Universidad de las Ciencias Informáticas
Facultad 3

Sistema de Gestión de Información para la Defensa

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

Autor:
Addiel Aguila Espinaco.

Tutor:
MSc. Sarianny Olivares Aguilar.

Co-Tutores:
Ing. Yenier Figueroa Machado.
Ing. Robin Sencial Terrero.

La Habana, Junio 2014
¨Año 56 de la Revolución¨

Declaración de autoría

Por este medio declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) para que haga el uso que estime pertinente.

Para que así conste firmo la presente a los ____ días del mes de _____ del 2014.

Autor:

_____

Addiel Aguila Espinaco

Tutor:

_____

MSc. Sarianny Olivares Aguilar

Co-Tutores:

_____ _____

Ing. Yenier Figueroa Machado Ing. Robin Sencial Terrero

Datos de contacto

Datos del Tutor:

Nombre: MSc. Sarianny Olivares Aguilar.

Máster en Educación.

Correo electrónico: solivares@uci.cu

Datos de los Co-Tutores:

Nombre: Ing. Yenier Figueroa Machado.

Ingeniero en Ciencias Informáticas.

Correo electrónico: yfigueroa@uci.cu

Nombre: Ing. Robin Sencial Terrero.

Ingeniero en Ciencias Informáticas.

Correo electrónico: rsencial@uci.cu

Datos del Autor:

Nombre: Addiel Aguila Espinaco.

Correo electrónico: aespinaco@estudiantes.uci.cu

". . .Y cuando cada casa sea una fortaleza, cuando cada calle, cada árbol, tenga un hombre que lo defienda y los obreros sepan combatir (…), en cada pedazo de tierra…
¿Quién podrá vencer esta Revolución?"

Fidel Castro Ruz

X Congreso de la CTC. 18 de noviembre de 1959

Agradecimientos

A mi mamá que gracias a su cariño, dedicación y esfuerzo hoy puedo alcanzar este logro.

A mis hermanos Jenniffer y Pablo Enrique, que solo el hecho de saber que puedo ser un ejemplo para ellos me enorgullece.

A Carlos Mario, por ser siempre un ejemplo a seguir y hacerme sentir como su hijo. Hoy estoy seguro que está orgulloso de mí.

A mi papá y a mi madrastra Noemí por el cariño y apoyo que me han brindado en todo momento.

A mis abuelos, en especial a mi abuela Mima y a Mony.

A mis tíos, mis primos y de forma general a toda mi familia que siempre se preocupan por mí y me ayudan a cumplir mis metas.

A Yunior por su apoyo incondicional y a su familia por hacerme sentir parte de ella en tan poco tiempo.

A mi tutora, Sarianny y su familia, que para mí ella ha sido como mi segunda madre en la universidad.

A todas mis amistades, mis Titis, Rodnier, Aniel, Dayán, Ronniel, Joenis y la Familia de amigos a la cual pertenezco.

A mis compañeros de la FEU y de los Festivales, los de la nueva y vieja escuela; por compartir tantos madrugones, dejar que yo soñara cosas casi imposible y ayudarme a cerrar en todo lo que hacíamos.

A mis amigas Yanete y Nidia que a pesar de la distancia siempre vamos a estar juntos y a todos mis compañeros de grupo con los que compartí estos 6 años.

A todos los profes del Centro Cultural por soportarme cada día, en especial a Malcom y Yenirsy, por darme la posibilidad de creerme artista.

A Rosalina, por su amistad y por contribuir a mi felicidad.

A Sailí y a Ruth, por su apoyo, cariño y cada día tratar de alegrarme la vida.

A mis amigos y conocidos que van dejando un poco de su experiencia en mi vida y por estar a mi lado en las buenas y el las malas.

A todos los que de una forma u otra me ayudaron.
A todos los que alguna vez se preocuparon y me dijeron: "¿Y la tesis?"

Dedicatoria

La vida más que un regalo de dios, es una bendición que nos dan nuestros padres en especial nuestras madres; por eso hoy quiero dedicar este logro a mi mamá. La persona que me mantiene firme y constante en mis propósitos, a su cariño, apoyo y su confianza en todo lo que hago.

RESUMEN

El Sistema de Defensa Nacional tiene como tarea primordial mantener las conquistas alcanzadas, bajo la concepción de la guerra de todo el pueblo; es por ello que en la Universidad de las Ciencias Informáticas se desarrollan actividades de preparación teóricas-prácticas asociadas a la defensa para cumplir eficientemente con el encargo social.

En la Facultad 3 de esta institución se llevan a cabo un grupo de tareas en relación con el tema, y asociadas a ellas existe un gran flujo de información que debe estar actualizada y disponible. Actualmente, se gestiona de manera manual, detectando insuficiencias en el proceso, como son: problemas en la actualización, ejecución, control de los datos lo que afecta el cumplimiento de las tareas y misiones del Batallón Atípico de MTT de la facultad.

Con el propósito de mejorar dicha gestión se propone en esta investigación el sistema informático SIGiD (Sistema de Gestión de Información para la Defensa). En este documento se describen las etapas de análisis, diseño, implementación y prueba del sistema antes mencionado. Para guiar el desarrollo del sistema, se escogió como metodología de desarrollo de software SXP y para la selección de las herramientas se eligió seguir la línea de desarrollo para aplicaciones web del Centro de Gobierno Electrónico.

Entre las facilidades que brinda el sistema está: emplantillamiento y ubicación del personal; planificación, ejecución y evaluación de actividades; y generación de reportes mediante una interacción directa y fácil con los usuarios.

Palabras claves: defensa, preparación para la defensa, sistemas de gestión de la información.

ÍNDICE DE CONTENIDO

RESUMEN.....VI
INTRODUCCIÓN.....11
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.....16
1.1 Antecedentes del Sistema de Defensa Nacional.....16
1.2 Sistema de Defensa de la UCI.....18
1.3 Gestión de la información.....20
1.4 Sistemas de gestión de la información.....20
1.4.1 Sistemas de gestión de información de recursos humanos.....21
1.5 Metodología de desarrollo de software.....25
1.6 Herramientas a utilizar en la propuesta de solución.....28
1.6.1 Lenguaje de Modelado UML 2.0.....29
1.6.2 Herramienta CASE (Computer Aided Software Environment).....30
1.6.3 Herramienta para modelado de prototipado web.....31
1.6.4 Marco de Trabajo.....31
1.6.5 Lenguaje de programación.....33
1.6.6 Sistema Gestor de Base de Datos.....35
1.6.7 Servidor web.....36
1.6.8 Entorno de Desarrollo Integrado.....36
1.7 Conclusiones Parciales.....37
CAPÍTULO 2: DESCRIPCIÓN DEL SISTEMA.....38
2.1 Descripción de los procesos del negocio.....38
2.2 Propuesta de solución.....39
2.2.1 Concepción inicial del sistema.....39
2.3 Captura de Requisitos.....39
2.3.1 Requisitos funcionales.....40
2.3.2 Requisitos no funcionales.....41
2.3.3 Lista de Reserva del Producto (LRP).....41
2.4 Diseño de metáforas.....42
2.4.1 Historias de usuarios del negocio.....42
2.4.2 Tareas de la Ingeniería.....44
2.5 Arquitectura.....45
2.6 Patrones de diseño.....47
2.7 Diagrama de paquetes.....51
2.8 Modelo de datos.....52

2.9 Conclusiones Parciales 54
CAPÍTULO 3: IPLEMETACIÓN Y VALIDACIÓN 55
3.1 Implementación 55
3.1.1 Diagrama de componentes 55
3.1.2 Diagrama de despliegue 57
3.1.3 Plan de Releases 58
3.1.4 Estándar de codificación empleado 59
3.2 Validación 60
3.2.1 Validación de requisitos 60
3.2.2 Validación del diseño de la aplicación web 61
3.3 Pruebas 64
3.3.1 Pruebas de funcionalidad 65
3.3.2 Pruebas unitarias 66
3.4 Valoración del comportamiento de las variables de la investigación 68
3.5 Conclusiones Parciales 71
CONCLUSIONES GENERALES 72
RECOMENDACIONES 73
BIBLIOGRAFÍA REFERENCIADA 74

ÍNDICE DE FIGURAS

Figura 1: Estructura del Sistema de Defensa Nacional. 17
Figura 2: Sistema de Defensa UCI. 18
Figura 3: Esquema de la metodología SXP. 28
Figura 4: Esquema de la arquitectura. 46
Figura 5: Uso del patrón Llaves Subrogadas en la entidad Persona. 48
Figura 6: Empleo del patrón Control de Acceso Basado en Roles (RBAC). 48
Figura 7: Empleo del patrón Experto. 49
Figura 8: Evidencia del patrón Creador en la clase PersonaController.php. 49
Figura 9: Evidencia del patrón Bajo Acoplamiento en las clases: TrabajadorInterno, TrabajadorExterno, Trabajador y Persona. 50
Figura 10: Evidencia del patrón Alta Cohesión en la clase PersonalController. 50
Figura 11: Diagrama paquetes del sistema. 51
Figura 12: Diagrama de Clases del paquete Core. 52
Figura 13: Diagrama de Clases del bundle PersonaBundle del paquete Defensa. 52
Figura 14: Modelo de datos. 53
Figura 15: Diagrama de componentes. 56
Figura 16: Diagrama de componentes en función de la arquitectura. 57
Figura 17: Diagrama de despliegue. 58
Figura 18: Cantidad de procedimiento por clases. 62
Figura 19: Resumen de cantidad de procedimientos. 62
Figura 20: Resultado de las variables: Responsabilidad, Complejidad y Reutilización. 63
Figura 21: Cantidad de relaciones de uso por clases. 63
Figura 22: Resumen de relaciones de uso. 63
Figura 23: Resultado de las variables Acoplamiento, Complejidad, Reutilización y Cantidad de Pruebas. 64
Figura 24: Resumen de Pruebas de Caja Negra. 66
Figura 25: Grafo de flujo del código de la función UbicarAction() y cálculo de su complejidad ciclomática. 67
Figura 26: Resultados obtenidos en la aplicación de la tlecnica de Caja Blanca. 68

ÍNDICE DE TABLAS

Tabla 1: Comparación entre Aplicaciones Web y Desktop 25
Tabla 2: Comparación entre las metodologías SCRUM, XP y SXP 27
Tabla 3: Herramientas y tecnologías de las líneas de desarrollo de CEGEL 29
Tabla 4: Fragmento de la Plantilla LRP (Prioridad: Muy Alta). 42
Tabla 5: Distribución de HU por ARF. 43
Tabla 6: HU_2: Mostrar Perfil Personal. 44
Tabla 7: T_6: Diseñar las interfaces requeridas para la funcionalidad Mostrar perfil personal. 44
Tabla 8: T_7: Implementar funcionalidad: Mostrar perfil personal. 44
Tabla 9: Plan de Releases. 59
Tabla 10: Valores de las variables de calidad: Responsabilidad, Complejidad y Reutilización. 62
Tabla 11: Valores de las variables: Acoplamiento, Complejidad, Reutilización y Cantidad de Pruebas. 64
Tabla 12: CP_3-6 correspondiente a HU_8: Reubicar personal. 65
Tabla 13: Caso de prueba para el camino básico 1. 67
Tabla 14: Caso de prueba para el camino básico 4. 68
Tabla 15: Medición de tiempo del proceso: Ubicar Personal (Antes del sistema). 69
Tabla 16: Medición de tiempo del proceso: Ubicar Personal (Con el sistema). 70

INTRODUCCIÓN

La preparación militar como parte de la formación docente de la sociedad cubana, sustentada en la concepción de la guerra de todo el pueblo; surge a raíz de la posibilidad real de que se produzca un ataque militar contra Cuba.

El Ministerio de las Fuerzas Armadas Revolucionarias (MINFAR), como institución militar del Estado Cubano, ha considerado como elemento clave: el empleo del capital humano, en interés de asignarle a cada ciudadano un lugar en la defensa. Esta institución tiene como prioridad los requisitos del sistema defensivo territorial y las condiciones físicas, psíquicas, políticas y sociales existentes. Además de la preparación militar y otras cualidades personales de modo que cada efectivo pueda ser empleado apropiadamente, de acuerdo con las misiones que le sean asignadas.

Para lograr que las nuevas generaciones se sientan identificadas y comprometidas con la Patria, el MINFAR ha considerado como estrategia vital, la preparación teórica y práctica en temas relacionados con la defensa. En aras de cumplir estos objetivos, se han trazado una serie de acciones de formación y preparación dirigidas a la juventud cubana. En el contexto educacional han sido insertadas diferentes disciplinas y actividades docente-educativas de perfil ideológico-militar como parte del proceso cubano de enseñanza en todos sus niveles, tras la aplicación de un convenio entre el MINFAR y el Ministerio de Educación (MINED).

En el ámbito universitario, se ha implementado y aprobado como parte del currículo de las diferentes carreras de pregrado, la disciplina docente: Preparación para la Defensa (PPD), la cual abarca contenidos teóricos e histórico-militares, dirigidos a desarrollar los conocimientos necesarios sobre defensa y seguridad nacional de los futuros profesionales. La PPD se imparte mediante diferentes asignaturas que divergen en dependencia de las condiciones y características de cada institución; y en función de la carrera que se estudie.

Dentro de los centros que pertenecen al Ministerio de Educación Superior (MES), se encuentra la Universidad de las Ciencias Informáticas (UCI), donde se aplica también la estrategia de formación militar, tanto para estudiantes como trabajadores. En el pregrado se gestiona la preparación teórica mediante la PPD, consolidándose en las asignaturas Seguridad Nacional y Defensa Nacional, y de forma práctica se convoca a ejercicios y actividades como: La Reunión de Estudios Militares (REM), Días de la Defensa Nacional, Bastiones Universitarios, Meteoros, entre otras. Todas con el

objetivo fundamental de tributar a la preparación de la defensa del país y mantener su soberanía.

La UCI, anualmente se convierte en sede de todos los ejercicios militares antes mencionados, para ello es imprescindible mantener activo su Sistema de Defensa, el cual acoge a más de 12000 personas, por lo que es necesario que cada área de la universidad gestione de manera eficiente la información que se genera de los Batallones Mixtos de las Milicias de Tropas Territoriales (MTT) Atípicos, estructura que acoge a cada efectivo de la institución.

La Facultad 3 como una de las áreas que integra el Sistema de Defensa de la UCI, al comienzo de cada curso escolar mediante el Vicedecanato de Extensión y Residencia (VDER) en estrecha relación con el Área de Atención Militar UCI, gestiona la información referente a la defensa. Se reconoce un alto nivel de compromiso y responsabilidad ante esta tarea, no obstante, se identifica como insuficiencia que no se realiza una adecuada gestión de la información debido a que:

- ✓ Se manejan grandes volúmenes de información, la cual es procesada de manera manual, lo que trae consigo una asignación mayor de personal y tiempo para llevar a cabo este proceso, demorándose aproximadamente un mes.
- ✓ Los datos son almacenados en formato duro y múltiples archivos de ofimática, por lo que se encuentran sujetos a la pérdida, el deterioro y duplicación de la información, así como errores de totalización.
- ✓ No existe interacción con la información generada del proceso, lo que limita el acceso y actualización permanente de los datos, así como la manipulación y divulgación de los resultados.
- ✓ Existen deficiencias en el emplantillamiento y la ubicación del personal, asociadas a la carencia de planillas digitales que permitan aglutinar los datos personales y militares de cada efectivo, lo que dificulta la disponibilidad del batallón ante cualquier evento y unido a esto el cumplimiento de sus tareas y misiones.

Lo antes expuesto constituye la problemática que origina esta investigación, por lo que se define el siguiente **problema a resolver**: ¿Cómo mejorar la inadecuada gestión de la información sobre la estructura militar y las actividades de la defensa del Batallón Mixto de MTT de la Facultad 3 de la UCI, que afecta el cumplimiento de sus tareas y misiones?

Al identificarse el problema, se determina como **objeto de investigación**: los sistemas de gestión de la información. Por lo que se concreta como **campo de acción:** las herramientas de gestión de la información.

Para sustentar el objeto de estudio y resolver las insuficiencias, se asume como **objetivo general**: Desarrollar un sistema informático que mejore la gestión de la información referente a la estructura militar y las actividades de la defensa del Batallón Mixto de MTT de la Facultad 3 de la UCI.

Se defiende la **idea principal** que el desarrollo del sistema informático para la gestión de la información referente a la estructura militar y las actividades de la defensa, permitirá mejorar el cumplimiento de las tareas y misiones del Batallón Mixto de MTT de la Facultad 3 de la UCI.

Para dar cumplimiento al objetivo general se plantean los siguientes **objetivos específicos**:

- ✓ Establecer los referentes teóricos de la investigación mediante un estudio sobre estructuras y actividades del Sistema de Defensa en la UCI, sistemas de gestión de la información y herramientas para la gestión de la información.
- ✓ Analizar herramientas para la gestión de la información para comparar sus características e identificar ventajas y limitaciones.
- ✓ Identificar tendencias y enfoques de herramientas para la gestión de la información en el proceso de desarrollo de software para establecer patrones de aplicación en las herramientas estudiadas.
- ✓ Implementar un sistema informático que permita la gestión de la información referente a la estructura militar y las actividades de la defensa del Batallón Mixto de MTT de la Facultad 3 de la UCI.
- ✓ Validar el sistema informático propuesto a partir de pruebas de funcionalidad y pruebas unitarias.

Para dar cumplimiento a los objetivos específicos planteados, se desarrollan las siguientes **tareas de la investigación**:

- ✓ Establecimiento del estado del arte de la investigación a partir del estudio de las estructuras y actividades del Sistema de Defensa UCI, sistemas de gestión de la información y herramientas para la gestión de la información.
- ✓ Caracterización de las herramientas para la gestión de la información en el desarrollo de software.

- ✓ Análisis de tecnologías y lenguajes a emplear para desarrollar el sistema informático.
- ✓ Identificación, especificación y validación de los requisitos funcionales y no funcionales a tener en cuenta para el desarrollo de los módulos del sistema informático, con el uso de técnicas especializadas y la métrica de especificación.
- ✓ Validación de la herramienta propuesta a partir de pruebas de funcionalidad y pruebas unitarias; mediante el método de Caja Negra con la técnica de Partición de Equivalencia y el método de Caja Blanca con la técnica de Camino Básico respectivamente.

Para dar cumplimiento a las tareas propuestas anteriormente, se utilizaron los siguientes métodos científicos de investigación teóricos y empíricos. Dentro de los **métodos teóricos** se utilizaron:

- ✓ *Histórico-Lógico:* permitió realizar un estudio sobre la estructura del Sistema de defensa en la UCI, los sistemas y herramientas para la gestión de la información existentes, las tendencias del uso actual de las mismas, con el fin de seleccionar los patrones más apropiados para implementar en la propuesta de solución y darle cumplimiento al objetivo general de la presente investigación.
- ✓ *Analítico-Sintético*: permitió realizar el estudio teórico de la investigación facilitando el análisis de documentos y la extracción de los elementos más importantes acerca del proceso de desarrollo de sistemas de gestión de la información.
- ✓ *Inductivo-Deductivo: fue utilizado para el razonamiento* de la información consultada y llegar a un grupo de conocimientos particulares y generales sobre los procesos de gestión de información.
- ✓ *Modelación:* permitió la creación de una representación o modelo de la gestión de la información referente a la defensa, haciendo posible el diseño de los prototipos funcionales y la representación de los requisitos del sistema informático.

Dentro de los **métodos de investigación empíricos** se emplearon:

- ✓ *Observación:* se empleó a través de la realización del estudio de los elementos que componen la gestión de la información de la defensa existente en la UCI, además en el análisis de algunas herramientas de gestión de información a nivel mundial y en Cuba, que brindan información sobre el proceso de gestión y seguridad de la información que poseen, así como la observación de la ejecución de actividades realizadas en la UCI tales como: la REM, el Bastión Universitario y movilizaciones de reservistas de Unidades Militares.

- ✓ *Entrevista:* se aplicó a sujetos para comprobar la forma en que se gestiona la información referente a la defensa, propiciando el levantamiento de requisitos del sistema y obtención de información necesaria para el desarrollo de la investigación.
- ✓ *Medición:* permitió medir la calidad de la especificación de los requisitos y el grado de ambigüedad de estos, además de obtener una medida de la calidad del diseño para su validación y el comportamiento de las variables de la investigación.

El presente trabajo está estructurado de la siguiente manera:

Capítulo 1: Fundamentación teórica

En este capítulo se da a conocer las cuestiones teóricas necesarias para la comprensión de la investigación, lo cual incluye el establecimiento del estado del arte a través del estudio de herramientas de gestión de información a nivel mundial y nacional. Se realiza el análisis y selección de la metodología de desarrollo en función de las necesidades y las características del grupo de desarrollo. Así como el estudio de las líneas de desarrollo del Centro de Gobierno Electrónico (CEGEL), para decidir que herramientas emplear en el desarrollo de la solución.

Capítulo 2: Descripción de la aplicación

En este capítulo se realiza una breve descripción de los principales procesos del negocio y el análisis de la propuesta de solución. Luego se definen los requisitos funcionales, no funcionales y se describen las principales historias de usuario. Se expone todo el diseño de la aplicación con los artefactos definidos por la metodología para esta fase. Además se muestran varios diagramas para esclarecer las características de la propuesta de solución. Por último se muestra la arquitectura a la cual responde el sistema, la evidencia de los patrones de diseño utilizados, el diagrama de paquetes y el modelo de datos.

