



Universidad de las Ciencias  
Informáticas

**Universidad de las Ciencias Informáticas**

**Facultad # 1**

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

**Módulo Asistente a la Gestión de los Recursos Humanos**

**para el Sistema de Información Primaria de Personas.**

**Autor:**

Yaidel Díaz Cáceres

**Tutores:**

Ing. Yarileidis Barcena Calzado

Ing. Dashiell Hondarez Laza

La Habana, junio de 2018

“Año 60 de la Revolución”

**Declaración de Autoría**

Declaro ser autor del presente trabajo de diploma y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo, con carácter exclusivo.

Para que así conste firmo la presente a los \_\_\_\_ días del mes de \_\_\_\_\_ del año \_\_\_\_\_.

Autor: \_\_\_\_\_

Yaidel Díaz Cáceres

Tutores: \_\_\_\_\_

Ing. Yarileidis Barcena Calzado

\_\_\_\_\_

Ing. Dashiel Hondarez Laza

## **RESUMEN**

Dentro de una organización los Recursos Humanos juegan un papel crítico puesto que son responsables de seleccionar, contratar, formar, emplear y retener a los trabajadores. Su objetivo básico es alinear sus políticas con las de la organización y para poder ejecutar ésta estrategia de forma satisfactoria es fundamental que los datos relacionados con el personal sean válidos y estén disponibles. La Universidad de las Ciencias Informáticas cuenta con una amplia gama de productos desplegados en varios sectores de Cuba y muchos de estos registran sus datos de manera diferente, situación que, si una misma entidad desea contar con más de un software desarrollado por la universidad, puede provocar duplicidad de información, inconsistencia de los datos y afectar el correcto funcionamiento de la empresa. Para mitigar esta situación se ha decidido desarrollar un sistema que registre los datos de los trabajadores y a la vez brinde a los sistemas que lo requieran dicha información. Este sistema contará con varios módulos entre los que se encuentran: captura de datos, directorio, control de acceso, administración, trazas y asistente de gestión de recursos humanos en el cual se centra este trabajo. Dicho módulo contará con varias funcionalidades y características que le permitirán gestionar los subprocesos de baja, cambio de área de trabajo y de cargo y la gestión de las plantillas de cargos y ocupaciones y de personal. Además, permitirá actualizar la información de los trabajadores. Estas funcionalidades contribuirán al control interno de los trabajadores.

**Palabras clave:** disponibilidad, empresa, información, recursos humanos.

INTRODUCCIÓN.....	1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.....	6
1.1 INTRODUCCIÓN.....	6
1.2 CONCEPTOS RELACIONADOS CON LA INVESTIGACIÓN.....	6
1.3 ESTUDIO DEL ESTADO DEL ARTE.....	7
1.3.1 ORACLE FUSION HCM.....	7
1.3.2 MICROSOFT DYNAMICS NAVISION.....	8
1.3.3 OPENERP/ODOO ENTERPRISE MÓDULO DE RRHH.....	9
1.3.4 FASTOS & PAGUS MÓDULO PERSONAL.....	10
1.3.5 MÓDULO DE RECURSOS HUMANOS PARA EL SISTEMA INTEGRAL DE GESTIÓN DEL FONDO CUBANO DE BIENES CULTURALES.....	12
1.3.6 SOLUCIÓN PARA INFORMATIZAR EL PROCESO DE BAJA EN LA UCI.....	13
1.3.7 VALORACIÓN DE LOS SISTEMAS ANÁLIZADOS.....	13
1.4 HERRAMIENTAS DE DESARROLLO.....	14
1.4.1 LENGUAJE DE PROGRAMACIÓN.....	14
1.4.2 MARCO DE TRABAJO PARA EL DESARROLLO.....	15
1.4.3 IDE DE DESARROLLO.....	15
1.4.4 SISTEMA GESTOR DE BASES DE DATOS.....	16
1.4.5 HERRAMIENTA CASE.....	17
1.5 HERRAMIENTA PARA PRUEBAS DE CARGA Y ESTRÉS.....	18
1.6 HERRAMIENTA PARA PRUEBAS DE SEGURIDAD.....	18
1.7 METODOLOGÍA PARA EL DESARROLLO.....	19
1.8 CONCLUSIONES PARCIALES.....	22

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN .....	23
2.1 INTRODUCCIÓN.....	23
2.2 DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN .....	23
2.3 REQUISITOS DE SOFTWARE .....	24
2.3.1 REQUISITOS FUNCIONALES.....	24
2.3.2 REQUISITOS NO FUNCIONALES .....	25
2.4 DESCRIPCIÓN DE LOS REQUISITOS FUNCIONALES.....	27
2.4.1 PLANIFICACIÓN DE LAS HISTORIAS DE USUARIO .....	29
2.4.2 PLAN DE ITERACIONES.....	30
2.5 ARQUITECTURA DE SOFTWARE .....	32
2.5 PATRONES DE DISEÑO.....	34
2.6 DIAGRAMA DE CLASES DEL DISEÑO .....	38
2.6 MODELO ENTIDAD – RELACIÓN.....	40
2.8 DIAGRAMA DE COMPONENTES .....	40
2.9 CONCLUSIONES PARCIALES .....	42
CAPÍTULO 3: IMPLEMENTACIÓN Y VALIDACIÓN DE LA PROPUESTA DE SOLUCIÓN .....	43
3.1 ESTÁNDARES DE CODIFICACIÓN .....	43
3.2 DIAGRAMA DE DESPLIEGUE .....	46
3.3 VALIDACIÓN DE REQUISITOS.....	47
3.3.1 TÉCNICAS DE VALIDACIÓN DE REQUISITOS.....	48
3.4 PRUEBAS .....	49
3.4.1 PRUEBAS ESTRUCTURALES.....	50
3.4.2 PRUEBAS FUNCIONALES.....	54

3.5 PRUEBAS DE INTEGRACIÓN .....	61
3.6 VALIDACIÓN DE LA HIPÓTESIS .....	62
3.7 CONCLUSIONES PARCIALES .....	63
CONCLUSIONES GENERALES.....	64
REFERENCIAS BIBLIOGRAFICAS .....	65
ANEXOS.....	69
ANEXO 1 ESTIMACIÓN .....	69
ANEXO 2: CUESTIONARIO A EXPERTOS.....	69
ANEXO 3: MATRIZ DE COEFICIENTES DE CONCORDANCIA POR INDICADOR.....	71

TABLA 1: VARIABLES, DIMENSIONES E INDICADORES PARA LA HIPÓTESIS.....	3
TABLA 2: COMPARACIÓN ENTRE AUP Y LA VARIANTE AUP-UCI.....	20
TABLA 3: ESCENARIOS PARA LA DISCIPLINA DE REQUISITOS.....	21
<i>TABLA 4: RF1 INSERTAR FLUJO DE BAJA.....</i>	<i>27</i>
TABLA 5: RF3 INSERTAR CAUSA DE BAJA.....	27
TABLA 6: RF5 CREAR BAJA.....	28
TABLA 7: PLANIFICACIÓN DE LAS HISTORIAS DE USUARIO.....	29
<i>TABLA 8: VARIABLES PARA LOS CASOS DE PRUEBA FUNCIONAL GESTIONAR BAJA.....</i>	<i>56</i>
TABLA 9: CASO DE PRUEBA FUNCIONAL INSERTAR BAJA.....	57
TABLA 10: CASO DE PRUEBA FUNCIONAL MODIFICAR BAJA.....	58
TABLA 11: CASO DE PRUEBA FUNCIONAL ELIMINAR BAJA.....	59
TABLA 12: FÓRMULAS.....	62
TABLA 13: ESTIMACIÓN.....	69
TABLA 28: CUESTIONARIO A EXPERTOS.....	69
TABLA 30: MATRIZ DE COEFICIENTE DE CONCORDANCIA POR INDICADOR.....	71
TABLA 31: TABLA DE FRECUENCIAS.....	71
TABLA 32: PUNTOS DE CORTE Y ESCALA DE LOS INDICADORES.....	72

FIGURA 1: MODELO (ELABORACIÓN PROPIA) .....	34
FIGURA 2: PLANTILLAS (ELABORACIÓN PROPIA).....	34
FIGURA 3: VISTAS (ELABORACIÓN PROPIA).....	34
FIGURA 4: PATRÓN EXPERTO (ELABORACIÓN PROPIA) .....	36
FIGURA 5: PATRÓN CREADOR (ELABORACIÓN PROPIA).....	37
FIGURA 6: PATRÓN DECORADOR (ELABORACIÓN PROPIA) .....	38
FIGURA 7: DIAGRAMA DE CLASES DEL DISEÑO HU5 (ELABORACIÓN PROPIA) .....	39
FIGURA 8: MODELO DE DATOS (ELABORACIÓN PROPIA).....	40
FIGURA 9: DIAGRAMA DE COMPONENTES (ELABORACIÓN PROPIA) .....	41
FIGURA 10: DIAGRAMA DE DESPLIEGUE (ELABORACIÓN PROPIA).....	47
FIGURA 11: PRUEBAS UNITARIAS (ELABORACIÓN PROPIA) .....	51
FIGURA 12: PRUEBAS DE SEGURIDAD (ITERACIÓN 1) (ELABORACIÓN PROPIA).....	52
FIGURA 13: PRUEBAS DE SEGURIDAD (ITERACIÓN 2) (ELABORACIÓN PROPIA).....	53
FIGURA 14: PRUEBAS DE SEGURIDAD (ITERACIÓN 3) (ELABORACIÓN PROPIA).....	53
FIGURA 15: PRUEBAS DE CARGA (ELABORACIÓN PROPIA) .....	56
FIGURA 16: PRUEBAS DE CAJA NEGRA (ELABORACIÓN PROPIA) .....	61



## INTRODUCCIÓN

En el mundo actual, el auge de la informática y las comunicaciones es cada vez mayor, acrecentando su uso en los diversos espacios de la vida y la gestión empresarial. La industria del software constituye una actividad económica de gran importancia a nivel internacional, pues brinda múltiples fuentes de negocio, facilita las comunicaciones y perfila como una de las oportunidades de crecimiento económico más factible para los países en vías de desarrollo. La evolución de la ciencia y la tecnología ha proporcionado novedosos sistemas que administran toda clase de datos, desde los datos contables, transacciones bancarias, controles de acceso hasta la información de las personas en las más diversas instituciones en todo el mundo. Cuba no es la excepción, pues a pesar de no contar con toda la infraestructura de países desarrollados en el ámbito de la ciencia y la tecnología si ha ganado campo en ese aspecto y son visibles los esfuerzos realizados, con el objetivo de mitigar al máximo la brecha digital existente en el mundo actual, donde los países desarrollados cuentan con toda la infraestructura necesaria para evolucionar en el campo de la tecnología y los subdesarrollados están cada vez más atrasados en ese mismo sentido.

En medio de esa revolución informática y tecnológica la Universidad de las Ciencias Informáticas (UCI), desde su fundación, ha contribuido con el proceso de informatización de la sociedad cubana. Ha desempeñado tareas que van desde el desarrollo de software de las más diversas temáticas hasta la capacitación del personal en diversas empresas, con el objetivo de facilitar y optimizar todos los procesos que se desarrollan en sus diferentes áreas. Un ejemplo de esto, lo tenemos en la actualidad donde más de un centenar de aplicaciones informáticas se han instalado en casi todos los ministerios y empresas del país. (de la Fuente y de los Ángeles Gil Estallo 2004)

Actualmente en muchas entidades los directivos, tanto de las empresas como de sus departamentos, no poseen toda la información necesaria de sus trabajadores y en ocasiones no reciben la información de sus departamentos en la forma y el momento preciso. Una organización está conformada por varios departamentos que trabajan en conjunto y están estrechamente relacionados, persiguiendo un mismo objetivo, los cuales llevan a cabo procesos que se pueden informatizar. Uno de estos departamentos, indispensables para el funcionamiento adecuado de toda empresa y el cumplimiento de sus metas, es el Departamento de Recursos Humanos (RRHH), este es el encargado de todo lo relacionado con la gestión del personal. Una de sus funciones implica controlar la relación que existe entre la organización y los

empleados. La gestión de los RRHH en Cuba, como parte del proceso de perfeccionamiento de la gestión empresarial, juega un papel esencial que se corresponde con las concepciones modernas de este campo, teniendo en cuenta que en temas de RRHH cada organización debería ajustar sus modelos a sus características propias. Los RRHH son considerados el activo más significativo de cualquier organización pues son las personas los pilares fundamentales del proceso productivo, por lo cual es de suma importancia que los datos asociados con estas sean reales y válidos. (Sabín 2005)(Morales Cartaya 2009)

Muchos de los sistemas que ha desplegado la UCI registran los datos primarios de las personas por separado, lo que conlleva a que exista duplicidad de los datos nominales de las personas de una misma entidad en varios sistemas, además muchos de los documentos que competen al departamento de RRHH se encuentran en archivos manuales y el volumen es considerable, afectando la disponibilidad de dicha información. En ocasiones la búsqueda de registros e información del personal requiere de tiempo extra para su localización y en algunos casos se encuentra desactualizada. Así, generar cualquier tipo de información se hace difícil, pues el proceso es manual e implica horas de trabajos, haciendo que muchas veces no se alcancen los resultados esperados. Los efectos de dicha situación repercuten no solo en el departamento de RRHH, sino también en aquellas áreas vinculadas a este departamento y en la organización en su totalidad.

Teniendo en cuenta los elementos mencionados anteriormente se plantea como **problema de investigación**:

¿Cómo mantener la información nominal de las personas actualizada y disponible en las diferentes entidades que deseen contar con software desarrollados por la Universidad de las Ciencias Informáticas para contribuir al control interno de los trabajadores?

Para dar solución a dicho problema se define como **objeto de estudio**: el proceso de gestión de los recursos humanos, especificando como **campo de acción**: los subprocesos asociados a la gestión de las plantillas de cargos y ocupaciones y de personal.

Por lo que se plantea como **objetivo general**: desarrollar un módulo asistente a la gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas, que permita mantener la información nominal de las personas actualizada y disponible en las entidades que deseen contar con software

## Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas.

desarrollados por la Universidad de las Ciencias Informáticas para contribuir al control interno de los trabajadores.

Para ello se establecen los siguientes **objetivos específicos**:

- I. Elaborar el marco teórico conceptual referente al proceso de gestión de los recursos humanos y el empleo de las tecnologías con este fin.
- II. Determinar las tecnologías, herramientas y la metodología de desarrollo a emplear como bases del Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas.
- III. Desarrollar los artefactos propuestos por la metodología de desarrollo seleccionada que permiten describir la solución propuesta.
- IV. Implementar un módulo que brinde asistencia a la gestión de recursos humanos para el Sistema de Información Primaria de Personas.
- V. Validar la solución a través de una estrategia de prueba.

Con el cumplimiento de lo antes expuesto se intenta demostrar la siguiente **hipótesis**:

El desarrollo de un módulo que brinde asistencia a la gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas, que permita mantener la información nominal de las personas actualizada y disponible, en las entidades que deseen contar con software desarrollados por la Universidad de las Ciencias Informáticas, podría contribuir al control interno de los trabajadores.

**Variable independiente:** Módulo Asistente a la Gestión de los Recursos Humanos y **variables dependientes:** información nominal de las personas actualizada y disponible y control interno de los trabajadores.

Para ofrecer una mejor caracterización de las variables, se procedió a su operacionalización:

**Tabla 1:** Variables, dimensiones e indicadores para la hipótesis.

<b>Variables conceptuales</b>	<b>Dimensiones</b>	<b>Indicadores</b>
Módulo Asistente a la Gestión de Recursos Humanos.	Entendimiento	Excelente = 5
	Aprendizaje	Bien = 4

**Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas.**

	Operabilidad	Regular = 3 Mal = 2
	Atracción	
	Esfuerzo del usuario	
	Facilidad de uso	
	Capacidad de la interfaz visual	
	Seguridad	
	Tolerancia a fallas	
	Recuperación	
	Tiempo de procesos	
	Capacidad de ser analizado	
	Facilidad de prueba	
	Posibilidad de actualización	
	Estabilidad	
	Coexistencia	
	Exportación	
	Aplicabilidad	
	Entendimiento	
Información nominal de las personas actualizada y disponible.	Posibilidad de actualizar los datos nominales	
	Disponibilidad de la información de los trabajadores	
	Información confiable	
Control interno de los trabajadores.	Información de puestos de trabajo	
	Información de área a la que pertenecen los trabajadores	
	Información de cargos de trabajadores	

Los **métodos teóricos** utilizados para cumplir con las tareas a desarrollar son:

**Análisis histórico-lógico:** permitió estudiar cómo se desarrolla la gestión de los recursos humanos en la universidad, así como los cambios que ha sufrido dicha gestión hasta el surgimiento de la necesidad de realización del presente trabajo.

**Análítico – Sintético:** se hace un estudio de los documentos que se manejan en el proceso de gestión del personal en las áreas de la Universidad de las Ciencias Informáticas para así identificar los problemas existentes en la misma, sus causas y lograr un mejor análisis y diseño.

Los **métodos empíricos** utilizados para cumplir con las tareas a desarrollar son:

**Observación:** se realizará un seguimiento sobre cómo se ejecutan las diferentes tareas en el Departamento de Recursos Humanos para hacer un registro visual de los procesos que se llevan a cabo en la actualidad en las diferentes áreas.

