [En línea] http://www.programacionextrema.org/.

50. Joskowicz, José. ***Reglas y prácticas en eXTREME Programming.*** **España : s.n., 2008.**

51. Beck, Kent. ***Una explicación de la programación extrema:aceptar el cambio.*** **s.l. : Addison Wesley Iberoamericana Espanya, S.A, 2002. pág. 216. ISBN:8478290559.**

52. Casas, Sandra y Reinaga, Héctor. ***Identificación y Modelado de los Aspectos Tempranos por Tarjetas de Responsabilidad y Colaboraciones.*** **Santa Cruz, Argentina : s.n.**

53. Bahit, Eugenia. ***POO y MVC en PHP.***

54. Larman, Craig. ***UML y Patrones Introducción al análisis y diseño orientado a objetos.*** **México : Prentice Hall Hispanoamericana, S.A, 1999. ISBN 970-1 7-0261-1.**

55. Calleja, Manuel Arias. ***Carmen.Estándares de Codificación.*** **Departamento de Inteligencia Artificial, Universidad Nacional de Educación a Distancia UNED. Madrid : s.n. Documento.**

56. Pflegeer, Charles P. ***Security in Computing.*** **Cuarta. Pensylvania,EE UU : Prentice Hall, 2006. pág. 880. ISBN-10:0-13-239077-9.**

57. Luis Miguel Echeverry Tobón, Luz Elena Delgado Carmona. ***Caso práctico de la metodología ágil XP al desarrollo de software.*** **Facultad de ingeniería:eléctrica,electrónica,física y ciencias de la computación, Universidad Tecnológica de Pereira. Pereira : s.n., 2007. pág. 110, Tesis para optar al título de Ingeniero en Sistemas y Computación.**

58. Pressman, Roger. ***Ingeniería de Software.*** **México : McGraw-Hill/INTERAMERICANA DE MEXICO, 2005. ISBN: 9789701054734.**

59. Medina, Manuel Ildefonso Ruiz. eumed.net. ***eumed.net.*** **[En línea] Universidad de Málaga, 28 de mayo de 2015. [Citado el: 28 de mayo de 2015.] http://www.eumed.net/tesis-doctorales/2012/mirm/validacion_confiabilidad.html.**

60. Hernández Sampieri, Roberto,Fernández Collado,Carlos,Bapotista Lucio,Pilar. ***Metodología de la Investigación.*** **México : Programas Educativos S.A. ISBN 970-10-1899-0.**

61. Rolando Alfredo, Sayda Coello González. ***El proceso de investigación científica.*** **Ciudad de la Hbana : Editorial Universitaria del Ministerio de Educación Superior, 2011.**

62. Perdita Stevens, Rob Pooley,Addison Wesley. *Utilización de UML en Ingeniería del Software con Objetos y Componentes.* 2002.

63. Ledesma, Rubén. *Introducción al Bootstrap.Desarrollo de un ejemplo acompañado de software de aplicación.* Mar de Plata,Argentina : s.n.

64. Meléndrez, Edelys Hernández. *Como escribir una tesis.* s.l. : Escuela Nacional de Salud Pública, 2006.

65. Letelier, Patricio y Penadés, Ma.Carmen. Metodologías ágiles para el desarrollo de software:eXtreme Programming(XP). *Metodologías ágiles para el desarrollo de software:eXtreme Programming(XP).* [En línea] [Citado el: 2014 de 03 de 03.] http://www.cyta%28xp%29.htm/.

66. Bagarotti Acebo, Yadira. *Gestión de la Calidad en el ciclo de Desarrollo del Software de proyectos que usan metodologías ágiles.* Granma : s.n., Agosto,2012.

67. Peñalver Romero, Gladys Marsi, García de la Puente, Sergio Jesús y Meneses Abad, Abel. *SXP,metodología de desarrollo de software SXP,software development methodology .* s.l. : Serie Científica de la Universidad de las Ciencias Informáticas, 2011.

68. Pantaleón, Dra. Marta Elena Zorilla. *Modelos de datos.* [Presentación Power Point] Cantabria, España : Universidad de Cantabria, 2011.

69. Pérez, Javier Eguíluz. librosweb. *introducción a javascript.* [En línea] [Citado el: 16 de marzo de 2015.] http://www.librosweb.es.

70. Pérez, Javier Egíluz. librosweb. *Intoducción a CCS.* [En línea] [Citado el: 16 de marzo de 2015.] http://www.liobrosweb.es.

Anexos

Anexo I

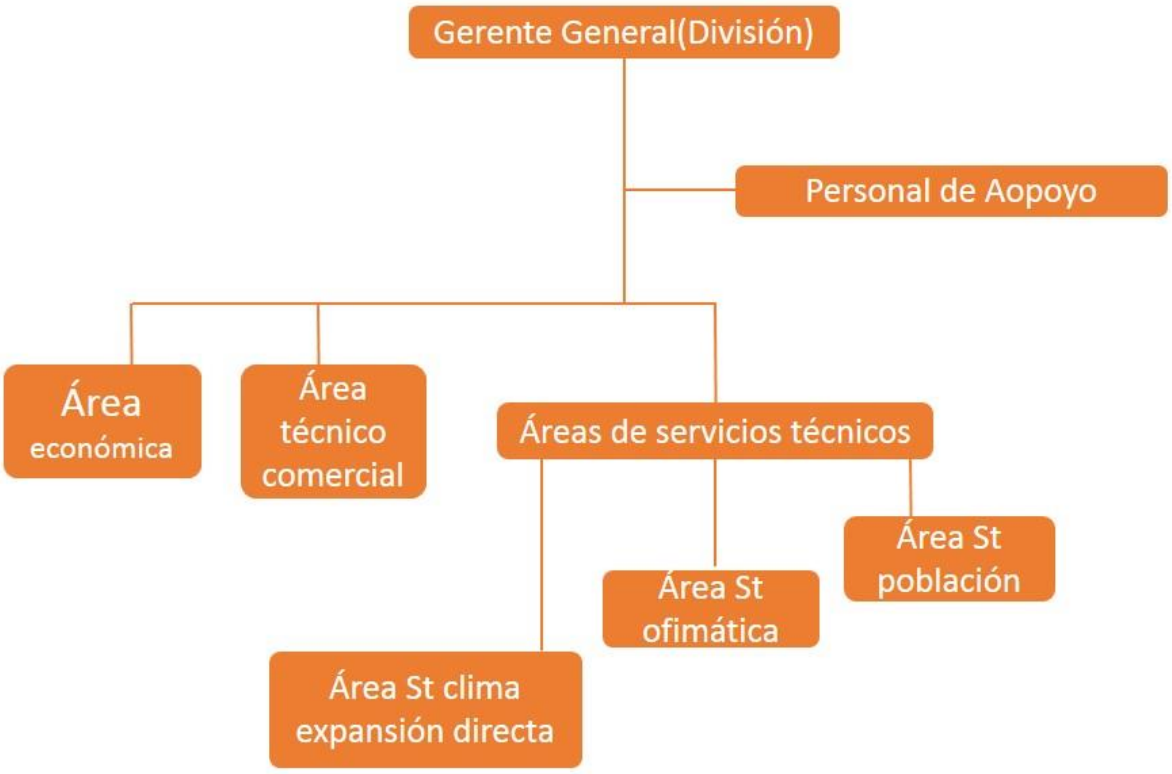


Figura 2: Organigrama de la división Norte de la empresa COPEXTEL

Anexo II

Guía de observación:

Estudio de las soluciones similares

- ✓ Para analizar cómo se realiza los procesos de pago por resultado a niveles nacionales y cuáles sirven de base para la investigación.