Capítulo 3: Implementación y validación de la propuesta

Se abordan aspectos significativos de la implementación del sistema, así como los artefactos generados del proceso. Se muestran los resultados de la validación de los requisitos y el diseño, a través de la aplicación de técnicas y métricas respectivamente. Además, se hace un resumen de los resultados obtenidos tras la aplicación de las pruebas de funcionamiento y unitarias realizadas a la al sistema. Por último se realiza una valoración de las variables de la investigación antes y después de la puesta en práctica de SGiD.

CAPÍTULO 1
FUNDAMENTACIÓN TEÓRICA

En el presente capítulo se exponen conceptos que permitirán comprender la esencia del negocio que se describe, así como la teoría que facilita entender lo fenomenológico asociado al objeto de investigación. Se abordan además; las tendencias, distribución y uso de los sistemas de gestión de información a nivel internacional y nacional. También se describen las principales características y funcionalidades de las tecnologías a emplear en la construcción de la propuesta de solución.

1.1 Antecedentes del Sistema de Defensa Nacional

La preparación militar como parte de la formación docente de la sociedad cubana, se inició en septiembre de 1975, al firmarse el convenio entre el MINFAR y el MINED. Luego al aprobarse la Directiva 29 del MINFAR se planteó perfeccionar el sistema de preparación para la defensa de los estudiantes universitarios. Lo que dio origen a la disciplina de PPD, con una proyección más específica, vinculada a las exigencias del futuro desempeño de los egresados, sus responsabilidades y acciones concretas en relación con la defensa.

Por ello, su misión principal es la formación de profesionales con calidad y pertinencia, capacidad de liderazgo político de forma continua y sistemática, con una amplia preparación técnica y humanística; así como su posterior superación de postgrado; identificados con la Revolución y el socialismo (1).

El Sistema de Defensa Nacional cubano tiene como tarea primordial mantener las conquistas alcanzadas por la Revolución, por lo que todo el esfuerzo militar se dedica estrictamente a la defensa del país. Este se constituye y prepara desde tiempo de paz, para dirigir el país en las condiciones de: estado de guerra, guerra, movilización general o estado de emergencia.

Mientras que el MINFAR es el organismo encargado de dirigir, ejecutar y controlar la aplicación de la política del Estado y del Gobierno para la preparación del país, la soberanía y la realización de la lucha armada en caso de guerra (2).

Por su parte, las Milicias de Tropas Territoriales constituyen una de las formas de organización popular para llevar a cabo la lucha armada y cumplir otras tareas de la defensa. Están integradas sobre la base de los principios de voluntariedad, selectividad y territorialidad. Pertenecen a las MTT más de un millón de cubanos y casi

la mitad son mujeres. Sus unidades están armadas en su totalidad con armamento de infantería, artillería y artillería antiaérea. Están organizadas en divisiones, regimientos, batallones, compañías independientes y Formaciones Especiales; que pueden cumplir misiones de carácter territorial, planteadas por los Consejos de Defensa Provinciales, Municipales y de Zona (3).

En la siguiente figura se muestra la estructura del Sistema de Defensa Nacional:

Figura 1: Estructura del Sistema de Defensa Nacional (Elaboración propia).

Según funge en el Artículo 56, Capítulo IV, de la Ley de Defensa Nacional: Las entidades civiles, centros educacionales y organismos del estado pueden crear, de acuerdo con las necesidades de la defensa territorial y a cuenta de sus propios recursos, agrupaciones de fuerzas y medios con el carácter de Formaciones Especiales de las MTT. Los jefes de las entidades mencionadas están en la obligación de prepararlas y darles los aseguramientos necesarios (4).

El autor de esta investigación concuerda en que las universidades cubanas deben integrarse de manera eficiente a esta organización, mantener una adecuada estructura y formación especial de las MTT, para llevar a cabo la formación y actualización constante de su personal, buscando estrategias que le permitan la capacitación, preparación y gestión de la información en todo lo referente a la defensa.

1.2 Sistema de Defensa de la UCI

Al ser la UCI una universidad de nuevo tipo que posee características especiales que la distinguen de las del resto del país, aglutina un número elevado de capital humano en función de la defensa. Cuenta con un Sistema de Defensa propio de la institución compuesto por la Dirección de Seguridad, Protección y Defensa (DSPD), la Cátedra de PPD, Área de Atención Militar y los VDER de las facultades; dirigido por el Consejo de Defensa UCI.

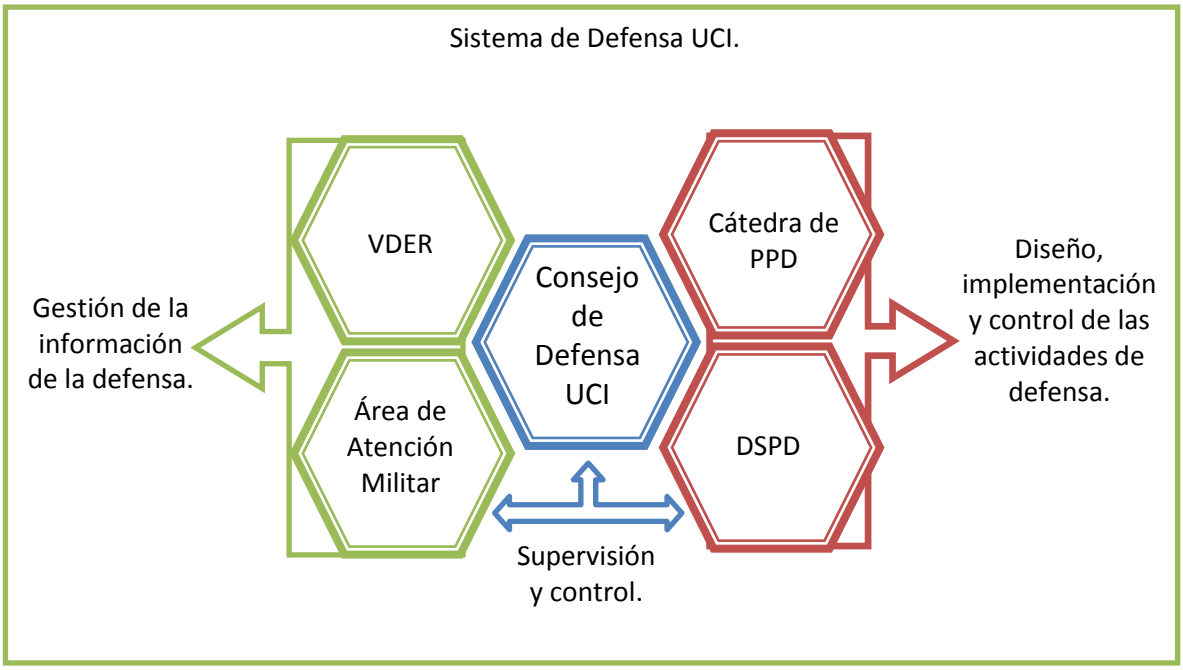


Figura 2: Sistema de Defensa UCI (Elaboración propia).

Como se muestra en la Figura 2, cada una de las áreas que integran el Sistema de Defensa UCI, tienen misiones específicas, donde se destaca la DSPD encargada del diseño, planificación y realización de las actividades de preparación militar en la universidad, en previa coordinación con la Cátedra de PPD.

Por otra parte los VDER de cada una de las facultades y el Área de Atención Militar son los encargados de gestionar toda la información referente a las estructuras militares y las actividades de la defensa de sus batallones, tales como:

- ✓ Realizar la ubicación del personal en las estructuras de sus batallones y la asignación de actividades para su realización.
- ✓ Archivar en múltiples documentos de ofimática el emplantillamiento de cada efectivo y los resultados del control y evaluación de las actividades ejecutadas.
- ✓ Mantener un carácter de confidencialidad sobre los datos almacenados, mediante un acceso limitado a los mismos.
- ✓ Preservar en buen estado las tarjetas que se encuentran en formato duro con la documentación personal de cada efectivo.

- ✓ Mantener la información actualizada y disponible para la movilización oportuna de los batallones ante situaciones excepcionales.

La existencia de batallones mixtos de MTT como estructura organizativa para la defensa; exige que la UCI y cada una de sus áreas utilicen como guía una serie de documentos rectores que dirigen, regulan y organizan todos sus procesos, entre los cuales destacan:

- ✓ Plantilla resumen de Plana Mayor y Cuadros de Mando de Batallón de MTT Atípico: específica de forma escalonada las funciones en cada uno de los niveles de mando dentro del batallón, así como sus misiones y objetivos ante situaciones excepcionales (Ver Anexo 1).
- ✓ Acta de Cooperación entre la UCI y el Sector Militar del municipio La Lisa: se establecen las acciones conjuntas entre el Sector Militar del municipio La Lisa y la UCI para garantizar el registro, control, aviso, presentación y traslado de los estudiantes y trabajadores de la universidad (Ver Anexo 2).
- ✓ La Resolución No.46 de del MINFAR: se disponen las normas para organizar el registro militar, empleo en la defensa y movilización de los trabajadores y estudiantes (Ver Anexo 3).
- ✓ Consejo de Defensa UCI y sus grupos de trabajo: documento institucional donde se describe la estructura de mando del Consejo de Defensa de la universidad, así como sus diferentes grupos de trabajo con sus respectivas funciones y misiones (Ver Anexo 4).
- ✓ Orden No.3 del Viceministro Jefe del Estado Mayor General: se dispone la aplicación de la política de empleo de los estudiantes de los centros de educación superior en el completamiento de las unidades de las FAR y la organización de las reservas listas de personal (Ver Anexo 5).

Si se realiza una valoración del Sistema de Defensa de la UCI, su estructura, organización, misiones y responsabilidades para con cada efectivo en la defensa, no cabe duda que cada área y específicamente las facultades por ser las que aglutinan mayor cantidad de personal entre estudiantes y trabajadores; tienen una gran responsabilidad en la gestión de la información que generan, pues de la calidad y pertinencia de esta, depende la efectividad del cumplimiento de las tareas y misiones de cada uno de los batallones.

1.3 Gestión de la información

Gestionar información es: ir en busca de nuevos significados, analizar y aplicar el principio de que el todo, es más que la suma de las partes. Es producir un impacto en el ambiente de cualquier organización (5).

La gestión de la información es el proceso de analizar y utilizar la información que se ha recabado y registrado, para permitir a los administradores (a todos los niveles) tomar decisiones documentadas, así como para mejorar los procesos, productos y servicios de la organización (5).

Por otro lado Woodman plantea "...la gestión de información es todo lo que tiene que ver con obtener la información correcta, en la forma adecuada, para la persona indicada, al costo correcto, en el momento oportuno, en el lugar indicado para tomar la acción precisa" (6).

Para Ponjuán cuando se menciona gestión de información se refiere a la gestión que se desarrolla en un Sistema de Información (si se trata de que el sistema tenga como propósito obtener salidas informacionales) y la define como: [...] el proceso mediante el cual se obtienen, despliegan o utilizan recursos básicos (económicos, físicos, humanos, materiales) para manejar información dentro y para la sociedad a la que sirve. Tiene como elemento básico la gestión del ciclo de vida de este recurso y ocurre en cualquier organización. Es propia también de unidades especializadas que manejan este recurso en forma intensiva, llamadas unidades de información (7).

De lo anteriormente planteado, el autor de esta investigación asume la posición de Woodman, ya que el gestionar la información de manera correcta y eficiente puede maximizar el valor y los beneficios derivados de su uso, minimizar el costo de adquisición y procesamiento, así como determinar responsabilidades para su uso efectivo, eficiente y económico, además de asegurar un suministro continuo de la información oportuna.

1.4 Sistemas de gestión de la información

Los sistemas de gestión de información constituyen la base de todos los demás sistemas, garantizan la disponibilidad de las herramientas y procedimientos organizativos necesarios para la generación, adquisición, procesamiento, almacenamiento, búsqueda, recuperación, transmisión y uso de la información interna y externa de interés para el trabajo de la organización (5).

Algunos autores como Davis y Olson conceptualizan los sistemas de gestión de información como un “sistema integrado y automatizado para proveer la información que sostenga las funciones de operatividad, gestión y toma de decisiones en una organización” (8).

Por otra parte Moreiro lo define como “el conjunto de políticas y normas relacionadas entre sí que se establecen para el acceso y tratamiento de los recursos de información, incluye los registros administrativos y los archivos, el soporte tecnológico de los recursos y el público a que se destina. En su evolución el sistema puede manejar la función de inteligencia corporativa y generar productos de inteligencia” (9).

Los autores cuando analizan en detalle las funciones de los sistemas de gestión de información, consideran que estos deben aprovechar al máximo sus recursos de información en función de la mejora continua y de la toma de decisiones organizacional a todos los niveles jerárquicos, desde la cúspide estratégica hasta la base operativa.

Es de interés para esta investigación analizar los sistemas de gestión de información de recursos humanos, debido a que se dificulta el acceso al estudio de sistemas de gestión de información relacionadas con datos militares por el estricto carácter de confidencialidad que mantienen todos los países con respecto a este tema. Además la solución que se propone se basa en la manipulación y control de datos personales que ayudan a gestionar la información referente a la estructura militar y las actividades de la defensa de cada efectivo del batallón de la Facultad 3 de la UCI.

1.4.1 Sistemas de gestión de información de recursos humanos

La Gestión de Información de Recursos Humanos, es la capacidad de mantener a la organización productiva, eficiente y eficaz, a partir del uso adecuado de su recurso humano (10). Por ello se define como Sistema Gestión de Información de Recursos Humanos (SGIRH), al sistema informático de gestión basado en un grupo de prácticas, técnicas y políticas en la integración y dirección de los recursos humanos para que consiga sus objetivos, desempeñando sus tareas de forma eficaz y eficiente (11).

A partir de lo antes planteado, se puede definir un SGIRH: como una herramienta informática que a partir del uso adecuado de la información almacenada de las personas, logre una gestión eficiente, en función del cumplimiento de los objetivos específicos de la organización. En el caso de la presente investigación sería: un sistema informático que a partir de las políticas, estatutos, indicaciones y documentos rectores del proceso de la defensa, realice la gestión eficiente de la información

referente a la estructura militar y las actividades de la defensa del Batallón Mixto de MTT Atípico de la Facultad 3 de la UCI.

Sistemas internacionales de gestión de información de recursos humanos

A continuación, se detalla un estudio sobre algunos sistemas internacionales de gestión de información asociados a los recursos humanos.

RRHH

La línea de productos de software RRHH brinda una solución integral y definitiva para las necesidades de las empresas que deseen optimizar los procesos de selección y búsqueda de personal para lograr rapidez, mayor efectividad, menores costos y mejorar la imagen empresarial.

Estos sistemas cuentan con fichas de perfil personal donde se recogen los datos de los empleados, sus evaluaciones y sanciones recibidas; lo que resulta de alto valor para contribuir a la toma de decisiones. Además de ello, cuenta con un sistema de puntos acumulativos para cada trabajador, ubicándolos en un ranking a nivel empresarial, útil para asignar nuevos puestos de trabajo u otorgar estimulaciones salariales (12).

Cezanne OnDemand

Los productos Cezanne proveen soluciones avanzadas de gestión del capital humano ayudando a las organizaciones a mejorar, gestionar, recompensar y retener sus recursos más importantes: las personas.

Cezanne OnDemand es un potente sistema de recursos humanos online, equipado con excelentes funcionalidades que reducen las tareas administrativas, dinamiza la gestión de las personas e involucra a los empleados de forma más productiva. Con una navegación intuitiva, pantallas fáciles de usar, robusta seguridad, sistema de alertas y notificaciones, flexibilidad en la elaboración de reportes y sistema de autoservicio integrado para directivos y empleados. Ofrece además un entorno ágil y cómodo para la gestión del personal en una empresa.

Este producto facilita la recopilación de información sobre los empleados. De forma centralizada, incluye datos personales y profesionales, historial laboral, habilidades y calificaciones, expedientes disciplinarios y formación profesional de cada persona vinculada al negocio. Reflejando toda la información sobre la gestión de las personas en el sistema de recursos humanos, de forma sencilla y de fácil acceso; muy similar a un perfil personal y psicológico.

Además, facilita el almacenamiento de documentos relevantes para la empresa, permitiendo subir todo tipo de archivos, como nóminas electrónicas o adjuntar documentos entre los que figuran los currículos y contratos de trabajo; e incluso, los propios empleados, pueden incorporar documentos como: partes de enfermedad o certificados de cursos completados. Los documentos se agrupan por categorías, pudiéndose crear diferentes, en función de las necesidades, facilitando su uso (13).

Sistema de Gestión de Personal Militar y Plantillas Orgánicas

Este sistema realizado por la Universidad Pontificia Comillas de Madrid, es una aplicación web válida para cualquier ejército en su país, ya sea de tierra o de aire. Está destinada a mejorar el sistema actual de la Jefatura de Personal y permite una mayor rapidez y comodidad mediante la gestión del personal militar. Mantiene un historial de la carrera militar y la trayectoria profesional de todo el personal militar.

La aplicación también permite realizar búsquedas de personal según determinados datos introducidos, facilita la realización de plantillas orgánicas según la unidad seleccionada, ya sea una fragata, una corbeta o un portaaviones.

El sistema lleva una gestión integral de la información del personal militar, en todo lo que se refiere a altas, bajas o modificaciones. Gestiona los ascensos en las diferentes categorías. Permite mostrar la estructura orgánica de las Fuerzas Armadas basada en la ordenación jerárquica de sus miembros, de forma que se logre la adecuada comprensión de los diferentes niveles de mando dentro de cada estructura.

Sistemas nacionales de gestión de información de recursos humanos

En el ámbito nacional, también se han desarrollado diversos sistemas asociados a la gestión de recursos humanos. Por lo que se describen brevemente a continuación los que más relación tienen con el propósito de esta investigación:

Sistema Integral de Gestión de Recursos Humanos (GREHU)

Desarrollada por el Centro de Estudios de Ingeniería en Sistemas (CEIS) de la Ciudad Universitaria José Antonio Echeverría (CUJAE), constituye una herramienta para la gestión integral de información de recursos humanos. Esta propone procedimientos y pautas a seguir para un mejor control y manipulación de los datos contenidos en el modelo principal del expediente laboral de los trabajadores, las actas de sanciones y medidas disciplinarias, entre otros, relacionados con la nueva concepción del Inventario de Personal.

Permite realizar preselección tanto de candidatos externos como de los trabajadores existentes según criterios preestablecidos para realizar promociones o nuevos contratos, basados en el grado de prioridad según el análisis de la información almacenada (14).

Sistema de Gestión Universitaria (XAUCE SGU)

El Sistema de Gestión Universitaria (SGU), desarrollado en el Centro de Informatización Universitaria (CENIA), está diseñado para ser una solución integral en la gestión de los procesos sustantivos de la universidad, incluyendo los recursos humanos asociados a los mismos.

El SGU gestiona información de las áreas de procesos: Pregrado, Postgrado, Desarrollo, Tecnologías, Investigación, Ingreso, Ubicación laboral, Residencia, Extensión universitaria, Cooperación Internacional, Biblioteca y Teleformación. Por lo que resulta de gran importancia para la universidad y mantiene alto nivel de satisfacción en la comunidad universitaria (15).

Después del estudio de los SGIRH antes mencionados, se concluye que ninguno se adecua a las especificidades del negocio inherente a la investigación, denotándose entonces la necesidad de desarrollar una solución para el problema planteado, referente a la gestión de la información de la estructura militar y las actividades de la defensa del Batallón Mixto de MTT Atípico de la Facultad 3 de la UCI.

Aunque es preciso reconocer que el estudio de dichas soluciones aportó elementos significativos a tomar en consideración para el desarrollo de la propuesta de solución como es el caso de:

- ✓ *Confección de fichas de perfil personal.* (Presente en los sistemas RRHH, Cezanne OnDemand, Sistema de Gestión de Personal Militar y Plantillas Orgánicas y GREHU).
- ✓ *Almacenamiento de documentos relevantes para la organización.* (Presente en el sistema Cezanne OnDemand).
- ✓ *Búsqueda de personal según determinados criterios.* (Presente en los sistemas RRHH, Sistema de Gestión de Personal Militar y Plantillas Orgánicas).
- ✓ *Mostrar la estructura orgánica basada en la ordenación jerárquica.* (Presente en el Sistema de Gestión de Personal Militar y Plantillas Orgánicas).
- ✓ *Arquitectura basada en módulos.* (Tomado como característica de los sistemas GREHU y XAUCE SGU).

Luego de conceptualizar los principales elementos del negocio y establecer un análisis de herramientas similares existentes, lo cual respalda la necesidad de implementar un sistema de gestión acorde a las necesidades. Se hace necesario decidir el enfoque de la solución, mediante el estudio de la factibilidad de implementación en caso de ser web o desktop, en consecuencia con las necesidades planteadas.

Características	Web	Desktop
Personalización, actualización y soporte.	Es suficiente con realizar los cambios en el servidor WEB.	Los cambios hay que realizarlos en cada PC donde se tenga la aplicación.
Accesibilidad y cobertura.	Cualquier lugar con acceso a la red.	Solo en la PC donde se haya instalado previamente la aplicación.
Capacidad de usuarios concurrentes.	Alta, debido a la arquitectura basada en servicios web.	Baja, ya que la forma de diseño es centrada en un único usuario local.
Portabilidad.	El sistema puede ser usado a través de un navegador, sin necesitar grandes prestaciones de recursos en la PC.	Puede presentar incompatibilidades con el sistema operativo, así como dependencia de los recursos que posea la PC.
Infraestructura y movilidad.	Solo se tiene que conectar a la red.	Está restringido a la ubicación de la PC local.
Seguridad eléctrica y lógica.	Es responsabilidad del proveedor de servicio.	Es responsabilidad de cada usuario que usa el sistema localmente.

Tabla 1: Comparación entre Aplicaciones Web y Desktop (16).