La presente investigación se encuentra estructurada en tres capítulos:

**Capítulo 1:** Fundamentación teórica metodológica de la investigación. Se enuncian los conceptos asociados al dominio del problema, se analizan las soluciones existentes hasta el momento ante problemas similares. Además, se precisan las tecnologías y herramientas a utilizar para dar solución al problema planteado, así como la metodología de desarrollo de software que se empleará para implementar de la propuesta de solución.

**Capítulo 2:** Análisis y diseño de la propuesta de solución. Se enuncian y describen los requisitos funcionales y no funcionales y la arquitectura seleccionada, así como los respectivos modelos de diseño y de datos, correspondientes a la propuesta de solución. También se describen los patrones arquitectónicos y de diseño empleados, de acuerdo a la arquitectura seleccionada.

**Capítulo 3:** Implementación y validación a la propuesta de solución. Se muestran y describen los artefactos ingenieriles relacionados con la metodología de desarrollo seleccionada. Entre los principales elementos se encuentran el modelo de despliegue y los estándares de codificación. Además, se detallan las pruebas realizadas al módulo.

## **CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA**

### **1.1 INTRODUCCIÓN**

Con el fin de lograr una mayor comprensión del alcance de la investigación, se presentan en este capítulo los elementos teóricos fundamentales que la respaldan. Se realiza un estudio de los conceptos asociados, así como de las herramientas existentes que permiten la gestión de los recursos humanos. Además de un estudio de las características y elementos esenciales de la metodología empleada para guiar el proceso de desarrollo de la solución propuesta; al igual que de las herramientas y tecnologías con las que se desarrolla.

### **1.2 CONCEPTOS RELACIONADOS CON LA INVESTIGACIÓN**

#### **Recursos humanos**

Una empresa cuenta con diversos tipos de recursos que le permiten funcionar y alcanzar sus metas. Los empleados, trabajadores y colaboradores son quienes conforman lo que se conoce como recursos humanos de una entidad.

El concepto también se utiliza para nombrar al departamento, la oficina o la persona que se dedica a analizar, elegir, contratar, formar y retener a los trabajadores de una compañía.(ASALE 2014)

#### **Gestión**

Hace referencia a la acción y a la consecuencia de administrar o gestionar algo. Gestionar es llevar a cabo diligencias que hacen posible la realización de una actividad. Administrar abarca las ideas de gobernar, disponer, dirigir, ordenar u organizar una determinada cosa o situación.(ASALE 2014)

#### **Gestión de recursos humanos**

Actividades coordinadas para dirigir y controlar una organización, que permiten materializar la política laboral, que se aplican con la participación activa y efectiva de los trabajadores en la planificación, organización, dirección, control y evaluación de los recursos humanos, que determinan o inciden en el desempeño de la organización. Los objetivos que intenta cumplir la gestión de recursos humanos es

desarrollar y administrar diferentes políticas y programas para de esta manera poder brindarle a la estructura administrativa empleados capaces.(Alles 2012)

### **1.3 ESTUDIO DEL ESTADO DEL ARTE**

#### **1.3.1 ORACLE FUSION HCM**

HCM Fusion es un software de gestión de recursos humanos. Es una solución desarrollada por Oracle. Abarca la mayoría de los aspectos de los recursos humanos. Es un software privativo, para su adquisición es necesario el pago de licencia, además permite la integración con otros productos, especialmente los de Oracle.

#### **Funcionalidades más destacadas de Fusion HCM en sus módulos:**

1. Administración de puestos y ascensos.
2. Administración global de recursos humanos.
3. Asignación de puestos de trabajo.
4. Autoservicio del empleado.
5. Autoservicio del mánager.
6. Cálculo de impuestos y prestaciones.
7. Control de asistencia.
8. Evaluación del rendimiento.
9. Gestión de contratación.
10. Gestión de flujos de trabajo.
11. Gestión de nóminas.
12. Gestión de períodos vacacionales.
13. Gestión de personal.
14. Gestión del talento.
15. Planificación y gestión de la organización.

#### **Entre las ventajas de Oracle Fusion HCM se pueden citar:**

1. Varias opciones para la implementación.

2. Frecuentes actualizaciones.
3. Módulos especializados.
4. Fuertes opciones para la seguridad de los datos.
5. Herramientas de inteligencia organizacional completamente integradas con los datos de RRHH.
6. Si tiene un entorno empresarial de Oracle, la implementación es mucho más sencilla.

**Y algunas de sus desventajas son:**

1. Necesita una gran cantidad de memoria para su versión *on-premise*.
2. La terminología de Oracle puede no resultar atractiva para los usuarios.
3. No tiene opciones para la gestión de formación a pesar de tener módulo de gestión del talento.
4. Su módulo de contratación es bastante limitado.

Las necesidades y características de cada empresa determinarán el precio final del software. Dependiendo del proveedor elegido, módulos adquiridos o número de usuarios, existirán variaciones. El precio de este sistema, al ser SaaS (*Software como servicio*), se determina por número de usuarios por mes y número de módulos contratados.(Smith 2016)

### **1.3.2 MICROSOFT DYNAMICS NAVISION**

**Microsoft Dynamics NAV permite controlar:**

1. Información de los colaboradores.
2. Ubicación laboral.
3. Riesgos laborales.
4. Previsiones salariales.
5. Reclutamiento de trabajadores.
6. Selección de trabajadores.
7. Planes y acciones de capacitación.

**Funcionalidades más destacadas de Microsoft Dynamics Navision en sus módulos:**

1. Inventario de personal.



2. Disponibilidad de plantilla.
3. Puestos de trabajo.
4. Riesgos laborales.
5. Previsiones salariales.
6. Selección de personal.
7. Planes de formación.
8. Enlace con *Infojobs*.

**Ventajas:**

1. Ahorro de tiempo al buscar información de los empleados.
2. Ofrece constantes actualizaciones a sus usuarios.
3. Control en tiempo real de los recursos materiales.
4. Se integra de manera nativa con otros productos, como Excel, Dynamics CRM y SharePoint.
5. Sobresale los cortos tiempos de formación que necesitan los usuarios para su correcto funcionamiento.
6. Gran flexibilidad y adaptabilidad.
7. El tiempo de implantación es menor al que requieren otros ERP.
8. Inclusión de flujos de trabajo.

**Desventajas:**

1. Uno de los inconvenientes para su adquisición es su costo económico, ligeramente mayor al de otros programas de gestión.

**1.3.3 OPENERP/ODOO ENTERPRISE MÓDULO DE RRHH**

Este módulo de gestión de recursos humanos permite manejar: gastos, asistencias, faltas, jornadas de reclutamiento y evaluaciones periódicas. Para su implementación se contó con diferentes herramientas, entre las que se pueden mencionar las siguientes:

1. Lenguaje Python para el desarrollo del producto dando la posibilidad de hacer el producto portable, además, se hizo uso de lenguajes como lenguaje de marcado extensible (XML), JavaScript y hojas de estilo en cascada (CSS).
2. Postgres SQL como servidor de Base de Datos.
3. Pycharm y PyDev o Eclipse para la implementación.

**Entre las funcionalidades más destacadas de OPENERP/ODOO en sus módulos podemos citar:**

1. Seguimiento de asistencia.
2. Gestión de empleados.
3. Reclutamiento.
4. Bajas.
5. Asignaciones.
6. Hojas de trabajo.
7. Evaluaciones.
8. Gastos.
9. Reportes.
10. Gestión de requerimientos de empleados.(Moss 2017)

#### **1.3.4 FASTOS & PAGUS MÓDULO PERSONAL**

Fastos & Pagus es un sistema que permite controlar las informaciones fundamentales de los empleados de una entidad. Conformado por los módulos: Configuración, Personal, Capacitación, Cuadros, además genera y controla las nóminas y se configura acorde a las necesidades de pago de cada cliente.

#### **Funcionalidades del Sistema Fastos & Pagus:**

1. Registro de los empleados: se guardan los datos de los empleados, así como informaciones referentes a los reportes de vacaciones, certificados médicos, licencias y resolución.
2. Control de la plantilla: permite establecer la estructura organizativa de las plazas de la entidad.
3. Control de asistencia: incorpora el control de claves de asistencias, turnos de trabajos, horarios, tarjeta de asistencia e incidencias de cada empleado.

4. Informes y modelos: permite obtener un total de 11 modelos y 56 informes, por ejemplo: cierre del período, análisis de fondo de tiempo, estadísticos, entre otros.
5. Control de la capacitación: acciones de capacitación, estudios realizados, cursos, eventos, experiencia docente, publicaciones, conocimientos, idiomas extranjeros, plan de desarrollo e informes.
6. Control de la información de los cuadros: se establece el registro de los cuadros, dirigentes y reserva, referente a evaluaciones, inspecciones, sanciones, necesidades de capacitación, entre otros.
7. Evaluación de Desempeño: Establece un procedimiento general que posibilita definir de forma dinámica los modelos evaluativos y sus respectivos indicadores, en función de las necesidades de cada entidad, también permite asociar el o los modelos evaluativos que se utilizan en este proceso para cada trabajador y efectuar la correspondiente evaluación del período indicado, manteniendo el control de dicho proceso. Si se instala el Módulo de Nómina, es posible configurar una nómina para el pago por resultados, tomando como base el cierre de evaluaciones del período.
8. Nómina: posee facilidades para su configuración y adaptación a los intereses de los clientes. Se enlaza de forma automática con la información generada por el módulo de Personal.

**Funcionalidades del módulo Personal:**

1. Definición de los nomencladores a utilizar.
  - a. Áreas y Departamentos.
  - b. Cargos.
  - c. Movimiento de nómina y otros.
2. Registro de los empleados y sus datos.
3. Control de asistencia.
  - a. Claves de asistencias.
  - b. Turnos de trabajo.
  - c. Horarios.
  - d. Tarjeta de asistencia.
  - e. Incidencias.
4. Obtención de Informes.
5. Cálculo de la pre nómina.(Colectivo de autores 2014)

### **1.3.5 MÓDULO DE RECURSOS HUMANOS PARA EL SISTEMA INTEGRAL DE GESTIÓN DEL FONDO CUBANO DE BIENES CULTURALES**

El módulo de RRHH para el sistema integral de gestión del Fondo Cubano de Bienes Culturales fue desarrollado por el centro de desarrollo de software de la Facultad Regional Granma de la Universidad de las Ciencias Informáticas. Dicho módulo tiene como objetivo agilizar los procesos que se ejecutan en el departamento de Recursos Humanos del Sistema Integral de Gestión del Fondo Cubano de Bienes Culturales, permitiendo que estos se efectúen de forma rápida con eficiencia y calidad. El sistema de gestión fue desarrollado en el Sistema Operativo GNU/Linux (Ubuntu), haciendo uso de la metodología SXP para su implementación fue necesario contar con diferentes herramientas, entre las que se pueden mencionar:

1. Marco de trabajo Django como tecnología principal, propia del lenguaje Python, para su implementación, utilizado para acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones.
2. Lenguaje Python para el desarrollo del producto dando la posibilidad de hacer el producto portable. Es un lenguaje interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece ventajas como la rapidez de desarrollo, además, se hizo uso de lenguajes como HTML, JavaScript y CSS.
3. Postgres SQL como gestor de Base de Datos.
4. Visual Paradigm como herramienta CASE para el modelado de diagramas.

Dentro de los servicios que brinda el sistema de gestión creado se pueden mencionar los siguientes:

1. Registrar: permite a los usuarios realizar cambios en dependencia de los privilegios que tenga cada uno.
2. Prenómina: el sistema permite gestionar los datos correspondientes a las prenominas (pago adicional de los trabajadores, horas descontadas, certificados médicos, etc.). Además, permite obtener la información detallada de cada trabajador.
3. Reportes: el sistema de gestión brinda la posibilidad de realizar reportes estáticos y dinámicos en formato de documento portable, en inglés *Portable Document Format* (PDF) y Excel. Estos están presente en todo el sitio, realizándose los mismos en tiempo real.(Pompa Nuñez 2013)

### **1.3.6 SOLUCIÓN PARA INFORMATIZAR EL PROCESO DE BAJA EN LA UCI**

Este módulo permite integrar el proceso de baja al Sistema de Gestión Universitaria de la universidad, cuyo objetivo radica en garantizar la devolución de los recursos, así como el cierre de los servicios que fueron puestos a disposición del personal cuando entró a la universidad. Para la elaboración de dicha propuesta, el proceso fue guiado por la metodología DAC (Desarrollo Ágil con Calidad) y su construcción se basó en herramientas y tecnologías libres como: NetBeans, PostgreSQL y Visual Paradigm.

**Algunas de las funcionalidades de dicho módulo son:**

1. Mostrar solicitudes de baja.
2. Crear solicitud de baja.
3. Generar documento de baja.
4. Configurar áreas de visita obligatoria.
5. Mostrar estados de solicitud.
6. Modificar causa de baja.
7. Configurar personas autorizadas a dar baja en el área de baja.(Ruiz Llanes 2015)

### **1.3.7 VALORACIÓN DE LOS SISTEMAS ANALIZADOS**

A pesar de ser múltiples los sistemas desarrollados para la gestión de los recursos humanos en el mundo, la gran mayoría están encaminados a resolver el problema en una empresa específica. Los procesos que se llevan a cabo en el área de RRHH difieren en dependencia de las legislaciones propias de cada país en este tema, así como por las particularidades de la empresa en que se desarrollan.

Después del estudio realizado se llega a la conclusión de que la elección de las herramientas más convenientes para el desarrollo de la propuesta de solución depende de las necesidades de los clientes, así como de las funcionalidades que ofrezcan de base para soportar una implementación que responda a dichas necesidades. Durante el estudio de las diversas soluciones existentes que manejan información asociada a la gestión personal, se pudo constatar que poseen una gran variedad de funcionalidades en dependencia del entorno empresarial que se encuentre y aunque los sistemas estudiados no son aptos para integrarse al Sistema de Información Primaria de Personas sirvieron como guía para la implementación de la propuesta

de solución en especial la solución para informatizar el proceso de baja en la Universidad de las Ciencias Informáticas y el Módulo de RRHH de Odo Open ERP.

## **1.4 HERRAMIENTAS DE DESARROLLO**

### **1.4.1 LENGUAJE DE PROGRAMACIÓN**

Python 3.5 es un lenguaje de programación fácil de aprender y poderoso. Tiene estructuras de datos de alto nivel eficientes y un enfoque simple pero efectivo para la programación orientada a objetos. La elegante sintaxis y el tipado dinámico de Python, junto con su naturaleza interpretada, lo convierten en un lenguaje ideal para el desarrollo de aplicaciones web.

Python fue publicado por primera vez por Guido Van Rossum en 1991. El lenguaje tiene un modelo abierto de desarrollo basado en la comunidad, administrado por la organización sin fines de lucro Python Software Foundation. Existen varios intérpretes y compiladores que implementan el lenguaje Python, incluyendo uno en Java (Jython).

Python posee una amplia librería disponible de forma gratuita para todas sus plataformas. Además, posee varios módulos, programas y herramientas de tercero gratuitas que cuentan con documentación.

El intérprete de Python se extiende fácilmente con nuevas funciones y tipos de datos implementados en C o C ++. Python también es adecuado como lenguaje de extensión para aplicaciones personalizables.

Python soporta programación orientada a objetos, programación imperativa y funcional, además de otros paradigmas de programación. También tiene un sistema de tipado dinámico y manejo automatizado de memoria utilizando conteo de referencias.

Se puede ejecutar en Windows, Linux/Unix, Mac OS X, OS/2, Amiga, Palm Handhelds y teléfonos celulares Nokia. También, ha sido portado para las máquinas virtuales de Java y .NET.(Colectivo de autores 2015)

Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License.

### 1.4.2 MARCO DE TRABAJO PARA EL DESARROLLO

Django 1.10.3 es un marco de trabajo web Python de alto nivel que fomenta un desarrollo rápido y un diseño limpio y pragmático. Desarrollado por desarrolladores experimentados, se encarga de gran parte de las complicaciones del desarrollo web, por lo que puede concentrarse en escribir su aplicación sin necesidad de reinventar la rueda. Es gratis y de código abierto. Django fue diseñado para ayudar a los desarrolladores a llevar las aplicaciones desde el concepto hasta su finalización lo más rápido posible, toma en serio la seguridad y ayuda a los desarrolladores a evitar muchos errores comunes de seguridad. Algunos de los sitios más concurridos de la web aprovechan la capacidad de Django para escalar de forma rápida y flexible. Django también cuenta con:

1. Servidor web incluido para entorno de desarrollo.
2. Potente ORM (Mapeo Objeto-Relacional) incluido.
3. Sistema de plantillas.
4. Agradables *urls*.
5. Base de datos embebida.

Utiliza además una variación de la arquitectura MVC (Model-View-Controller/Modelo-Vista-Controlador) llamada MTV (Model-Template-View/Modelo-Plantilla-Vista), así se puede modificar algo en un archivo sin afectar otros ficheros. Todo esto bajo el principio de una y sólo una vez.(Forcier, Bissex y Chun 2008)

### 1.4.3 IDE DE DESARROLLO

Un entorno de desarrollo integrado o entorno de desarrollo interactivo, en inglés *Integrated Development Environment* (IDE), es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.

Normalmente, un IDE consiste de un editor de código fuente, herramientas de construcción automáticas y un depurador. La mayoría de los IDE tienen auto-completado inteligente de código.

PyCharm 2017.2.2 es un IDE que se utiliza para la programación en Python. Proporciona análisis de código, un depurador gráfico, un probador de unidad integrada, integración con sistemas de control de versiones, y apoya el desarrollo web con Django.