Desarrollo de la investigación

- ✓ Observar si las funcionalidades implementadas cumplen con los artefactos elaborados durante el flujo de trabajo de análisis y diseño.

Resultados de la investigación

- ✓ Se aprecia el grado de complejidad de las no conformidades detectadas y cómo erradicarlas.

Anexo III

Preguntas realizadas en la entrevista al gerente de la división Norte del a empresa COPEXTEL:

1- ¿Qué es la resolución 17 del código de trabajo?

2- ¿Cómo COPEXTEL realiza el pago?
3- ¿Cómo lo hacen en la división Norte?
4- ¿Cuantas áreas tiene la división?
5- ¿A partir de que indicadores lo establece?
6- ¿Cómo se encuentra la infraestructura tecnológica de la empresa?
7- ¿Qué tiempo necesita para la entrega del producto?

Anexo IV

Historias de usuario

Historia de Usuario	
Número: 7	**Nombre HU:** Gestionar evaluación de desempeño
Usuario: Gerente general, Jefe de área	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: Los Jefes de área tienen la posibilidad de gestionar las evaluaciones de desempeño de cada trabajador perteneciente a su división, en el caso del administrador gestiona todas las evaluaciones.	
Observaciones: la funcionalidad gestionar incluye adicionar, modificar y eliminar las evaluaciones.	

Historia de Usuario	
Número: 8	**Nombre HU:** Mostrar evaluaciones
Usuario: Gerente general, Jefe de área, Trabajador	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: Los trabajadores tienen la posibilidad de acceder a las evaluaciones de desempeño de cada trabajador perteneciente a su división, y chequearlas en el sistema.	
Observaciones**:**	

Historia de Usuario	
Número: 9	**Nombre HU:** Gestionar valores de entrada
Usuario: Gerente general	
Prioridad de Negocio: Alto	**Riesgo de desarrollo**: Medio
Puntos estimados: 0.5	**Iteración asignada**: 1

Descripción: El Gerente general tiene la posibilidad de gestionar los valores de entrada por los cuales el sistema se va a regir para realizar los cálculos de pago.
Observaciones: la funcionalidad gestionar incluye insertar, modificar y eliminar los valores de entrada.

Historia de Usuario	
Número: 10	**Nombre HU:** Gestionar Método de Pago
Usuario: Gerente general	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: El gerente general tiene la posibilidad de gestionar las formas de pago que adoptará la división de acuerdo al rendimiento de la empresa.	
Observaciones: la funcionalidad gestionar incluye adicionar, modificar y eliminar los métodos de pago.	

Historia de Usuario	
Número: 11	**Nombre HU:** Efectuar Pago
Usuario: Gerente general, Jefe de área	
Prioridad de Negocio: Alta	**Riesgo de desarrollo**: Alto
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: el gerente general y los jefes de área tienen la posibilidad de gestionar el pago de los trabajadores pertenecientes a su área.	
Observaciones: la funcionalidad gestionar incluye modificar y mostrar los pagos de los trabajadores.	

Historia de Usuario	
Número: 12	**Nombre HU:** Mostrar salario histórico a los trabajadores.
Usuario: Gerente general, Jefe de área, Trabajador	
Prioridad de Negocio: Media	**Riesgo de desarrollo**: Medio
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: todos los usuarios del sistema tienen la posibilidad de ver el salario histórico por meses o años de los trabajadores de la división.	
Observaciones:	

Historia de Usuario	
Número: 13	**Nombre HU:** Mostrar monto de salario histórico de la división
Usuario: Gerente general, Jefe de área, Trabajador	
Prioridad de Negocio: Media	**Riesgo de desarrollo**: Medio
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: todos los usuarios del sistema tienen la posibilidad de ver el monto a pagar por meses o años de la división.	
Observaciones:	

Historia de Usuario	
Número: 14	**Nombre HU:** Crear reporte del cumplimiento de la empresa.
Usuario: Gerente general, Jefe de área, Trabajador	
Prioridad de Negocio: Media	**Riesgo de desarrollo**: Medio
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: todos los usuarios del sistema tienen la posibilidad de ver un reporte con todos los datos asociados al cumplimiento de la empresa.	
Observaciones:	

Historia de Usuario	
Número: 15	**Nombre HU:** Mostrar datos estadísticos en forma de gráfico.
Usuario: Gerente general, Jefe de área, Trabajador	
Prioridad de Negocio: Media	**Riesgo de desarrollo**: Alta
Puntos estimados: 0.5	**Iteración asignada**: 1
Descripción: todos los usuarios del sistema tienen la posibilidad de consultar datos estadísticos de la empresa en forma de gráficas de comportamiento.	
Observaciones:	

Historia de Usuario	
Número: 16	**Nombre HU:** Exportar a pdf
Usuario: Gerente general, Jefe de área, Trabajador	
Prioridad de Negocio: Medio	**Riesgo de desarrollo**: Medio

Puntos estimados: 0.5	**Iteración asignada**: 4
Descripción: Los usuarios tienen la posibilidad de exportar a pdf las evaluaciones individuales de cada uno de los trabajadores de la división.	
Observaciones:	

Historia de Usuario	
Número: 17	**Nombre HU:** Exportar a Excel
Usuario: Gerente general, Jefe de área, Trabajador	
Prioridad de Negocio: Medio	**Riesgo de desarrollo**: Medio
Puntos estimados: 0.5	**Iteración asignada**: 4
Descripción: Los usuarios tienen la posibilidad de exportar a Excel las evaluaciones individuales de cada uno de los trabajadores de la división.	
Observaciones:	

Anexo V

Tarjetas CRC

Nombre: Área	
Responsabilidad	**Colaboraciones**
✓ Crear área ✓ Editar área ✓ Eliminar área	Area

Nombre: Cargo	
Responsabilidad	**Colaboraciones**
✓ Crear cargo ✓ Editar cargo ✓ Eliminar cargo	

Nombre: Indicadoreficiencia	
Responsabilidad	**Colaboraciones**
✓ Crear indicador de eficiencia ✓ Editar indicador de eficiencia ✓ Eliminar indicador de eficiencia	IndicadorEspecífico

Nombre: Trabajador	
Responsabilidad	**Colaboraciones**
✓ Crear un trabajador ✓ Insertar un trabajador en la lista de trabajadores ✓ Editar un trabajador ✓ Eliminar un trabajador	Area Cargo Usuario

Nombre: Forma de pago	
Responsabilidad	**Colaboraciones**
✓ Crear una forma de pago ✓ Insertar una forma de pago ✓ Editar una forma de pago ✓ Eliminar una forma de pago	

Nombre: Valores de entrada	
Responsabilidad	**Colaboraciones**
✓ Crear un valor de entrada ✓ Insertar un valor de entrada ✓ Editar un valor de entrada ✓ Eliminar un valor de entrada	

Anexo VI

Tareas de ingeniería

Tarea	
Número de tarea: 7	**Número de HU :** 4
Nombre de tarea: implementación de la funcionalidad Autenticar usuario	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 13/3/2015	**Fecha fin :**14/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite al usuario autenticarse en el sistema con un usuario y una contraseña definida por el administrador del sistema.	