A partir de la comparación establecida anteriormente y en correspondencia con las necesidades existentes, entre las que resalta: la necesidad que la solución sea accesible desde distintas áreas de la universidad; el autor de este trabajo determina desarrollar la propuesta con enfoque web, lo cual brinda múltiples beneficios en cuanto a portabilidad, actualización y soporte e infraestructura.

Para proseguir, se hace necesaria la selección de la metodología de desarrollo de software, y las herramientas y tecnologías a emplear durante el desarrollo de la propuesta.

1.5 Metodología de desarrollo de software

Se puede definir como metodología de desarrollo de software: un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda en la construcción de un software (17). En la actualidad existen diversas metodologías de desarrollo de software para guiar a los equipos de trabajo en la creación de un nuevo software*, atendiendo a su complejidad* son separadas en dos grupos: metodologías tradicionales o formales y las ágiles o ligeras.

Las metodologías tradicionales se caracterizan por centrarse en llevar una documentación exhaustiva de todo el desarrollo, generando gran cantidad de artefactos. Además son resistentes a cualquier cambio en el proceso de desarrollo de software. Para la realización de todas las actividades definidas por estas, es necesario contar con un equipo de trabajo grande capaz de administrar un proceso complejo en varias etapas.

En aras de simplificar el trabajo y hacerlo más dinámico, surgen las metodologías ágiles. Las que se caracterizan por ser más orientadas a la inclusión del usuario como parte del equipo de desarrollo de software, el cual cuenta con pocos miembros. La formalidad y complejidad de los documentos disminuye, así como la cantidad de entregables. Dichas metodologías son más adaptables a los cambios durante el desarrollo y permiten que el equipo adapte la solución que se construye a nuevas especificidades que se soliciten.

A partir de lo descrito, teniendo en cuenta las necesidades y condiciones presentes para la realización de la propuesta de solución, donde:

- El equipo de desarrollo es pequeño.
- Los requisitos son altamente cambiantes.
- El tiempo de desarrollo es corto.
- El negocio no requiere grandes volúmenes de documentación.
- El cliente es parte del grupo de desarrollo.

Se decide seleccionar una metodología ágil de desarrollo de software. Entonces debe definirse cuál de las existentes, es la más ventajosa para el desarrollo de la propuesta de solución de este trabajo.

Entre las metodologías ágiles más populares de acuerdo a las comunidades internacionales de desarrollo de software se encuentra: XP (*Extreme Programming*) y SCRUM. Las cuales proveen un conjunto de buenas prácticas y beneficios al equipo de trabajo que las emplee. XP está más orientada al desarrollo, mientras que SCRUM se especializa en la gestión de los proyectos. Por lo que, con el objetivo de aumentar los beneficios de uso de dichas metodologías, en el 2008 se desarrolló un híbrido de ambas denominada SXP (18).

Para una mejor comprensión de los criterios anteriormente expuestos, se relaciona la comparación siguiente:

SCRUM	XP	SXP
Tiempo de duración de una iteración.		
Una semana o un mes (19).	Aproximadamente 2 meses (20).	Una semana o un mes (19).
Cambios en las iteraciones.		
Los equipos no permiten cambios en sus Sprint. Una vez que la reunión de planificación de iteración se ha completado y se ha contraído un compromiso con la entrega de un conjunto de elementos del *backlog* (lista de tareas identificadas por el equipo SCRUM durante la planificación del Sprint) del producto, estos se mantienen sin cambios hasta el final del Sprint (19).	Los equipos son más susceptibles al cambio dentro de sus iteraciones, siempre y cuando el equipo no ha empezado a trabajar en una característica particular (20).	No es posible introducir cambios durante el Sprint, por lo tanto para planificar su duración hay que pensar en cuánto tiempo se puede comprometer a mantener los cambios fuera del Sprint (19).
Prioridad de desarrollo.		
El propietario del producto SCRUM (cliente) prioriza la acumulación de productos, pero es el equipo quien determina la secuencia en la que se desarrollarán los elementos del *backlog* (18).	Los equipos trabajan en un orden de prioridad estricta determinado por el cliente (21).	Los equipos trabajan en un orden de prioridad estricta determinado por el cliente (19).
Prácticas de ingeniería		
No prescribe prácticas de ingeniería (18).	Establece prácticas de ingeniería como el desarrollo basado en pruebas, el enfoque en pruebas automatizadas, la programación en parejas, diseño simple y refactorización (21).	Establece prácticas de ingeniería como el desarrollo basado en pruebas, metáforas, la programación en parejas, diseño simple y refactorización (19).

Tabla 2: Comparación entre las metodologías SCRUM, XP y SXP (Elaboración propia).

Queda evidenciado que SXP toma los elementos particulares de SCRUM y XP, y los combina logrando mayor productividad, mayores resultados y una organización documental orientada al registro de los elementos más importantes del desarrollo de las soluciones sin generar una carga excesiva al equipo de trabajo. Lo que influye en que el equipo de desarrollo se mantenga unido, colaborativo y motivado por un desarrollo ágil, con objetivos claros y un seguimiento diario del progreso del desarrollo del software (18).

SXP consta de cuatro fases principales: Planificación-Definición donde se establece la concepción inicial del negocio y su visión. Se fijan además, las expectativas y se

realiza el aseguramiento del financiamiento del proyecto. En la segunda fase: Desarrollo es donde se realiza la implementación del sistema hasta que esté listo para ser entregado. La fase de Entrega constituye la puesta en marcha del sistema; y por último la fase de Mantenimiento, es donde se realiza el soporte para el cliente.

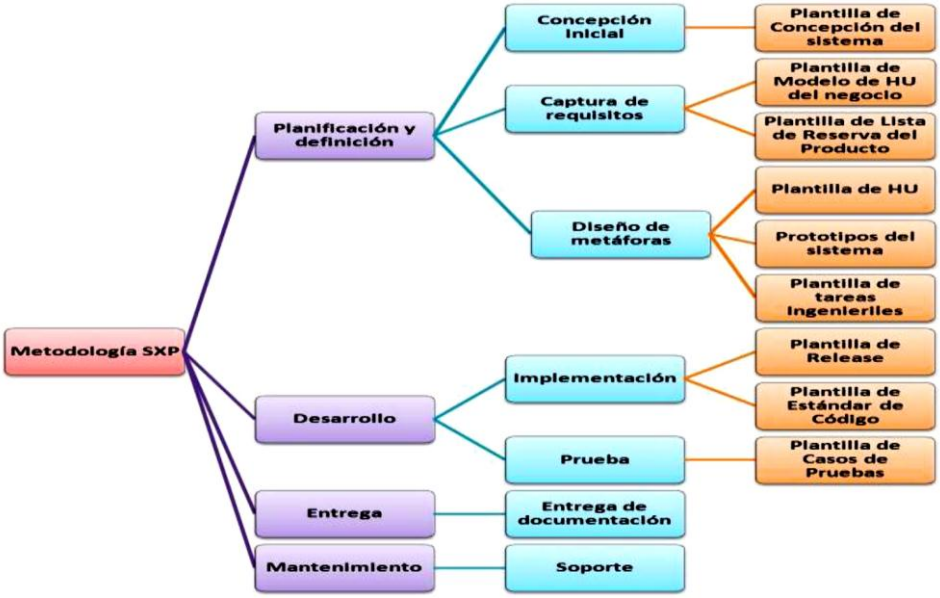


Figura 3: Esquema de la metodología SXP (Elaboración propia).

De cada una de ellas se despliegan siete flujos de trabajo: Concepción Inicial, Captura de Requisitos, Diseño de Metáforas, Implementación, Prueba, Entrega de la Documentación, Soporte e investigación, utilizándose este último por el equipo de desarrollo cuando sea necesario, es decir, es un flujo que se puede mover y utilizarlo en cualquier parte del ciclo de vida del proyecto.

1.6 Herramientas a utilizar en la propuesta de solución

Para definir las herramientas y tecnologías a utilizar durante el desarrollo de la propuesta, se realizó un estudio sobre las líneas de desarrollo utilizadas en el centro CEGEL (Centro de Gobierno Electrónico) para aplicaciones web de gestión. Con el fin de determinar cuál de las líneas se adecúa mejor a las necesidades y especificidades del problema planteado, se describen a continuación los principales elementos que las caracterizan, y las herramientas y tecnologías que proponen en cada caso.

En consecuencia con el proceso de reestructuración de la producción en la universidad que establece el Proceso de Mejora, como conclusión de la certificación del nivel II del Modelo CMMI (Modelo Integrado de Madurez de la Capacidades), el centro contiene una línea de desarrollo orientada a realización de aplicaciones web mediante la utilización del marco de trabajo Symfony2, con lenguaje PHP y otra línea orientada a aplicaciones desktop que utiliza la tecnología Java; aunque es necesario

aclarar que con la tecnología Java también se pueden realizar aplicaciones web mediante uso de marcos de trabajo como Spring[1].

Luego de la realización de una entrevista al Asesor de Tecnología, Seguridad y Arquitectura del centro, se logró definir la esencia de cada una de estas líneas de desarrollo, utilizadas actualmente por algunos proyectos de CEGEL; debido a que el documento oficial que establece formalmente su utilización se encuentra actualmente en elaboración. A continuación se muestran las tecnologías a utilizar que proponen cada una de estas líneas de desarrollo:

Herramientas y tecnologías	Línea de desarrollo 1	Línea de desarrollo 2
Lenguaje de Modelado.	UML 2.0.	UML 2.0.
Herramienta CASE.	Visual Paradigm 8.0 Enterprise Edition.	Visual Paradigm 8.0 Enterprise Edition.
Marcos de trabajo.	Symfony 2.3.7, Bootstrap.	Spring, GEFORT.
Lenguajes de programación.	PHP 5.3, JavaScript 1.8.5, CCS3, HTML5.	Java.
Entorno de Desarrollo Integrado (IDE).	NetBeans 7.3 (para PHP).	NetBeans 7.2 (para Java).
Servidor de aplicaciones web.	Apache 2.2.	GlassFish 3.1.2.
Sistema Gestor de Base de Datos.	PostgreSQL 9.3.	PostgreSQL 9.3.

Tabla 3: Herramientas y tecnologías de las líneas de desarrollo de CEGEL (elaboración propia).

Luego de analizar la propuesta de cada una de estas líneas, el autor de esta investigación decide acogerse a la línea de desarrollo para aplicaciones web que utiliza el marco de trabajo Symfony2, debido a la experiencia de usuario que se posee del uso de dicha línea para la creación de aplicaciones de este tipo. A continuación se realiza un análisis de las principales característica y beneficios que brindarán las herramientas y tecnologías seleccionadas en correspondencia con la línea de desarrollo seleccionada.

1.6.1 Lenguaje de Modelado UML 2.0

El Lenguaje Unificado de Modelado (UML por sus siglas en inglés *Unified Modeling Language*), es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software orientado a objetos. Tiene como objetivo principal, entregar un material de apoyo que permita definir diagramas propios, como también poder entender el modelamiento de diagramas ya existentes. Es un lenguaje estándar de modelado para software, para la visualización, especificación, construcción y documentación de los artefactos de sistemas en los que

[1] **Spring:** Marco de trabajo para el desarrollo de aplicaciones y contenedor de inversión de control, de código abierto para la plataforma Java.

el software juega un papel importante. Permite a los desarrolladores visualizar los resultados de su trabajo en esquemas o diagramas estandarizados (22).

Este lenguaje además, proporciona un conjunto de elementos de modelado, anotaciones, relaciones y normas que pueden aplicarse a una actividad de desarrollo de software. Sin embargo, UML también puede ser empleado para modelar otros dominios, tales como el modelado de sistemas y el modelado de negocio. (23)

Para facilitar el uso de este lenguaje de modelado se utiliza la herramienta Visual Paradigm lo que permite aprovechar las facilidades de uso y la amplia gama de utilidades que brinda, a continuación una descripción de esta herramienta.

1.6.2 Herramienta CASE (Computer Aided Software Environment)

Las herramientas CASE comprenden un conjunto de programas de diferentes tipos empleados para ayudar a las actividades del proceso del software como el análisis de requisitos, el modelado de sistemas, la depuración y las pruebas (20).

Permiten incrementar la productividad, comunicarse de manera más eficiente con los usuarios, integrar las actividades y proporcionar continuidad de una fase a la siguiente durante todo el ciclo de vida del desarrollo de sistemas e integrar el trabajo que desempeñan en el sistema, educiendo de esta forma el costo en términos de tiempo, recursos y dinero del proceso de desarrollo del software (24).

Visual Paradigm 8.0 Enterprise Edition.

Es una de las herramientas UML CASE del mercado, considerada como muy completa y fácil de usar, con soporte multiplataforma y que proporciona excelente facilidades de interoperabilidad con otras aplicaciones.

Fue creada para el ciclo vital completo del desarrollo de software que lo automatiza y acelera, lo que permite la captura de requisitos, análisis, diseño e implementación. Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la definición de base de datos a partir del esquema de las clases.

Permite invertir código fuente de programas, archivos ejecutables y binarios en modelos UML al instante, creando de manera simple toda la documentación. Está diseñada para usuarios interesados en sistemas de software de gran escala con el uso del acercamiento orientado a objeto, además apoya los estándares más recientes de las notaciones de Java y de UML. Incorpora el soporte para trabajo en equipo, que

permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros (25).

1.6.3 Herramienta para modelado de prototipado web

Para el modelado de los prototipos web se decide emplear Axure RP Pro 5.5 pues, permite crear prototipos navegables, con mejor calidad visual. Además la herramienta brinda un conjunto de librerías muy atractivas y la posibilidad de crearlas nuevas para satisfacer las necesidades del modelado.

Permite además, generar prototipos de diseño HTML, lo que permite la interacción de cualquier miembro del equipo de desarrollo o las partes interesadas con estos sin necesidad de instalar Axure. De esta forma se puede comprobar si el sistema de navegación de la aplicación web es el deseado y además el cliente podrá tener una idea clara y precisa de cómo será la aplicación final. (26)

Luego de la presentación de esta herramienta CASE para la representación visual de los principales componentes del desarrollo del software y para el modelado de prototipado web el Axure RP 6.5; se da paso a las tecnologías y herramientas propuestas por la línea de desarrollo, lo que establece el uso del marco de trabajo Symfony2 con el lenguaje de programación PHP, como servidor de aplicaciones Apache 2.2, Gestor de Base de Datos PostgreSQL 9.3 y como IDE de desarrollo el NetBeans en su versión 7.3.

1.6.4 Marco de Trabajo

Según Jorge Naula, un *framework*[2], es: "… *una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación*". Es decir; un *marco de trabajo* web se puede definir como un conjunto de componentes (por ejemplo: clases, descriptores y archivos de configuración en XML[3]) que componen un diseño reutilizable que facilita y agiliza el desarrollo de sistemas web y es precisamente el que se utiliza en estos estudios (27).

Entre los objetivos principales de un *marco de trabajo* se encuentran: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo, como el uso de patrones.

[2] ***Framework*:** La traducción de *framework* (o en plural *frameworks*) en idioma español es marco de trabajo.
[3] **XML:** Siglas en inglés de e***X****tensible* ***M****arkup* ***L****anguage* ('lenguaje de marcas extensible').

Symfony 2.3.7

Este marco de trabajo fue diseñado con el objetivo de optimizar la creación de las aplicaciones web con el uso de sus características. Puede ser utilizado en plataformas Unix, Linux y Windows; y está desarrollado en lenguaje PHP. Posee además una librería de clases que permite reducir el tiempo de desarrollo; pero requiere de instalación, configuración y utilización de líneas de comando.

Este marco de trabajo contiene abundante documentación y está diseñado para trabajar de forma modular en bundles[4] y ha sido ideado para aprovechar todas las nuevas características de PHP 5.3. Entre sus ventajas se puede mencionar que incorpora el patrón Modelo-Vista-Controlador (MVC)[5], soporta AJAX[6], plantillas y conexión y trabajo con la bases de datos con la integración del ORM Doctrine (28).

ORM Doctrine 2

Doctrine 2 es el mapeador de objetos relacionales (ORM) para PHP 5.3.0+ que provee una persistencia transparente para los objetos PHP. Este usa el patrón de mapeo de datos, lo que proporciona una completa separación entre la lógica del dominio/negocio de la persistencia en un sistema de gestión de bases de datos relacionales.

El beneficio de Doctrine para los programadores es la habilidad de concentrarse únicamente en la lógica de negocio orientada a objetos y preocuparse de la persistencia solo como una tarea secundaria. Lo cual no significa que la persistencia no sea importante para Doctrine 2, sin embargo se sostiene el criterio de que existen considerables beneficios para la programación orientada a objetos si la persistencia y las entidades son perfectamente separadas (29).

Twig 2.1

Twig es un motor de plantillas, integrado con Symfony2 para crear las plantillas de la aplicación, las cuales tienen una sintaxis concisa, fáciles de leer y de escribir. La característica más importante de Twig es que está implementado con herencia entre plantillas, lo que permite crear un “esqueleto” base que contiene todos los rasgos comunes de las interfaces ahorrando así código y tiempo en el trabajo. Twig se caracteriza por ser rápido, seguro y flexible:

Rápido: Twig compila las plantillas hasta código PHP regular optimizado. El costo general en comparación con código PHP regular se ha reducido al mínimo.

[4] **Bundles**: Conjunto de archivos que implementan una única funcionalidad.
[5] **MVC:** Siglas de Modelo Vista Controlador. Es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación.
[6] **AJAX:** Acrónimo de ***A****synchronous* ***J****avaScript* ***A****nd* ***X****ML* (JavaScript asíncrono y XML).

Seguro: Twig tiene un modo de recinto de seguridad para evaluar el código de plantilla que no es confiable. Esto permite utilizar Twig como un lenguaje de plantillas para aplicaciones donde los usuarios pueden modificar el diseño de la plantilla.

Flexible: Twig es alimentado por flexibles analizadores léxico y sintáctico. Esto permite al desarrollador definir sus propias etiquetas y filtros personalizados (30).

Twitter Bootstrap 2.1

Twitter Bootstrap es una colección de herramientas de software libre para la creación de sitios y aplicaciones web el cual es compatible con gran parte de los navegadores web. Bootstrap 2.1 es un *framework* diseñado para simplificar el proceso de creación de diseños web. Para ello ofrece una serie de plantillas CSS y de ficheros JavaScript, los cuales permiten obtener:

- ✓ Interfaces que funcionen de forma perfecta en los navegadores actuales y correctamente en los no tan actuales.
- ✓ Un diseño que pueda ser visualizado de forma correcta en distintos dispositivos y a distintas escalas y resoluciones.
- ✓ Una mejor integración con las bibliotecas que se suelen usar habitualmente, como por ejemplo jQuery.
- ✓ Un diseño sólido basado en herramientas actuales y potentes (31).

1.6.5 Lenguaje de programación

Un lenguaje de programación es un lenguaje artificial que se utiliza para expresar programas de ordenador. Está formado por un conjunto de símbolos, palabras claves utilizables y por reglas gramaticales para construir sentencias sintáctica y semánticamente correctas (32).

A continuación se hace un análisis de las características de los lenguajes de programación a utilizar en la solución.

Lenguaje de programación del lado del servidor

PHP 5.3

PHP es un lenguaje interpretado de alto nivel, multiplataforma, embebido en páginas HTML (*Hyper Text Markup Language*, Lenguaje para el Formato de Documentos de Hipertexto). La característica más potente y destacable de este lenguaje es su soporte para una gran cantidad de bases de datos, además de otros servicios que usen protocolos como IMAP[7], SNMP[8], NNTP[9], POP3[10], HTTP y derivados.

[7] **IMAP:** Protocolo de Acceso a Mensajes Electrónicos, por sus siglas en inglés.
[8] **SNMP:** Protocolo Simple de Administración de Red, por sus siglas en inglés.

La mayoría de su sintaxis es similar a C, Java y Perl; y la meta de este lenguaje es permitir escribir a los desarrolladores web, páginas dinámicas de una manera rápida y fácil (33).

Además posee gran capacidad de procesamiento de texto, donde se incluyen las Expresiones Regulares Compatibles de Perl, muchas extensiones, y herramientas para el acceso y análisis de documentos XML (Lenguaje de Marcado Extensible, por sus siglas en inglés). También es potente, de alto rendimiento, de fácil aprendizaje y de escaso consumo de recursos (34).

Lenguajes de programación del lado del cliente

JavaScript 1.8.5

JavaScript es uno de los lenguajes de script u orientado a documento, creado por la empresa Netscape, para añadir interactividad a las páginas web. Es compacto y basado en objetos, diseñado para el desarrollo de aplicaciones cliente-servidor (35).

Es un lenguaje interpretado e independiente de plataforma que permite incluir macros en páginas web. Estas macros se ejecutan del lado del cliente y no en el servidor. La característica de JavaScript que más simplifica la programación es que, aunque el lenguaje soporta varios tipos de datos, no es necesario declarar el tipo de las variables, argumentos de funciones ni valores de retorno de las funciones. El tipo de las variables cambia implícitamente cuando es necesario, lo que ayuda a programar con rapidez macros sencillas.

La principal ventaja de JavaScript es que eso sucede sin ningún tipo de transmisión de datos a través de Internet, de tal forma que cuando un usuario escribe algo en un formulario, no es necesario que sea enviado al servidor, verificado y devuelto (36).

HTML 5

HTML, siglas de HyperText Markup Language (lenguaje de marcado de hipertexto), en su versión 5, es usado para describir la estructura y el contenido en forma de texto. Este puede describir hasta un cierto punto, la apariencia de un documento. Además se ha convertido en el formato más fácil para la creación de páginas web debido a su sencillez y no hay que compilar el código para ver si funciona. Se puede ver en forma inmediata el resultado del trabajo y también es usado para complementar el texto con objetos tales como imágenes (37).