Funciona en Windows, Mac OS o Linux con una única clave de licencia. Posee un depurador de código integrado y corredor de pruebas gráficas. Además, incorpora un terminal y permite la integración con Git (Sistema concurrente de control de versiones), Apache Subversion, Mercurial y herramientas integradas de base de datos.

Permite desarrollar, depurar, probar e implementar aplicaciones en servidores remotos o máquinas virtuales, con intérpretes remotos, intérprete de órdenes seguro, y la integración Vagrant. ofrece un gran apoyo específico para los marcos de desarrollo web como Django, Frasco, Google App Engine, Pirámide, y web2py.

Presenta una asistencia inteligente de codificación con la finalización de código inteligente, inspecciones de código, resaltado de error sobre la marcha, auto-correcciones, junto con refactorizaciones de código automatizadas y capacidades de navegación, ricos para las principales implementaciones de Python.(Islam 2015)

### **1.4.4 SISTEMA GESTOR DE BASES DE DATOS**

PostgreSQL 9.4 es un potente sistema de base de datos, de código abierto objeto-relacional. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación por su fiabilidad, integridad de datos y corrección. Se ejecuta en todos los sistemas operativos, incluyendo Linux, UNIX y Windows. Cuenta además con varios tipos de datos, incluyendo entero, numéricos, booleanos, char, varchar, date, intervalo y timestamp. También es compatible con el almacenamiento de grandes objetos binarios, como imágenes, sonidos y vídeos. Cuenta con las interfaces de programación nativas para C / C ++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros, además de poseer una amplia documentación.

PostgreSQL cuenta con características sofisticadas como control de concurrencia de varias versiones (MVCC por sus siglas en inglés), punto en el tiempo de recuperación, *tables-paces*, replicación asincrónica, transacciones anidadas (puntos de retorno), en línea, copias de seguridad y un sofisticado planificador de consulta. Es compatible con los juegos de caracteres internacionales, codificación de caracteres *multibytes*, *unicode*, y es consciente de la configuración regional de clasificar, mayúsculas y minúsculas, y el formato. Es altamente escalable tanto en la enorme cantidad de datos que puede manejar como en el número de



usuarios simultáneos que puede acomodar. Hay sistemas activos de PostgreSQL en entornos de producción que manejan más de cuatro *terabytes* de datos.(Dar et al. 2015)

#### **1.4.5 HERRAMIENTA CASE**

Las herramientas del proceso de ingeniería de software proporcionan un soporte automático o semiautomático para el proceso y los métodos, a estas herramientas se les llaman herramientas CASE (*Computer-Aided Software Engineering* o Ingeniería de Software Asistida por Computadora).

Se puede definir a las Herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los procesos del ciclo de vida de desarrollo de un software.

Visual Paradigm 8.0 para UML es una herramienta CASE que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, implementación y pruebas. Ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite construir diagramas de diversos tipos, código inverso, generar código desde diagramas y generar documentación.

Se integra con diversos IDE´s como: NetBeans (de Sun), Eclipse (de WebShere), JDeveloper, entre otros. Está disponible en varias ediciones: *Enterprise, Professional, Community, Standard, Modeler* y *Personal*. Genera código y realiza ingeniería inversa para diferentes lenguajes de programación como: PHP, Java, Python, entre otros.(Oscar 2013)

Según las políticas trazadas por el país y la universidad, sobre el uso de herramientas multiplataforma y de código abierto, se decide utilizar el Visual Paradigm para UML como herramienta CASE.

## 1.5 HERRAMIENTA PARA PRUEBAS DE CARGA Y ESTRÉS

Apache JMeter es un software de código abierto, diseñada para cargar el comportamiento funcional de aplicaciones y medir su rendimiento. Originalmente fue diseñado para probar aplicaciones web, pero desde entonces se ha expandido a otras funciones de prueba. (Halili 2008)

### Algunas de las características que incluye son:

1. Capacidad para cargar y probar el rendimiento de aplicaciones, servidores y protocolos.
2. IDE de prueba con todas las funciones que permite la grabación rápida de plan de prueba (desde navegadores o aplicaciones nativas), compilación y depuración.
3. Modo de línea de comandos (modo no guiado / sin cabeza) para cargar la prueba desde cualquier sistema operativo compatible con Java.
4. Presenta informe HTML dinámico completo.
5. Fácil correlación mediante la capacidad de extraer datos de los formatos de respuesta más populares, HTML, *JavaScript Object Notation* (JSON), XML o cualquier formato de texto.
6. Portabilidad completa.
7. El marco completo de *multi-threading* permite el muestreo concurrente por muchos hilos y el muestreo simultáneo de diferentes funciones por grupos de hilos separados.
8. Almacenamiento en caché y fuera de línea / reproducción de resultados de pruebas.
9. Núcleo altamente extensible.
10. Se pueden elegir varias estadísticas de carga con temporizadores conectables.
11. Los complementos de análisis y visualización de datos permiten una gran extensibilidad y personalización.
12. Las funciones se pueden usar para proporcionar una entrada dinámica a una prueba o para proporcionar manipulación de datos.
13. Fácil integración continua a través de bibliotecas de código abierto de terceros.

## 1.6 HERRAMIENTA PARA PRUEBAS DE SEGURIDAD

Acunetix Web Vulnerability Scanner 9 es una herramienta muy utilizada en el escáner de vulnerabilidades en aplicaciones web. Es capaz de escanear cualquier sitio web o aplicación web que sea accesible a través

del protocolo HTTP / HTTPS. Permite realizar análisis automáticos para la realización de pruebas, pero no todas se pueden realizar de forma automática, por lo que proporciona herramientas de pruebas manuales para análisis particulares. Además, escanea los sitios usando algunas de las vulnerabilidades y anuncia todos los problemas y métodos de infiltración.(Russell y Cohn 2012)

### **Características de Acunetix Web Vulnerability Scanner:**

1. Escáner web: un escáner web es una verificación de seguridad automática. El escaneo de seguridad del sitio web generalmente incluye dos fases:
  - 1.1 Rastreo: analiza el sitio automáticamente para construir la estructura del sitio.
  - 1.2 Escaneo: lanzamiento de una serie de ataques contra archivos y programas basados en vulnerabilidades web.
2. Escaneo de puertos: realizar un escaneo de puertos contra un sitio de alojamiento de servidor web.
3. Buscador de objetivos: especifica el servidor web en los puertos 80, 443 en un rango de direcciones IP.
4. Escáner de subdominio: escanea subdominios usando varias técnicas.
5. Inyector de SQL ciego: Ideal para pruebas de penetración, Blind SQL Injector es una herramienta de extracción de datos que puede analizar errores de inyección sql.
6. Editor http: crea, analiza y edita solicitudes http y respuestas de servidor.
7. Sniffer http: verifica y modifica el tráfico http entre el cliente http y un servidor web.
8. Admite sitios HTML5.
9. Detección de vulnerabilidad JSON, datos XML y encabezados HOST http.
10. Detección de vulnerabilidad *Cross-site scripting* (XSS).
11. Detección de vulnerabilidad de XSS ciego.

### **1.7 METODOLOGÍA PARA EL DESARROLLO**

En todo sistema de calidad de software es necesario la utilización de procedimientos, guías y técnicas que definan el modo de construcción del software. La misma se basa en tres pilares básicos: qué hay que hacer y en qué orden, cómo y con qué deben llevarse a cabo. Esto es, básicamente, qué etapas, actividades y tareas se deben acometer, qué técnicas deben emplearse para realizar estas actividades y cuáles son las herramientas de software a utilizar en cada caso.

Existen metodologías ágiles y robustas. Las ágiles son convenientes para guiar proyectos de escaso volumen que demanden una rápida implementación. Por otro lado, las robustas pueden ser empleadas para guiar el proceso de desarrollo de proyectos grandes o pequeños.

La metodología empleada para el desarrollo de la presente investigación se basó en el método de desarrollo ágil: Proceso Unificado Ágil (AUP). Esta es una versión simplificada de RUP (Proceso Unificado de Software), que describe, de una manera simple y fácil de entender, la forma de desarrollar aplicaciones de software de negocio, usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. (Jacobson, Booch y Rumbaugh 2000)

En la UCI, se decide realizar una variación de esta metodología, de forma tal que se adapte al ciclo de vida definido para la actividad productiva en la universidad, logrando estandarizar el proceso de desarrollo de software. De las cuatro fases que propone AUP (Inicio, Elaboración, Construcción, Transición), se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola titulada Ejecución y se agrega la fase de Cierre.

A continuación, se muestra la comparación entre AUP y la variante AUP-UCI con relación a sus fases y los objetivos específicos de las mismas para esta última (Ver Tabla 2).

**Tabla 2:** Comparación entre AUP y la variante AUP-UCI

Fases AUP	Fases Variación AUP-UCI	Objetivos de las fases (Variación AUP-UCI)
Inicio	Inicio	Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
Elaboración	Ejecución	En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. Durante esta fase el producto es transferido al ambiente de los usuarios finales o entregado al cliente. Además, en la
Construcción		
Transición		

		transición se capacita a los usuarios finales sobre la utilización del software.
	Cierre	En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

**Descripción de las disciplinas.**

AUP propone 7 disciplinas (Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyecto y Entorno). Se decide para el ciclo de vida de los proyectos de la UCI tener 7 disciplinas también, pero a un nivel más atómico que el definido en AUP. Los flujos de trabajos: Modelado de negocio, Requisitos y Análisis y diseño en AUP están unidos en la disciplina Modelo, en la variación para la UCI se consideran a cada uno de ellos disciplinas.

**Escenarios para la disciplina Requisitos.**

El esfuerzo principal, en la disciplina Requisitos, es desarrollar un modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto. Existen tres formas de encapsular los requisitos [Casos de Uso del Sistema (CUS), Historias de usuario (HU) y Descripción de requisitos por proceso (DRP)], agrupados en cuatro escenarios condicionados por el Modelado de negocio.(Rodríguez Sánchez 2014)

A continuación, se presentan los escenarios para la disciplina de requisitos de acuerdo a la variante AUP-UCI (Ver Tabla 3).

**Tabla 3:** Escenarios para la disciplina de requisitos

<b>Escenarios</b>	<b>Características</b>
Escenario No 1	Proyectos que modelen el negocio con CUN solo pueden modelar el sistema con CUS.
Escenario No 2	Proyectos que modelen el negocio con MC solo pueden modelar el sistema con CUS
Escenario No 3	Proyectos que modelen el negocio con DPN solo pueden modelar el sistema con DRP.
Escenario No 4	Proyectos que no modelen negocio solo pueden modelar el sistema con HU.

Para el caso de la presente investigación, por las características que presenta el sistema, se utiliza el Escenario No 4: Historias de Usuario.

### **1.8 CONCLUSIONES PARCIALES**

Los sistemas consultados que de alguna manera tributan a la investigación, no satisfacen el problema de la misma, aunque aportaron elementos significativos para el desarrollo de la propuesta de solución y permitió identificar las tendencias actuales de los sistemas que soportan la gestión de los RRHH. Los elementos significativos de las herramientas, tecnologías y tendencias a utilizar, satisfacen las necesidades existentes para el desarrollo de la propuesta de solución. Además, se puede afirmar que las mismas poseen características y ventajas con respecto a otras que facilitan la implementación del mismo.

## **CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN**

### **2.1 INTRODUCCIÓN**

En el presente capítulo se describe la propuesta de solución, además, se enuncian los requisitos funcionales y no funcionales y la arquitectura seleccionada. De ellos se presentan las historias de usuario correspondientes. Igualmente se describen los patrones arquitectónicos y de diseño a utilizar, de acuerdo a la propuesta arquitectónica.

### **2.2 DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN**

Como solución a la problemática descrita anteriormente se propone la implementación de un módulo que permita informatizar el proceso de baja y la gestión de la información del personal, que sea configurable y adaptable a diversos entornos. El módulo permitirá configurar el flujo que debe seguir el proceso de baja a través de los diferentes departamentos establecidos a autorizar la misma, siendo obligatorio para la culminación de dicho proceso la aprobación por todos los departamentos involucrados. Cada uno de los procesos de baja debe tener una persona autorizada a darle inicio y culminación al mismo. El módulo gestionará todo lo relacionado con el flujo de las bajas donde es de vital importancia el área y el cargo de la persona que autoriza la baja. Además, gestionará las causas de las bajas dado el motivo que les da origen. Luego de estar definido el flujo de bajas y las causas se le puede dar inicio al proceso de la baja, después de haber seleccionado la persona que solicita la baja, así como la causa que origina la misma, se registrarán dichos datos, además del estado, donde por defecto será inicio, fecha de solicitud y la persona que inicia el proceso. Se generarán reportes de bajas por área y por causas, donde se mostrará todo lo relacionado con las bajas del área que se seleccione o la causa que da inicio a la misma. Otra de las funcionalidades con que contará el módulo, es gestionar la información nominal del trabajador de un área, donde se permitirá modificar los datos personales del trabajador, así como actualizar el área a la que pertenece y el cargo que ocupa en caso de que esta cambie. Se mostrarán las plazas por cargo de cada área y de ellas las que están cubiertas. Así mismo la lista de trabajadores por el área a la que pertenecen, el cargo, el rol que desempeñan en el sistema y las observaciones.

## 2.3 REQUISITOS DE SOFTWARE

Los requisitos de software constituyen las necesidades de los clientes, las funcionalidades y restricciones que debe cumplir el software. Los mismos se clasifican en funcionales y no funcionales.(Wieggers 2009)

### 2.3.1 REQUISITOS FUNCIONALES

Los Requisitos Funcionales (RF) 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. A continuación, se identifican los RF del módulo a desarrollar.

El módulo debe cumplir con los siguientes RF:

- RF1. Agregar área al flujo de baja
- RF2. Eliminar área del flujo de baja
- RF3. Insertar causa de baja
- RF4. Modificar causa de baja
- RF5. Crear baja.
- RF6. Insertar estado de la solicitud de baja
- RF7. Modificar estado de la solicitud de baja
- RF8. Cancelar baja
- RF9. Reporte bajas por áreas
- RF10. Reporte bajas por causa
- RF11. Modificar cargo de un trabajador
- RF12. Modificar dirección de un trabajador
- RF13. Modificar teléfono de un trabajador
- RF14. Realizar movimiento de trabajadores de un área
- RF15. Reporte de trabajadores de un área
- RF16. Gestionar plantilla de cargos y ocupaciones de un área (P2)
- RF17. Gestionar plantilla de personal de un área (P4)



### **2.3.2 REQUISITOS NO FUNCIONALES**

#### **Requisitos no funcionales de software para estaciones clientes**

RnF1. Sistema Operativo Linux o Windows.

RnF2. Navegador web Mozilla Firefox v17.0 o superior, Google Chrome v20.0 o superior, o versiones actuales de Opera, Internet Explorer y Safari.

#### **Requisitos no funcionales de software para estaciones servidores**

RnF3. Sistema operativo Linux o Windows.

RnF4. Gestor de bases de datos: PostgreSQL 9.4.

RnF5. Lenguaje de programación: Python 3.5.

#### **Requisitos no funcionales de hardware para estaciones clientes**

RnF6. PC Pentium 4 a 1GHz o superior, mínimo 512 MB de RAM.

#### **Requisitos no funcionales de hardware para estaciones servidores**

RnF7. PC Pentium 4 a 2 GHz o superior, mínimo 2 GB de RAM, 20 GB o superior de disco duro.

#### **Requisitos no funcionales de usabilidad**

RnF8. Facilidad de uso por parte de los usuarios: el sistema debe presentar una interfaz amigable que permita la fácil interacción con el mismo y llegar de manera rápida y efectiva a la información buscada. Además, debe ser una interfaz de manejo cómodo que posibilite a los usuarios sin experiencias una rápida adaptación.

RnF9. El sistema podrá ser utilizado por usuarios con las siguientes características:

1. Conocimientos básicos relativos al uso de una computadora.
2. Conocimientos sólidos relativos a los procesos de negocio acorde al rol que desempeñe.

RnF10. El sistema será distribuido en idioma español.

RnF11. Los términos utilizados se establecerán acorde al negocio correspondiente para facilitar la comprensión de la herramienta de trabajo.

RnF12. El sistema poseerá estructura y diseño homogéneos en todas sus pantallas, que facilite la navegación:

1. Menús laterales y desplegados que permitan el acceso rápido a la información por parte de los usuarios.

### **Restricciones en el diseño y la implementación**

RnF13. Lenguaje de programación: Python.

RnF14. Plataforma de desarrollo: JetBrains Pycharm 2017 2.2.

RnF15. Marco de trabajo: Django 1.10.

RnF16. Para el acceso a datos se utilizará Django ORM.

RnF17. Para el modelado de UML 2.0 se utilizará Visual Paradigm 8.0.

RnF18. Gestor de base de datos se utilizará PostgreSQL 9.4.

### **Requisitos no funcionalidades de seguridad**

RnF19. La seguridad se define a nivel de roles, con el fin de mantener la integridad de los datos en función del acceso de cada uno de ellos, trayendo consigo además la protección de la información.

RnF20. Autenticación segura para acceder a la aplicación.

### **Requisitos no funcionalidades de fiabilidad**

RnF21. Si ocurren errores en el sistema no se mostrarán detalles de los mismos al cliente.