Tarea	
Número de tarea: 8	**Número de HU :**5
Nombre de tarea: Configuración de la HU Gestionar indicador de eficiencia	

Tipo de tarea: configuración	**Puntos estimados:** 1 día
Fecha de inicio: 17/3/2015	**Fecha fin :** 18/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se seleccionan los datos con los que se debe trabajar en esta funcionalidad.	

Tarea	
Número de tarea: 9	**Número de HU :** 5
Nombre de tarea: implementación de la funcionalidad Gestionar indicador de eficiencia	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 18/3/2015	**Fecha fin :**19/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite crear un indicador de eficiencia para la evaluación de los trabajadores en el sistema. De igual forma, brinda la posibilidad de adicionar un indicador, modificarlo, mostrarlo y eliminarlo.	

Tarea	
Número de tarea: 10	**Número de HU :**6
Nombre de tarea: Configuración de la HU Gestionar indicador específico	
Tipo de tarea: configuración	**Puntos estimados:** 1 día
Fecha de inicio: 19/3/2015	**Fecha fin :** 11/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se seleccionan los datos con los que se debe trabajar en esta funcionalidad.	

Tarea	
Número de tarea: 11	**Número de HU :** 6
Nombre de tarea: implementación de la funcionalidad Gestionar indicador específico	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 21/1/2015	**Fecha fin :**22/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite crear un indicador de eficiencia específico para la evaluación. De igual forma, brinda la posibilidad de adicionar un indicador específico, modificarlo, mostrarlo y eliminarlo.	

Tarea	
Número de tarea: 12	**Número de HU :**7
Nombre de tarea: Configuración de la HU Gestionar evaluación de desempeño	
Tipo de tarea: configuración	**Puntos estimados:** 1 día

Fecha de inicio: 22/3/2015	**Fecha fin :** 23/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se seleccionan los datos con los que se debe trabajar en esta funcionalidad.	

Tarea	
Número de tarea: 13	**Número de HU :** 7
Nombre de tarea: implementación de la funcionalidad Gestionar evaluación de desempeño	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 23/3/2015	**Fecha fin :**24/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite crear una evaluación de desempeño para un trabajador. De igual forma, brinda la posibilidad de adicionar una evaluación, modificarlo y mostrarlo.	

Tarea	
Número de tarea: 14	**Número de HU :** 8
Nombre de tarea: implementación de la funcionalidad Mostrar evaluaciones	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 24/3/2015	**Fecha fin :**25/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite mostrar las evaluaciones de un trabajador o un área específica.	

Tarea	
Número de tarea: 15	**Número de HU :**9
Nombre de tarea: Configuración de la HU Gestionar valores de entrada	
Tipo de tarea: configuración	**Puntos estimados:** 1 día
Fecha de inicio: 25/3/2015	**Fecha fin :** 26/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se seleccionan los datos con los que se debe trabajar en esta funcionalidad.	

Tarea	
Número de tarea: 16	**Número de HU :** 9
Nombre de tarea: implementación de la funcionalidad Gestionar valores de entrada	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 26/3/2015	**Fecha fin :**27/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	

Descripción: se implementa la funcionalidad, la cual permite agregar los valores de entrada para poder efectuar el pago a un trabajador. De igual forma, brinda la posibilidad de adicionar un valor de entrada, modificarlo, mostrarlo y eliminarlo.

Tarea	
Número de tarea: 17	**Número de HU :**10
Nombre de tarea: Configuración de la HU Gestionar método de pago	
Tipo de tarea: configuración	**Puntos estimados:** 1 día
Fecha de inicio: 28/3/2015	**Fecha fin :** 29/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se seleccionan los datos con los que se debe trabajar en esta funcionalidad.	

Tarea	
Número de tarea: 18	**Número de HU :** 10
Nombre de tarea: implementación de la funcionalidad Gestionar método de pago	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 30/3/2015	**Fecha fin :**31/3/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite crear un método de pago para poder efectuar el pago a un trabajador. De igual forma, brinda la posibilidad de adicionar un método de pago, modificarlo, mostrarlo y eliminarlo.	

Tarea	
Número de tarea: 19	**Número de HU :** 11
Nombre de tarea: implementación de la funcionalidad Efectuar pago	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 1/4/2015	**Fecha fin :**2/4/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: implementa la funcionalidad efectuar pago, la cual permite pagar a los trabajadores de acuerdo a una evaluación dada.	

Tarea	
Número de tarea: 20	**Número de HU :** 12
Nombre de tarea: implementación de la funcionalidad Mostrar salario histórico a los trabajadores	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 3/4/2015	**Fecha fin :** 4/4/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	

Descripción: se implementa la funcionalidad, la cual muestra el salario obtenido por los trabajadores por meses, años, así como la evaluación asociada a los mismos.

Tarea	
Número de tarea: 21	**Número de HU :** 13
Nombre de tarea: implementación de la funcionalidad Mostrar monto a pagar histórico en la división	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 4/4/2015	**Fecha fin :** 5/4/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual muestra el monto a pagar en la división por meses y años.	

Tarea	
Número de tarea: 22	**Número de HU :** 14
Nombre de tarea: implementación de la funcionalidad Crear reporte del cumplimiento de la empresa	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 5/4/2015	**Fecha fin :** 6/4/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual crea un reporte acerca del cumplimiento de la empresa (Productividad, sobrecumplimiento).	

Tarea	
Número de tarea: 23	**Número de HU :** 15
Nombre de tarea: implementación de la funcionalidad Mostrar datos estadísticos en forma de gráfico	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 6/4/2015	**Fecha fin :** 7/4/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual muestra los datos obtenidos por la empresa en forma de un gráfico asociado a todos los reportes generales, dígase productividad, y monto a pagar asociado a la misma.	

Tarea	
Número de tarea: 24	**Número de HU :** 16
Nombre de tarea: implementación de la funcionalidad Exportar a pdf	

Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 8/4/2015	**Fecha fin :** 9/4/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite exportar a pdf todas las evaluaciones, así como los reportes en forma de gráfico.	