[9] **NNTP:** Protocolo para la Transmisión de Noticias en Red, por sus siglas en inglés.
[10] **POP3**: Protocolo de Oficina de Correo u Oficina Postal.

Este lenguaje se ha elegido por sus innumerables características, de las cuales se destacan que es un lenguaje estático para el desarrollo de sitios web que permite describir hipertexto presentando el texto de forma estructurada y agradable. Se utiliza para definir texto, tablas, y otros elementos que forman parte del diseño de la página web.

CSS 3

Las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML por sus siglas en inglés de eXtensible Markup Language (lenguaje de marcas extensible) y por extensión en XHTML (por sus siglas del inglés eXtensible HyperText Markup Language). Permite separar la estructura de un documento de su presentación. La información de estilo puede ser adjuntada tanto como un documento separado o en el mismo documento HTML (38).

Las ventajas de utilizar CSS es que tienen un control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo. Por otro lado los navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad ya que una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario. Facilitando que el documento HTML en sí mismo sea más claro de entender y se consiga reducir considerablemente su tamaño (37).

1.6.6 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) es un conjunto de programas que administran y gestionan la información contenida en una base de datos. Ayuda a realizar la definición de los datos, el mantenimiento de la integridad de estos dentro de la base de datos, el control de su seguridad y privacidad; así como la manipulación de los mismos (39).

PostgreSQL 9.3

Es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo la licencia BSD y con su código fuente disponible libremente. Utiliza un modelo cliente/servidor y usa multiprocesos para garantizar la estabilidad del sistema ya que un fallo en uno de los procesos no afecta el resto y el sistema continúa con su funcionamiento. Funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema (40).

1.6.7 Servidor web

Un servidor web es un programa que sirve datos en forma de páginas web, hipertextos o páginas HTML: textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonidos. La comunicación de estos datos entre cliente y servidor se hace por medio del protocolo HTTP. Con esto, un servidor Web se mantiene a la espera de peticiones HTTP, que son ejecutadas por un cliente HTTP; lo que solemos conocer como un navegador web (41).

Apache 2.2

El Servidor Apache 2.2 es un servidor web de código abierto para plataformas Unix, GNU/Linux, Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1. Entre sus principales ventajas se encuentran que es modular, código abierto, multiplataforma, extensible y fácil de proveer soporte (42).

1.6.8 Entorno de Desarrollo Integrado

Un Entorno de Desarrollo Integrado (*Integrated Development Environment*) es un programa informático compuesto por un conjunto de herramientas de programación que les permite a los programadores escribir, compilar, depurar y ejecutar programas (43).

NetBeans 7.3

Es un producto libre y gratuito sin restricciones de uso, teniendo el código fuente disponible para su reutilización bajo las licencias CDDL (Licencia Común de Desarrollo y Distribución) y la GPL (General Public License). Facilita la programación proporcionando el auto-completamiento y la inserción automática de código, una descripción sencilla sobre la estructura del programa, opciones de refactorización, depuración, entre otros.

Netbeans permite la reutilización de módulos así como el control de versiones a través de la integración con Subversion[11]. Permite también crear aplicaciones web con PHP 5 y además viene con soporte para el marco de trabajo Symfony2 (44).

[11] **Subversion**: Herramienta de control de versiones basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros.

1.7 Conclusiones Parciales

El análisis de los principales conceptos asociados al dominio del problema facilitó una mejor comprensión del negocio. Además el establecimiento del estado del arte permitió obtener las tendencias actuales de las aplicaciones y sistemas de gestión de información de recursos humanos tanto nacionales como internacionales. Lo que ratificó la necesidad de crear una solución para el problema planteado e incluir y adaptar algunas de las funcionalidades estudiadas de dichos sistemas.

Luego del estudio de factibilidad entre aplicaciones web y desktop ante el problema plateado, se decidió desarrollar la solución de tipo web. Para ello se realizó un estudio sobre las metodologías de software existentes, lo que concluyó con la selección de la metodología ágil SXP, debido a los beneficios que brinda al combinar los elementos de SCRUM para la gestión eficiente del proyecto con las facilidades de XP pata la ingeniería de software.

Por otra parte, se realizó una comparación entre las líneas de desarrollo del centro CEGEL, donde se logró seleccionar como línea de desarrollo a seguir: la línea para aplicaciones web que propone una serie de herramientas y tecnologías con la utilización del marco de trabajo Symfony2. Además se realizó un análisis detallado de las herramientas y tecnologías propuestas para lograr una mejor comprensión de cada una de ellas.

CAPÍTULO 2
DESCRIPCIÓN DE LA APLICACIÓN

En el presente capítulo se hace una descripción de la propuesta de solución a través de los requisitos funcionales y no funcionales, así como las particularidades del diseño. Se muestran aspectos esenciales de la arquitectura del sistema y los patrones de diseño empleados. También se hace alusión a los artefactos inherentes a la fase Planificación y Definición.

2.1 Descripción de los procesos del negocio

Para lograr una total comprensión del negocio; a continuación se hace una breve descripción de las características y el funcionamiento de cada uno de los principales procesos que involucran:

- ✓ *Emplantillamiento del personal:* el proceso de emplantillamiento no es más que la recopilación y agrupación de todos los datos personales y de la defensa de cada efectivo del batallón. El emplantillamiento se realiza en el Área de Atención Militar UCI, donde se reciben de la Secretaría General los datos actualizados de cada facultad, y a partir de ello, realiza el emplantillamiento del personal nuevo y actualiza los registros ya existentes. Este proceso es conducido por el especialista del área.
- ✓ *Ubicación del personal:* la ubicación del personal dentro de la estructura del batallón, es tarea del personal del VDER de la facultad en coordinación con el Área de Atención Militar. El vicedecanato luego de recibir las indicaciones pertinentes para el proceso, realiza la ubicación del personal de la facultad, a partir de las necesidades existentes y en correspondencia con el resultado del análisis de los datos personales en cada caso.
- ✓ *Planificación, ejecución y evaluación de las actividades de la defensa:* el Plan de Actividades del Batallón se diseña a partir del Plan de Actividades de la Defensa UCI emitido por la DSPD. En la facultad, cada actividad es el resultado de un trabajo colaborativo ente el VDER, profesores pertenecientes a la Cátedra de PPD y la máxima dirección del batallón (Plana Mayor). Luego de lograr un diseño de actividades acorde con las misiones y tareas del batallón, para asegurar la realización correcta y una buena participación, es responsabilidad de la Plana Mayor, a todos sus niveles, notificar y asegurar la calidad de las mismas. El último paso de este proceso es la evaluación de los resultados, para ello el VDER elabora

un informe sobre la actividad donde se resumen de las principales observaciones y emite una evaluación.

- ✓ *Controlar grupos especiales:* se denominan como grupos especiales a la Brigada de Respuesta Rápida y el Plan de Localización y Aviso; ambas estructuras funcionan paralelas al batallón donde cumplen tareas y misiones específicas ante cualquier evento adverso. El control del personal y la correcta manipulación y confidencialidad de los datos correspondientes a estos grupos, es función del VDER de la facultad. Del conjunto de datos que se gestiona, resultan de vital importancia los datos de contacto y aviso en cada caso.

2.2 Propuesta de solución

Teniendo en cuenta los objetivos que persigue el presente trabajo de diploma y en correspondencia con los procesos descritos en el epígrafe anterior, se propone como solución desarrollar un sistema informático que mejore la gestión de información referente a la defensa de la Facultad 3 de la UCI; para ello la solución propuesta debe ser capaz de:

- Crear y almacenar fichas de información personal, donde se aglutinan todos los datos de los efectivos del batallón.
- Estructurar de manera jerárquica la composición del batallón, lo que posibilita la asignación de misiones y tareas específicas.
- Diseñar y evaluar actividades dirigidas a cualquier estructura dentro del batallón.
- Controlar de forma eficiente la información referente a los grupos especiales, manteniendo un alto nivel de confidencialidad y discreción.
- Generar reportes informativos, para favorecer la divulgación de los resultados del proceso.
- Contener una pequeña biblioteca virtual, donde se encuentren disponibles los principales documentos rectores de las actividades de la defensa.

2.2.1 Concepción inicial del sistema

En la primera fase de SXP se genera la plantilla Concepción del Sistema, en ella se especifican los aspectos generales organizativos y de concepción del sistema, su objetivo, principales involucrados y otros aspectos que permitan la posterior organización del desarrollo del proyecto (Anexo 6).

2.3 Captura de Requisitos

La ingeniería de requisitos es el conjunto de actividades implicadas en descubrir, documentar y mantener un conjunto de requisitos del software a desarrollar. La

captura de los mismos es un proceso en el cual los datos son extraídos a partir de la aplicación de técnicas de captura de información (45).

Las técnicas aplicadas fueron la entrevista y la tormenta de ideas. Para las cuales el cliente fue incluido como parte del equipo de desarrollo, aportando y corrigiendo las ideas preliminares sobre las funcionalidades identificadas. También se realizó un procedimiento arqueológico documental a los archivos históricos almacenados, para identificar coincidencias en el funcionamiento de la estructura militar del Batallón Mixto de MTT Atípico de la Facultad 3.

2.3.1 Requisitos funcionales

Los requisitos funcionales son aquellos que indican lo que debe hacer el producto, son las capacidades con las que debe cumplir el mismo. Como resultado de la aplicación de las técnicas de captura de información se obtuvieron 42 requisitos funcionales (RF) recogidos en el Anexo 7. Estos requisitos fueron agrupados por objetivos según el estándar para la Agrupación de Requisitos Funcionales (ARF) de la IEEE 830:

- **ARF1-Batallón:** en esta agrupación se encuentran las funcionalidades relacionadas con la administración del batallón y su estructura. De manera que se pueda crear la estructura del batallón, listar su personal, mostrar y gestionar estructuras dentro de las Unidades de Aseguramiento Material y Unidades de Aseguramiento Defensivo; mostrar la estructura del batallón mediante un esquema lógico y filtrar la información según criterios determinados.
- **ARF2-Personal:** esta agrupación le permite al usuario crear, modificar y eliminar la información de un estudiante (cadete/civil) o de un trabajador (interno/externo), hacer búsquedas avanzadas de personas y mostrar su perfil. También brinda la posibilidad de listar las personas que han causado baja en el sistema.
- **ARF3-Actividades:** en esta agrupación el usuario puede mostrar, planificar, modificar y eliminar las actividades. Así como realizar búsquedas avanzadas y mostrar los participantes de una actividad determinada.
- **ARF4-Grupos Especiales:** el usuario puede en esta agrupación mostrar y gestionar el personal de los grupos especiales (Brigada de Respuesta Rápida y Plan de Localización y Aviso), así como, emitir avisos y alarmas en situaciones determinadas.
- **ARF5-Reportes:** la agrupación de reportes permite al usuario generar reportes informativos sobre la estructura del batallón, perfil personal, información del personal de sus unidades y Plana Mayor; así como reportes informativos sobre las actividades, grupos especiales y bajas del sistema.

- **ARF6-Documentación:** esta agrupación le permite al usuario consultar y gestionar la documentación referente a la defensa.
- **ARF7-Administración:** esta última agrupación le permite al administrador general gestionar los usuarios de la aplicación, los cuales a partir del rol que desempeñen se les otorgarán privilegios para trabajar sobre los módulos antes descritos.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales (RNF) se refieren a las características o cualidades que debe tener el producto para hacerlo atractivo, usable, rápido y confiable. Para el desarrollo de la aplicación web propuesta se definieron 40 RNF, los que se clasificaron en RNF de usabilidad, confiabilidad, eficiencia, soporte, restricciones de diseño, de interfaz, de licencia, estándares aplicables y seguridad (Ver Anexo 7).

2.3.3 Lista de Reserva del Producto (LRP)

La LRP, es una colección organizada y priorizada de los requisitos del producto que se describen como funcionalidades que el sistema debe cumplir en su desarrollo y especifica las cualidades requeridas por el software. Además de determinar el orden en que se le irá dando cumplimiento a cada requisito recogido según la prioridad establecida en cada uno.

La plantilla LRP (adjunta en el Anexo 8), contiene de cada requisito funcional el número identificativo, una breve descripción, la estimación puntual (EP) y los roles que realizaron la estimación. La tabla siguiente es un fragmento de la LRP, para una mejor comprensión de la misma:

Prioridad				
Ítems *	**Requisito**	**Descripción**	**Estimación**	**Estimado por:**
Muy Alta				
RF#1	Mostrar listado del Batallón.	Permite mostrar el listado del personal de todo el Batallón, agrupado por unidades y la Plana Mayor por separado.	3	Programador
RF#4	Añadir Unidades de Aseguramiento Combativo.	Permite añadir una nueva Unidad de Aseguramiento Combativo.	3	Programador
RF#5	Eliminar Unidad de Aseguramiento Combativo.	Permite eliminar una Unidad de Aseguramiento Combativo.	3	Programador
RF#7	Añadir Unidades de Aseguramiento Material.	Permite añadir una nueva Unidad de Aseguramiento Material.	3	Programador
RF#8	Eliminar Unidad	Permite eliminar una Unidad de	3	Programador

	de Aseguramiento Material.	Aseguramiento Material.		
RF#10	Mostrar estructura del Batallón.	Permite mostrar un esquema que refleja la estructura organizativa de la estructura del Batallón.	4	Programador
RF# 41	Autenticar usuario.	Permite autenticar un usuario en la aplicación con su usuario y contraseña en el dominio uci.	1	Programador
RF#42	Cargar información de los estudiantes y profesores.	Cargar la información personal de los estudiantes y profesores.	2	Programador

Tabla 4: Fragmento de la Plantilla LRP (Prioridad: Muy Alta).

2.4 Diseño de metáforas

A partir de la definición de las funcionalidades descritas en la LRP es posible establecer el Diseño de Metáforas, mediante las historias de usuarios, prototipos del sistema y las tareas ingenieriles que permiten su desarrollo siendo estas las actividades que se realizan y se describen a continuación.

2.4.1 Historias de usuarios del negocio

En la plantilla del modelo de historias de usuarios se describen los actores y trabajadores del negocio, además se presenta un diagrama de historias de usuarios del negocio que permite ver la relación entre los usuarios y las actividades que se realizan, la misma está descrita en el Anexo 9.

Las Historias de Usuario (HU) son equivalentes a los casos de uso en el proceso unificado y constituyen la base para las pruebas funcionales. Básicamente una historia de usuario es una lista priorizada de requisitos o funcionalidades, descritas usando la terminología del cliente. En la siguiente tabla se puede apreciar el listado de las mismas organizadas según las ARF:

ARF	Historias de Usuarios	Sprint[12]
Batallón	HU_3: Mostrar personal del Batallón.	1
	HU_4: Mostrar estructura del Batallón.	1
	HU_5: Modificar estructura del Batallón.	1
Personal	HU_2: Mostrar perfil personal.	2
	HU_6: Administrar personal.	2
	HU_7: Buscar personal.	2
	HU_8: Reubicar personal.	2
	HU_9: Mostrar bajas.	2
Actividades	HU_10: Administrar actividad.	2
	HU_11: Asignar personal a actividades.	2
	HU_12: Mostrar Plan de Actividades.	2

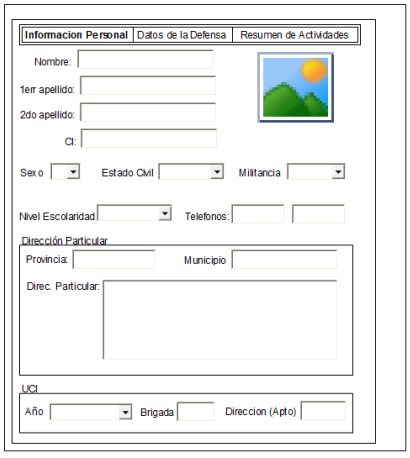
[12] **Sprint:** Terminología para definir las iteraciones en la metodología SXP.

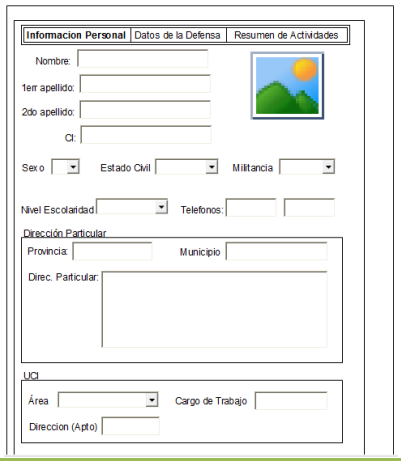
Grupos Especiales	HU_13: Mostrar Brigada de Respuesta Rápida.	3
	HU_14: Modificar Brigada de Respuesta Rápida.	3
	HU_15: Mostrar Plan de Localización y Aviso.	3
	HU_16: Modificar Plan de Localización y Aviso.	3
Reportes	HU_17: Realizar reporte del personal del Batallón.	3
	HU_18: Realizar reporte de Perfil Personal.	3
	HU_19: Realizar reporte de Actividades.	3
	HU_20: Realizar reporte de Grupos Especiales.	3
Documentación	HU_21: Administrar documento.	3
	HU_22: Mostrar listado de documentos.	3
Administración	HU_1: Autenticar Usuario.	1
	HU_23: Administrar usuario.	1
	HU_24: Administrar rol.	1
	HU_25: Administrar permiso.	1

Tabla 5: Distribución de HU por ARF.

A continuación se ilustra la historia de usuario HU_2: Mostrar Perfil Personal

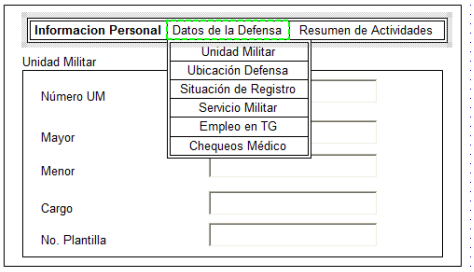
Historia de Usuario	
Número: *HU_2*	**Nombre Historia de Usuario**: Mostrar perfil personal
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: Addiel Aguila Espinaco.	**Iteración Asignada**: Sprint 1.
Prioridad en Negocio: Alta.	**Puntos Estimados**: 2.
Riesgo del Desarrollo: Alto.	**Puntos Reales**: 1.
Descripción: El usuario una vez autenticado en la aplicación, pasa directamente a su página de perfil de información personal. El perfil cuenta con tres secciones donde se agrupan los datos de cada persona: ✓ *Información Personal:* Es donde aparecen todos los datos personales de cada usuarios, tales como nombre, apellidos, CI, dirección particular, militancia. ✓ *Datos de la Defensa:* Aparecen los datos correspondientes a la defensa tales como datos de la unidad militar a la que pertenecen, la situación de registro que presenta; así como datos sobre cumplimiento de servicio militar y chequeos médicos. ✓ *Resumen de Actividades:* Muestra un resumen de todas las actividades de la defensa en las que ha participado la persona, de las cuales se conoce el nombre de la actividad, la fecha de realización y la evaluación en dicha actividad.	
Prototipo de Interface: - Información Personal. Vista para Estudiantes - Trabajadores:	



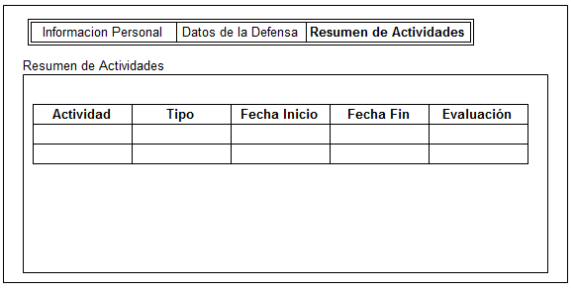


correspondiente a la ARF2-Personal. El resto de las HU con su descripción y prototipo están contenidas en el Anexo 10.

- Datos de la Defensa: Unidad Militar.



- Resumen de Actividades.



✓ **RF#16:** Mostrar perfil personal.

Tabla 6: HU_2: Mostrar Perfil Personal.

2.4.2 Tareas de la Ingeniería

Luego de la descripción de las historias de usuarios del sistema, se hace necesario la definición un grupo de tareas ingenieriles dirigidas a dar cumplimiento a la implementación de las funcionalidades definidas en ellas, lo cual sienta las bases para el posterior desarrollo de la solución. Por ejemplo, las tareas relacionadas con la HU_2: Mostrar perfil personal, mostrada anteriormente:

Tareas de la Ingeniería	
Número Tarea: T_6	**Número Historia de Usuario**: HU_2.
Nombre Tarea: Diseñar las interfaces requeridas para la funcionalidad Mostrar perfil personal.	
Tipo Tarea: Desarrollo.	**Puntos Estimados**: 1.
Fecha Inicio: 15 – febrero - 2014	**Fecha Fin**: 18 – febrero - 2014
Programador Responsable: Addiel Aguila Espinaco.	
Descripción: Se crean las interfaces necesarias, teniendo en cuenta los prototipos definidos en la Historia de Usuario.	

Tabla 7: T_6: Diseñar las interfaces requeridas para la funcionalidad Mostrar perfil personal.

Tareas de la Ingeniería	
Número Tarea: T_7	**Número Historia de Usuario**: HU_2.
Nombre Tarea: Implementar funcionalidad: Mostrar perfil personal.	
Tipo Tarea: Desarrollo.	**Puntos Estimados**: 1.
Fecha Inicio: 18– febrero - 2014	**Fecha Fin**: 24- febrero- 2014
Programador Responsable: Addiel Aguila Espinaco.	
Descripción: Se crean las clases y algoritmos necesarios para la implementación de la funcionalidad de autenticación de usuario.	

Tabla 8: T_7: Implementar funcionalidad: Mostrar perfil personal.

El resto de las tablas que describen las Tareas de la Ingeniería están recogidas en el Anexo 11.