## 2.4 DESCRIPCIÓN DE LOS REQUISITOS FUNCIONALES

Los RF son la entrada esencial para realizar el análisis, diseño, implementación y pruebas de un sistema. Es por ello que realizar la descripción de los mismos, es de gran importancia. (Cardozzo y Academy 2016)

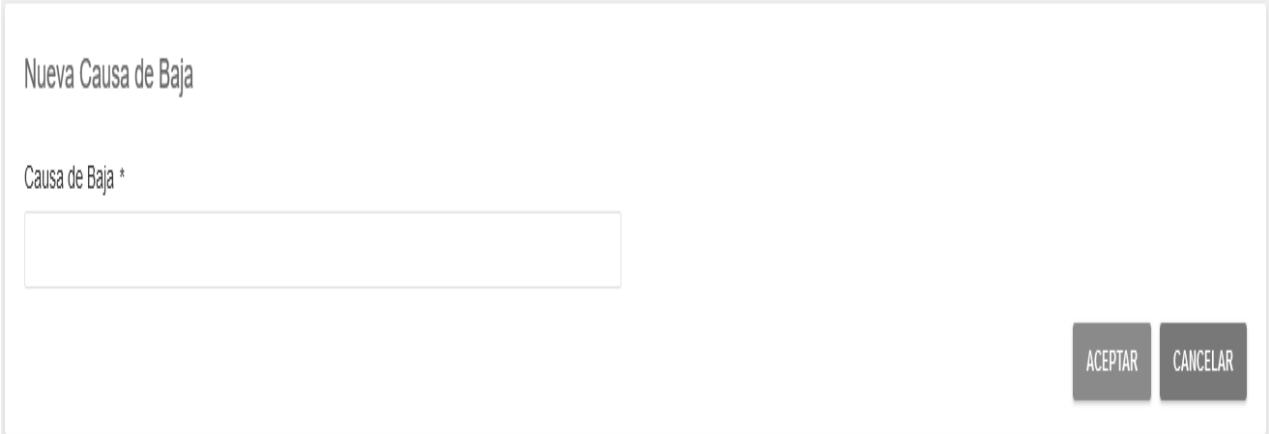
A continuación, en las Tablas 4, 5 y 6, se presenta la descripción de los RF del Módulo Asistente a la Gestión de los Recursos Humanos a partir del modelo propuesto por la HU.

**Tabla 4:** RF1 Insertar flujo de Baja

Historia de Usuario	
Número: 1	Requisito: 1
Programador: Yaidel Díaz Cáceres	Iteración asignada: 1
Prioridad: Alta	Tiempo estimado: 5 días
Riesgo en desarrollo: Medio	Tiempo real:
<b>Descripción:</b> Permitirá definir las diferentes áreas de la empresa que intervienen en el proceso de baja, así como la persona que puede autorizar la continuidad del proceso en las mismas. Del flujo de baja se registrarán los siguientes datos: <ul style="list-style-type: none"> <li>• Áreas que intervienen en el proceso de baja</li> <li>• Cargo autorizado a dar continuidad al proceso de baja</li> </ul>	
<b>Observaciones:</b>	
<b>Prototipo de interfaz:</b> 	

**Tabla 5:** RF3 Insertar causa de baja

Historia de Usuario	
Número: 2	Requisito: 3
Programador: Yaidel Díaz Cáceres	Iteración asignada: 1

<b>Prioridad:</b> Alta	<b>Tiempo estimado:</b> 6 días
<b>Riesgo en desarrollo:</b> Medio	<b>Tiempo real:</b>
<b>Descripción:</b> Permitirá insertar las diferentes causas que pueden dar inicio a los procesos de baja. De las causas se registrará: <ul style="list-style-type: none"> <li>• Descripción de la causa</li> </ul>	
<b>Observaciones:</b>	
<b>Prototipo de interfaz:</b> 	

**Tabla 6:** RF5 Crear baja

<b>Historia de Usuario</b>	
<b>Número:</b> 3	<b>Requisito:</b> 5
<b>Programador:</b> Yaidel Díaz Cáceres	<b>Iteración asignada:</b> 2
<b>Prioridad:</b> Alta	<b>Tiempo estimado:</b> 8 días
<b>Riesgo en desarrollo:</b> Medio	<b>Tiempo real:</b>
<b>Descripción:</b> Permitirá dar comienzo a un proceso de baja. De las bajas se registrarán los siguientes datos: <ul style="list-style-type: none"> <li>• Nombre de la persona que solicita la baja</li> <li>• Nombre de la persona autorizada que da inicio al proceso de baja</li> <li>• Fecha en que inicia el proceso de baja</li> <li>• Fecha de finalización del proceso de baja</li> <li>• Estado de la solicitud de baja</li> <li>• Causa de la solicitud de baja</li> </ul>	
<b>Observaciones:</b>	
<b>Prototipo de interfaz:</b>	

Nuevo Proceso de Baja

Persona que solicita la baja

Yaidel Diaz

Causa de la solicitud de baja

Problemas familiares

ACEPTAR
CANCELAR

### 2.4.1 PLANIFICACIÓN DE LAS HISTORIAS DE USUARIO

Para el desarrollo del módulo, se realizó una estimación de cada una de las historias de usuario identificadas (Ver Tabla 12), la cual arrojó los resultados que se muestran a continuación:

**Tabla 7:** Planificación de las historias de usuario

No	Nombre de la HU	Prioridad	Complejidad	Riesgo	Esfuerzo(días)	Puntos estimados	Iteración
1	Agregar área al flujo de baja	Alta	Alta	Medio	5	18	1
2	Eliminar área del flujo de baja	Alta	Baja	Medio	3	13	1
3	Insertar causa de baja	Alta	Media	Medio	6	16	1
4	Modificar causa de baja	Media	Media	Medio	4	9	1
5	Crear baja	Alta	Alta	Medio	8	18	2
6	Insertar estado de la solicitud de baja	Alta	Media	Medio	3	15	2
7	Modificar estado de la solicitud de baja	Media	Media	Medio	3	9	2
8	Cancelar baja	Alta	Baja	Medio	2	13	2

9	Reporte de bajas por área	Media	Media	Bajo	4	9	3
10	Reporte de bajas por causa	Media	Media	Bajo	4	9	3
11	Modificar cargo de un trabajador	Alta	Media	Medio	3	15	2
12	Modificar dirección de un trabajador	Media	Media	Bajo	3	9	2
13	Modificar teléfono de un trabajador	Media	Media	Bajo	3	9	2
14	Realizar movimiento de trabajador de un área	Alta	Media	Medio	4	15	1
15	Reporte de trabajadores de un área	Media	Media	Medio	4	9	3
16	Gestionar plantilla de cargos y ocupaciones de un área	Alta	Alta	Bajo	7	18	3
17	Gestionar plantilla de personal de un área	Alta	Media	Bajo	5	16	3

#### **2.4.2 PLAN DE ITERACIONES**

Después de haber sido identificadas y definidas las HU y realizado un estudio de la estimación propuesta para la construcción de las mismas, se procede a la planificación de la fase de implementación, en la cual se va a definir cuales historias de usuario se implementarán por cada iteración definida.

Las funcionalidades son planificadas en esta fase, generando al final de cada una, un entregable funcional que implementa las historias de usuario asignadas a la iteración. Como las historias de usuario no tienen suficiente detalle como para permitir su análisis y desarrollo, el cliente debe participar activamente durante el desarrollo de cada iteración; estas son utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida de avance.

En el caso del módulo actual se definieron tres iteraciones para proceder con las historias de usuarios precisadas. A continuación, se muestra como estará estructurada cada iteración.

## Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas.

Iteración 1: Se propone codificar las historias de usuario, que proveen las funcionalidades básicas del módulo:

HU1. Agregar área al flujo de baja

HU2. Eliminar área del flujo de baja

HU3. Insertar causa de baja

HU6. Insertar estado de la solicitud de baja

HU14. Realizar movimiento de trabajador de un área

Iteración 2: Se codifican las historias de usuario con prioridad alta y riesgo medio que requieren de mucha interacción con el usuario:

HU5. Crear baja

HU4. Modificar causa de baja

HU7. Modificar estado de la solicitud de baja

HU8. Cancelar baja

HU11. Modificar cargo de un trabajador

HU12. Modificar dirección de un trabajador

HU13. Modificar teléfono de un trabajador

Iteración 3: Se continúa la codificación de las HU que necesitan de la finalización de otras HU:

HU9. Reporte bajas por áreas

HU10. Reporte bajas por causa

HU15. Reporte de trabajadores de un área

HU16. Gestionar plantilla de cargos y ocupaciones de un área

HU17. Gestionar plantilla de personal de un área

## 2.5 ARQUITECTURA DE SOFTWARE

La arquitectura de software es una vista del sistema que incluye los componentes principales del mismo, el comportamiento de estos componentes según se les percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del mismo.

El patrón MVC (Modelo-Vista-Controlador) surge con el objetivo de reducir el esfuerzo de programación, necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos, a partir de estandarizar el diseño de las aplicaciones. El patrón MVC es un paradigma que divide las partes que conforman una aplicación en el Modelo, las Vistas y los Controladores, permitiendo la implementación por separado de cada elemento, garantizando así la actualización y mantenimiento del software de forma sencilla y en un reducido espacio de tiempo. A partir del uso de marcos de trabajo basados en el patrón MVC, como lo es Django, aunque en este se refiere a las vistas como *templates*(Plantillas) y al controlador como *views*(Vistas), se puede lograr una mejor organización del trabajo y mayor especialización de los desarrolladores y diseñadores.(Pillai 2017)

En Django la M, V y C se separan de la siguiente manera:

1. M, la porción de acceso a la base de datos, es manejada por la capa de la base de datos de Django, la cual se describe en este capítulo.
2. V, la porción que selecciona qué datos mostrar y cómo mostrarlos, es manejada por la vista y las plantillas.
3. C, la porción que delega a la vista dependiendo de la entrada del usuario, es manejada por el marco de trabajo siguiendo las *urls* y llamando a la función apropiada de Python para la URL obtenida.



Debido a que la "C" es manejada por el mismo marco de trabajo y la parte más importante se produce en los modelos, las plantillas y las vistas, Django es conocido como un marco de trabajo MTV (Modelo-Vista-Plantillas). En el patrón de diseño MTV:

1. M significa "*Model*" (Modelo), la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los ellos.
2. T significa "*Template*" (Plantilla), la capa de presentación. Esta capa contiene las decisiones relacionadas a la presentación: como algunas cosas son mostradas sobre una página web u otro tipo de documento.
3. V significa "*View*" (Vista), la capa de la lógica de negocios. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada: puedes pensar en esto como un puente entre el modelo y las plantillas.

En la interpretación de Django de MVC, la "vista" describe los datos que son presentados al usuario, no necesariamente el cómo se mostrarán, pero si cuáles datos son presentados. En contraste, los marcos de trabajo MVC y similares sugieren que el trabajo del controlador incluya la decisión de cuales datos son presentados al usuario, mientras que la vista sea estrictamente el cómo serán presentados y no cuáles. (Holovaty y Kaplan-Moss 2010)

En la propuesta de solución se aprecia la arquitectura de software MVT como se muestra en las figuras 1,2 y 3:

```
class estado(models.Model):
    estado=models.CharField(max_length=140, unique=True, validators=[validate_estado])

    def __str__(self):
        return self.estado
```

Figura 1: Modelo (Elaboración propia)



Figura 2: Plantillas (Elaboración propia)

```
class listar_estados(RequiredSecurityMixin, SuccessMessageMixin, ListView):
    need_login = True
    permission = RequiredSecurityMixin.CHANGE
    template_name = 'listar_estados.html'
    model = estado

    def get_context_data(self, **kwargs):
        user = self.request.user.id
        context = super(listar_estados, self).get_context_data(**kwargs)
        listar_estado = estado.objects.all()
        lista_personas = persona.objects.all()

        context['estados'] = listar_estado
        context['bajaspendientes'] = bajasPendientes(self)

        return context
```

Figura 3: Vistas (Elaboración propia)

## 2.5 PATRONES DE DISEÑO

Los patrones de diseño constituyen la base para la búsqueda de soluciones a problemas comunes en el proceso de desarrollo de software. Se caracterizan por su reusabilidad, flexibilidad y aplicabilidad a diferentes problemas de diseño en diversas circunstancias.(Gamma 2002)

El patrón es un esquema de solución que se aplica a un tipo de situación, adaptándose a los cambios específicos que pueda tener. Se considera que un patrón de diseño describe la estructura comúnmente recurrente de los componentes en comunicación, resuelve un problema general de diseño en un contexto particular y tiende a ser independientes de los lenguajes y paradigmas de programación.(Gamma 2002)

**Patrones de Asignación de Responsabilidades (GRASP) utilizados:**

**Experto:** es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Da origen a diseños donde el objeto de software realiza las operaciones que normalmente se aplican al elemento de la realidad que representa, por lo que ofrece una analogía con el mundo real. Con la utilización de este patrón se conserva el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clases sencillas y más cohesivas que son más fáciles de comprender y mantener. (Chazallet 2016). En la Figura 4 se puede apreciar el uso del patrón en la solución desarrollada.

```
85 class bajas(models.Model):
86     id_baja = models.BigIntegerField(primary_key=True, unique=True)
87     nombre=models.CharField(max_length=140)
88     apellidos= models.CharField(max_length=140)
89     estado = models.ForeignKey(estado, null=True, blank=True, on_delete=models.SET_NULL)
90     causa_id=models.ForeignKey(causa_baja, null=True, blank=True, on_delete=models.SET_NULL)
91     persona_autoriza_id=models.ForeignKey(persona, null=True, blank=True, on_delete=models.SET_NULL)
92     fecha = models.DateTimeField(default=timezone.now, editable=True)
93     area = models.ManyToManyField(area)
94
95     def __str__(self):
96         return self.id_baja
97
98     def __str__(self):
99         return self.nombre
100
101     def __str__(self):
102         return self.apellidos
```

Figura 4: Patrón experto (Elaboración propia)

**Creador:** este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Brinda un soporte bajo lo que supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización. (Chazallet 2016). En la Figura 5 se puede apreciar el uso del patrón en la solución desarrollada.

```
42     @login_required
43     class agregarFlujo(RequiredSecurityMixin, SuccessMessageMixin, CreateView):
44         need_login = True
45         permission = RequiredSecurityMixin.CREATE
46         model = flujo
47         template_name = 'crear.html'
48         form_class = FormularioInsertarFlujo
49         success_url = '/rrhh/listarflujo'
50         success_message = "Área insertada al flujo satisfactoriamente."
51
52         def form_invalid(self, form):
53             messages.Error(self.request, 'Por favor corrija los errores.')
54             return super(agregarFlujo, self).form_invalid(form)
55
```

*Figura 5: Patrón creador (Elaboración propia)*

**Alta cohesión:** este patrón es la meta principal que ha de tenerse en cuenta en todo momento. Una clase de alta cohesión posee un número relativamente pequeño de responsabilidades, con una importante funcionalidad relacionada y poco trabajo que hacer. Colabora con otros objetos para compartir el esfuerzo si la tarea es grande. Con el uso de este patrón mejoran la claridad y la facilidad con que se entiende el módulo. Se simplifica la complejidad del mantenimiento y las mejoras en funcionalidades. Puede generar un bajo acoplamiento. La ventaja de una alta cohesión es la capacidad de reutilización, porque una clase muy cohesiva puede destinarse a un propósito muy específico.(Gamma 1995)

**Bajo Acoplamiento:** es un principio que se debe tener siempre en cuenta durante las decisiones de diseño. Es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño. Este patrón estimula asignar una responsabilidad de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento. Soporta el diseño de clases más independientes, que reducen el impacto de los cambios, y también más reutilizables, que acrecientan la oportunidad de una mayor productividad. No puede considerarse en forma independiente de otros patrones como Experto o Alta cohesión, sino que más bien ha de incluirse como uno de los principios del diseño que influyen en la decisión de asignar responsabilidades. Con el uso de este patrón los componentes no se afectan por cambios de otros componentes, son fáciles de entender por separado y fáciles de reutilizar.(Gamma 1995)

**Patrones Gang of Four (GoF o Pandilla de Cuatro):**

**Decorador:** Permite añadir funcionalidad a una clase dinámicamente. (Gamma 1995). En la Figura 6 se puede apreciar el uso del patrón en la solución desarrollada.

```
@login_required
def listar_areas(request): ...

@login_required()
def listar_cargo(request): ...

@login_required
def listar_flujo(request): ...
```

*Figura 6: Patrón decorador (Elaboración propia)*

## 2.6 DIAGRAMA DE CLASES DEL DISEÑO

Un diagrama de clases del diseño es un modelo físico y concreto de las clases del sistema, es un plano de la implementación. Debe ser mantenido durante todo el ciclo de vida del software y da forma al mismo pues representa abstracciones de las clases directamente utilizables en la implementación del módulo como se presenta en la Figura 7.(Amo, Normand y Pérez 2005)



## 2.6 MODELO ENTIDAD – RELACIÓN

Un modelo de datos se puede definir como un conjunto de conceptos, reglas y convenciones bien definidos que permiten aplicar una serie de abstracciones a fin de describir y manipular los datos de un cierto mundo real que se desea almacenar en la base de datos. Por ello, se le considera fundamental para el desarrollo de cualquier aplicación que necesite almacenar datos. En la Figura 8 se presenta el diagrama correspondiente al modelo entidad-relación de base de datos utilizada para el manejo de información del módulo propuesto. (Siau 2004)