Tarea	
Número de tarea: 25	**Número de HU :** 17
Nombre de tarea: implementación de la funcionalidad Exportar a Excel	
Tipo de tarea: desarrollo	**Puntos estimados:** 1 día
Fecha de inicio: 10/4/2015	**Fecha fin :** 11/4/2015
Programador responsable: Alberto Martínez Oquendo y Juan Carlos Casadesús Rades	
Descripción: se implementa la funcionalidad, la cual permite exportar a Excel todas las evaluaciones.	

Anexo VII

Casos de prueba

HU Gestionar Área

Caso de prueba de aceptación	
Código: HU2_P1	**Historia de usuario:**2
Nombre: Adicionar área.	
Descripción: prueba para la funcionalidad adicionar área al sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: Se selecciona el menú Recursos Humanos, en el subíndice áreas la opción añadir área, se llenan los datos solicitados y se añade al sistema.	
Resultado esperado: se adiciona un área en el sistema.	
Evaluación de la prueba: prueba satisfactoria.	

Caso de prueba de aceptación	
Código: HU2_P2	**Historia de usuario:** 2

Nombre: Mostrar área
Descripción: prueba para la funcionalidad que permite mostrar el listado de áreas de la división.
Condiciones de ejecución: el usuario debe estar autenticado en el sistema
Pasos de ejecución: al seleccionar la pestaña Recursos Humanos, dentro de la misma se puede acceder al subíndice áreas, seguidamente se muestra un listado con todos las áreas existentes en el sistema, luego la opción ver.
Resultado esperado: el sistema debe mostrar el listado de áreas existentes.
Evaluación de la prueba: prueba satisfactoria.

Caso de prueba de aceptación	
Código: HU2_P3	**Historia de usuario:**2
Nombre: Modificar área.	
Descripción: prueba para la funcionalidad modificar área en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: al seleccionar la pestaña Recursos Humanos, dentro de la misma se puede acceder al subíndice áreas, seguidamente se muestra un listado con todos las áreas existentes en el sistema, luego la opción Editar.	
Resultado esperado: el sistema muestra un formulario con los campos de un área.	
Evaluación de la prueba: prueba satisfactoria.	

Caso de prueba de aceptación	
Código: HU2_P4	**Historia de usuario:** 3
Nombre: Eliminar un área	
Descripción: prueba para la funcionalidad que permite eliminar un área.	

Condiciones de ejecución: ✓ el usuario debe estar autenticado como Gerente general. ✓ debe existir al menos un área en el sistema.
Pasos de ejecución: se selecciona la pestaña Recursos Humanos, dentro de la misma el subíndice área, se selecciona el área que se desea eliminar, se presiona el botón eliminar y se muestra una alerta de confirmación de eliminación, seguidamente se da clic en el botón aceptar.
Resultado esperado: el área fue eliminada satisfactoriamente.
Evaluación de la prueba: prueba satisfactoria.

HU 3 Gestionar Cargo.

Caso de prueba de aceptación	
Código: HU3_P1	**Historia de usuario:**3
Nombre: Adicionar cargo	
Descripción: prueba para la funcionalidad adicionar cargo.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: se selecciona la pestaña Recursos Humanos, dentro de la misma el subíndice Cargos se selecciona la opción Adicionar, se muestra un formulario con los campos asociados a un cargo, luego se selecciona el botón adicionar.	
Resultado esperado: se adiciona un cargo en el sistema.	
Evaluación de la prueba: prueba satisfactoria	

Caso de prueba de aceptación	
Código: HU3_P2	**Historia de usuario:** 3
Nombre: Mostrar cargo	
Descripción: prueba para la funcionalidad que permite mostrar el listado de cargos de los trabajadores de un área determinada.	

Condiciones de ejecución: el usuario debe estar autenticado como Gerente general en el sistema
Pasos de ejecución: al seleccionar la pestaña Recursos Humanos, dentro de la misma se puede acceder al subíndice cargos, seguidamente se muestra un listado con todos los cargos existentes en el sistema, y se selecciona la opción ver.
Resultado esperado: el sistema debe mostrar el listado de cargos existentes.
Evaluación de la prueba: prueba satisfactoria

Caso de prueba de aceptación	
Código: HU3_P3	**Historia de usuario:**3
Nombre: Modificar cargo	
Descripción: prueba para la funcionalidad modificar cargo en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: al seleccionar la pestaña Recursos Humanos, dentro de la misma se puede acceder al subíndice cargos, seguidamente se muestra un listado con todos los cargos existentes en el sistema, y se selecciona la opción modificar.	
Resultado esperado: se observan los datos editados del cargo seleccionado en el sistema.	
Evaluación de la prueba: prueba satisfactoria	

Caso de prueba de aceptación	
Código: HU3_P4	**Historia de usuario:** 3
Nombre: Eliminar un cargo.	
Descripción: prueba para la funcionalidad que permite eliminar un trabajador.	
Condiciones de ejecución: ✓ el usuario debe estar autenticado como administrador. ✓ debe existir al menos un cargo en el sistema.	

Pasos de ejecución: se selecciona la pestaña Recursos Humanos, dentro del subíndice cargos se selecciona el cargo que se desea eliminar, se da sobre la pestaña eliminar y se muestra una alerta de confirmación de eliminación, seguidamente se da en el botón aceptar.
Resultado esperado: el cargo fue eliminado satisfactoriamente.
Evaluación de la prueba: prueba satisfactoria

HU 4 Autenticar usuario

Caso de prueba de aceptación	
Código: HU4_P1	**Historia de usuario:**4
Nombre: Autenticar usuario	
Descripción: prueba para la funcionalidad Autenticar usuario al sistema.	
Condiciones de ejecución: debe existir el usuario creado en el sistema con una contraseña asociada al mismo.	
Pasos de ejecución: cuando se accede el sistema mediante el navegador, se llenan los parámetros de identificación y autenticación con el usuario y la contraseña definida antes en el mismo.	
Resultado esperado: se accede a la vista principal del sistema.	
Evaluación de la prueba: prueba satisfactoria	

HU 5 Gestionar indicador de eficiencia

Caso de prueba de aceptación	
Código: HU5_P1	**Historia de usuario:**5
Nombre: Adicionar indicador de eficiencia	
Descripción: prueba para la funcionalidad Adicionar indicador de eficiencia al sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	

Pasos de ejecución: Se selecciona el menú Gestión de indicadores, luego se escoge la opción indicador de eficiencia, se selecciona la opción adicionar, se llenan los datos solicitados y se añade al sistema.
Resultado esperado: se adiciona un indicador de eficiencia en el sistema.
Evaluación de la prueba: prueba satisfactoria

Caso de prueba de aceptación	
Código: HU5_P2	**Historia de usuario:**5
Nombre: Modificar indicador de eficiencia	
Descripción: prueba para la funcionalidad Adicionar indicador de eficiencia al sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: Se selecciona el submenú Indicadores, luego se selecciona la opción eficiencia y se modifican los datos del indicador de eficiencia y se guardan en el sistema.	
Resultado esperado: se observan los datos editados del indicador seleccionado en el sistema.	
Evaluación de la prueba: prueba satisfactoria	

Caso de prueba de aceptación	
Código: HU5_P3	**Historia de usuario:** 5
Nombre: Mostrar indicador de eficiencia	
Descripción: prueba para la funcionalidad que permite mostrar el listado de indicadores de eficiencia para evaluar a los trabajadores.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema	
Pasos de ejecución: Se selecciona el submenú Indicadores, luego se selecciona la opción eficiencia y seguidamente se muestran los indicadores de eficiencia existentes en el sistema.	
Resultado esperado: el sistema debe mostrar el listado de indicadores de eficiencia existentes.	