2.5 Arquitectura

La Arquitectura de Software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, su conducta, y las formas en que interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la mayor parte de las abstracciones (46).

El desarrollo de la solución define, a partir del marco de trabajo Symfony2, una arquitectura basada en cuatro capas: (Capa de Presentación, la Capa de Negocio, la Capa de Acceso a Datos y la Capa de Datos), con el uso del patrón arquitectónico **Modelo-Vista-Controlador (MVC),** muy beneficioso en aplicaciones que manejan gran cantidad de datos y transacciones complejas, donde se requiere una mejor separación de conceptos facilitando la programación en diferentes capas de manera paralela e independiente (47).

MVC sugiere la separación del software en 3 estratos: Modelo, Vista y Controlador, que serán explicados brevemente:

- ✓ **Modelo:** Es la representación de la información que maneja la aplicación. Básicamente son los datos puros que puestos en contexto del sistema proveen de información al usuario a la aplicación misma (47).
- ✓ **Vista:** Es la representación del modelo en forma gráfica disponible para la interacción con el usuario. En el caso de una aplicación web, la “vista” es una página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones (47).
- ✓ **Controlador:** Es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando el modelo en caso de ser necesario (47).

La arquitectura propuesta, además posee la característica de tener varios complementos transversales a las capas para garantizar seguridad, tratamiento de excepciones, entre otros aspectos. Cada uno de los módulos o bundles se estructuran arquitectónicamente igual. Las entidades del dominio son accesibles en la sub-capa servidor de la Capa de Presentación, la Capa de Negocio y la Capa de Acceso a Datos (Ver Figura 4).

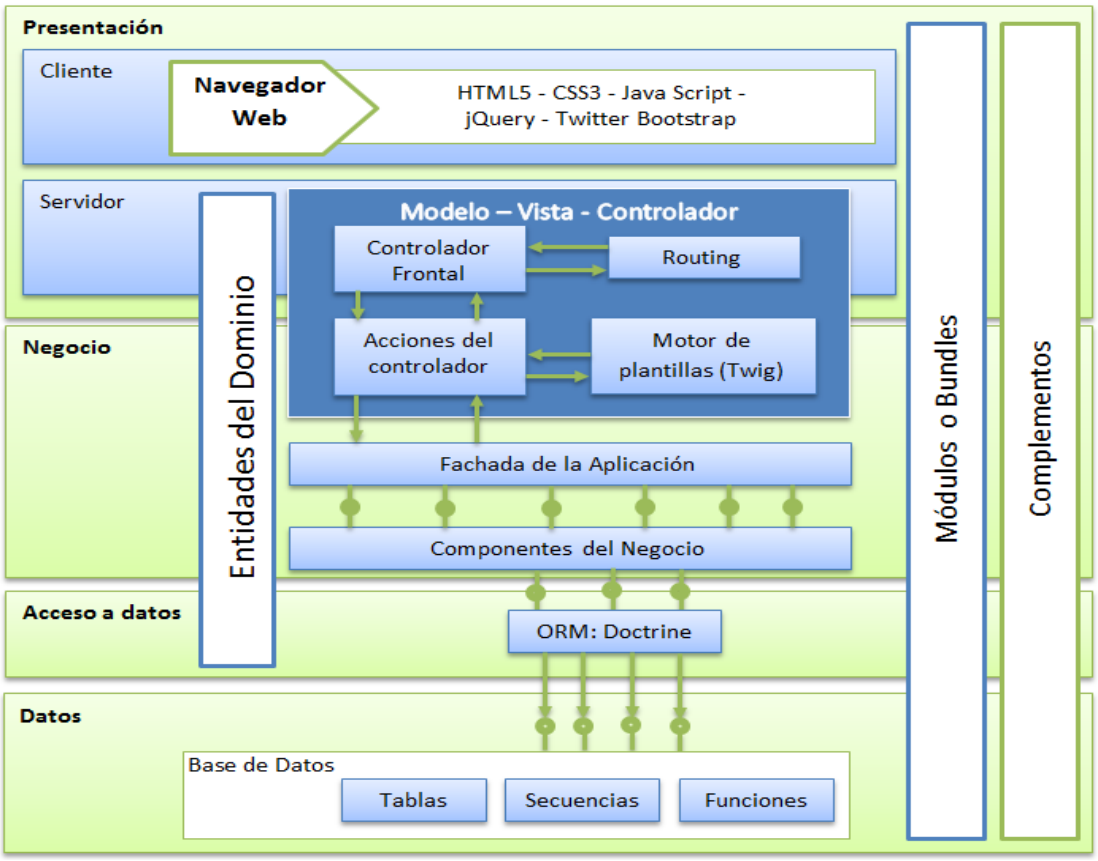


Figura 4: Esquema de la arquitectura (Elaboración propia) **(47)**.

- ✓ **Capa de Presentación:** contiene los componentes con los que el usuario va a interactuar (páginas). Permite alcanzar las funcionalidades que brinda el controlador y mostrar o capturar la información a través de los diferentes elementos que comprende, es decir: *twigs*, formularios, *grids*, entre otros.

  Se divide en dos sub-capas: cliente y servidor. En la primera se visualiza mediante el navegador web (utilizando tecnologías como HTML5, CSS3, JavaScript, Twitter-Bootstrap y JQuery) datos procesados. En la segunda se maneja la lógica de control así como la construcción de páginas y formularios (48).
- ✓ **Capa de Negocio:** Su diseño depende directamente del negocio específico al que se refiera cada sistema. Esta capa recibe una petición del nivel superior y gestiona o procesa la misma. Finalmente envía una respuesta continuando el proceso en el punto donde se inició dicha petición. Contiene las clases gestoras, encargadas del manejo de la lógica de negocio. Dichas clases son gestionadas mediante una fachada de aplicación que desacopla los componentes del negocio de las clases controladoras (48).
- ✓ **Capa de Acceso a datos:** Contiene las entidades y repositorios que mapea Doctrine como marco de trabajo para la comunicación con el servidor de datos. Gestiona las peticiones de la capa de negocio consultando la BD y retornando los

datos que se recuperan al gestor correspondiente. Es importante destacar que las entidades del dominio se encuentran en esta capa pero son accesibles además desde las capas superiores (48).

- ✓ **Capa de datos:** En esta capa se encuentra el gestor de base de datos PostgreSQL y en él un conjunto de esquemas, tablas, vistas y procedimientos almacenados que permiten persistir la información con la que trabaja la aplicación y manejar su almacenamiento (48).
  - ✓ **Complementos:** Los Complementos son un grupo de facilidades y mejoras que permiten lograr una arquitectura más flexible y adaptable tales como la gestión de seguridad, las validaciones, la mensajería y la configuración así como tratamiento de excepciones y otras (48).
  - ✓ **Módulos (Bundles):** Cada módulo constituye una estructura de carpetas que se organizan según la arquitectura que se describe en la figura 5. Permiten utilizar funcionalidades construidas por terceros o empaquetar sus propias funcionalidades para distribuirlas y reutilizarlas (48).

2.6 Patrones de diseño

En términos generales, un patrón es un conjunto de información que proporciona respuesta a un grupo de problemas similares, es decir, un patrón es una solución a un problema en un contexto (49).

Además del uso del patrón arquitectónico MVC, en la solución propuesta se emplean distintos tipo de patrones de diseño como son: los patrones de diseño de base de datos, de asignación de responsabilidad y de comportamiento; los cuales se describen a continuación:

Patrones de diseño de base de datos:

- ✓ **Llaves subrogadas:** El uso de este patrón es bastante generalizado en la base de datos pues la mayoría de las entidades contienen una llave única entera auto-incremental, lo que facilita reducir el costo de las búsquedas en la base de datos.

  EL uso de este patrón constituye una protección ante los cambios debido a que la lógica del negocio no está en las llaves y evita la contención (bloqueo) de la base de datos, debido a la rapidez de los mecanismos de generación que provee el sistema.

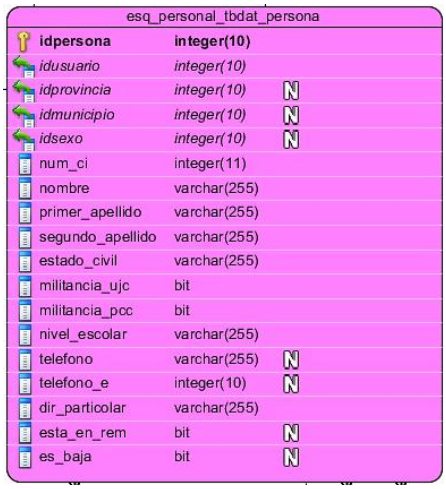


Figura 5: Uso del patrón Llaves Subrogadas en la entidad Persona.

- ✓ **Control de Acceso Basado en Roles (RBAC, por sus siglas en inglés):** este patrón se implementa mediante la asignación de roles a los diferentes usuarios de la aplicación. Cada rol posee permisos para realizar determinadas funciones de acuerdo a las restricciones del negocio. Por lo que cada usuario tiene estrictamente los permisos que les son conferidos a través del rol que desempeña.

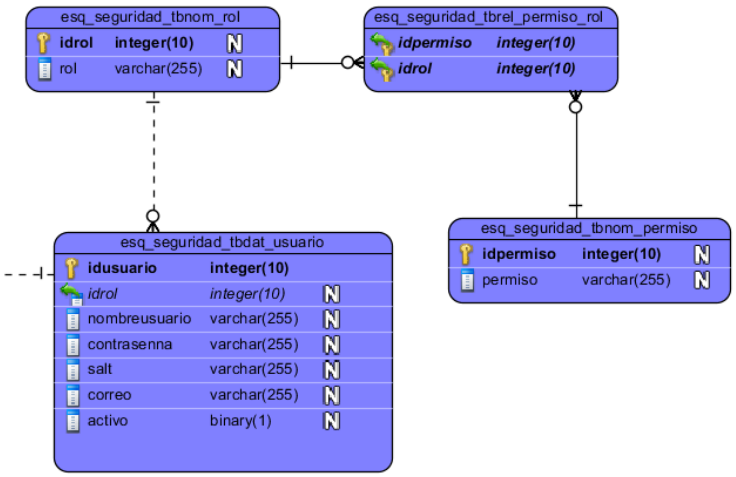


Figura 6: Empleo del patrón Control de Acceso Basado en Roles (RBAC).

Patrones de Diseño de Asignación de Responsabilidades o GRASP (Patrones Generales de Software para Asignar Responsabilidades):

- ✓ **Experto:** este patrón es aplicado en todas las clases debido a que cada una de ellas es experta en gestionar la información que les compete. Lo que facilita el entendimiento, extensión y mantenimiento del sistema. Además de conservar el encapsulamiento y contribuir al bajo acoplamiento (49). A continuación se muestra la clase *Actividad* la cual es experta en información referente a una actividad.

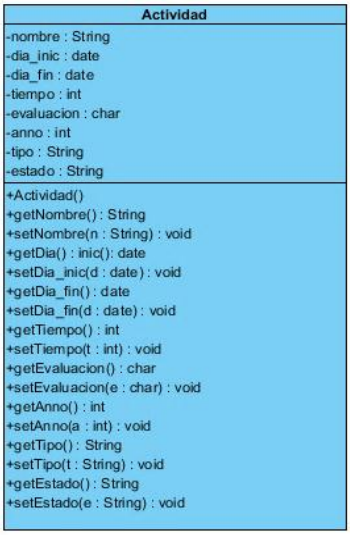


Figura 7: Empleo del patrón Experto.

- ✓ **Creador:** el patrón se evidencia en las clases controladoras que, para cada uno de los módulos o funcionalidades de la aplicación, son las encargadas de crear las instancias de los objetos que manejan, favoreciendo así la reutilización y el bajo acoplamiento (48). A continuación se muestra el uso del patrón en la clase controladora *PersonaController.php,* perteneciente al *PersonalBundle.*

```
public function newCadeteAction() {
    $renderDir = 'PersonalBundle:persona:newCadete.html.twig';
    $entity = new EstudianteCadete();
    $tarjeta = new TarjetaUbicacionEspecial();

    $form = $this->createForm(new EstudianteCadeteType(), $entity);
    $formTarjeta = $this->createForm(new TarjetaUbicacionEspecialType(), $tarjeta);

    return $this->render($renderDir, array(
                'form' => $form->createView(),
                'formTarjeta' => $formTarjeta->createView(),
    ));
}
```

Figura 8: Evidencia del patrón Creador en la clase *PersonaController.php*.

- ✓ **Bajo Acoplamiento:** se proporciona un bajo acoplamiento en el diseño debido a que las clases existentes tienen asignadas responsabilidades de tal forma que estas no dependan en gran medida de otras, permitiendo de esta forma tener sistemas más robustos y de fáciles de mantener (49).

  Esta característica permitió potenciar la reutilización y disminuyó la dependencia entre las clases. Se evidencia en las clases *TrabajadorInterno, TrabajadorExterno que* heredan solamente de *Trabajador;* de igual forma sucede con las clases *EstudianteCadete* y *EstudianteCivil* que heredan de Estudiante; y a su vez las clases *Trabajador* y *Estudiante* heredan de la clase *Persona* que es una clase

estable en cuanto a su implementación; lo que garantiza un bajo acoplamiento entre las clases.

Figura 9: Evidencia del patrón Bajo Acoplamiento en las clases: *TrabajadorInterno, TrabajadorExterno, Trabajador y Persona*.

- ✓ **Alta Cohesión:** el patrón se evidencia debido a que a cada una de las clases se le asignaron responsabilidades de tal forma, que estén estrechamente relacionadas entre sí y no realicen un trabajo excesivo (49).

  El patrón se evidencia en cada clase que realiza una labor única dentro del sistema y colabora con las otras para llevar a cabo una tarea, básicamente en las clases que forman parte de las capas controladora y modelo. Como es el caso de la clase *PersonalController*.

```
use CORE\SeguridadBundle\Anotaciones\PermisoSeguridad;
use CORE\CoreBundle\Controller\CoreController;
use DEFENSA\PersonalBundle\Entity\EstudianteCadete;
use DEFENSA\PersonalBundle\Entity\EstudianteCivil;
use DEFENSA\PersonalBundle\Form\EstudianteCadeteType;
use DEFENSA\UbicacionBundle\Entity\TarjetaUbicacionEspecial;
use DEFENSA\UbicacionBundle\Form\TarjetaUbicacionEspecialType;
use DEFENSA\UbicacionBundle\Entity\TarjetaUbicacionDefensa;
use DEFENSA\UbicacionBundle\Form\TarjetaUbicacionDefensaType;
use DEFENSA\PersonalBundle\Form\EstudianteCivilType;
use DEFENSA\PersonalBundle\Entity\TrabajadorExterno;
use DEFENSA\PersonalBundle\Form\TrabajadorExternoType;
use DEFENSA\PersonalBundle\Entity\TrabajadorInterno;
use DEFENSA\PersonalBundle\Form\TrabajadorInternoType;
use CORE\SeguridadBundle\Entity\Usuario;
use CORE\AdminBundle\Form\UsuarioRejistroType;

/**
 */
class PersonaController extends CoreController {

    private $nameService = 'defensa.personal.business.persona';
    private $indexRouting = 'persona_buscar';
```

Figura 10: Evidencia del patrón Alta Cohesión en la clase *PersonalController*.

Patrones de Comportamiento (Patrones GOF):

- ✓ **Observador**: Cuando uno de los objetos cambia su estado, notifica este cambio a todos los dependientes. Un complemento debe ser capaz de agregar métodos, o hacer algo antes o después de ejecutar un método, sin interferir con otros complementos (49).

- ✓ **Decorador:** aplicado a la generación de vistas, la solución que ofrece dicho patrón es la de añadir funcionalidad adicional a las plantillas. Por ejemplo, añadir el menú y el pie de página a las plantillas que lo requieran, se trata de decorar las plantillas con elementos adicionales reutilizables. El sistema de plantillas *twig*, está provisto de un mecanismo de herencia gracias al cual la decoración de plantillas resulta de una flexibilidad y versatilidad total (49).
- ✓ **Fachada:** es una interfaz que actúa como mediadora entre dos capas en la aplicación. Este patrón se empleó en las clases controladoras que actúan como intermediarias entre el modelo y las restantes capas.

2.7 Diagrama de paquetes

El propósito del diagrama es el de representar los paquetes fundamentales del sistema, que agrupan las funcionalidades y permiten una mejor estructuración y comunicación entre clases.

Como parte del artefacto Modelo de Diseño; se generan el diagrama de paquetes y el diagrama clases de la solución propuesta. En la Figura 11 se evidencia la existencia de dos grandes paquetes: el Core que aglutina 3 bundles (AdminBundle, Seguridad Bundle y CoreBundle); y el paquete Defensa contiene los 5 bundles (BatallonBundle, PersonalBundle, UbicacionBundle, ActividadBundle y DocumentacionBundle).

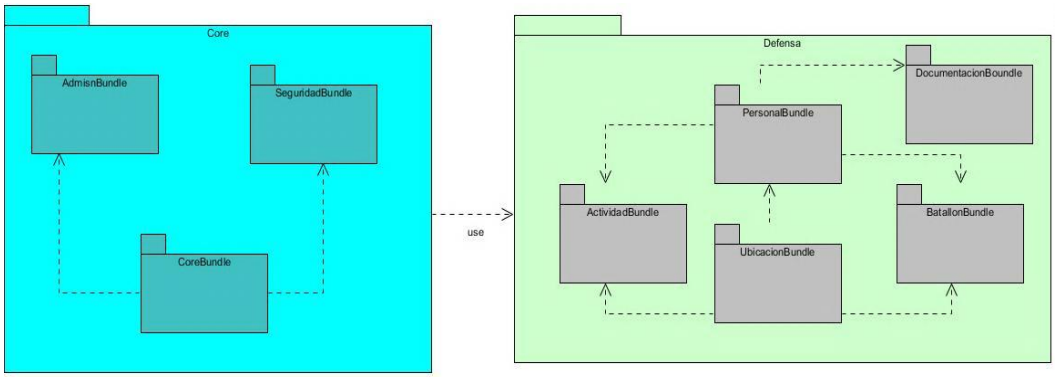


Figura 11: Diagrama paquetes del sistema.

Dentro de estos bundles existe una distribución de sub-paquetes que representan la arquitectura propuesta para el desarrollo del sistema (Patrón arquitectónico Modelo-Vista-Controlador) y el sub-paquete Entity, el cual encapsula las clases entidades del sistema. Además se muestran las relaciones existentes entre las clases y los paquetes. Las figuras 12 y 13 muestran el Diagrama de Clases correspondiente al paquete Core y el bundle PersonalBundle respectivamente.

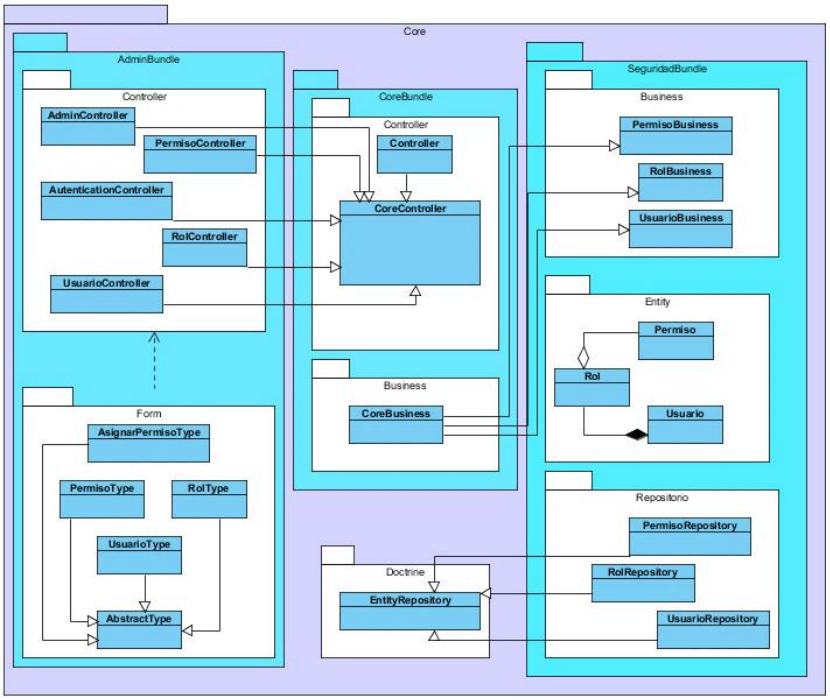


Figura 12: Diagrama de Clases del paquete Core.

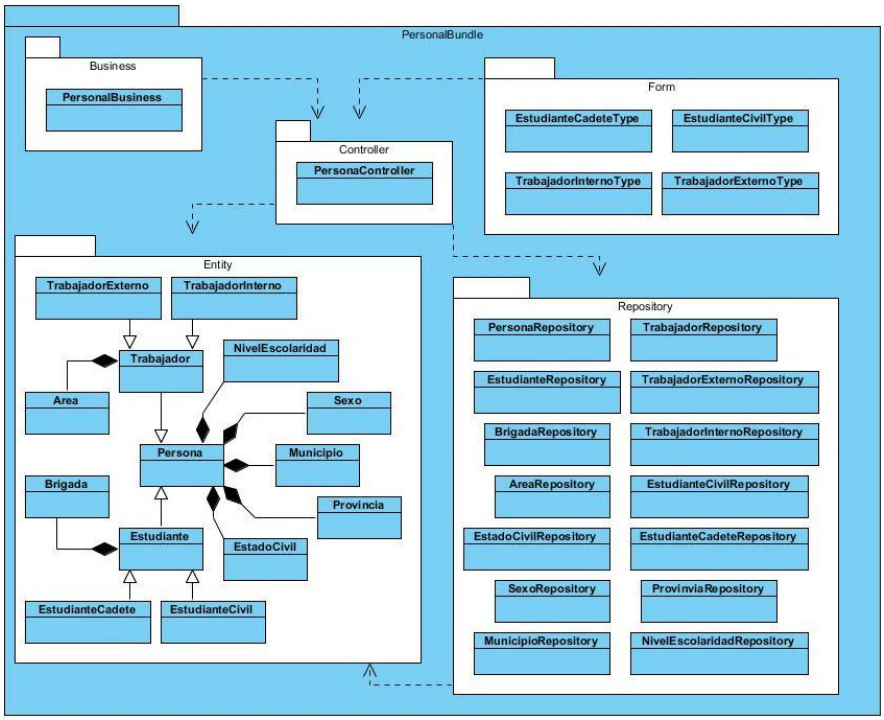


Figura 13: Diagrama de Clases del bundle PersonaBundle del paquete Defensa.