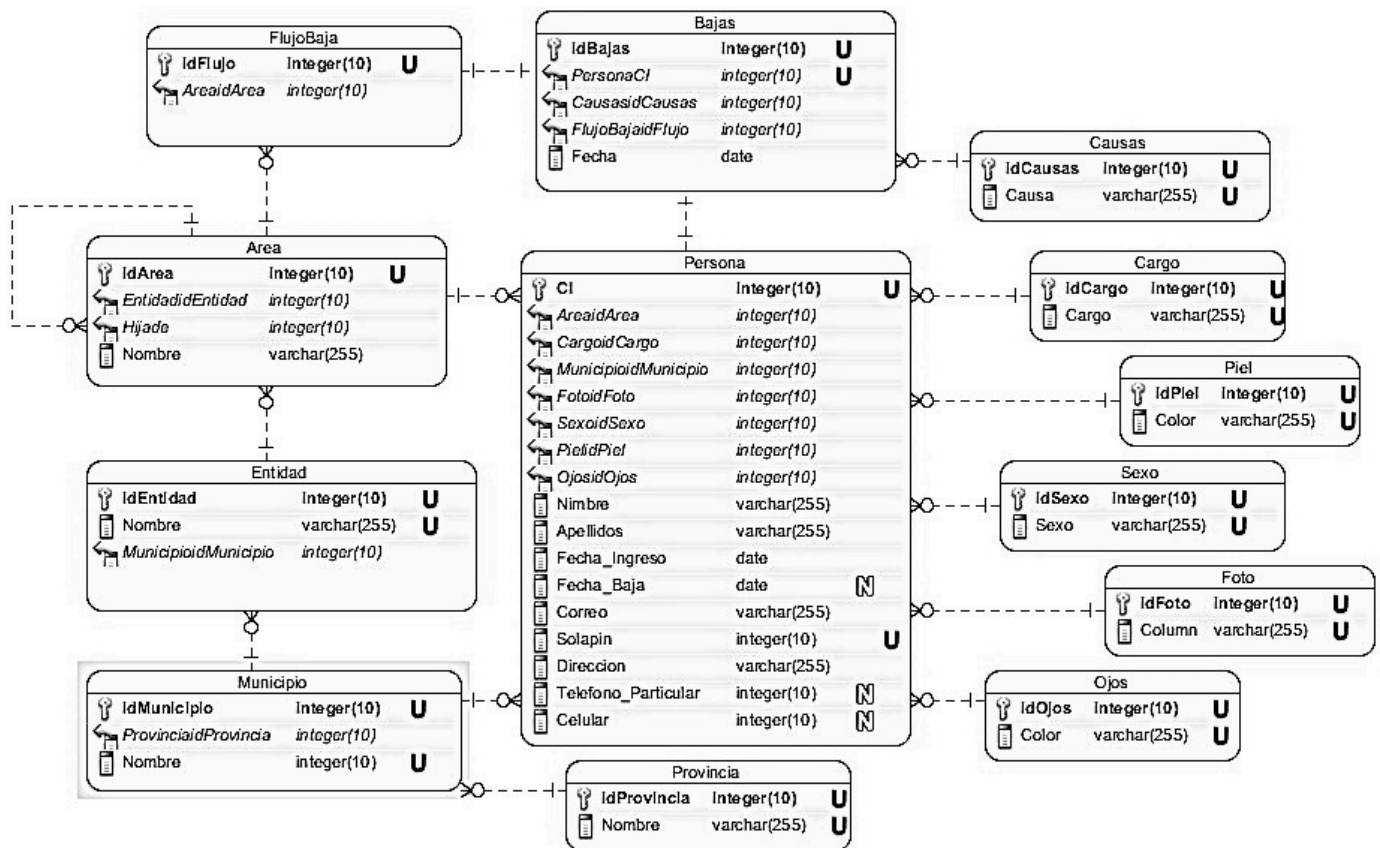


Figura 8: Modelo de datos (Elaboración propia)

## 2.8 DIAGRAMA DE COMPONENTES

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes del software, sean estos componentes de código fuente, binarios o ejecutables. Desde el punto de vista del



diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo.(Pressman 2005)

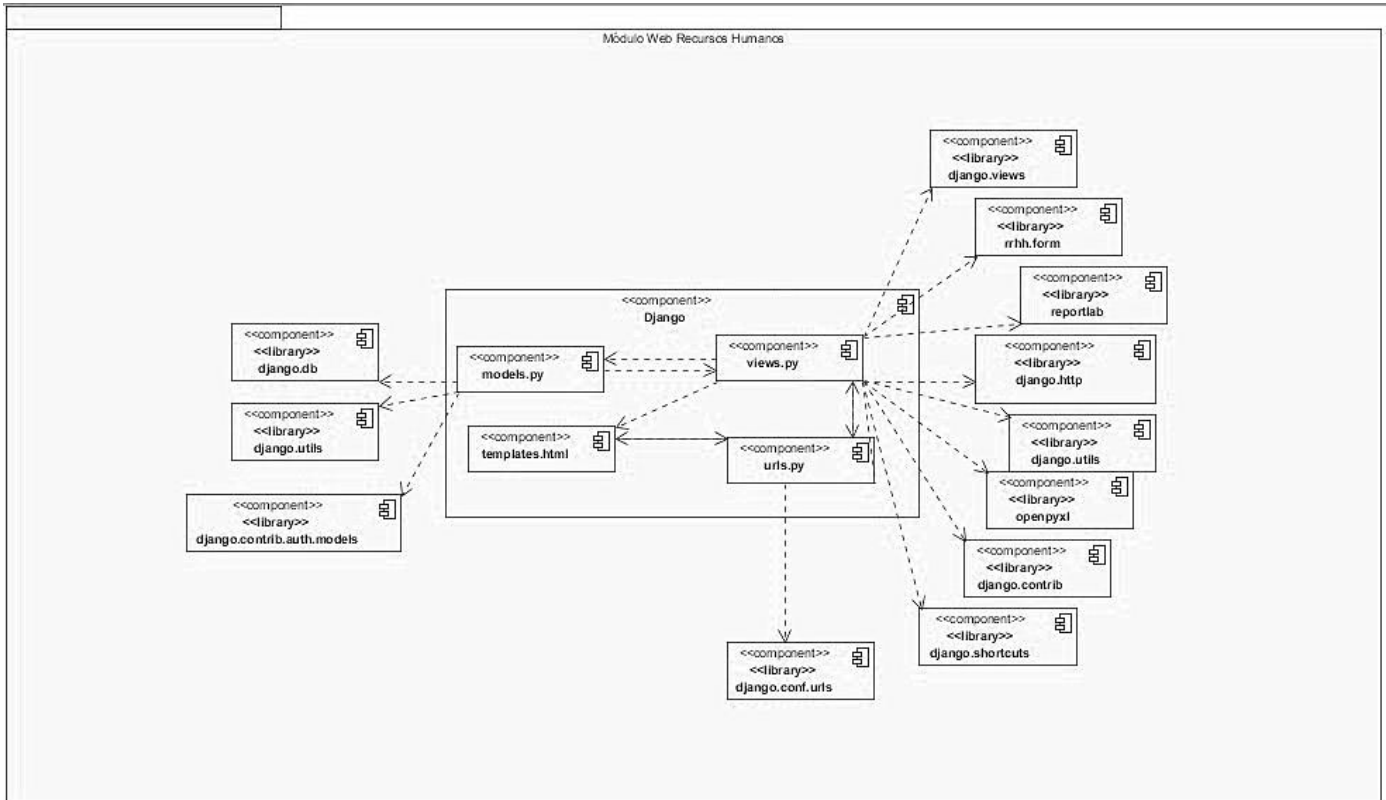


Figura 9: Diagrama de componentes (Elaboración propia)

Para el desarrollo de la propuesta de solución se utilizaron librerías estándar de Django entre las que se encuentran: django.db para la interacción con la base de datos, django.utils para las validaciones, django.contrib.auth.models para la interacción con el modelo de usuarios propuesto por el marco de trabajo, django.conf.urls para las direcciones url y otras librerías del marco de trabajo. Además, se utilizaron las librerías reportlab para la generación de reportes en formato PDF y openpyxl para los reportes en formato Excel.

## **2.9 CONCLUSIONES PARCIALES**

Los requisitos funcionales y no funcionales definidos, además de las HU son la base para el desarrollo de las funcionalidades y características con que debe contar el módulo. Los diagramas de clases del diseño del software a desarrollar permitieron la implementación de la propuesta de solución, además se desarrolló un plan de iteraciones para el desarrollo de las HU. El modelo de datos representa la estructura que tendrán los datos, sus atributos y la relación que existe entre ellos. El diagrama de componentes representa los componentes que forman el módulo y muestra las dependencias entre ellos.

## **CAPÍTULO 3: IMPLEMENTACIÓN Y VALIDACIÓN DE LA PROPUESTA DE SOLUCIÓN**

Un producto listo para ser entregado, requiere el completo desarrollo y validación de las funcionalidades previamente definidas. En el actual capítulo se describe la construcción de la propuesta teniendo en cuenta las técnicas de programación, estrategias de codificación y la integración con otros módulos. Se define y aplica la estrategia de prueba del producto, especificando las evaluaciones realizadas a la propuesta y los resultados.

### **3.1 ESTÁNDARES DE CODIFICACIÓN**

Los estándares de codificación son un conjunto de reglas a seguir por los desarrolladores con el objetivo de establecer un orden y un formato común en el código fuente del software en desarrollo. Son de vital importancia durante la etapa de construcción del software ya que permite que el personal del proyecto pueda entender de forma fácil el código, garantizándose la organización y estructura del código fuente.

Para el desarrollo de la solución se utilizan los estándares de codificación definidos en el documento PEP 8 Style Guide for Python Code.(Rossum, Warsaw y Coghlan 2001)

A continuación, quedan definidos:

#### **Indentación:**

1. Las líneas de continuación deben alinearse verticalmente con el carácter que se ha utilizado (paréntesis, llaves, corchetes).
2. Utilizar una indentación de una tabulación para cada línea con excepción de la primera.
3. La indentación se realizará solamente con tabulaciones, no deben utilizarse nunca los cuatro (4) espacios.

#### **Máxima longitud entre líneas:**

1. Todas las líneas deben estar limitadas a un máximo de setenta y nueve caracteres.

2. Dentro de paréntesis, corchetes o llaves se puede utilizar la continuación implícita para cortar las líneas largas.
3. En cualquier circunstancia se puede utilizar el carácter “\” para cortar las líneas largas.

**Líneas en blanco:**

4. Separar las funciones de alto nivel y definiciones de clases con dos líneas en blanco.
5. Las definiciones de métodos dentro de una clase deben separarse por una línea en blanco.
6. Se puede utilizar líneas en blanco escasamente para separar secciones lógicas.

**Codificaciones:**

Utilizar la codificación UTF-8.

Se pueden incluir cadenas que no correspondan a esta codificación utilizando “\x”, “\u” o “\U”.

**Importaciones:**

7. Las importaciones deben estar en líneas separadas.
8. Siempre deben colocarse al comienzo del archivo.
9. Deben quedar agrupadas de la siguiente forma:
  - a. Importaciones de la librería estándar.
  - b. Importaciones terceras relacionadas.
  - c. Importaciones locales de la aplicación/librerías.
  - d. Cada grupo de importaciones debe estar separado por una línea en blanco.
10. Evitar utilizar espacios en blanco en las siguientes situaciones:
  - a. Inmediatamente dentro de paréntesis, corchetes y llaves.
  - b. Inmediatamente antes de una coma, un punto y coma o dos puntos.
  - c. Antes del paréntesis que comienza la lista de argumentos en la llamada a una función.
  - d. Inmediatamente antes de un corchete que empieza una indexación.
  - e. Más de un espacio alrededor de un operador de asignación (u otro) para alinearlo con otro.

**Espacios en blancos en expresiones y sentencias:**

1. Deben rodearse con exactamente un espacio los siguientes operadores binarios:
  - a. Asignación (=).
  - b. Asignación de aumentación (+=, -=, etc.).
  - c. Comparación (==, <, >, >=, <=, !=, <>, in, not in, is, is not).
  - d. Expresiones lógicas (and, or, not).
2. Si se utilizan operadores con prioridad diferente se aconseja rodear con espacios a los operadores de menor prioridad.
3. No utilizar espacios alrededor del igual (=) cuando es utilizado para indicar un argumento de una función o un parámetro con un valor por defecto.

#### **Comentarios:**

1. Los comentarios deben ser oraciones completas.
2. Si un comentario es una frase u oración su primera palabra debe comenzar con mayúscula a menos que sea un identificador que comience con minúscula.
3. Nunca cambiar las minúsculas y mayúsculas en los identificadores de clases, objetos, funciones, etc.
4. Si un comentario es corto el punto final puede omitirse.

#### **Cadenas de documentación:**

1. Deben quedar documentados todos los módulos, funciones, clases y métodos públicos.
2. Para definir una cadena de documentación debe quedar encerrada dentro de ("").
3. Los ("" ) que finalizan una cadena de documentación deben quedar en una línea a no ser que la cadena sea de una sola línea.

#### **Convenciones de nombramiento:**

1. Nunca se deben utilizar como simple caracteres para nombres de variables los caracteres de minúscula "l", o mayúscula "O", o mayúscula "L" ya que en algunas fuentes son indistinguibles de los números uno y cero.
2. Los módulos deben tener un nombre corto y en minúscula.

3. Los nombres de clases deben utilizar la convención “CapWords” (palabras que comienzan con mayúsculas).
4. Los nombres de las excepciones deben estar escrito también en la convención “CapWords” utilizando el sufijo “Error”.
5. Los nombres de las funciones deben estar escrito en minúscula separando las palabras con un guion bajo “\_”.
6. Las constantes deben quedar escritas con letras mayúsculas separando las palabras por un guion bajo (\_).

### **3.2 DIAGRAMA DE DESPLIEGUE**

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo.(Ver Figura 10)(Jacobson, Booch y Rumbaugh 2000)

El entorno de despliegue estará compuesto por:

Cliente: se refiere a los sistemas de cómputo que se utilizarán para la interacción con el sistema.

Servidor de aplicación: es el encargado de atender todas las peticiones de los clientes y controlar su interacción con los datos.

Servidor de base de datos: es el entorno donde se almacenan los datos.

Webcam: dispositivo para la captura de imágenes.

Impresora: dispositivo para la impresión de los reportes.

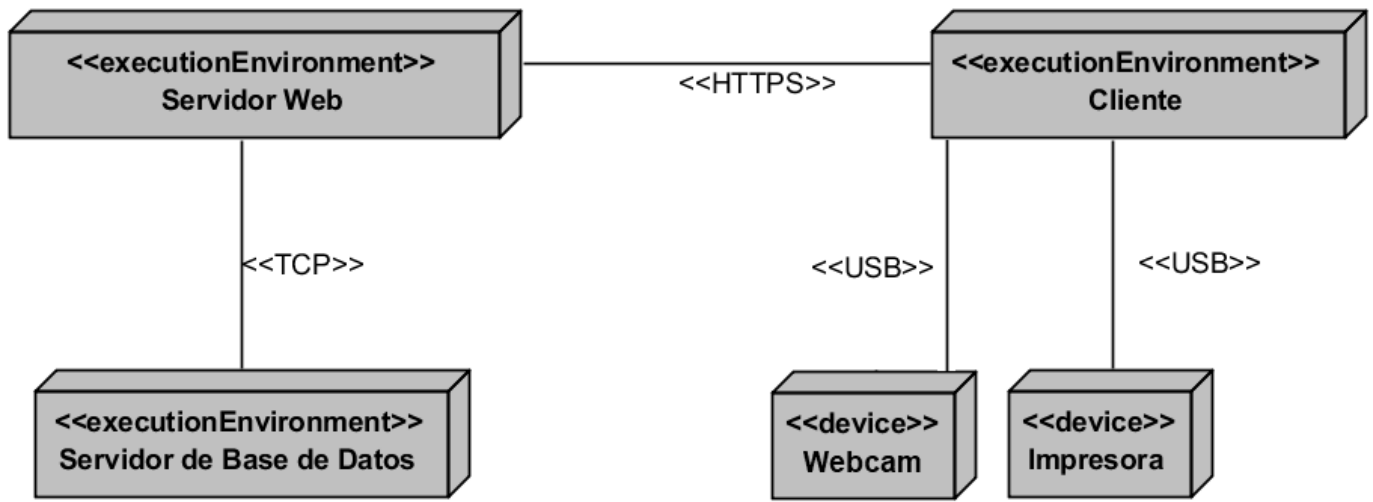


Figura 10: Diagrama de despliegue (Elaboración propia)

### 3.3 VALIDACIÓN DE REQUISITOS

La validación de requisitos examina las descripciones para asegurar que todos los requisitos de la propuesta de solución han sido establecidos sin ambigüedad, sin inconsistencias, sin omisiones, que los errores detectados hayan sido corregidos y que el resultado del trabajo se ajusta a los estándares establecidos para el proceso, el proyecto y el producto. (Pressman y Bruce R. Maxim 2014)

#### Crterios para validar los requisitos:

Para validar los requisitos según (Pressman y Murrieta 2006), se pueden chequear mediante un cuestionario guiado por un conjunto de interrogantes con el objetivo de descubrir la mayor cantidad de errores posibles. A continuación, se muestra las interrogantes utilizadas por ser consideradas necesarias para la validación de los requisitos del módulo:

1. ¿Está el requisito claramente definido? ¿Puede interpretarse mal?
2. ¿Está identificado el origen del requisito (por ejemplo: persona, norma, documento)?
3. ¿El planteamiento final del requisito ha sido contrastado con la fuente original?
4. ¿El requisito está delimitado en términos cuantitativos?
5. ¿Qué otros requisitos hacen referencia al requisito estudiado?
6. ¿El requisito incumple alguna restricción definida?