Evaluación de la prueba: prueba satisfactoria

Caso de prueba de aceptación	
Código: HU5_P4	**Historia de usuario:** 5
Nombre: Eliminar indicador de eficiencia	
Descripción: prueba para la funcionalidad que permite eliminar un indicador de eficiencia.	
Condiciones de ejecución: ✓ el usuario debe estar autenticado como Gerente general. ✓ debe existir al menos un indicador de eficiencia en el sistema.	
Pasos de ejecución: se selecciona la pestaña Indicadores, luego en el submenú indicador de eficiencia y se selecciona el indicador que se desea eliminar, se da sobre la pestaña eliminar y se muestra una alerta de confirmación de eliminación, seguidamente se da en el botón aceptar.	
Resultado esperado: el indicador de eficiencia fue eliminada satisfactoriamente.	
Evaluación de la prueba: prueba satisfactoria	

HU 6 Gestionar indicador específico

Caso de prueba de aceptación	
Código: HU6_P1	**Historia de usuario:**6
Nombre: Adicionar indicador específico	
Descripción: prueba para la funcionalidad Adicionar indicador específico al sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: Se selecciona el menú Gestión de indicadores, luego se selecciona la opción añadir indicador, se llenan los datos solicitados y se añade al sistema.	
Resultado esperado: se adiciona un indicador específico en el sistema.	
Evaluación de la prueba: prueba satisfactoria.	

Caso de prueba de aceptación	
Código: HU6_P2	**Historia de usuario:**6
Nombre: Modificar indicador específico.	
Descripción: prueba para la funcionalidad Adicionar indicador específico al sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: Se selecciona la pestaña Indicadores, el subíndice indicador específico y se busca el indicador a modificar, se presiona en el botón editar y se modifican los datos del indicador específico y se guarda en el sistema.	
Resultado esperado: se observan los datos editados del indicador específico seleccionado en el sistema.	
Evaluación de la prueba: prueba satisfactoria	

Caso de prueba de aceptación	
Código: HU6_P2	**Historia de usuario:** 6
Nombre: Mostrar indicador específico	
Descripción: prueba para la funcionalidad que permite mostrar el listado de indicadores de eficiencia para evaluar a los trabajadores.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema	
Pasos de ejecución: al seleccionar el submenú Indicadores, dentro de la misma se puede acceder al subíndice indicador de eficiencia, seguidamente se muestra un listado con todos los indicadores específicos existentes en el sistema y se selecciona la opción ver.	
Resultado esperado: el sistema debe mostrar el indicador de eficiencia existente.	
Evaluación de la prueba: prueba satisfactoria	

Caso de prueba de aceptación

Código: HU6_P4	**Historia de usuario:** 6
Nombre: Eliminar indicador específico	
Descripción: prueba para la funcionalidad que permite eliminar un indicador específico.	
Condiciones de ejecución: ✓ el usuario debe estar autenticado como administrador. ✓ debe existir al menos un indicador de eficiencia en el sistema.	
Pasos de ejecución: se selecciona el submenú Indicadores, luego en el subíndice indicador específico y se selecciona el indicador que se desea eliminar, se da sobre el botón eliminar y se muestra una alerta de confirmación de eliminación, seguidamente se da en el botón aceptar.	
Resultado esperado: el indicador específico fue eliminado satisfactoriamente.	
Evaluación de la prueba: prueba satisfactoria	

HU 7 Gestionar evaluación de desempeño

Caso de prueba de aceptación	
Código: HU7_P1	**Historia de usuario:**7
Nombre: Crear evaluación de desempeño	
Descripción: prueba para la funcionalidad Crear evaluación de desempeño en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general o como Jefe de área.	
Pasos de ejecución: Se selecciona el módulo evaluaciones, luego se selecciona el área donde radica el trabajador deseado, se selecciona al trabajador y se accede a la opción evaluar, se llenan los datos de la evaluación del mismo y se guardan en el sistema.	
Resultado esperado: se crea una evaluación de un trabajador.	
Evaluación de la prueba: prueba satisfactoria.	

Caso de prueba de aceptación	
Código: HU7_P2	**Historia de usuario:**7

Nombre: Modificar evaluación de desempeño
Descripción: prueba para la funcionalidad modificar evaluación de desempeño al sistema.
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general o como Jefe de área.
Pasos de ejecución: Se selecciona el módulo evaluaciones, luego se selecciona el área donde radica el trabajador a evaluar, se selecciona el mismo y se le da a la opción editar evaluación, se modifican los datos de la evaluación del mismo y se guardan en el sistema.
Resultado esperado: se observan los datos editados de la evaluación del trabajador seleccionado en el sistema.
Evaluación de la prueba: prueba satisfactoria

Caso de prueba de aceptación	
Código: HU7_P2	**Historia de usuario:** 7
Nombre: Mostrar evaluación de desempeño	
Descripción: prueba para la funcionalidad que permite mostrar el listado de evaluaciones de desempeño de los trabajadores en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema	
Pasos de ejecución: al seleccionar el módulo evaluaciones, se selecciona el área donde está el trabajador a evaluar, seguidamente se muestra un listado con todas las evaluaciones de los trabajadores en el sistema.	
Resultado esperado: el sistema debe mostrar el listado de evaluaciones de los trabajadores existentes.	
Evaluación de la prueba: prueba satisfactoria	

HU 8 Mostrar evaluaciones

Caso de prueba de aceptación	
Código: HU8_P2	**Historia de usuario:** 8

Nombre: Mostrar evaluaciones
Descripción: prueba para la funcionalidad que permite mostrar el listado de evaluaciones de desempeño de los trabajadores en el sistema.
Condiciones de ejecución: el usuario debe estar autenticado en el sistema
Pasos de ejecución: al seleccionar la pestaña Evaluaciones, se selecciona un área y seguidamente se muestra un listado con todas las evaluaciones de los trabajadores en el sistema.
Resultado esperado: el sistema debe mostrar el listado de evaluaciones de los trabajadores existentes.
Evaluación de la prueba: prueba satisfactoria.