2.8 Modelo de datos

El modelo de datos es un conjunto de conceptos que sirven para describir la estructura, semántica y las relaciones existen entre las entidades de una base de datos importantes para el funcionamiento del negocio y muestra los datos que serán contenidos en el sistema (50).

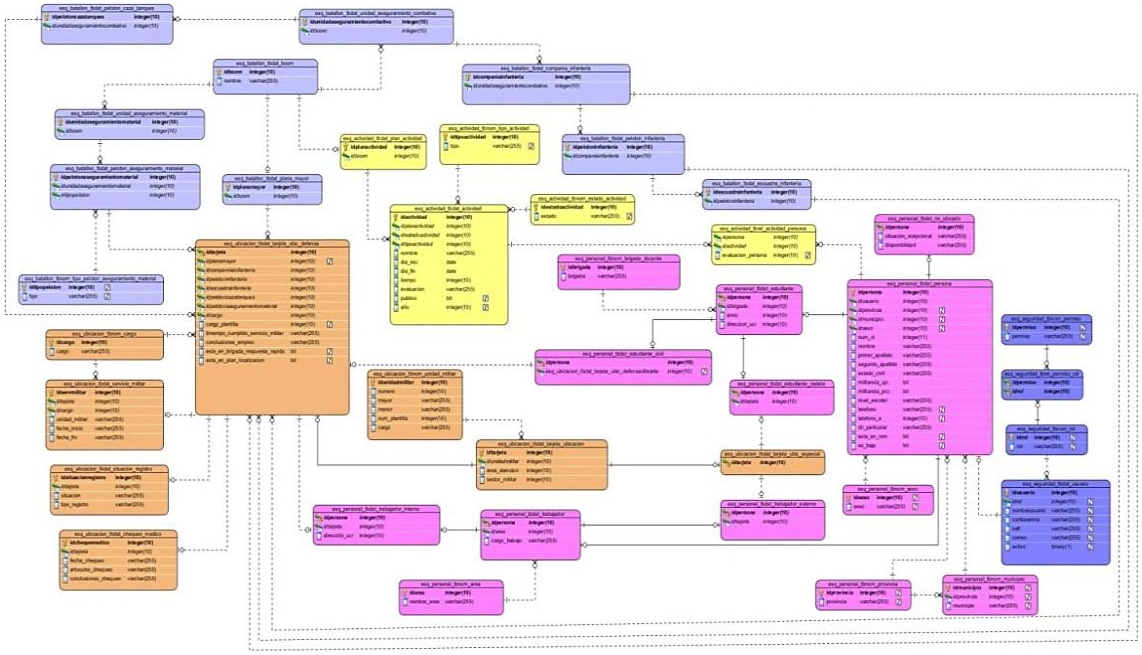


Figura 14: Modelo de datos.

El modelo de datos obtenido cuenta con un total de 45 tablas donde 12 de ellas son nomencladores encargados de gestionar conceptos específicos del negocio ya predefinidos, por ejemplo la tabla *esq_actividad_tbnom_tipo_actividad*, que se relaciona con la tabla *esq_actividad_tbdat_actividad* y se definen los tipos de actividades que existirán en el sistema (Postgrado, Ejercicio Práctico, REM, Meteoros, Día de la Defensa, Ejercicio Teórico, Otros).

Las tablas existentes en el modelo se encuentran divididas por áreas lógicas dentro del negocio donde se le asocia a cada color un concepto determinado (Ver Anexo 12), donde:

- ✓ **Violeta claro:** representa la estructura organizativa del batallón.
- ✓ **Amarillo:** tablas referentes a las actividades.
- ✓ **Naranja claro:** representa la parte del negocio referente a las tarjetas de ubicación y sus diferentes tipos.
- ✓ **Rosado:** las tablas referentes a la información del personal.
- ✓ **Azul oscuro:** tablas referentes a la seguridad del sistema.

2.9 Conclusiones Parciales

En el presente capítulo se realizó la descripción de la aplicación mediante el uso de la metodología seleccionada en su primera etapa (*Planificación y Definición)*.

Durante la concepción inicial del sistema se realizó una descripción detallada de los principales procesos del negocio y se presentó la propuesta de solución que sugiere la investigación, basada en el problema y el estado del arte establecido en el Capítulo 1. El flujo de trabajo generó la plantilla de Concepción Inicial del Sistema, la cual reúne los aspectos más significativos de la aplicación a desarrollar, entre los que figuran la misión y visión del sistema, así como las herramientas seleccionadas.

Como parte de la realización del segundo flujo (Captura de Requisitos), se aplicaron técnicas de ingeniería de requisitos para su captura. Fueron definidos un total de 42 RF, agrupados en 7 ARF por objetivo bajo el estándar de la IEE 830, mientras que los 40 RNF definidos; fueros descritos y agrupados por categorías. Además, se generó el artefacto LRP, en el cual se agruparon los RF según la estimación de su complejidad por parte del desarrollador.

En el Diseño de Metáfora, se realizó un estudio sobre todas las HU definidas como punto de partida para el diseño y realización de los prototipos del sistema. Lo que se concretó con la elaboración de las Plantillas de Historias de Usuario, que aglutinan su descripción, junto al prototipo y los requisitos a los que responde la HU. A partir de las HU se definieron las TI orientadas a guiar el posterior desarrollo de la solución y sentar las bases para las restantes fases de la metodología.

Luego se realizó un análisis sobre la arquitectura a la cual responde el sistema, resaltando el uso del patrón arquitectónico MVC, y el comportamiento de los componentes que interactúan de forma transversales a las capas para garantizar seguridad, tratamiento de excepciones, entre otros aspectos. Luego, se realizó un análisis sobre el uso de los patrones de diseño utilizados durante el desarrollo de la aplicación.

Se realizó el diagrama de paquetes con el propósito de representar los paquetes fundamentales del sistema, que agrupan las funcionalidades y permiten una mejor estructuración y comunicación entre clases. Por último, para describir la estructura, semántica y las relaciones existen entre las entidades de una base de datos se realizó el Modelo de Datos de la Aplicación, con la descripción detallada de cada una de sus áreas.

CAPÍTULO 3
IMPLEMENTACIÓN Y VALIDACIÓN

En la realización de este capítulo se describe el proceso de implementación de la aplicación en términos de componentes; donde se detalla mediante el diagrama de despliegue como quedará distribuida la aplicación. Se muestran los resultados obtenidos de la aplicación de las métricas y técnicas empleadas para validar los requisitos y el diseño del sistema; además de los resultados alcanzados luego de la realización las pruebas de funcionalidad y unitarias. De ello, se generan los principales artefactos definidos por la metodología, correspondientes a la fase de Desarrollo, en sus dos flujos de trabajo (Implementación y Prueba).

3.1 Implementación

La implementación constituye una de las fases más importantes del desarrollo de software. En ella se toman como punto de partida los resultados obtenidos en el diseño, implementándose el sistema en términos de componentes como ficheros de código binario, código fuente, scripts y ejecutables. Su importancia reside en que se obtiene como consecuencia un sistema ejecutable, siendo esto uno de los principales objetivos en el desarrollo de software (51).

3.1.1 Diagrama de componentes

Un diagrama de componentes representa la separación de un sistema de software en componentes físicos (por ejemplo archivos, módulos, paquetes, base de datos, código fuente, binario o ejecutable etc.) y muestra las dependencias y organización existente entre estos componentes (51).

En el diagrama de componentes de la Figura 15, se evidencian las interacciones entre el los componentes de Symfony2 y los del sistema SGiD. Symfony2 cuenta con los componentes *Security* (para la seguridad), *Routing* (para las rutas de acceso), *Config* (para las configuraciones), *Mensajería* (para la gestión de mensajes informativos y de errores), *Excepciones* (para el manejo de las excepciones generadas) y *Ctr_Frontal* es el controlador frontal que recibe cada petición realizada al sistema, entre otros. Estos componentes permiten un adecuado manejo de la seguridad, las configuraciones, las excepciones y errores del subsistema.

Mientras que el sistema SGiD está compuesto por tres paquetes fundamentales, Controlador (encargado de dar respuesta a las peticiones de los usuarios), Modelo

(contiene las clases entidades de la herramienta y lo relacionado con la lógica de negocio), Vista (encargado de la construcción y manejo de las interfaces de usuario).

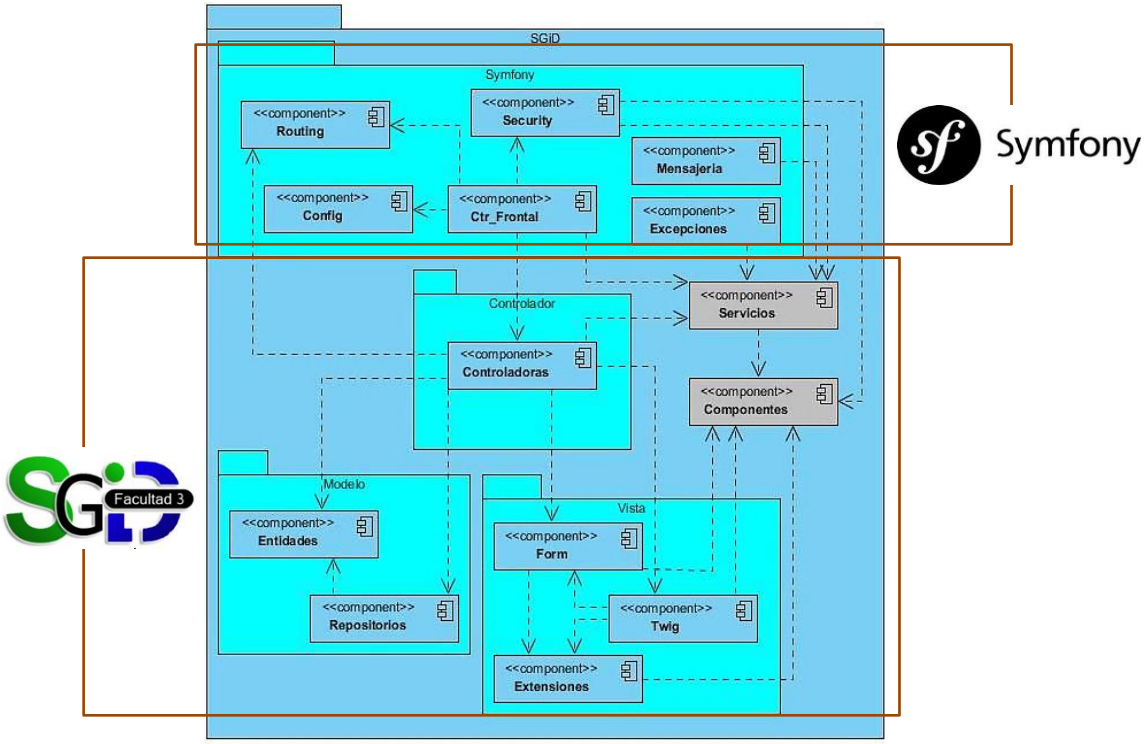


Figura 15: Diagrama de componentes.

El paquete Controlador contiene el componente Controladoras, responsable de dar respuesta a las peticiones realizadas por los usuarios, posee una relación con el componente Routing para el manejo de las rutas definidas para el módulo y el componente Servicios que ofrece los servicios públicos.

El componente Controladoras se relaciona, además, con el componente Entidades, que a su vez es usado por el componente Repositorio el cual contiene las clases repositorios encargadas de realizar las consultas a la base de datos. El paquete Vista incluye los componentes Form (encargado del trabajo con los formularios), Twig (para las interfaces de usuarios y plantillas del módulo) y Extensiones (posee los JavaScript y CSS).

De igual forma, estos componentes pueden ser agrupados y organizados atendiendo a la arquitectura del sistema para lograr un mejor entendimiento de sus funciones, distribución e interacciones entre sí. En la Figura 16, se muestran el diagrama de componentes a partir de la arquitectura.

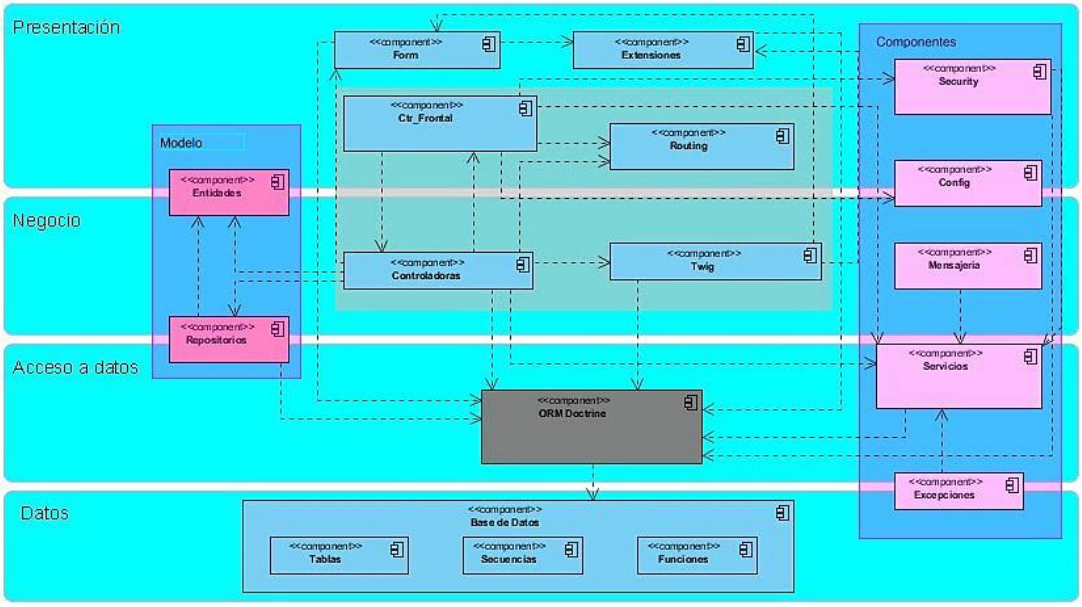


Figura 16: Diagrama de componentes en función de la arquitectura.

3.1.2 Diagrama de despliegue

Los diagramas de despliegue proveen una vista de cómo los componentes se despliegan en el transcurso de la infraestructura del sistema. Permite comprender la correspondencia entre la arquitectura de software y la de hardware.

El siguiente diagrama de despliegue muestra la forma en que se van a comunicar los componentes del sistema. La PC cliente (usuario) se va a comunicar por el protocolo HTTP[13] para utilizar las funcionalidades que se encuentran en el Servidor web. Este a su vez se comunica con el Servidor de Base de Datos por medio del protocolo TCP/IP[14] para la obtención de la información que maneja el sistema y al Servidor que provee los servicios de LDAP, también con el protocolo TCP/IP.

Además de ello, para el despliegue del sistema a implementar se tiene conectada al Servidor web por el protocolo HTTP la PC_Cliente (Admin) la cual cuenta con una impresora con conexión USB donde se podrá obtener de forma física los reportes e información generada en la aplicación.

[13] HTTP: Protocolo de transferencia de hipertexto.
[14] TCP/IP: Protocolo de Control de Transmisión.

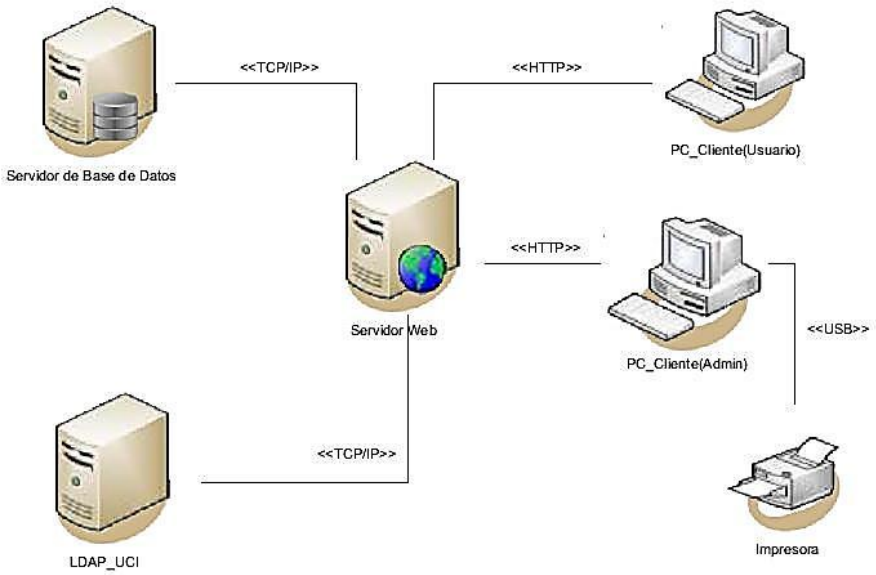


Figura 17: Diagrama de despliegue.

3.1.3 Plan de Releases

Entre los artefactos definidos por las metodologías SXP para el flujo de trabajo Implementación, de la fase de Desarrollo, se encuentra el Plan de Releases; donde se describen cada una de las iteraciones en que se divide la implementación del sistema, especificando en cada caso la duración aproximada y el orden a realizar las historias de usuario correspondientes.

El Plan de Releases del sistema SGiD, cuenta con 3 iteraciones para su implementación, donde previo a cada iteración se realiza una reunión de planificación de iteración para determinar en qué funcionalidad del producto trabajará el equipo, teniendo como resultado el Plan de Releases que se muestra a continuación.

Release	Descripción de la Iteración	Orden de HU a implementar	Duración
1	Implementación de los módulos Seguridad y Batallón.	HU_1: Autenticar Usuario.	30 días.
		HU_23: Gestionar usuario.	
		HU_24: Gestionar rol.	
		HU_25: Gestionar permiso.	
		HU_3: Mostrar personal del Batallón.	
		HU_4: Mostrar estructura del Batallón.	
		HU_5: Modificar estructura del Batallón.	
2	Implementación de los módulos Personal y Actividades.	HU_2: Mostrar perfil personal.	40 días.
		HU_6: Gestionar personal.	
		HU_7: Buscar personal.	
		HU_8: Reubicar personal.	
		HU_9: Mostrar bajas.	
		HU_10: Gestionar actividad.	
		HU_11: Asignar personal a actividades.	
		HU_12: Mostrar Plan de Actividades.	

3	Implementación de los módulos Grupos Especiales, Reporte y Documentación.	HU_13: Mostrar Brigada de Respuesta Rápida.	30 días.
		HU_14: Modificar Brigada de Respuesta Rápida.	
		HU_15: Mostrar Plan de Localización y Aviso.	
		HU_16: Modificar Plan de Localización y Aviso.	
		HU_17: Realizar reporte del personal del Batallón.	
		HU_18: Realizar reporte de Perfil Personal.	
		HU_19: Realizar reporte de Actividades.	
		HU_20: Realizar reporte de Grupos Especiales.	
		HU_21: Gestionar documento.	
		HU_22: Mostrar listado de documentos.	

Tabla 9: Plan de Releases.

3.1.4 Estándar de codificación empleado

Con el objetivo de lograr una estandarización en la programación de la aplicación web se decide aplicar el estándar de codificación *CamelCase*, específicamente la variante *lowerCamelCase*. El cual facilitará la lectura, comprensión y mantenimiento del código.

A continuación se describe a grandes rasgos las convenciones de nomenclatura.

General:

- ✓ Se exceptúan el uso de las tildes y la letra ñ, la que será sustituida por nn.
- ✓ En todo momento se utilizarán nombres que sean claros, concretos y libres de ambigüedades. Ejemplo: "idpersona" y no solamente "id".
- ✓ El nombre de todas las variables y métodos comenzarán con letra minúscula y si este está compuesto por varias palabras, todas las palabras internas que lo componen comienzan con mayúscula. Ejemplo: “buscarPersonasUbicar()”.

Identación:

- ✓ El contenido siempre se identará con *tabs*, nunca utilizando espacios en blanco.

Clases:

- ✓ El nombre de las clases comenzará con mayúscula si este está compuesto por varias palabras, todas las palabras internas que lo componen comienzan con mayúscula también. Ejemplo: “Persona”.
- ✓ Intentar mantener los nombres de las clases descriptivos y simples. Usar palabras completas, evitar acrónimos y abreviaturas, a no ser que la

abreviatura sea mucho más conocida que el nombre completo, como URL o HTML.

Nombre de variables:

- ✓ No se utilizarán nombres de variables que puedan ser ambiguos.
- ✓ Se procurará evitar dar nombres sin sentido a variables temporales. Por ejemplo: temp, i, tmp.
- ✓ Las variables booleanas deben tener nombres que sugieran respuestas o contenidos de tipo Sí/No, por ejemplo: “esBaja”.
- ✓ Los nombres de las variables booleanas deben ser positivos, por ejemplo: “esBaja”.