7. ¿El requisito es verificable? Si es así, ¿se podrá efectuar pruebas para verificar el requisito?
8. ¿Se puede seguir el requisito en el modelo de la solución que se ha desarrollado?
9. ¿Está el requisito asociado con los rendimientos del módulo o con su comportamiento?
10. ¿El requisito está implícitamente definido?
11. ¿El requisito es modificable?
12. ¿El requisito está completo?
13. ¿El requisito puede ser implementado?
14. ¿El requisito puede ser probado?

#### **Resultado de aplicar los criterios de validación:**

Luego de aplicar el conjunto de interrogantes para validar los requisitos definidos para el desarrollo del módulo, se obtuvo el 100 % de aprobación por parte del cliente.

#### **3.3.1 TÉCNICAS DE VALIDACIÓN DE REQUISITOS**

Con el objetivo de obtener una mayor calidad y demostrar que los requisitos definidos realmente describen al módulo que el cliente necesita; se utilizaron las siguientes técnicas para la validación de requisitos:

1. **Revisión técnica formal de requisitos:** se realizan reuniones donde se examinan las descripciones del módulo buscando errores en el contenido o en la interpretación, ideas donde se necesitan aclaraciones, información incompleta, inconsistencias, requisitos contradictorios o requisitos imposibles o inalcanzables.
2. **Construcción de prototipos:** la confección de los prototipos permite que el usuario pueda conocer cómo es la propuesta que el equipo de desarrollo implementa y puede aprobar la idea o corregir los elementos ajenos a los requisitos funcionales descritos.
3. **Generación de casos de prueba:** la elaboración de casos de prueba permite comprobar el cumplimiento de los requisitos funcionales y que estos presentan la calidad requerida.

#### **Resultado de aplicar las técnicas de validación**



Al concluir el proceso de revisión de requisitos se detectaron algunas inconsistencias que fueron erradicadas de inmediato. Los errores detectados son:

1. Se interpretaron de forma incorrecta algunas de las funcionalidades y características solicitadas por el cliente.
2. Errores ortográficos o tipográficos.

### **3.4 PRUEBAS**

Una vez finalizada la implementación de la solución, se hace necesario comprobar que su funcionamiento sea correcto, verificando que las funcionalidades se ajusten a las especificaciones planteadas. Para ello se realizan las pruebas de software, con el fin de detectar defectos y asegurar que estos sean corregidos antes de la entrega del producto al cliente.

Las pruebas son actividades en las cuales un sistema o componente es ejecutable bajo condiciones o requerimientos específicos permitiendo que los resultados sean observados y registrados, estas se realizan con el objetivo de encontrar deficiencias existentes en el software.(Jacobson, Booch y Rumbaugh 2000). Durante el flujo de trabajo de pruebas se verifica el resultado final de la implementación, probando la estructura; tanto en la construcción interna como intermedia, así como las versiones finales.

#### **Las pruebas de validación de software se clasifican en:**

Funcionales o de Caja Negra: En este tipo de pruebas se prueban las funcionalidades del programa contra las especificaciones, para lo que se observa el sistema como una caja negra, y completamente ajeno a su estructura interna. Son un conjunto de entradas, condiciones de ejecución y salidas esperadas para un objetivo en particular, con la idea de ejercitar la ruta de un programa específico o para verificar el cumplimiento de un requisito específico. Una ventaja de las pruebas funcionales es que están dirigidas a lo que el programa se supone que debe hacer, lo que es natural y comprensible para todos. Tiene la limitante de que no es factible probar todos los posibles valores de entrada, además, dado que no se conoce la estructura interna o la lógica podrían existir anomalías insertadas deliberadamente por el desarrollador, algo que posiblemente la prueba no detectará.

Estructurales o de Caja Blanca: Las condiciones de prueba se diseñan para examinar los caminos de la lógica con lo que es posible analizar la estructura interna del sistema. Una ventaja de este tipo de pruebas es que son exhaustivas y que se centran en el código, por lo que existe mayor posibilidad de detectar los errores deliberados. Su desventaja es que no comprueba la corrección de las especificaciones ya que se centra solo en la lógica interna y no verifica la consistencia entre la lógica y la especificación. Otro inconveniente es que no existe forma de detectar los caminos perdidos que, sin estar en el código, pueden recorrer los datos; tampoco es posible recorrer todas las posibles rutas lógicas a través de un programa ya que esto implica un gran número de casos de prueba.

Funcionales/Estructurales o de Caja Gris: La comunicación entre el probador y el desarrollador para el estudio de la especificación de requisitos, es necesaria para comprender la estructura interna del sistema y para clarificar las especificaciones ambiguas. Estas pruebas son una combinación de las dos anteriores.(Sommerville y Galipienso 2005; Pressman y Murrieta 2006)

### **3.4.1 PRUEBAS ESTRUCTURALES**

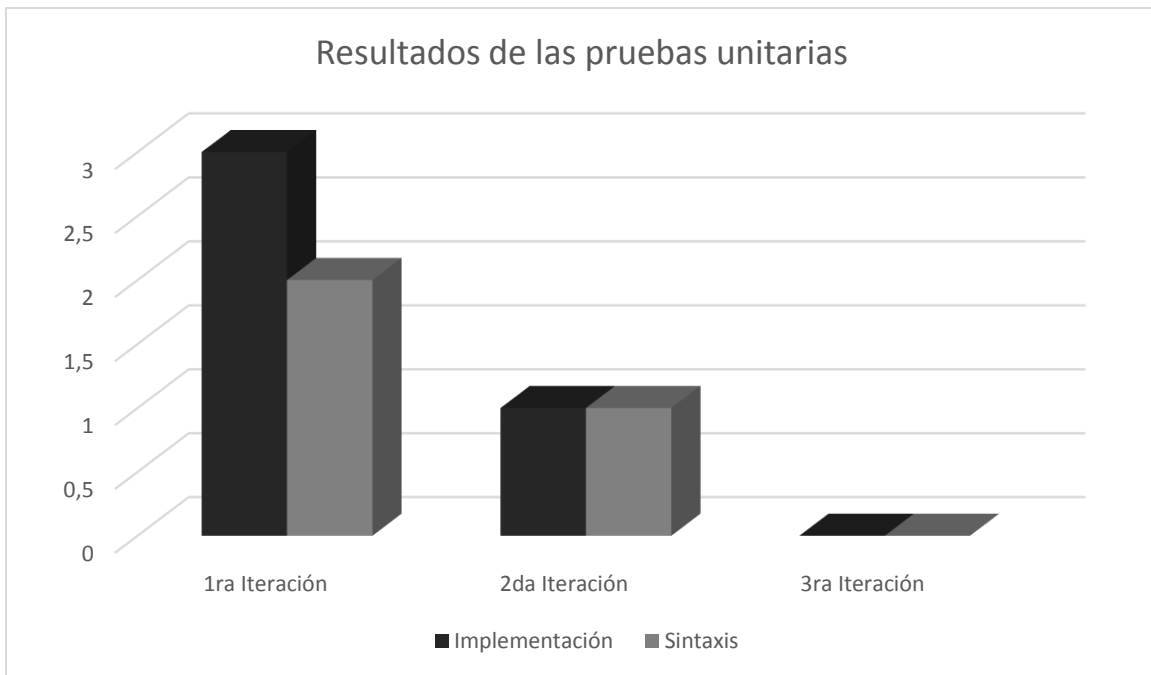
Estas pruebas, también suelen ser llamadas estructurales o de cobertura lógica. En ellas se pretende investigar sobre la estructura interna del código, exceptuando detalles referidos a datos de entrada o salida, para probar la lógica del programa desde el punto de vista algorítmico. Realizan un seguimiento del código fuente según se va ejecutando los casos de prueba, determinándose de manera concreta las instrucciones y/o bloques; que han sido ejecutados.(Salazar Martínez 2015)

Además, con su realización, se pretende indagar sobre la estructura interna del código, omitiendo detalles referidos a datos de entrada o salida. Su objetivo principal es probar la lógica del programa desde el punto de vista algorítmico.(Salazar Martínez 2015)

#### **Pruebas de unidad:**

Este tipo de prueba se centra en ejecutar cada módulo o unidad mínima a ser probada; lo cual provee un mejor modo de manejar la integración de las unidades en componentes mayores. Busca asegurar que el código funciona de acuerdo con las especificaciones y que el módulo lógico es válido.(Chazallet 2016)

Para la presente investigación se utiliza este tipo de prueba mediante el uso de la librería unittest. Se realizaron tres iteraciones en las cuales se detectaron un total de 5 no conformidades de sintaxis e implementación. En la primera iteración se encontraron 3 no conformidades, las cuales fueron resueltas de forma satisfactoria. En una segunda iteración 2, a las cuales se les dio solución. Posteriormente en una tercera iteración no fueron encontrados nuevos errores.



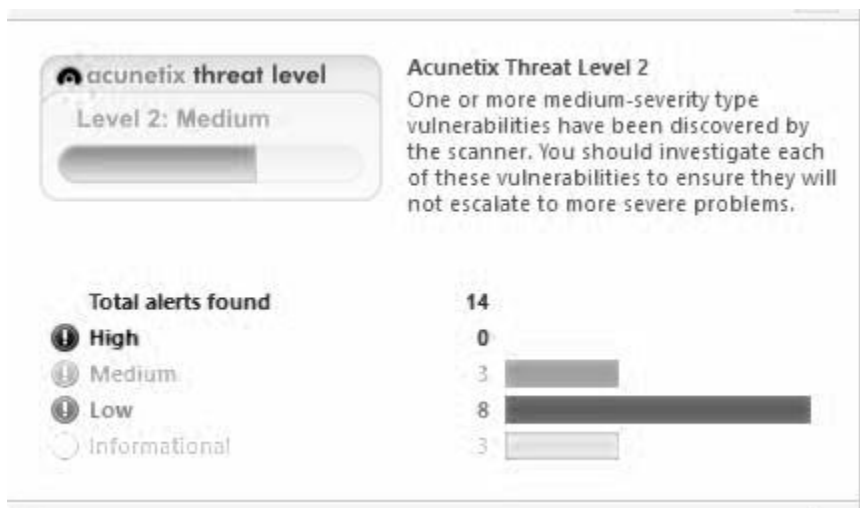
**Figura 11:** Pruebas unitarias (Elaboración propia)

**Pruebas de seguridad:**

En el desarrollo de software la seguridad es un requisito indispensable para garantizar la calidad del producto que se desarrolla. Las pruebas de seguridad se realizan con el objetivo de encontrar vulnerabilidades en los sistemas de cómputo para posteriormente mitigarlas en la medida de lo posible. Si bien es casi imposible garantizar este aspecto de calidad al 100%, si se pueden realizar una serie de pruebas que permitan conocer donde se encuentran las debilidades y corregirlas (Sommerville y Galipienso 2005). Para probar la realidad del Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas se utilizó la herramienta Acunetix obteniendo los siguientes resultados.

**1era iteración:**

Fueron detectadas un total de 14 vulnerabilidades (Ver Figura 11) asociadas básicamente a las credenciales, los directorios y vulnerabilidades a los ataques de fuerza bruta.



**Figura 12:** Pruebas de seguridad (Iteración 1) (Elaboración propia)

**2da Iteración:**

Fueron detectadas un total de 2 vulnerabilidades (Ver Figura 12) asociadas básicamente a las credenciales en texto plano y los directorios no seguros.

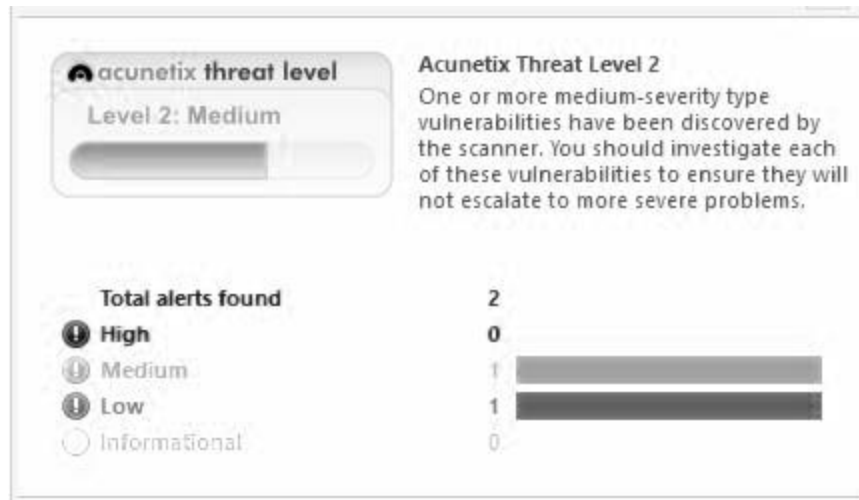


Figura 13: Pruebas de seguridad (Iteración 2) (Elaboración propia)

Todas las vulnerabilidades detectadas en las iteraciones anteriores fueron corregidas de manera satisfactoria.

### 3ra Iteración:

En esta iteración no fueron detectadas amenazas de seguridad. (Ver Figura 13)

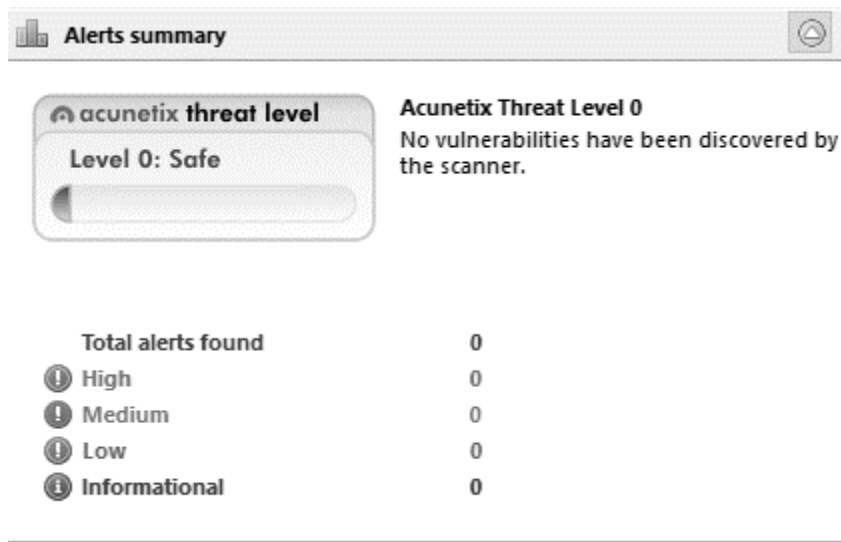


Figura 14: Pruebas de seguridad (Iteración 3) (Elaboración propia)

### **3.4.2 PRUEBAS FUNCIONALES**

En las pruebas funcionales, las condiciones de prueba se desarrollan de acuerdo con las bases del programa o la funcionalidad del sistema; es decir, el probador requiere información sobre los datos de entrada y las salidas esperadas, pero no sabe cómo funciona el sistema. No es necesario conocer la estructura interna de un programa para ejecutarlo; el probador se centra en probar la funcionalidad del programa contra la especificación. Estas pruebas son una importante estrategia para probar el software. Su objetivo es ser completamente indiferente acerca del comportamiento y la estructura interna del programa y, a cambio, se concentran en la búsqueda de las circunstancias en las que no se comporta de acuerdo con sus especificaciones.(Montoya 2013)

El proceso para realizar este tipo de pruebas es:

1. Analizar los requisitos y especificaciones.
2. Seleccionar entradas válidas con base en la especificación, para determinar que los procesos del software bajo prueba son correctos. También se deben elegir entradas no válidas para comprobar que esos mismos procesos detectan y se ocupan correctamente de ellas.
3. Determinar la salida esperada para cada entrada.
4. Diseñar las pruebas con las entradas seleccionadas.
5. Ejecutar las pruebas.
6. Comparar las salidas halladas con las salidas esperadas.
7. Tomar determinación acerca del funcionamiento apropiado del software.

#### **Pruebas de carga y estrés:**

Una prueba de carga se ejecuta para comprender el comportamiento de una aplicación ante una carga determinada, puede ser el número de usuarios esperado ejecutando una cifra de transacciones durante un tiempo determinado. El resultado dará el tiempo de respuesta de todas las transacciones críticas. Si la base de datos, servidor de aplicación también se monitorizan, entonces esta prueba puede mostrar potenciales problemas en la aplicación.(Montoya 2013)

Las pruebas de estrés son utilizadas normalmente para someter a la aplicación al límite de su funcionamiento, mediante la ejecución de un número de usuarios muy superior al esperado. Esta prueba tiene como finalidad determinar la robustez de una aplicación cuando la carga es extrema y ayuda a administradores a determinar si la aplicación se comportará correctamente en dichas situaciones.

La prueba de estrés determina el punto en que un sistema comienza a brindar un desempeño inaceptable. Este punto puede ser llamado punto de quiebre. La idea es estresar a un sistema al punto de quiebre para encontrar errores que podrían ser potencialmente perjudiciales. Esta es la única forma de poder detectar este tipo de errores. No se espera que el sistema procese la sobrecarga sin los recursos adecuados, pero sí que se comporte de una manera razonable. (Montoya 2013)

Resultado de las pruebas de carga:

**Hardware de prueba:**

1. Tipo de procesador: Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz 2.60GHz.
2. RAM: 8 GB DDR3.

**Software instalado:**

1. Tipo de servidor *web*: Apache2.4.
2. Plataforma: Sistema Operativo x 64 bits Windows 10.
3. Servidor de BD: PostgreSQL9.4
4. Servidor de prueba: ApacheJmeter3.2

Una vez configurado el entorno de prueba se realizaron 2 iteraciones con 100 peticiones cada una para comprobar la respuesta del módulo ante una carga aceptable de usuarios. Los resultados de la prueba fueron aceptados, pues no se presentaron errores y el tiempo máximo de respuesta fue de 3,3 segundos. En la Figura 14 se muestran los resultados de dichas iteraciones.

## Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas.

Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Mín	Máx	% Error	Rendimiento	Kb/sec	Sent KB/sec
/rrhh/listarbajas/	200	23853	31142	45160	47308	48610	224	49192	0,00%	1,3/sec	1,70	0,13
/rrhh/listarcau...	200	13214	2505	33734	42572	47471	211	48182	0,00%	1,6/sec	1,89	0,12
/rrhh/listaresta...	200	5539	1297	19292	30090	36223	210	38977	0,00%	1,6/sec	1,79	0,12
/rrhh/listarflujo/	200	6372	1301	22177	25923	32657	210	32900	0,00%	1,5/sec	2,14	0,12
/rrhh/listarp2/	200	6801	1288	21093	30151	35058	210	35571	0,00%	1,5/sec	2,24	0,12
/rrhh/listarp4/	200	4199	872	9262	13080	28372	210	30594	0,00%	1,5/sec	2,23	0,12
/rrhh/insertard...	200	4033	1157	9942	12376	29214	210	31200	0,00%	1,5/sec	2,19	0,11
/rrhh/insertarc...	200	4543	1298	9443	16023	28429	209	35810	0,00%	1,5/sec	2,37	0,11
/rrhh/insertare...	200	4440	1266	8901	11701	31137	208	35183	0,00%	3,0/sec	2,32	0,11
/rrhh/addflujo/	200	4547	1357	8515	12196	30104	211	33071	0,00%	3,0/sec	2,42	0,11
/rrhh/modificar...	200	4145	1340	8275	10184	23723	208	34125	0,00%	2,8/sec	2,44	0,11
/rrhh/modificar...	200	4351	3906	8341	10370	24538	209	30761	0,00%	2,8/sec	2,56	0,11
/rrhh/modificar...	200	4621	4942	7721	11444	29279	208	31883	0,00%	2,6/sec	2,60	0,11
/rrhh/modificar...	200	4404	4842	7867	9209	23141	210	30243	0,00%	2,6/sec	2,67	0,11
/rrhh/modificar...	200	4385	4768	7919	10235	28039	211	30286	0,00%	2,6/sec	2,72	0,11
/rrhh/eliminar...	200	4261	4810	7561	8273	17722	212	30384	0,00%	2,9/sec	2,75	0,11
/rrhh/eliminar...	200	4168	4641	7571	8043	12001	210	33988	0,00%	3,3/sec	2,79	0,11
/rrhh/eliminar...	200	4461	4475	7642	8531	23870	210	37319	0,00%	2,5/sec	2,83	0,11
/rrhh/eliminarfl...	200	4303	4468	7691	8528	21267	208	30062	0,00%	2,2/sec	2,94	0,11
/rrhh/autorizar/2	200	4478	4716	7399	9581	21390	211	30773	0,00%	1,8/sec	3,00	0,11
/rrhh/reporte...	200	4212	4213	7653	8644	12516	210	29819	0,00%	1,6/sec	3,04	0,11
/rrhh/reporte...	200	4379	4066	7511	8757	12758	212	43828	0,00%	1,5/sec	3,08	0,11
/rrhh/reporte...	200	3965	3627	7010	7866	16256	210	32177	0,00%	1,4/sec	3,10	0,11
/rrhh/pdf/2	200	3796	3131	6936	7730	15955	211	26296	0,00%	1,3/sec	3,17	0,11
/rrhh/assignarc...	200	4110	2899	7718	9768	26792	209	38471	0,00%	1,3/sec	3,21	0,11
Total	5000	5663	3628	11018	26376	37530	208	49192	0,00%	6,4/sec	59,50	2,56

Figura 15: Pruebas de carga (Elaboración propia)

### Diseño de casos de pruebas funcionales:

Tabla 8: Variables para los casos de prueba funcional Gestionar baja

No.	Nombre del campo	Alias para el caso de prueba	Clasificación	Valor nulo	Descripción
1	Nombre de la persona que solicita la baja	NP	Campo seleccionable	No	Se selecciona el nombre de la persona que ha solicitado la baja
2	Causa de la solicitud de baja	CB	Campo seleccionable	No	Se selecciona la causa por la cual la persona ha solicitado la baja
3	Persona que autoriza el proceso de baja	PA	Campo automático	No	El valor por defecto es la persona que realiza el proceso
4	Fecha de inicio del proceso de baja	FI	Campo automático	No	El valor por defecto es la fecha en que se realiza el proceso
5	Estado de la solicitud de baja	EB	Campo automático	No	El valor de este campo es el estado en que se encuentra el proceso de baja



**Tabla 9: Caso de prueba funcional Insertar Baja**

<b>Descripción general</b>								
Permitirá iniciar un nuevo proceso de baja								
<b>Condiciones de ejecución</b>								
Para iniciar un nuevo proceso de baja el usuario debe estar logueado en el sistema y poseer los permisos necesarios, además, debe ser insertada previamente una persona en el sistema en la misma área del usuario, así como una o más causas de baja.								
<b>Escenario</b>	<b>Descripción</b>	<b>NP</b>	<b>CB</b>	<b>PA</b>	<b>FI</b>	<b>EB</b>	<b>R/ del módulo</b>	<b>Flujo central</b>
ESC 1.1 Introducir datos de Proceso de baja correctamente.	El usuario debe seleccionar los dos campos del formulario.	Leonardo Lázaro Oduardo Pulido	Retiro	Usuario logueado en el sistema(Campo automático).	1 de Abril de 2018 (Campo automático).	Inicio (Campo automático).	El módulo debe crear una baja satisfactoriamente y muestra notificación de éxito.	En el menú a la izquierda se sigue la siguiente ruta: "Recursos Humanos - Proceso de Bajas- Iniciar Proceso de Baja". Aparecerá un formulario con dos campos seleccionables en los cuales se encontrarán todas las personas pertenecientes a la misma área del usuario logueado en el sistema y todas las causas de baja registradas respectivamente.

								ente. Una vez insertada la causa debe mostrarse una lista con todas las bajas incluyendo la insertada.
--	--	--	--	--	--	--	--	--------------------------------------------------------------------------------------------------------

**Tabla 10:** Caso de prueba funcional Modificar Baja

<b>Descripción general</b>								
Permitirá modificar un proceso de baja								
<b>Condiciones de ejecución</b>								
Para modificar un proceso de baja el usuario debe estar logueado en el sistema y poseer los permisos necesarios, además, debe ser insertada previamente una baja en el sistema por el usuario.								
<b>Escenario</b>	<b>Descripción</b>	<b>NP</b>	<b>CB</b>	<b>PA</b>	<b>FI</b>	<b>EB</b>	<b>R/ del módulo</b>	<b>Flujo central</b>
ESC 1.2 Modificar proceso de baja	El usuario debe seleccionar el botón editar	No	Enfermedad	No	No	No	El módulo debe mostrar un formulario para que el usuario seleccione una de las causas de baja que hay registrada en el mismo. Una vez modificada la baja el sistema debe mostrar un mensaje de éxito así como el listado de las bajas incluyendo	En el menú a la izquierda se sigue la siguiente ruta: "Recursos Humanos - Proceso de Bajas- Listar Proceso de Baja". Aparecerá un listado con todas las bajas creadas por el usuario y una columna acciones en la parte

							la modificación.	derecha de la lista, las acciones a ejecutar. El usuario selecciona la opción modificar.
--	--	--	--	--	--	--	------------------	------------------------------------------------------------------------------------------

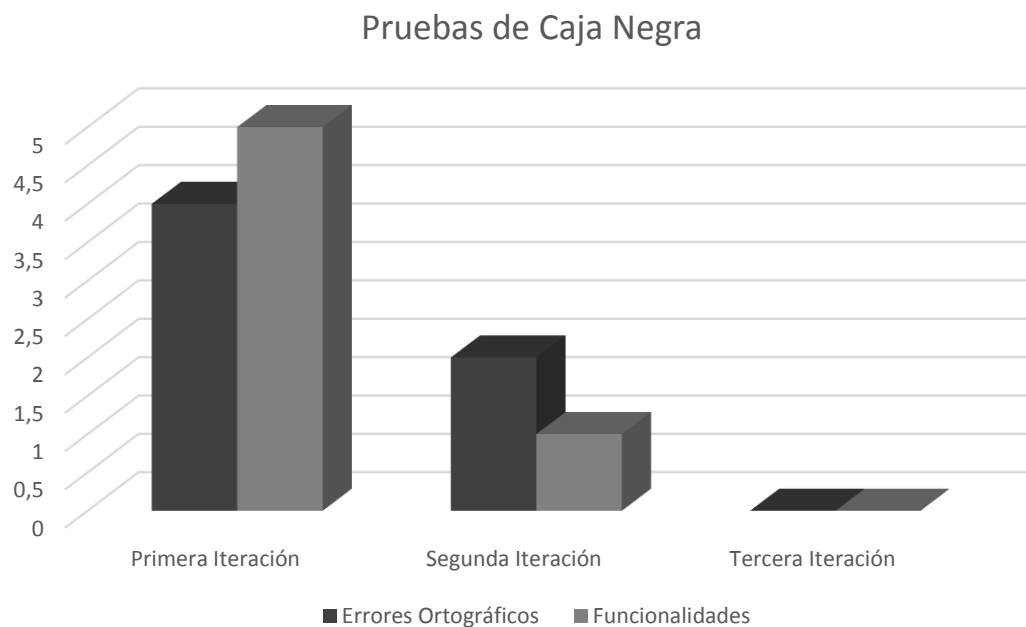
**Tabla 11:** Caso de prueba funcional Eliminar Baja

<b>Descripción general</b>								
Permitirá modificar un proceso de baja								
<b>Condiciones de ejecución</b>								
Para eliminar un proceso de baja el usuario debe estar logueado en el sistema y poseer los permisos necesarios, además, debe ser insertada previamente una baja en el sistema por el usuario.								
Escenario	Descripción	NP	CB	PA	FI	EB	R/ del módulo	Flujo central
ESC 1.3 Cancelar proceso de baja	El usuario debe seleccionar el botón de eliminar.	No	No	No	No	No	El módulo debe mostrar una confirmación antes de ejecutar la acción y una vez confirmada mostrar un mensaje de éxito y listar las bajas.	En el menú a la izquierda se sigue la siguiente ruta: "Recursos Humanos - Proceso de Bajas-Listar Proceso de Baja". Aparecerá un listado con todas las bajas creadas por el usuario y una columna acciones en la parte derecha de la lista, las acciones a ejecutar. El usuario

Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas.

								selecciona la opción eliminar y confirma la eliminación.
--	--	--	--	--	--	--	--	----------------------------------------------------------

El Módulo Asistente a la Gestión de los RRHH para el Sistema de Información Primaria de Personas fue sometido a tres iteraciones de pruebas. En una primera iteración, fueron detectadas 9 no conformidades: 4 errores ortográficos y 5 funcionalidades incorrectas. Luego de la segunda iteración se detectaron un total de 3, de ellas: 2 errores ortográficos y 1 funcionalidad incorrecta, que fueron corregidas a medida que fue avanzando el proceso de pruebas. Posteriormente una tercera iteración con el objetivo de verificar que las no conformidades antes detectadas estuvieran totalmente resueltas y no generaran otras, obteniéndose resultados satisfactorios.



**Figura 16:** Pruebas de caja negra (Elaboración propia)

### 3.5 PRUEBAS DE INTEGRACIÓN

Durante la prueba de integración, se probó de manera combinada el módulo de RRHH con los módulos que componen el Sistema de Información Primaria de Persona. Dichos módulos fueron acoplados progresivamente en conjuntos. Una vez verificado que el conjunto funciona de acuerdo con lo previsto, fue sumado un nuevo módulo. Luego del acoplamiento de cada uno, se efectuaron pruebas para comprobar la correcta interacción entre los mismos. Este proceso fue realizado para todos los componentes del software. Una vez relacionados se realizaron nuevamente pruebas para verificar los requerimientos funcionales, de rendimiento y de seguridad definidos en las primeras etapas del ciclo de vida del software para comprobar que en el proceso de integración no se introdujeran errores en los elementos integrados.

El Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas fue integrado satisfactoriamente. Durante la integración no fueron detectados nuevos errores y la comunicación entre los módulos fue exitosa.

### 3.6 VALIDACIÓN DE LA HIPÓTESIS

La validación de la hipótesis planteada al inicio de la investigación se realizó utilizando el criterio de expertos, aplicando su variante Delphi. La aplicación de la misma permitió la selección de un conjunto de expertos para realizar la validación de la solución propuesta. Dicha selección estuvo basada en una serie de competencias definidas por el investigador, que a su consideración le otorgaban a los entrevistados la categoría de expertos.

Entre las características a tener en consideración se pueden citar: categoría docente, relación con el desarrollo de software, conocimiento de las funcionalidades del módulo desarrollado y conocimientos de RRHH.

Para la validación de la hipótesis se desarrolló un cuestionario (Ver Anexo 2), que recoge las métricas a evaluar y fue expuesto a la consideración de los expertos para la evaluación del Módulo Asistente a la Gestión de los RRHH para el Sistema de Información Primaria de Personas.

Se presentaron un total de veinticinco indicadores, los cuales fueron medidos en una escala del 2 al 5, donde 2 es la puntuación de deficiente y 5 la puntuación de excelencia. Con las respuestas obtenidas fue confeccionada una matriz donde se registraron los criterios de los expertos por cada uno de los indicadores. (Ver Anexo 3).

Una vez definida la matriz se calcula el coeficiente de concordancia mediante la siguiente fórmula.

**Tabla 12: Fórmulas**

<b>Coeficiente de concordancia (C)</b>	$C = 100 * \left(1 - \frac{D_s}{X_m}\right)$
<b>Donde:</b>	$D_s$
<b>Desviación estándar (<math>D_s</math>)</b>	$= \sqrt{\frac{1}{N-1} \sum (X_i - X_m)^2}$
<b>Media del criterio de los expertos por indicador (<math>X_m</math>)</b>	$X_m = \frac{\sum X_i}{N}$

Como resultado de la aplicación del método Delphi se obtuvo la aceptación para los 25 indicadores presentados a los expertos, con una puntuación de excelente. (Ver Anexo 3)

### **3.7 CONCLUSIONES PARCIALES**

La elaboración del modelo de despliegue permitió representar la distribución física de la solución y los elementos de hardware por los que estará compuesto. La aplicación de la estrategia de prueba permitió identificar y resolver los errores en la implementación, garantizando el correcto funcionamiento del producto final. La validación de la hipótesis planteada al inicio de la investigación, utilizando el criterio de expertos en su variante Delphi, demuestra la validez de la misma.

## CONCLUSIONES

A partir del contenido de la investigación presentada, de los antecedentes revisados en la literatura y su análisis, se arriba a las siguientes conclusiones:

1. La elaboración del marco teórico conceptual referente al proceso de gestión de los recursos humanos y el empleo de las tecnologías permitió analizar seis soluciones existentes, que, aunque no satisfacen el problema de la investigación, aportaron algunos elementos tenidos en cuenta para el desarrollo de la propuesta de solución.
2. El estudio de herramientas y tecnologías permitió la selección de Python, HTML, JavaScript y CSS como lenguaje de programación. PyCharm como IDE de programación, PostgreSQL como servidor de base de datos y Apache como servidor de aplicación. El proceso de desarrollo estuvo guiado por la metodología AUP-UCI en su escenario número 4.
3. Con la implementación del sistema, se logró obtener un software que permite el apoyo a la toma de decisiones en el proceso de gestión de los RRHH, a través de la gestión y representación de los datos nominales de los trabajadores.
4. La estrategia de prueba aplicada permitió identificar y resolver los errores en la implementación, garantizando el correcto funcionamiento del producto final, obteniéndose una solución que cumple con los requisitos definidos y satisface las necesidades establecidas en los mismos. Además, la validación de la hipótesis a través del criterio de experto en su variante Delphi arrojó una aceptación de excelente para cada uno de los criterios definidos.



## REFERENCIAS BIBLIOGRÁFICAS

- ALLES, M., 2012. *Diccionario de términos de Recursos Humanos*: S.I.: Panorama Editorial. ISBN 978-950-641-643-0.
- AMO, F.A., NORMAND, L.M. y PÉREZ, F.J.S., 2005. *Introducción a la ingeniería del software*. S.I.: Delta. ISBN 978-84-96477-00-1.
- ASALE, R.-, 2014. Diccionario de la lengua española - Edición del Tricentenario. *Diccionario de la lengua española - Edición del Tricentenario* [en línea]. [Consulta: 22 mayo 2018]. Disponible en: <http://dle.rae.es/?w=diccionario>.
- CARDOZZO, D.R. y ACADEMY, I.T.C., 2016. *Desarrollo de Software: Requisitos, Estimaciones y Análisis. 2 Edición*. S.I.: CreateSpace Independent Publishing Platform. ISBN 978-1-5300-8861-4.
- CHAZALLET, S., 2016. *Python 3: los fundamentos del lenguaje*. S.I.: Ediciones ENI. ISBN 978-2-409-00614-2.
- COLECTIVO DE AUTORES, 2015. *Python 3.5: Tutorial*. S.I.: Samurai Media Limited.
- COLECTIVO DE AUTORES, F. y P., 2014. *Manual de Usuario. Sistema Automatizado de Recursos Humanos – Nómina Versión 5.4.55*. 2014. S.I.: s.n.
- DAR, U., KROSING, H., MLODGENSKI, J. y ROYBAL, K., 2015. *PostgreSQL Server Programming - Second Edition*. S.I.: Packt Publishing. ISBN 978-1-78398-059-8.
- DE LA FUENTE, F.G. y DE LOS ÁNGELES GIL ESTALLO, M., 2004. *Los sistemas de información en la sociedad del conocimiento*. S.I.: Esic. ISBN 978-84-7356-370-3.
- FORCIER, J., BISSEX, P. y CHUN, W.J., 2008. *Python Web Development with Django*. S.I.: Pearson Education. ISBN 978-0-13-270181-5.
- GAMMA, E., 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*. S.I.: Pearson Education. ISBN 978-81-317-0007-5.