HU9 Gestionar valores de entrada

Caso de prueba de aceptación	
Código: HU9_P1	**Historia de usuario:**9
Nombre: Adicionar valores de entrada	
Descripción: prueba para la funcionalidad Adicionar valores de entrada al sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: Se selecciona un trabajador a evaluar, cuando se llenan los datos que se ponen a evaluar, se llenan los datos solicitados en la ventana emergente y se añade al sistema.	
Resultado esperado: se adiciona un valor de entrada al sistema.	
Evaluación de la prueba: prueba satisfactoria.	

Caso de prueba de aceptación	
Código: HU9_P2	**Historia de usuario:**9
Nombre: Modificar valores de entrada.	
Descripción: prueba para la funcionalidad modificar valores de entrada al sistema.	

Condiciones de ejecución: el usuario debe estar autenticado como Gerente general o como Jefe de área.
Pasos de ejecución: Se selecciona un trabajador a evaluar, cuando se llenan los datos que se ponen a evaluar, se modifican los datos solicitados en la ventana emergente y se añaden al sistema.
Resultado esperado: se observan los datos editados de la evaluación del trabajador seleccionado en el sistema.
Evaluación de la prueba: prueba satisfactoria.

Caso de prueba de aceptación	
Código: HU9_P3	**Historia de usuario:** 9
Nombre: Mostrar valores de entrada.	
Descripción: prueba para la funcionalidad que permite mostrar el listado de indicadores de eficiencia para evaluar a los trabajadores.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema	
Pasos de ejecución: al seleccionar la pestaña Gestión de indicadores, dentro de la misma se puede acceder al subíndice áreas, seguidamente se muestra un listado con todos los indicadores de eficiencia existentes en el sistema.	
Resultado esperado: el sistema debe mostrar el listado de indicadores de eficiencia existentes.	
Evaluación de la prueba: prueba satisfactoria.	

HU10 Gestionar método de pago

Caso de prueba de aceptación	
Código: HU10_P1	**Historia de usuario:**10
Nombre: Adicionar método de pago.	
Descripción: prueba para la funcionalidad Adicionar método de pago sistema.	

Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.
Pasos de ejecución: Se selecciona el submenú Pago, luego se selecciona la opción Formas de pago, se selecciona la opción adicionar y se llenan los datos solicitados y se añade al sistema.
Resultado esperado: se adiciona un indicador específico en el sistema.
Evaluación de la prueba: prueba satisfactoria.

Caso de prueba de aceptación	
Código: HU10_P2	**Historia de usuario:**10
Nombre: Mostrar método de pago.	
Descripción: prueba para la funcionalidad Adicionar método de pago sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general o Jefe de área.	
Pasos de ejecución: Se selecciona el submenú Pago, luego se selecciona la opción Formas de pago.	
Resultado esperado: se muestra un listado con los métodos de pago existentes en el sistema.	
Evaluación de la prueba: prueba satisfactoria.	

Caso de prueba de aceptación	
Código: HU10_P3	**Historia de usuario:**10
Nombre: Modificar método de pago.	
Descripción: prueba para la funcionalidad modificar método de pago sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general.	
Pasos de ejecución: Se selecciona el submenú Pago, luego se selecciona la opción Formas de pago, se selecciona la opción editar y se llenan los datos a modificar.	

Resultado esperado: se muestra el método de pago con los nuevos parámetros existentes en el sistema.
Evaluación de la prueba: prueba satisfactoria.

Caso de prueba de aceptación	
Código: HU10_P4	**Historia de usuario:** 10
Nombre: Eliminar método de pago	
Descripción: prueba para la funcionalidad que permite eliminar un método de pago en el sistema.	
Condiciones de ejecución: ✓ el usuario debe estar autenticado como Gerente general. ✓ debe existir al menos un indicador de eficiencia en el sistema.	
Pasos de ejecución: Se selecciona el submenú Pago, luego se selecciona la opción Formas de pago, se selecciona el indicador a eliminar y se selecciona la opción eliminar, luego muestra un cartel de confirmación y se selecciona la opción aceptar.	
Resultado esperado: el método de pago fue eliminada satisfactoriamente.	
Evaluación de la prueba: prueba satisfactoria.	

HU11 Efectuar pago

Caso de prueba de aceptación	
Código: HU11_P1	**Historia de usuario:**11
Nombre: Efectuar pago	
Descripción: prueba para la funcionalidad efectuar pago en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado como Gerente general o Jefe de área.	
Pasos de ejecución: Cuando se selecciona un trabajador en el sistema y se le pasan los parámetros correspondientes, se le da al botón evaluar.	
Resultado esperado: se muestra el pago asociado al trabajador evaluado en el sistema.	

Evaluación de la prueba: prueba satisfactoria.

HU12 Mostrar salario histórico de pago a los trabajadores

Caso de prueba de aceptación	
Código: HU12_P1	**Historia de usuario:** 12
Nombre: Mostrar salario histórico de pago a los trabajadores.	
Descripción: prueba para la funcionalidad Mostrar salario histórico de pago a los trabajadores en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Pasos de ejecución: Cuando se selecciona un trabajador en el sistema se muestra el salario histórico mensual y anual adquirido por el mismo.	
Resultado esperado: se muestra todos los pagos mensuales asociados al trabajador evaluado en el sistema.	
Evaluación de la prueba: prueba satisfactoria.	

HU13 Mostrar monto a pagar histórico de la división

Caso de prueba de aceptación	
Código: HU13_P1	**Historia de usuario:**13
Nombre: Mostrar monto a pagar histórico en la división.	
Descripción: prueba para la funcionalidad Mostrar monto a pagar histórico en la división en el sistema.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Pasos de ejecución: Cuando se selecciona un trabajador en el sistema se muestra el salario histórico mensual y anual adquirido por la división.	
Resultado esperado: se muestra todos los pagos mensuales asociados a la división en el sistema.	

Evaluación de la prueba: prueba satisfactoria.

HU14 Crear reporte del cumplimiento de la empresa

Caso de prueba de aceptación	
Código: HU14_P1	**Historia de usuario:**14
Nombre: Crear reporte del cumplimiento de la empresa.	
Descripción: prueba para la funcionalidad crear reporte del cumplimiento de la empresa en el sistema	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Pasos de ejecución:	
Resultado esperado: se crea un reporte con los datos asociados al cumplimiento de la empresa.	
Evaluación de la prueba: prueba satisfactoria.	

HU15 Mostrar datos estadísticos en forma de gráfico

Caso de prueba de aceptación	
Código: HU15_P1	**Historia de usuario:**15
Nombre: Mostrar datos estadísticos en forma de gráfico.	
Descripción: prueba para la funcionalidad Mostrar datos estadísticos en forma de gráfico.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Pasos de ejecución: cuando el usuario entra al sistema puede ver las gráficas asociadas al rendimiento de la empresa, dígase forma de pago y monto a pagar.	
Resultado esperado: se muestran los datos del cumplimiento de la empresa y el pago en la misma en forma de gráficos en la ventana principal.	
Evaluación de la prueba: prueba satisfactoria.	