3.2 Validación

Una vez terminada la implementación del sistema durante las tres iteraciones planificadas en el Palan de Releases, cumpliendo con el tiempo estimado para el desarrollo cada una y con la utilización de estándar *CamelCase* anteriormente descrito. Se hace necesario, la verificación de los requisitos y el diseño propuesto, proceso que se describe a continuación.

3.2.1 Validación de requisitos

Con el objetivo de verificar si los requisitos del software obtenidos definen el sistema que el cliente desea, se llevó a cabo un proceso de validación de los mismos, para el cual se emplearon las siguientes técnicas:

- ✓ **Revisiones de los requisitos**: se realizaron revisiones a cada uno de los requisitos por parte del equipo de desarrollo y por el cliente. Las revisiones internas generaron un total de 9 no conformidades de tipo técnicas y de ortografía, las cuales fueron corregidas satisfactoriamente en tiempo. Con el cliente se desarrollaron 2 revisiones. En la primera, el mismo quedó satisfecho con el trabajo efectuado por el equipo de desarrollo, sin embargo se añadieron detalles a 6 de los requisitos funcionales para su perfeccionamiento. En el segundo encuentro, se añadieron especificaciones a 3 requisitos funcionales y se aprobaron finalmente los requisitos, generándose de este encuentro el Acta de Aceptación por parte del cliente (Ver Anexo 13).
- ✓ **Construcción de prototipos**: mediante los prototipos se le mostró al cliente un modelo ejecutable del portal web que le permitió tener una visión preliminar de cómo sería el sistema y a través de la interacción con los mismos se comprobó si satisfacía sus necesidades, los mismos fueron aprobados por el cliente con un alto nivel de satisfacción.

Métrica para Calidad de Especificación

Una de las métricas propuestas por la metodología de desarrollo seleccionada es la métrica de Calidad de la especificación (CE), la cual permite obtener cuán entendibles y precisos son los requisitos, primeramente se calcula el total de requisitos de la especificación como se muestra a continuación:

Nr = Nf + Nnf
Nr = 42 + 40
Nr = 82

Donde:
Nr: *número de requisitos de especificación*.
Nf: *número de requisitos funcionales.*
Nnf: *número de requisitos no funcionales*.

Para determinar, finalmente, la Especificidad de los Requisitos (ER) o ausencia de ambigüedad en los mismos se realiza la siguiente operación: **ER = Nui / Nr.**

Donde *Nui* es el número de requisitos para los cuales todos los revisores tuvieron interpretaciones idénticas. Mientras más cerca de 1 esté el valor de ER, menor será la ambigüedad. Para el caso de los requisitos obtenidos para la aplicación solo uno produjo contradicción en las interpretaciones. Sustituyendo las variables se obtiene:

ER = Nui / Nr
ER = 81 / 82
ER = 0,987 ≈ 0.99

Arrojando un resultado final satisfactorio, indicando que el grado de ambigüedad de los requisitos es sumamente bajo (1%) ya que el 99% es entendible. El requisito ambiguo fue modificado y validado para garantizar su correcta comprensión.

3.2.2 Validación del diseño de la aplicación web

Para comprobar la calidad y fiabilidad del diseño del sistema SGiD, se emplearon las métricas basadas en clases: Tamaño Operacional de Clase (TOC) y Relaciones entre Clases (RC), permitiendo el resultado de ambas métricas validar el diseño propuesto en la investigación. La aplicación detallada de la métrica se encuentra recogida en el Anexo 14. A continuación se muestra un resumen de los resultados de la aplicación de ambas métricas.

Tamaño Operacional de Clases (TOC)

La métrica TOC permite medir la responsabilidad, la complejidad de implementación y la reutilización de las clases del diseño. Es importante destacar que para esta métrica, la responsabilidad y la complejidad son inversamente proporcionales a la reutilización, por lo que a mayor responsabilidad y complejidad de implementación de una clase, menor será su nivel de reutilización. El tamaño operacional de una clase se puede determinar empleando medidas para saber el número total de operaciones que contiene.

A continuación se muestra el resultado de la aplicación de la métrica TOC en las 25 clases que juegan un papel fundamental en los procesos principales del sistema informático.

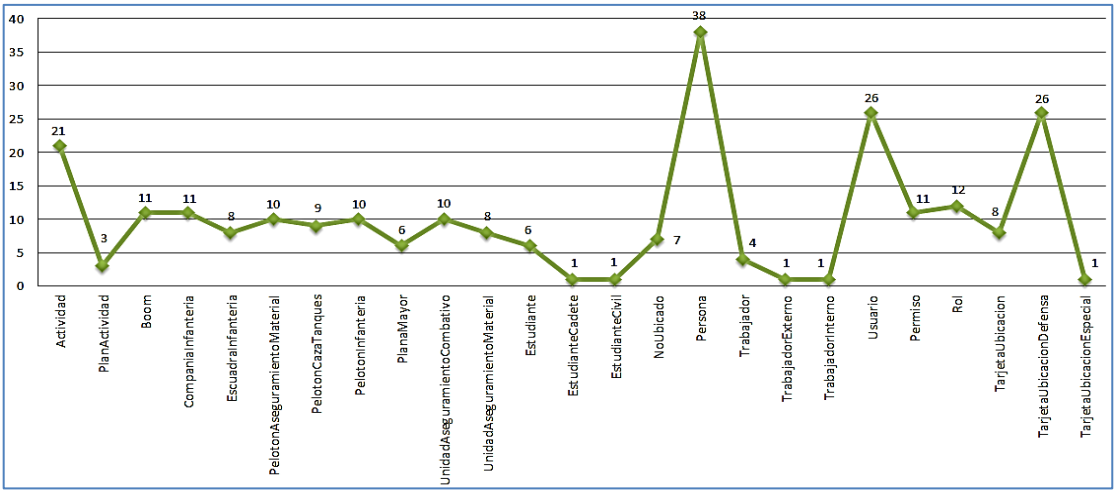


Figura 18: Cantidad de procedimiento por clases.

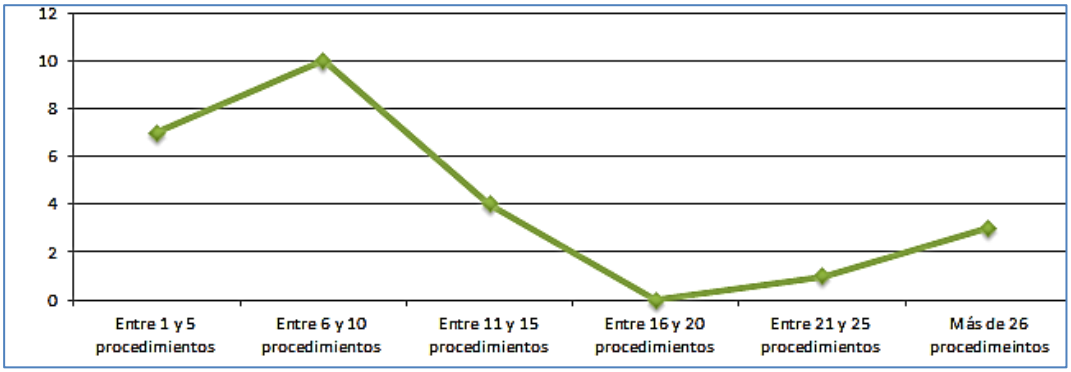


Figura 19: Resumen de cantidad de procedimientos.

Aplicando como umbral un promedio de 10 procedimientos (250 procedimientos en total / 25 clases existentes), se obtienen los valores de los atributos de calidad: responsabilidad de las clases, complejidad al implementar las mismas, así como sus niveles de reutilización.

	Responsabilidad		Complejidad		Reutilización	
Nivel	Cantidad de Clases	Promedio	Cantidad de Clases	Promedio	Cantidad de Clases	Promedio
Baja	17	68	17	68	17	16
Media	4	16	4	16	4	16
Alta	4	16	4	16	4	68

Tabla 10: Valores de las variables de calidad: Responsabilidad, Complejidad y Reutilización.

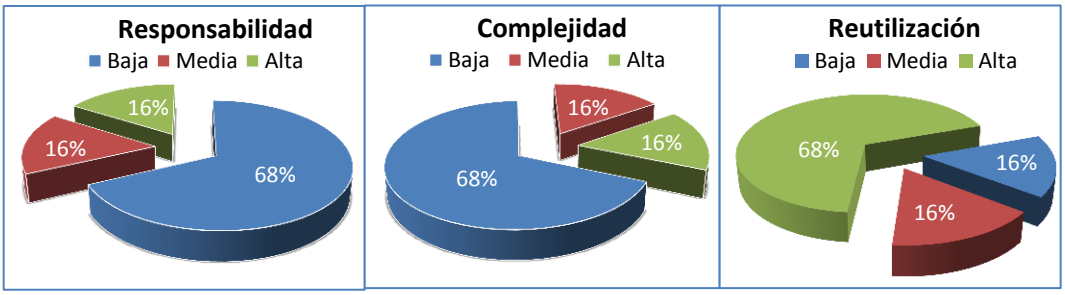


Figura 20: Resultado de las variables: Responsabilidad, Complejidad y Reutilización.

Luego de aplicada la métrica TOC se observa que las clases del diseño de la aplicación web no se encuentran sobrecargadas en cuanto a responsabilidades y el nivel de complejidad de las mismas no es muy alto, lo que favorece en gran medida la reutilización de estas.

Relaciones entre Clases (RC)

La métrica RC permite evaluar el acoplamiento, la complejidad de mantenimiento, la reutilización y la cantidad de pruebas de unidad necesarias para probar una clase teniendo en cuenta las relaciones existentes entre ellas. Luego de aplicar la métrica RC, se arrojaron los siguientes resultados:

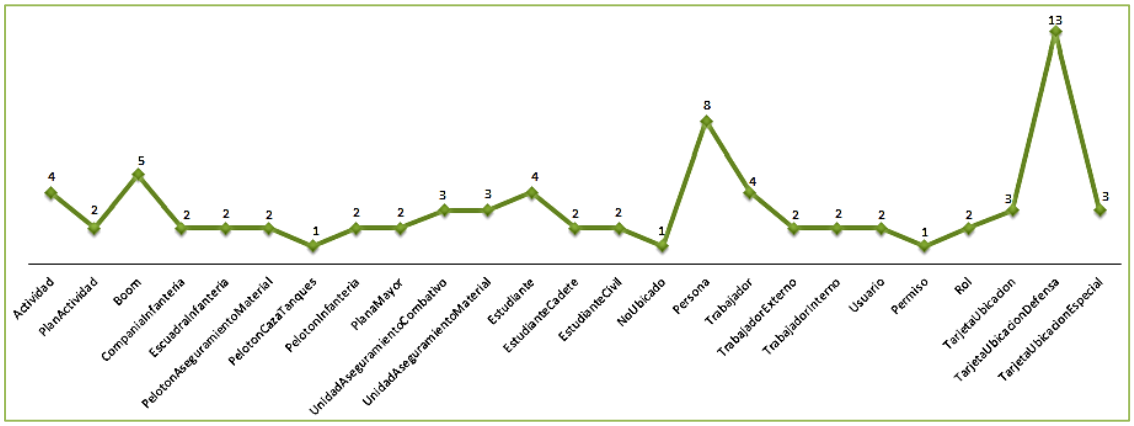


Figura 21: Cantidad de relaciones de uso por clases.

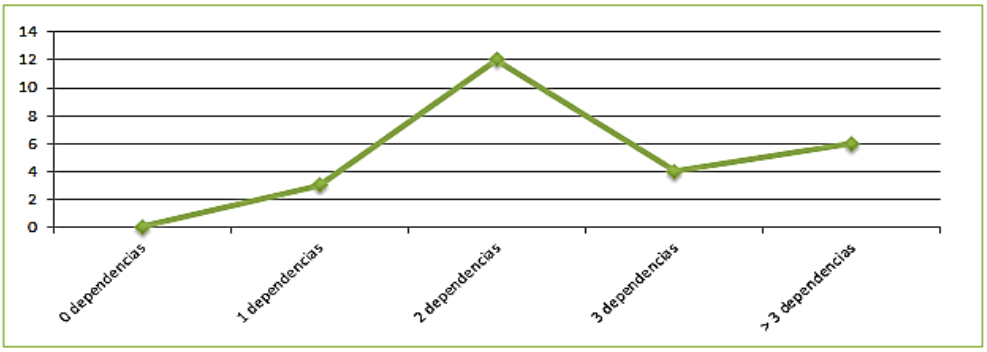


Figura 22: Resumen de relaciones de uso.

Luego de calcular los valores de las variables de calidad: acoplamiento, complejidad de mantenimiento, reutilización y cantidad de pruebas, aplicando como umbral un promedio 3 asociaciones de uso por clase; obteniendo los siguientes resultados:

Nivel	Acoplamiento		Complejidad		Reutilización		Cant. Pruebas	
	Cantidad de Clases	Promedio	Cantidad de Clases	Promedio	Cantidad de Clases	Promedio	Cantidad de Clases	Promedio
Ninguno	0	0	-	-	-	-	-	-
Bajo	15	60	19	76	2	8	19	76
Medio	4	16	4	16	4	16	4	16
Alto	6	24	2	8	19	76	2	8

Tabla 11: Valores de las variables: Acoplamiento, Complejidad, Reutilización y Cantidad de Pruebas.

Una vez aplicada la métrica RC y teniendo en cuenta el umbral definido para validar el diseño, se obtiene como resultado que las clases promueven el bajo acoplamiento; la complejidad de mantenimiento y la cantidad de pruebas no son altas, y en consecuencia, el grado de reutilización es mayor.

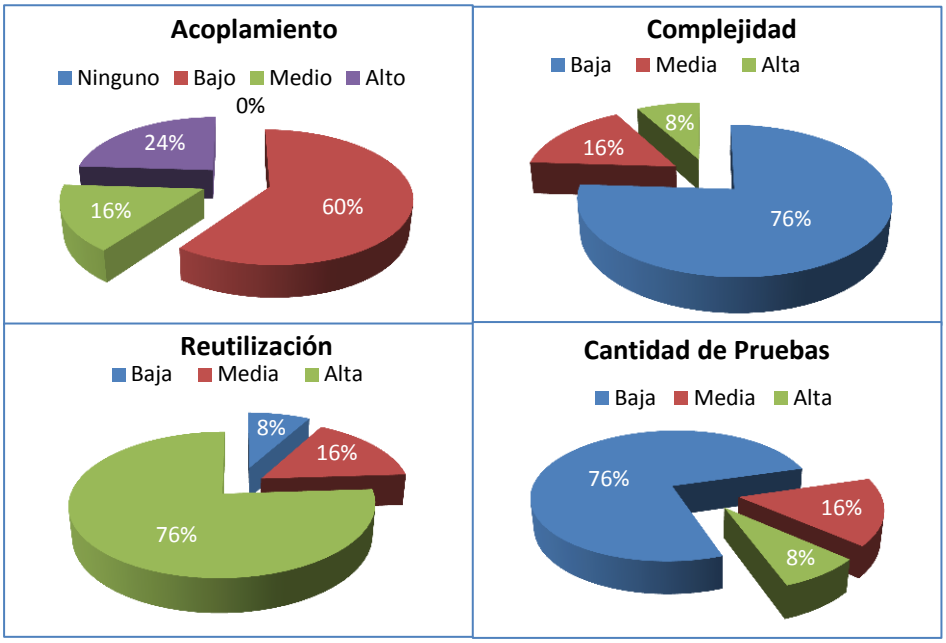


Figura 23: Resultado de las variables Acoplamiento, Complejidad, Reutilización y Cantidad de Pruebas**.**

En sentido general, los resultados obtenidos de la aplicación de las métricas TOC y RC demuestran que el diseño del sistema SGiD no es complejo, que las clases presentan bajo acoplamiento y un alto grado de reutilización.

3.3 Pruebas

Las pruebas son un instrumento adecuado para verificar el estado de la calidad de un producto en las cuales un sistema o componente es ejecutado bajo condiciones o requisitos especificados. Una de las principales fortalezas de la metodología de desarrollo SXP es el proceso de pruebas, el cual permite asegurar el éxito del producto al realizarse de manera continua, proporcionando la obtención de un producto de mayor calidad pues los errores son detectados en un corto plazo de tiempo y se corrigen de una manera más sencilla (52).

A la aplicación se le realizaron pruebas de funcionalidad y unitarias. Para las pruebas de funcionalidad se aplicó el método de Caja Negra con la técnica de Partición de Equivalencia, obteniendo el artefacto Diseño de Caso de Prueba. Mientras que para las de unidad el método aplicado fue el de Caja Blanca con la técnica de Camino Básico, generando el artefacto Caso de Prueba. A continuación se muestran los resultados de la aplicación de dichas pruebas.

3.3.1 Pruebas de funcionalidad

El principal objetivo de técnica de Caja Negra es comprobar que las funcionalidades de la aplicación se realizan de forma correcta y responden a las necesidades del cliente, apoyándose en los casos de pruebas diseñados para cada funcionalidad, los cuales aparecen en los Casos de Prueba de Aceptación, artefacto generado en el flujo de trabajo prueba de la Etapa de Desarrollo de SXP. En el Anexo 15, se muestra de forma detallada la aplicación de dicha prueba.

La tabla 12 muestra el caso de prueba CP_3-6 correspondiente a la HU_8: Reubicar personal.

Caso de Prueba de Aceptación	
Código Caso de Prueba: CP_3-6	**Nombre Historia de Usuario:** HU_8: Reubicar personal
Nombre de la persona que realiza la prueba: Addiel Aguila Espinaco**.**	
Descripción de la Prueba: Se ejecuta la prueba y se verifica que se pueda cambiar la ubicación en la defensa del usuario deseado.	
Condiciones de Ejecución: La persona a reubicar debe existir en la base de datos de la aplicación.	
Entrada / Pasos de ejecución: 1. Se realiza una búsqueda para encontrar la persona a quien se quiere reubicar. 2. Se selecciona la personal. 3. Se modifican los datos de su ubicación en la defensa de forma correcta. 4. Se guardan los cambios.	
Resultado Esperado: El sistema almacena los datos de la nueva ubicación de la persona y notificar que el proceso se realizó de forma correcta.	
Evaluación de la Prueba: Satisfactoria.	

Tabla 12: CP_3-6 correspondiente a HU_8: Reubicar personal.

Luego de la realización de 2 iteraciones de pruebas, se obtuvieron los siguientes resultados en cada uno de los módulos:

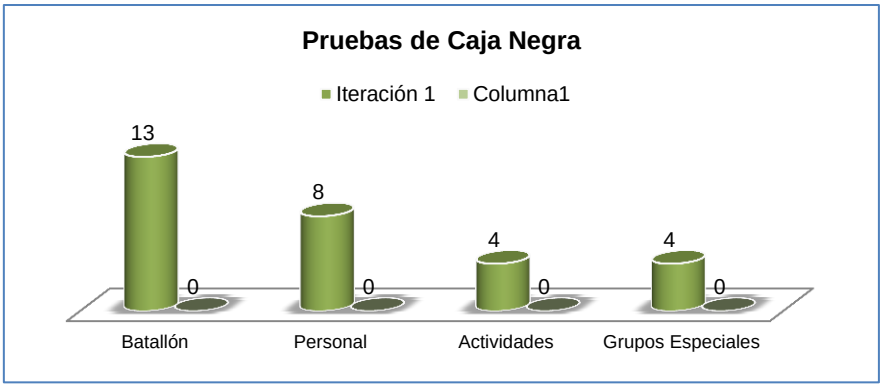


Figura 24: Resumen de Pruebas de Caja Negra.

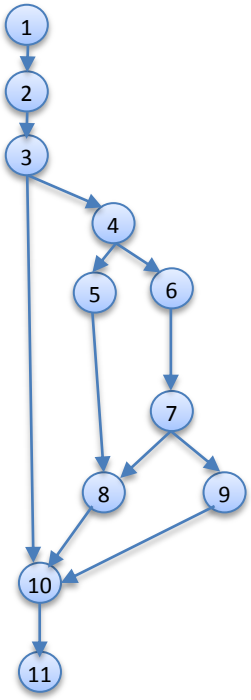
En una primera iteración se detectaron un total de 29 no conformidades entre los 4 módulos, generalmente errores ortográficos y de validación, las cuales se resolvieron de forma rápida y eficiente lo que posibilitó que durante la segunda iteración se obtuvieran resultados satisfactorios al no detectarse no conformidades. Finalmente se le fue otorgada el Acta de Liberación del Producto por parte del Grupo de Calidad del centro CEGEL (Ver Anexo 16).

3.3.2 Pruebas unitarias

Para comprobar que cada sentencia de código se ejecuta al menos una vez, se realizaron pruebas al código de las funcionalidades más complejas desde el punto de vista de la programación en cada uno de los módulos. Para ello se empleó el método de caja blanca con la técnica del C*amino Básico*; a partir de la obtención de la medida de la complejidad de un procedimiento o algoritmo y la obtención de un conjunto básico de caminos de ejecución de este, los cuales son utilizados para obtener los casos de prueba.

Seguidamente se muestra el proceso de pruebas realizado a la funcionalidad *ubicarAction()* perteneciente al módulo PersonalBoundle, específicamente a la controladora PersonaController.

Para obtener los casos de prueba a partir de la técnica seleccionada se debe construir el grafo de flujo correspondiente al código de la funcionalidad. Luego se determina la complejidad ciclomática V (G) del grafo resultante, la cual es un indicador del número de caminos independientes que existen en un grafo, es decir, es cualquier camino dentro del código que introduce por lo menos un nuevo conjunto de sentencias de proceso o una nueva condición (52).



La complejidad ciclomática puede ser calculada de 3 formas:

- V (G) = a - n + 2, siendo a el número de arcos o aristas del grafo y n el número de nodos.
- V (G) = r, siendo r el número de regiones cerradas del grafo.
- V (G) = c + 1, siendo c el número de nodos de condición.