- GAMMA, E., 2002. *Patrones de diseño: elementos de software orientado a objetos reutilizable*. S.I.: Pearson Educación. Addison-Wesley professional computing series. ISBN 978-84-7829-059-8.
- HALILI, E.H., 2008. *Apache JMeter: A Practical Beginner's Guide to Automated Testing and Performance Measurement for Your Websites*. S.I.: Packt Publishing. ISBN 978-1-84719-296-7.
- HOLOVATY, A. y KAPLAN-MOSS, J., 2010. *La guía definitiva de Django*. S.I.: Anaya Multimedia. Títulos Especiales. ISBN 978-84-415-2694-5.
- ISLAM, Q.N., 2015. *Mastering PyCharm*. S.I.: Packt Publishing. ISBN 978-1-78355-132-3.
- JACOBSON, I., BOOCH, G. y RUMBAUGH, J., 2000. *El proceso unificado de desarrollo de software*. S.I.: Pearson Educación. ISBN 978-84-7829-036-9.
- MONTORO, A.F., 2013. *Python 3 al descubierto*. S.I.: RC Libros. ISBN 978-84-939450-4-6.
- MONTOYA, E.S., 2013. *Prueba funcional del software: un proceso de verificación constante*: S.I.: s.n. ISBN 978-958-8743-41-7.
- MORALES CARTAYA, A., 2009. CONTRIBUCIÓN PARA UN MODELO CUBANO DE GESTIÓN INTEGRADA DE RECURSOS HUMANOS. ,
- MOSS, G., 2017. *Working with Odo 10*. S.I.: Packt Publishing. ISBN 978-1-78646-974-8.
- OSCAR, S., 2013. *Visual Paradigm for Uml*. S.I.: International Book Market Service Limited. ISBN 978-613-9-16653-4.
- PILLAI, A.B., 2017. *Software Architecture with Python*. S.I.: Packt Publishing. ISBN 978-1-78646-722-5.
- POMPA NUÑEZ, S., 2013. Módulo Recursos Humanos para el Sistema Integral de Gestión del Fondo Cubano de Bienes Culturales | eGov UFSC. [en línea]. [Consulta: 24 marzo 2018]. Disponible en: <http://www.egov.ufsc.br/portal/conteudo/m%C3%B3dulo-recursos-humanos-para-el-sistema-integral-de-gesti%C3%B3n-del-fondo-cubano-de-bienes-cult>.

- PRESSMAN, R.S., 2005. *Software Engineering: A Practitioner's Approach*. S.l.: Boston. ISBN 978-0-07-301933-8.
- PRESSMAN, R.S. y BRUCE R. MAXIM, D., 2014. *Software Engineering: A Practitioner's Approach*. S.l.: McGraw-Hill Education. ISBN 978-0-07-802212-8.
- PRESSMAN, R.S. y MURRIETA, J.E.M., 2006. *Ingeniería del software: un enfoque práctico*. S.l.: McGraw-Hill. ISBN 978-970-10-5473-4.
- RODRÍGUEZ SÁNCHEZ, T., 2014. *Metodología de desarrollo para la Actividad productiva de la UCI*. 2014. S.l.: s.n.
- ROSSUM, G., WARSAW, B. y COGHLAN, N., 2001. PEP 8 -- Style Guide for Python Code. *Python.org* [en línea]. [Consulta: 4 abril 2018]. Disponible en: <https://www.python.org/dev/peps/pep-0008/>.
- RUIZ LLANES, E., 2015. Solución para informatizar el proceso de baja en la Universidad de las Ciencias Informáticas. [en línea], [Consulta: 6 abril 2018]. Disponible en: <http://repositorio.uci.cu/jspui/handle/ident/8875>.
- RUSSELL, J. y COHN, R., 2012. *Acunetix*. S.l.: Book on Demand. ISBN 978-5-512-06368-2.
- SABÍN, R.G., 2005. *Nuevas tecnologías aplicadas a la gestión de RRHH: las TIC's como herramienta de mejora permanente del capital humano*. S.l.: Ideaspropias. ISBN 978-84-934547-5-3.
- SALAZAR MARTÍNEZ, E., 2015. Propuesta de Procedimiento para realizar pruebas de Caja Blanca a las aplicaciones que se desarrollan en lenguaje Python. [en línea]. Granma, Cuba: Facultad Regional Granma de la Universidad de las Ciencias Informáticas. Disponible en: <http://revistas.unab.edu.co/index.php?journal=rcc&page=article&op=view&path%5B%5D=1833&path%5B%5D=1662>.
- SANTOS, A.C., 1999. La toma de decisiones consensuales: instrumentos y experiencias en gestión organizacional. *Dirección y Organización* [en línea], vol. 0, no. 22. [Consulta: 14 mayo 2018]. ISSN 2171-6323. Disponible en: <http://www.revistadyo.com/index.php/dyo/article/view/283>.

SIAU, K., 2004. *Advanced Topics in Database Research*. S.I.: Idea Group Publishing. ISBN 978-1-59140-296-1.

SMITH, J., 2016. *Oracle Fusion Human Capital Management Essentials*. S.I.: CreateSpace Independent Publishing Platform. ISBN 978-1-5234-2494-8.

SOMMERVILLE, I. y GALIPIENSO, M.I.A., 2005. *Ingeniería del software*. S.I.: Pearson Educación. ISBN 978-84-7829-074-1.

WIEGERS, K.E., 2009. *Software Requirements*. S.I.: Microsoft Press. ISBN 978-0-7356-3708-5.

## ANEXOS

### ANEXO 1 ESTIMACIÓN

*Tabla 13: Estimación*

Prioridad	Complejidad	Esfuerzo (días)	Puntos estimados
Alta	Alta	5-8	18
Alta	Alta	0-4	17
Alta	Media	5-8	16
Alta	Media	0-4	15
Alta	Baja	5-8	14
Alta	Baja	0-4	13
Media	Alta	5-8	12
Media	Alta	0-4	11
Media	Media	5-8	10
Media	Media	0-4	9
Media	Baja	5-8	8
Media	Baja	0-4	7
Baja	Alta	5-8	6
Baja	Alta	0-4	5
Baja	Media	5-8	4
Baja	Media	0-4	3
Baja	Baja	5-8	2
Baja	Baja	0-4	1

### ANEXO 2: CUESTIONARIO A EXPERTOS

La presente entrevista tiene como objetivo la validación del Módulo Asistente a la Gestión de RRHH para el Sistema de Información Primaria de Personas. En su condición de experto sería importante que plasmara sus consideraciones y valoración acerca de los indicadores que a continuación se le presentan marcando con una (X) la opción que considere apropiada.

*Tabla 14: Cuestionario a expertos*

Variable	No.	Indicadores	5	4	3	2
	I1	Entendimiento				
	I2	Aprendizaje				
	I3	Operabilidad				
	I4	Atracción				
	I5	Esfuerzo del usuario				

**Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas.**

Módulo asistente a la gestión de RRHH.	I6	Facilidad de uso				
	I7	Capacidad de la interfaz visual				
	I8	Seguridad				
	I9	Tolerancia a fallas				
	I10	Recuperación				
	I11	Tiempo de procesos				
	I12	Capacidad de ser analizado				
	I13	Facilidad de prueba				
	I14	Posibilidad de actualización				
	I15	Estabilidad				
	I16	Coexistencia				
	I17	Exportación				
	I18	Aplicabilidad				
	I19	Precisión de los datos				
Información nominal de las personas actualizada y disponible.	I20	Posibilidad de actualizar los datos nominales				
	I21	Disponibilidad de la información de los trabajadores				
	I22	Información confiable				
Control interno de los trabajadores.	I24	Información de puestos de trabajo				
	I25	Información de área a la que pertenecen los trabajadores				
	I26	Información personal de los trabajadores				

Sus consideraciones acerca del análisis realizado puede expresarlas a continuación:

---



---



---



---

Para un mejor entendimiento de los indicadores planteados se presenta un glosario de términos.

**ANEXO 3: MATRIZ DE COEFICIENTES DE CONCORDANCIA POR INDICADOR**

*Tabla 15: Matriz de coeficiente de concordancia por indicador.*

Exp Ind	E1	E2	E3	E4	E5	E6		X <sub>m</sub>	D <sub>s</sub>	C
I1	5	3	5	4	4	5		4,33	0,82	81,6
I2	5	3	5	4	5	5		4,5	0,84	81,41
I3	4	3	4	5	5	4		4,17	0,75	81,93
I4	3	5	5	4	4	5		4,33	0,82	81,16
I5	5	4	4	5	5	5		4,67	0,52	88,93
I6	5	4	4	5	5	4		4,5	0,55	87,83
I7	5	4	5	4	4	5		4,5	0,55	87,83
I8	4	5	4	4	4	4		4,17	0,41	90,2
I9	4	5	5	4	5	5		4,67	0,52	88,93
I10	5	5	5	4	5	4		4,67	0,52	88,93
I11	5	5	5	5	4	5		4,83	0,41	91,55
I12	5	4	5	4	5	5		4,67	0,52	88,93
I13	5	4	5	5	4	5		4,67	0,52	88,93
I14	5	5	5	5	5	4		4,83	0,41	91,55
I15	5	5	5	5	5	5		5	0	100
I16	5	5	5	5	5	5		5	0	100
I17	5	5	5	5	5	5		5	0	100
I18	5	4	5	4	5	5		4,67	0,52	88,93
I19	5	5	5	4	4	4		4,5	0,55	87,83
I20	5	5	5	4	5	5		4,83	0,41	91,55
I21	5	5	5	5	5	5		5	0	100
I22	5	5	5	5	5	5		5	0	100
I23	5	5	5	5	5	5		5	0	100
I24	5	5	5	5	5	5		5	0	100
I25	5	5	5	5	5	5		5	0	100

*Tabla 16: Tabla de frecuencias*

Ind.	Frecuencia absoluta						Frecuencias acumuladas					Frecuencias relativas			
	Categorías				Total		Categorías					Categorías			
	E	B	R	M			E	B	R	M		E	B	R	M
I1	3	2	1	0	6	3	5	6	3	0,5	0,8	1	0		
I2	4	1	1	0	6	4	5	6	4	0,7	0,8	1	0		
I3	2	3	1	0	6	2	5	6	2	0,3	0,8	1	0		
I4	3	2	1	0	6	3	5	6	3	0,5	0,8	1	0		
I5	4	2	0	0	6	4	6	0	4	0,7	1	0	0		

I6	3	3	0	0	6	3	6	0	3	0,5	1	0	0
I7	3	3	0	0	6	3	6	0	3	0,5	1	0	0
I8	1	5	0	0	6	1	6	0	1	0,2	1	0	0
I9	4	2	0	0	6	4	6	0	4	0,7	1	0	0
I10	4	2	0	0	6	4	6	0	4	0,7	1	0	0
I11	5	1	0	0	6	5	6	0	5	0,8	1	0	0
I12	4	2	0	0	6	4	6	0	4	0,7	1	0	0
I13	4	2	0	0	6	4	6	0	4	0,7	1	0	0
I14	5	1	0	0	6	5	6	0	5	0,8	1	0	0
I15	6	0	0	0	6	6	0	0	6	1	0	0	0
I16	6	0	0	0	6	6	0	0	6	1	0	0	0
I17	6	0	0	0	6	6	0	0	6	1	0	0	0
I18	4	2	0	0	6	4	6	0	4	0,7	1	0	0
I19	3	3	0	0	6	3	6	0	3	0,5	1	0	0
I20	5	1	0	0	6	5	6	0	5	0,8	1	0	0
I21	6	0	0	0	6	6	0	0	6	1	0	0	0
I22	6	0	0	0	6	6	0	0	6	1	0	0	0
I23	6	0	0	0	6	6	0	0	6	1	0	0	0
I24	6	0	0	0	6	6	0	0	6	1	0	0	0
I25	6	0	0	0	6	6	0	0	6	1	0	0	0

**Tabla 17:** Puntos de corte y escala de los indicadores

Cálculo de los puntos de corte y escala de los indicadores							Evaluación
Ind.	E	B	R	Suma	P	N-P	
I1	0,000	0,967	3,490	4,457	1,486	-0,529	Excelente
I2	0,431	0,967	3,490	4,888	1,629	-0,673	Excelente
I3	-0,431	0,967	3,490	4,027	1,342	-0,386	Excelente
I4	0,000	0,967	3,490	4,457	1,486	-0,529	Excelente
I5	0,431	3,490	0	3,921	1,307	-0,351	Excelente
I6	0,000	3,490	0	3,490	1,163	-0,207	Excelente
I7	0,000	3,490	0	3,490	1,163	-0,207	Excelente
I8	-0,967	3,490	0	2,523	0,841	0,116	Excelente
I9	0,431	3,490	0	3,921	1,307	-0,351	Excelente
I10	0,431	3,490	0	3,921	1,307	-0,351	Excelente
I11	0,967	3,490	0	4,457	1,486	-0,529	Excelente
I12	0,431	3,490	0	3,921	1,307	-0,351	Excelente
I13	0,431	3,490	0	3,921	1,307	-0,351	Excelente
I14	0,967	3,490	0	4,457	1,486	-0,529	Excelente
I15	3,490	0	0	3,490	1,163	-0,207	Excelente
I16	3,490	0	0	3,490	1,163	-0,207	Excelente
I17	3,490	0	0	3,490	1,163	-0,207	Excelente
I18	0,431	3,490	0	3,921	1,307	-0,351	Excelente
I19	0,000	3,490	0	3,490	1,163	-0,207	Excelente
I20	0,967	3,490	0	4,457	1,486	-0,529	Excelente
I21	3,490	0	0	3,490	1,163	-1,163	Excelente
I22	3,490	0	0	3,490	1,163	-0,207	Excelente



**Módulo Asistente a la Gestión de los Recursos Humanos para el Sistema de Información Primaria de Personas.**

I23	3,490	0	0	3,490	1,163	-0,207	Excelente
P.C	1,298	1,970	0,558	95,638			

## GLOSARIO DE TÉRMINOS

**Entendimiento:** posibilidad de comprender el funcionamiento del módulo.

**Aprendizaje:** dificultad de los usuarios para aprender a utilizar el módulo.

**Operabilidad:** posibilidad de los usuarios de utilizar el módulo sin interrupciones técnicas.

**Atracción:** nivel de atracción del módulo para los usuarios.

**Esfuerzo del usuario:** esfuerzo que debe emplear un usuario en comprender el uso del Módulo Asistente a la Gestión de RRHH y poder usarlo.

**Facilidad de uso:** nivel en conocimiento que debe tener el usuario para poder interactuar con el módulo.

**Capacidad de la interfaz visual:** legibilidad de los datos que se muestran al usuario.

**Seguridad:** capacidad de evitar los accesos no autorizados a la información.

**Tolerancia a fallas:** se refiere a la habilidad de mantener un nivel específico de funcionamiento en caso de fallas del Módulo Asistente a la Gestión de RRHH o en caso de ocurrencia de infracciones en su interfaz.

**Recuperación:** se refiere a la capacidad de restablecer el nivel de operación y recobrar los datos que fueron afectados directamente por una falla, así como el tiempo y el esfuerzo necesarios para lograrlo.

**Tiempo de procesos:** tiempo de respuesta a las peticiones del usuario.

**Capacidad de ser analizado:** capacidad del Módulo Asistente a la Gestión de RRHH para atenerse a diagnósticos de deficiencias o causas de fallas en el software o la identificación de las partes a ser modificadas.

**Facilidad de prueba:** nivel de esfuerzo necesario para realizar pruebas a las funcionalidades del módulo.

**Estabilidad:** capacidad del Módulo Asistente a la Gestión de RRHH para evitar efectos inesperados por modificaciones del software.

**Coexistencia:** capacidad del Módulo Asistente a la Gestión de RRHH para coexistir con otros productos de software, compartiendo recursos comunes.

**Exportación:** posibilidad de generar reportes en los formatos predefinidos.

**Aplicabilidad:** nivel de aplicación del Módulo Asistente a la Gestión de RRHH.

**Posibilidad de actualizar los datos nominales:** posibilidad de actualizar la información laboral de los trabajadores.

**Disponibilidad de la información de los trabajadores:** posibilidad de acceder al módulo y consultar la información requerida.

**Información confiable:** restricciones para realizar acciones que afecten los datos del módulo a personal autorizado solamente.

**Información de puestos de trabajo:** información de los puestos de trabajo que ocupan los trabajadores en la entidad.

**Información de área a la que pertenecen los trabajadores:** información del área de trabajo que ocupan los trabajadores en la entidad.

**Información personal de los trabajadores:** posibilidad de consultar datos personales de los trabajadores.