HU16 Exportar a pdf

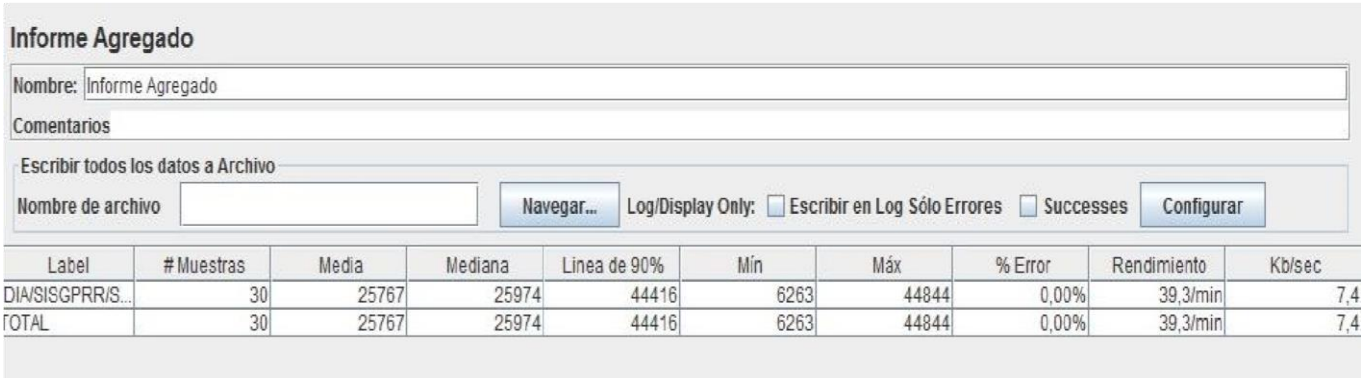
Caso de prueba de aceptación	
Código: HU16_P1	**Historia de usuario:**16
Nombre: Exportar a pdf	
Descripción: prueba para la funcionalidad exportar a pdf.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Pasos de ejecución: cuando el usuario entra al sistema, en el botón exportar a pdf puede exportar las evaluaciones asociadas a un trabajador, o las evaluaciones de un área en este formato.	
Resultado esperado: se muestran los datos exportados en un documento en formato pdf.	
Evaluación de la prueba: prueba satisfactoria.	

HU17 Exportar a Excel

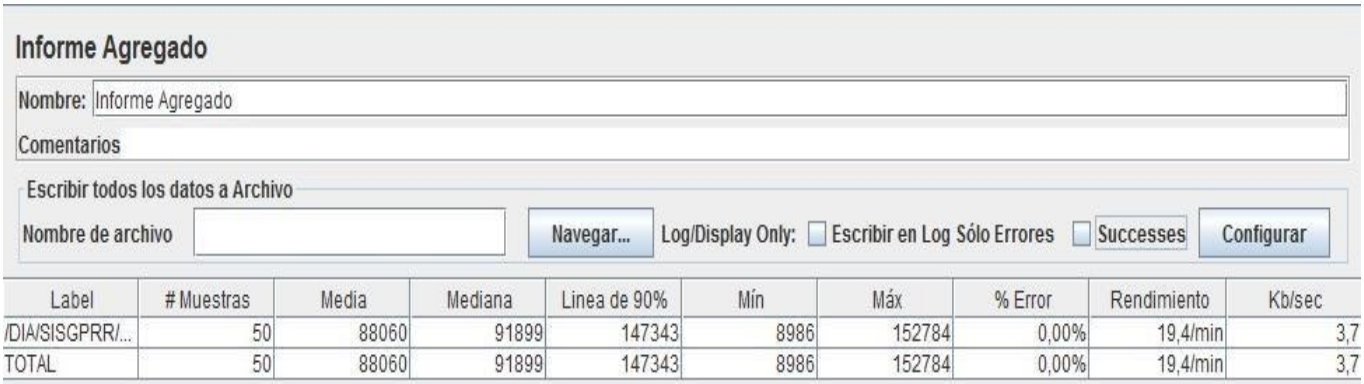
Caso de prueba de aceptación	
Código: HU17_P1	**Historia de usuario:**17
Nombre: Exportar a Excel	
Descripción: prueba para la funcionalidad exportar a Excel.	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema.	
Pasos de ejecución: cuando el usuario entra al sistema, en el botón exportar a pdf puede exportar las evaluaciones asociadas a un trabajador, o las evaluaciones de un área en este formato.	
Resultado esperado: se muestran los datos exportados en un documento en formato Excel.	
Evaluación de la prueba: prueba satisfactoria.	

Anexo VIII

Pruebas de carga y estrés para 30 y 50 usuarios.



Label	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
DIA/SISGPRR/S...	30	25767	25974	44416	6263	44844	0,00%	39,3/min	7,4
TOTAL	30	25767	25974	44416	6263	44844	0,00%	39,3/min	7,4



Label	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
/DIA/SISGPRR/...	50	88060	91899	147343	8986	152784	0,00%	19,4/min	3,7
TOTAL	50	88060	91899	147343	8986	152784	0,00%	19,4/min	3,7

Anexo IX

Encuesta para validar la presente propuesta de solución:

Encuesta

La presente encuesta tiene como objetivo conocer la necesidad de la implementación de un sistema para la gestión del pago por resultado en la división Norte de la empresa COPEXTEL. Gracias

1. El siguiente formulario, muestra en forma ordenada las tareas que intervienen en el proceso de pago en la división. Según su criterio y experiencia:

a) ¿Qué tiempo se emplea, para cada tarea a lo largo del proceso?

Tenga en cuenta para llenar el formulario que el tiempo puede ser expresado en horas, días, o semanas.

Tarea	Tiempo estimado
1. Consultar los datos de los trabajadores en el Hércules y tabularlos.	
2. Emitir las evaluaciones de los trabajadores de su área según la forma de pago definida.	

3. Enviar evaluaciones de tabuladas de cada trabajador al gerente general.	
4. Reunión con el gerente general para establecer el cierre del mes.	
5. proceder a efectuar el pago haciendo los cálculos pertinentes.	
Duración total	

2. ¿Considera usted que con sea necesario el uso de un sistema para la gestión del proceso de pago por resultado?

___ Sí __ No

2.2 ¿En caso positivo diga cuál o cuáles de los siguientes problemas constituyen la causa fundamental de esta necesidad?

___ Pérdida excesiva de tiempo en este proceso.

___ Se hace muy tedioso el trabajo con hojas de cálculo para la gestión de las evaluaciones.

___ El trabajo con hojas de cálculo está sujeto a cometer errores en las evaluaciones.

___ Otra causa.

___ Ninguna de las anteriores.

4. ¿Qué tiempo aproximado se emplea en el proceso de pago, desde que se comienza el proceso de evaluación hasta el día del cierre mensual?