V (G) = r

= 4

Realizando los cálculos correspondientes se obtiene por cualquiera de las variantes el siguiente resultado:

Figura 25: Grafo de flujo del código de la función UbicarAction() y cálculo de su complejidad ciclomática.

Por lo que el conjunto de caminos básico sería:

Camino Básico 1: 1-2-3-10-11

Camino Básico 2: 1-2-3-4-5-8-10-11

Camino Básico 3: 1-2-3-4-6-7-8-10-11

Camino Básico 4: 1-2-3-4-6-7-9-10-11

Luego se definen los casos de prueba para cada uno de los caminos básicos obtenidos. A continuación se muestra el resultado de las pruebas aplicadas a los caminos básicos 1 y 4.

Descripción: Se verificará que el sistema no permita ubicar una persona que no existe en la base de datos.		
Condición de Ejecución:	**Entrada.**	**Resultados esperados.**
La persona a ubicar no debe existir en la base de datos de la aplicación.	No se selecciona ninguna persona.	El sistema debe mostrar un mensaje notificando que se debe seleccionar una persona para poder efectuar la ubicación.
Resultado: Satisfactorio.		

Tabla 13: Caso de prueba para el camino básico 1.

Descripción: Se verificará se realice la ubicación de una persona de forma correcta.		
Condición de Ejecución:	**Entrada.**	**Resultados esperados.**
La persona debe existir en la base de datos de la aplicación y debe haber al menos una estructura creada dentro del batallón.	El identificador de la persona y la estructura donde será ubicada.	El sistema debe asignar la ubicación deseada a la persona, mostrando una página con su ubicación.
Resultado: Satisfactorio.		

Tabla 14: Caso de prueba para el camino básico 4.

De forma general se le realizaron las pruebas de caja blanca a 12 funcionalidades (3 de Batallón, 4 de Personal, 3 de Actividades y 2 de Grupos Especiales), seleccionadas a partir de su complejidad de implementación (Ver Anexo 16). Se realizaron dos iteraciones, las cuales arrojaron los siguientes resultados:

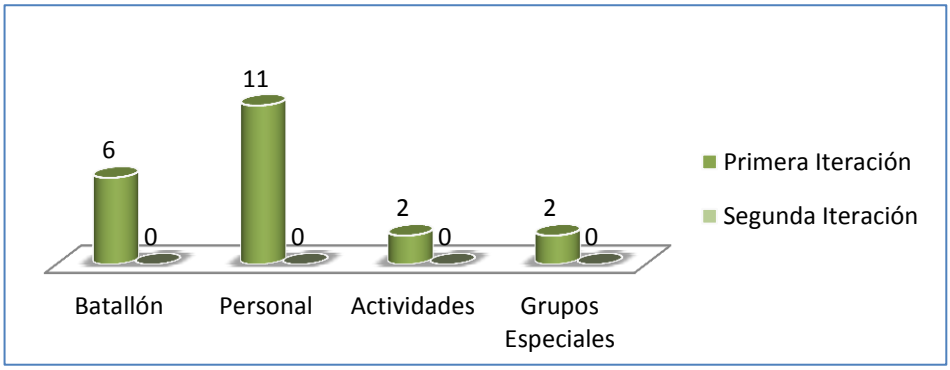


Figura 26: Resultados obtenidos en la aplicación de la técnica de Caja Blanca.

En una primera iteración se detectaron 21 no conformidades de forma general, las cuales fueron corregidas en su totalidad, por lo que una segunda iteración arrojó resultados satisfactorios con cero no conformidades.

3.4 Valoración del comportamiento de las variables de la investigación

Antes de la implantación del sistema, la gestión de la información sobre la estructura militar y las actividades de la defensa del Batallón Mixto de MTT Atípico de la Facultad 3, se realizaba de manera manual, resultando limitado el acceso a la información generada debido a que:

- ✓ Se destinaba gran cantidad de tiempo al proceso de emplantillamiento y ubicación de todo el personal.
- ✓ La información se almacenaba en archivos impresos o múltiples documentos de ofimática; lo que provocaba la centralización de los datos y el escaso conocimiento de los resultados de los procesos de la defensa.

✓ No existía un diseño y gestión eficiente de las actividades del batallón.

Todas estas deficiencias afectaban directamente el cumplimiento de las tareas y misiones de este batallón. Para demostrar la existencia del problema y aplicar los instrumentos seleccionados se definieron las siguientes variables:

Variable Dependiente: el cumplimiento de las tareas y misiones del Batallón Mixto de MTT Atípico de la Facultad 3 de la UCI.

Variable Independiente: la gestión de la información referente a la estructura militar y las actividades de la defensa.

A partir de los resultados del diagnóstico inicial, obtenido a través de la aplicación de los diferentes instrumentos y una vez introducida la herramienta informática SGiD; la gestión de la información referente a la estructura militar y las actividades de la defensa se logró mejorar, evidenciándose una descentralización de la misma, evidenciándose a través de la interacción generada mediante la web de los usuarios finales del proceso.

Se registra una reducción considerable del tiempo en el procesamiento de la información. La estimación del tiempo en horas, se realizó a través de la medición del mismo por la técnica (Cálculo del Tiempo Estándar o Tipo), donde se midió el tiempo estándar en la ejecución de una de las operaciones más complejas: *Ubicación del personal.*

Con el uso de la técnica se realizaron un total de tres mediciones en diferentes momentos, antes de implantar el sistema y luego de su puesta en práctica. Los resultados fueron los siguientes:

Medición 1	Tt (Tiempo Total)
Ubicación del personal	140.4 horas
Medición 2	**Tt (Tiempo Total)**
Ubicación del personal	161.46 horas
Medición 3	**Tt (Tiempo Total)**
Ubicación del personal	147.42 horas
Tiempo Estándar (Σ(Tt))	**449.28 horas**

Tabla 15: Medición del tiempo del proceso: Ubicar Personal (Antes del sistema).

Operaciones	Tt (Tiempo Total)
Ubicación del personal	56.16 horas
Operaciones	**Tt (Tiempo Total)**
Ubicación del personal	63.18 horas
Operaciones	**Tt (Tiempo Total)**
Ubicación del personal	49.14 horas
Tiempo Estándar (Σ(Tt))	**168.48 horas**

Tabla 16: Medición del tiempo del proceso: Ubicar Personal (Con el sistema).

La medición anterior permite demostrar que existe una reducción aproximada del tiempo en la ejecución del proceso de 280.8 horas.

Además, la creación de un archivo histórico permitió que el grado de persistencia de la información se elevara en un 100%, ya que fue almacenada la totalidad de los perfiles de los efectivos de la defensa, situación que no se daba antes de la creación del sistema debido a pérdida de los datos físicos que se generaban como resultado del proceso.

Se logra una estructura del batallón con calidad a partir del modelo que brinda el sistema para el emplantillamiento y ubicación del personal; un estricto control del Plan de Localización y Aviso, y la Brigada de Respuesta Rápida; permitiendo de manera sistemática contar con la disponibilidad de un organigrama con la estructura militar y la actualización de los datos necesarios de cada efectivo, lo que favorece sustancialmente el cumplimiento de las tareas y misiones del Batallón Mixto de MTT Atípico de la Facultad 3.

3.5 Conclusiones Parciales

Al finalizar el presente capítulo se puede arribar a las siguientes conclusiones:

- Se realizó el diagrama de componentes y de despliegue para mostrar una vista de cómo quedó la aplicación a nivel de componentes y su distribución.
- Se trataron los aspectos más significativos del proceso de implementación, donde se mostró el Plan de Releases y el estándar de codificación empleado en la solución.
- Se emplearon Técnicas de Validación de requisitos lo que permitió asegurar la fiabilidad de los mismos antes de comenzar el desarrollo del software.
- Se aplicaron las métricas TOC y RC que arrojaron resultados positivos en la valoración de la validación de la aplicación propuesta.
- De forma general, con las pruebas y validaciones realizadas se puede concluir que la herramienta desarrollada cumple con las especificaciones y requisitos definidos por los clientes en la etapa de concepción del sistema.
- Por último, se realizó una valoración del comportamiento de las variables que forman parte del problema de la investigación, demostrando que con el sistema desarrollado, la gestión de la información referente a la estructura militar y las actividades de la defensa se logró mejorar, evidenciándose una reducción del tiempo en el procesamiento de la información generada y descentralización de la misma, un correcto emplantillamiento del personal, mayor interacción de los usuarios para conocer y cumplir con sus tareas y misiones.

CONCLUSIONES GENERALES

Sobre la base del análisis, interpretación y sistematización de las indagaciones teóricas y empíricas, a continuación se presentan las siguientes conclusiones de la investigación:

1. El estudio teórico de los presupuestos actuales en los que se sustenta la gestión de la información referente a la defensa, estructuras militares y las actividades de la defensa de Batallones Mixtos de MTT, así como sus tareas y misiones permitió al autor de la investigación fundamentar la elaboración de una herramienta para la mejora de este proceso.

2. El análisis de un conjunto de herramientas y tecnologías permitió definir el marco tecnológico en ajuste a las políticas de migración a software libre por la cual aboga el país.

3. Se demuestra que el desarrollo y puesta en práctica de la herramienta SGiD, permitió mejorar la gestión de la información referente a la estructura militar, las actividades de la defensa y misiones del Batallón Mixto de MTT de la Facultad 3 de la UCI.

RECOMENDACIONES

El autor de este trabajo recomienda:

1. Divulgar los principales resultados de la investigación, a partir de la presentación y aplicación de la propuesta a otras facultades y/o áreas de la UCI.

2. Integrar futuras versiones con la incorporación de nuevas funcionalidades referentes a la evaluación cualitativa de actividades de la defensa propias de la Facultad 3.

BIBLIOGRAFÍA REFERENCIADA

1. Perales, Pedro E. Campos, Valladares, Ariel Guilarete y PREPARACIÓN, Angel Luis Cos Palmero. LA.

2. Fuerzas Armadas Revolucionarias. Sitio Web de la defensa de Cuba. www.cubadefensa.cu. [En línea] [Citado el: 12 de Noviembre de 2013.] 2013. http:// www.cubadefensa.cu/far.html. Ley de la defensa nacional, Nº 75 - 21/12/1994, Art. 34. [En línea]

3. Atlas Comparativo de la Defensa en América Latina y Caribe/Edición 2010. [En línea] [Citado el: 12 de Noviembre de 2013.]. ttp://www.resdal.org/atlas/atlas10-15-cuba.pdf (pag.187). [En línea]

4. Milicias de Tropas Territoriales. Sitio Web de la defensa de Cuba. www.cubadefensa.cu. [En línea] [Citado el: 12 de Noviembre de 2013.] 2013. http:// www.cubadefensa.cu/mtt.html. [En línea]

5. Garrucha, MSc. Manuel Piloto. Sistema de Gestión del Conocimiento para una Consultoría de Inteligencia Empresarial. [En línea] [Citado el: 15 de Noviembre de 2013.] http://www.intempres.pco.cu/Intempres2006/Intempres2006/Evaluacion de trabajos/Manuel_Pilo. [En línea]

6. Woodman L. (1985) Information management in large organizations. En: Information management from strategies to action. London: ASLIB.

7. Ponjuán,G., (2000). Aplicaciones de Gestión de información en las organizaciones. El profesional de la información y su dominio de las técnicas y herramientas de la Gestión. Tesis de Doctorado. Cuba, Departamento de Bibliotecología y Ciencia de la Informa.

8. Davis,G. y Olsón., (1985) Management Information Systems: Conceptual foundations, Structure and Development. 2a ed.Nueva York: McGrawhill, Citado en Moreiro, G. (1998) Introducción al estudio de la información y la documentación. Medellín: Editorial Unive.

9. Moreiro,G., (1998) Introducción al estudio de la información y la documentación. Medellín: Editorial de Antioquía.

10. Villaverde, P. O. F. and R. A. P. Gutiérrez. "La Gestión por Procesos y los Recursos Humanos. Presentación del software RH-CITMA " Retrieved 5 de febrero de 2008.

11. Julio, G. d. J. and C. R. Cristóbal. (1999). "La Gestión del Conocimiento y el Factor Humano. Pasos para Equilibrar sus funciones.

12. NYS Diseño Interactivo. . RRHH. [En línea] 2001. [Citado el: 18 de 12 de 2013.] http://www.rrhhWeb.com/includes_contenido/que_es.html. [En línea]

13. Cezanne Software HR. Cezanne. [En línea] 2009-2014. [Citado el: 18 de 12 de 2013.] http://cezannehr.com/es/software-rrhh/gestion-de-las-personas/. [En línea]

14. Centro de Estudios de Ingeniería en Sistemas CEIS CUJAE. GREHU. Sistema de Gestión de Recursos Humanos. [En línea] 2003. [Citado el: 12 de 01 de 2014.] http://grehu.cujae.edu.cu/index.asp. [En línea]

15. http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_31TD_05231_12.pdf. [En línea]

16. OSD, Open Systems Development. http://www.osdglobal.com/faq/desarrollo-software/comparativo-web-vs-escritorio.

17. Pressman, Roger S. 2001. Ingeniería del Software: una tecnología estratificada. Ingeniería del Software. Un enfoque práctico. Quinta Edición. España : McGraw Hill, 2001. [En línea]

18. Leyva Samada, Lisandra Isabel . 2009. Flujo de Investigación para la Metodología Ágil SXP. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas. [En línea] 2009. http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_24. [En línea]

19. Romero, Gladys Marsi Peñalver. 2008. MA-GMPR-UR2 Metodología ágil para proyectos de software libre. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas. [En línea] 2008. http://repositorio_institucional.uci.cu/jspui/handle/id. [En línea]

20. Sommerville, Ian. 2005. Ingeniería del software. Séptima Edición. Madrid. España: Pearson Educación. S. A., 2005. 84-7829-074-5. [En línea]

21. Del Pino Valdarrama, Santiago Luis. 2005. Programación extrema en pocos minutos: planificando la transición. Cuba : Tono. Revista Técnica de la Empresa de Telecomunicaciones de Cuba, S.A., 2005. 3, págs. 41-44. 18135056. [En línea]

22. Rumbaugh, James , Jacobson, Ivar y Booch, Grady. 2007. El Lenguaje Unificado de Modelado Manual de Referencia. s.l. : Addison-Wesley Iberoameri, 2007. 9788478290871. [En línea]

23. Leffingwell, Dean y Widrig, Don. 2003. "Using Software Engineering Techniques for Business Modeling. The Unified Modeling Language". Managing software requirements : a use case approach. United States of America : Pearson Education, Inc., 2003. 0-321-1224. [En línea]

24. Kendall, Kenneth E. y Kendall, Julie E. 2005. Análisis y Diseño de Sistemas. Sexta Edición. México: Pearson Educación de México, S.A. de C.V., 2005. 970-26-0577-6. [En línea]

25. Leffingwell, Dean y Widrig, Don. 2003. "Using Software Engineering Techniques for Business Modeling. The Unified Modeling Language". Managing software requirements : a use case approach. United States of America : Pearson Education, Inc., 2003. 0-321-1224.

26. Axure RP Web Oficial. http://www.axure.com/features.

27. Euphoriait. [Online] June 10, 2009. [Cited: Noviembre 19, 2013.] http://euphoriait.com/articulos/framework_web. [En línea]

28. Alfonso, X. C. G. Y. J. M. Mosaic. Introducción a los Sistemas de Gestión de Contenido [Online] 2007. [Cited: Diciembre 7, 2013.] http://mosaic.uoc.edu/articulos/cms1204.html. [En línea]

29. Object Relational Mapper. http://www.doctrine-project.org/projects/orm.html.

30. Twig. [En línea] http://twig.sensiolabs.org/.

31. Twitter Bootstrap. Página Oficial. http://bootstrapdocs.com/v2.1.1/docs/.

32. Rodríguez Sala, Jesús Javier . 2003. Introducción a la programación: teoría y práctica. s.l. : Editorial Club Universitario, 2003. 9788484542742. [En línea]

33. Sæther Bakken, Stig, Aulbach, Alexander y Schmid, Egon . 2000. PHP Manual. s.l: iUniverse Inc., 2000. Vol. 2. 978-0595132287. [En línea]

34. Gutierrez, Andres Felipe. Kumbia PHP Framework,¿PorqueProgramar debería ser más fácil?

35. Sánchez Maza, Miguel Ángel. 2012. JavaScript. s.l. : IC Editorial, 2012. 978-8495733184. [En línea]

36. González Boticario, Jesús y Gaudioso Vázquez, Elena. 2001. "Capítulo 6. JavaScript". Aprender y formar en Internet. Madrid, España : International Thomson Editors Spain Paraninfo, S. A., 2001. 84-283-2743-2. [En línea]

37. Eguiluz, Javier. Introducción a JavaScript. .

38. Eguiluz, Javier. Introducción a CSS.

39. Álvarez, Sara. 2007. Sistemas gestores de bases de datos. Introducción a este concepto y características especiales. [En línea] 2007. http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html. [En línea]

40. Caswell, Brian, Beale, Jay y Baker, Andrew. 2007. Snort Intrusion Detection and Prevention Toolkit. Kurasa : Syngress, 2007. 978-1-59749-099-3. [En línea]

41. Enciclopedia Cubana en la Red. http://www.ecured.cu/index.php/Servidores_Web#.C2.BFQu.C3.A9_es_un_Servidor_Web.3F.

42. del Castillo San Félix, Alvaro. 2000. El servidor de web Apache: Introducción práctica: Apache 1.x y 2.0 alpha. [En línea] 2000. http://acsblog.es/articulos/trunk/LinuxActual/Apache/html/x31.html. [En línea]

43. Editorbfb. 2011. Qué es un entorno de desarrollo integrado, IDE. Programación Desarrollo. [En línea] 2011. http://programaciondesarrollo.es/que-es-un-entorno-de-desarrollo-integrado-ide/. [En línea]

44. Software-talk.org. 2012. Netbeans vs Eclipse: An IDE comparison. Software Talk. [En línea] 2012. http://software-talk.org/blog/2012/01/netbeans-vs-eclipse-an-ide-comparison/. [En línea]

45. Peñalver et al, G.M., A. García, S., SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE , in 1er Congreso Iberoamericano de Ingeniería de proyectos . 2010: Chile.

46. Clements, P., "A Survey of Architecture Description Languages", in Proceedings of the International Workshop on Software Specification and Design. 1996: Alemania.

47. Arquitectura del Software. Available from: http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/rivera_l_a/capitulo2.pdf.

48. Areces, G.A., Diaz, Marquez Iskael, "Diseño e implementación de las capas de negocio y acceso a datos de los módulos Planificación y Ejecución de Visitas Familiares". 2008.

49. Patrones Grasp. 2011; Available from: http://www.buenastareas.com/ensayos/Patrones-Grasp/1896730.html.

50. Fernández, R.J., Modelo de Datos.

51. ARIZACA, R. E. Artefacto: Diagrama de Componentes. La Paz, Bolivia, 2009.

52. ROJAS, J. A. B., EMILIO. 2007. [Disponible en: http://gemini.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node26.html .

53. Comparativo de la Defensa en América Latina y Caribe/Edición 2010. [En línea] [Citado el: 12 de Noviembre de 2013.]. ttp://www.resdal.org/atlas/atlas10-15-cuba.pdf (pag.182). [En línea]

54. Bartle, Phil. Información para la gestión y gestión de la información. [En línea] [Citado el: 15 de Noviembre de 2013.] http://www.scn.org/mpfc/modules/mon-miss.htm. [En línea]

55. Informática en Salud http://informatica2009.sld.cu/Members/mirnacabrera/plataforma-para-la-administracion-procesamiento-y-transmision-de-la-informacion-en-el-sistema-de-salud-sisalud/ [En línea] 2013. [Citado el: 12 de 01 de 2014.]. [En línea]

56. http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_0017_04/1/TD_0017_04.pdf. [En línea]

57. Puertas Ortega, Juan y Orellana Zubieta, Francisco Javier. 2011. Un-paseo-por-PHP. Scribd.com. [En línea] 2011. http://es.scribd.com/doc/51830143/Un-paseo-por-PHP. DOI: 51830143. [En línea]

58. http://librosweb.es/bootstrap_3/capitulo_1.html. [En línea]

59. Cepeda Asqui, Jessica Paulina y Tene Reino, Blanca Georgina . 2012. Investigación de la Herramienta Case para el Desarrollo del Sistema Académico Educativo en el Centro de Educación Básica “Dr. Nicanor Larrea León, Basada en la Arquitectura .Net Framework. [En línea]

60. Danysoft. 2013. Embarcadero ER/Studio. [En línea] 2013. http://www.codegear-shop.com/Embarcadero-ER/Studio/es. [En línea]

61. Patrones-Grasp (2011). from http://www.buenastareas.com/ensayos/Patrones-Grasp/1896730.html.

62. Larman, Craig. UML y Patrones. Introducción al análisis y diseño orientado a objetos. México : PRENTICE HALL, 1999. ISBN 970-17-0261-1.

63. Symfony. [En línea] [Citado el: 12 de noviembre de 2012.] http://www.symfony.com/.

64. Symfony en español. [En línea] 12 de marzo de 2013. http://gitnacho.github.com/symfony-docs-es/book/doctrine.html.

65. Diagrama de clases. 2005; Available from: http://www-2.dc.uba.ar/materias/isoft1/is1-2005_2/apuntes/SlidesDC.pdf.

66. Gutierrez, Andres Felipe. Kumbia PHP Framework,¿PorqueProgramar debería ser más fácil?

67. Twig. [En línea] http://twig.sensiolabs.org/.

68. Castro, Elizabeth. HTML con XHTML y CSS. s.l. : Amaya.