___ De 1 a 3 días.

___ De 4 a 6 días.

___ 1 semana o más.

5. ¿Considera usted necesario la existencia de un sistema que permita a cada trabajador consultar la forma en que se le evalúa y paga, así como el comportamiento histórico de sus evaluaciones y salarios devengados?

___ Sí __ No

6. ¿Usted cree que el sistema sea fácil de usar e intuitiva?

___ Sí __ No

7. ¿Usted cree que el proceso de familiarización con el sistema pueda retrasar el tiempo para llevar a cabo el proceso de pago?

___ Sí __ No

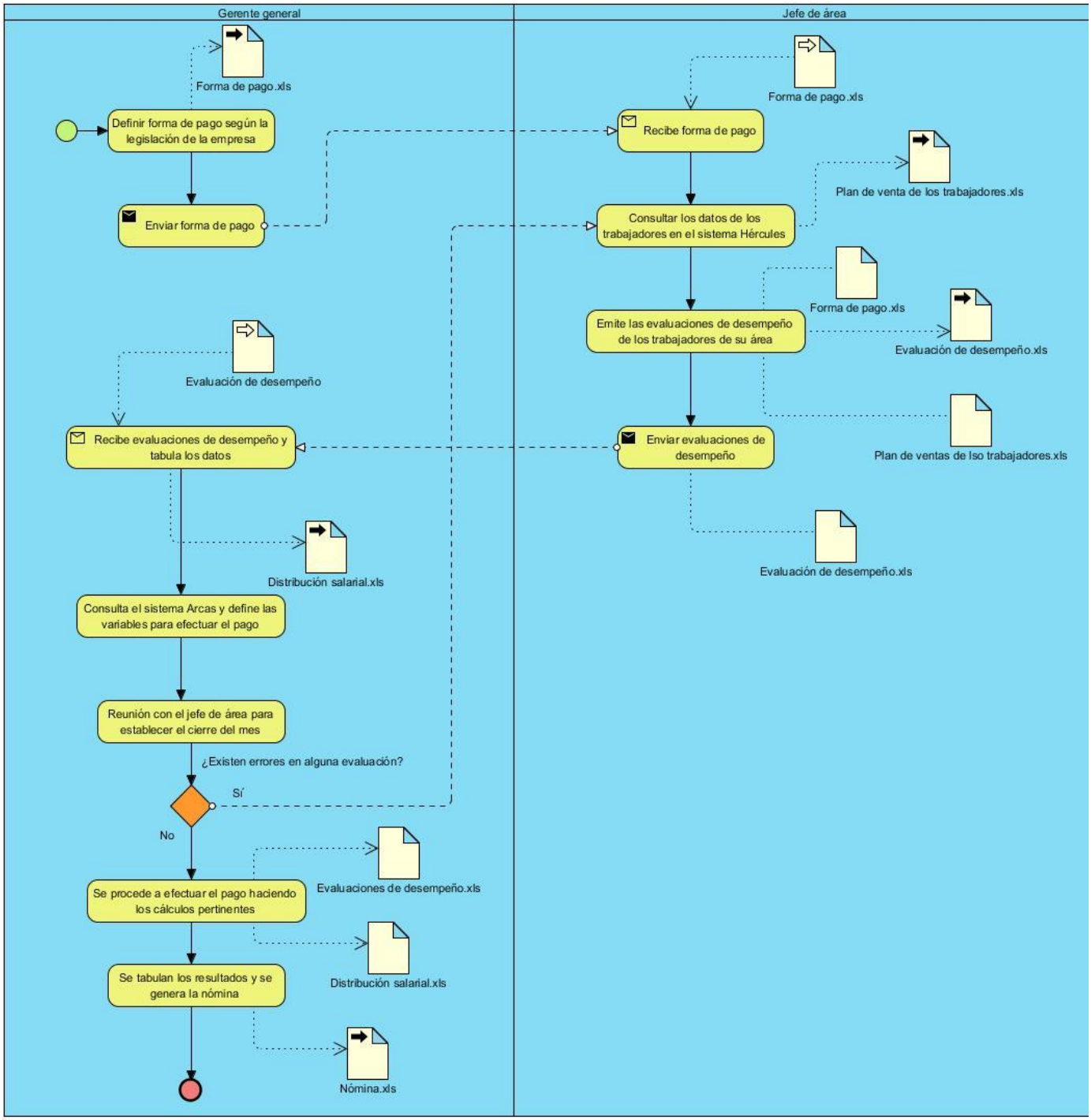
8. ¿Cree usted que el uso del sistema de gestión disminuye el tiempo del proceso de pago en la división?

___ Sí __ No

9. Otras consideraciones acerca del proceso de pago por resultado.

Anexo X

Diagrama de procesos que permite entender cómo se efectúa el proceso de pago por resultado en la división Norte de la empresa COPEXTEL.



Glosario de términos

COPEXTEL: siglas que identifican a la empresa comercializadora de soluciones tecnológicas integrales, bienes y servicios en Cuba y el exterior.

Artefactos ingenieriles: productos tangibles resultantes del proceso de desarrollo de software. Ayudan a la descripción de la función, la arquitectura o el diseño del software. Sin embargo otros se enfocan en el proceso de desarrollo en sí mismo como planes de proyecto.

Multiplataforma: es un atributo conferido a programas informáticos que son implementados e interoperan en múltiples plataformas informáticas.

Software libre: *software* que puede ser distribuido, modificado, redistribuido, copiado y usado libremente. Se basa en cuatro libertades: libertad para usarlo con cualquier propósito, libertad para modificarlo a las necesidades, libertad para distribuir copias y para mejorarlo. Que un *software* sea libre no quiere decir que sea gratuito, error que viene de la traducción *Free Software.*

Unix: sistema operativo comercializado en la década de los 70 que logró mucho éxito, sobre todo en las universidades y posteriormente en las empresas. Entre sus características presenta que es: portable, robusto y flexible. Actualmente goza de gran popularidad dentro de la tecnología de Internet.

Web: es un sistema para presentar información en Internet basado en hipertexto. Cuando se utiliza en género masculino (el Web, un Web) se refiere a un sitio Web entero, y si se utiliza en el género femenino (la Web, una Web) se refiere a una página Web concreta dentro del sitio Web.

WWW (World Wide Web): es el universo de información accesible a través de Internet, una fuente inagotable del conocimiento humano.

Código abierto: es la expresión con la que se conoce al software distribuido y desarrollado libremente.

GNOME: es un entorno de escritorio e infraestructura de desarrollo para sistemas operativos compuestos totalmente de software libre.

GNOME Foundation: en español Fundación GNOME, es una asociación englobada por el proyecto GNOME para facilitar una forma de comunicación con medios de comunicación, organizaciones comerciales y no comerciales interesados en GNOME.

XML: en inglés *Extensible Markup Language*, traducido al español como Lenguaje de Marcas Extensible. Es un lenguaje utilizado para almacenar datos en forma legible.