



**Universidad de San Carlos de Guatemala**  
**Facultad de Ingeniería**  
**Escuela de Ingeniería en Ciencias y Sistemas**

**SISTEMATIZACIÓN DE LA UNIDAD DE INFORMACIÓN  
DEL ARCHIVO HISTÓRICO DE LA POLICÍA NACIONAL,  
MINISTERIO DE CULTURA Y DEPORTES**

**ALVARO DANIEL CASTILLO CARRERA**  
**PEDRO LUIS DOMINGO VÁSQUEZ**  
**MAX ALEJANDRO CERNA FLORES**

Asesorado por: Inga. Floriza Felipa Ávila Pesquera de Medinilla

Guatemala, agosto de 2010

Universidad de San Carlos de Guatemala

Facultad de Ingeniería



**SISTEMATIZACIÓN DE LA UNIDAD DE INFORMACIÓN  
DEL ARCHIVO HISTÓRICO DE LA POLICÍA NACIONAL,  
MINISTERIO DE CULTURA Y DEPORTES**

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA

FACULTAD DE INGENIERÍA

POR

**Alvaro Daniel Castillo Carrera**

**Pedro Luis Domingo Vásquez**

**Max Alejandro Cerna Flores**

Asesorado por: Inga. Floriza Felipa Ávila Pesquera de Medinilla

AL CONFERÍRSELES EL TÍTULO DE

**INGENIERO EN CIENCIAS Y SISTEMAS**

GUATEMALA, AGOSTO DE 2010

Universidad de San Carlos de Guatemala

Facultado de Ingeniería



### **NÓMINA DE JUNTA DIRECTIVA**

DECANO	Ing. Murphy Olympo Paiz Recinos
VOCAL I	Inga. Glenda Patricia García Soria
VOCAL II	Inga. Alba Maritza Guerrero de López
VOCAL III	Ing. Miguel Ángel Dávila Calderón
VOCAL IV	Br. Luis Pedro Ortíz de León
VOCAL V	Br. José Alfredo Ortíz Herinex
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

### **TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO**

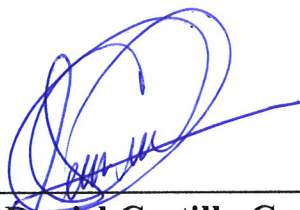
DECANO	Ing. Murphy Olympo Paiz Recinos
EXAMINADORA	Inga. Floriza Ávila
EXAMINADORA	Inga. Sonia Castañeda
EXAMINADOR	Ing. Marlon Pérez Turk
SECRETARIA	Inga. Marcia Ivonne Véliz

**HONORABLE TRIBUNAL EXAMINADOR**

Cumpliendo con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presentamos a su consideración el trabajo de graduación titulado:

**SISTEMATIZACIÓN DE LA UNIDAD DE INFORMACIÓN  
DEL ARCHIVO HISTÓRICO DE LA POLICÍA NACIONAL,  
MINISTERIO DE CULTURA Y DEPORTES,**

tema que nos fuera asignado por la Coordinación de la Carrera de Ingeniería en Ciencias y Sistemas, en abril de 2010.



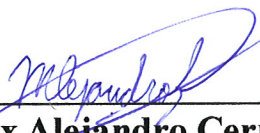
---

**Alvaro Daniel Castillo Carrera**



---

**Pedro Luis Domingo Vásquez**



---

**Max Alejandro Cerna Flores**



UNIDAD DE E.P.S.

Guatemala, 19 de mayo de 2010.  
REF.EPS.DOC.655.05.10.

Inga. Norma Ileana Sarmiento Zeceña de Serrano  
Directora Unidad de EPS  
Facultad de Ingeniería  
Presente

Estimada Ingeniera Sarmiento Zeceña.


Por este medio atentamente le informo que como Supervisora de la Práctica del Ejercicio Profesional Supervisado, (E.P.S) de los estudiantes universitarios de la Carrera de Ingeniería en Ciencias y Sistemas, **Alvaro Daniel Castillo Carrera** Carné No. **200312531**, **Pedro Luis Domingo Vásquez** Carné No. **200330475** y **Max Alejandro Cerna Flores** Carné No. **200413003** procedí a revisar el informe final, cuyo título es **“SISTEMATIZACIÓN DE LA UNIDAD DE INFORMACIÓN DEL ARCHIVO HISTÓRICO DE LA POLICIA NACIONAL, MINISTERIO DE CULTURA Y DEPORTES”**.

En tal virtud, **LO DOY POR APROBADO**, solicitándole darle el trámite respectivo.

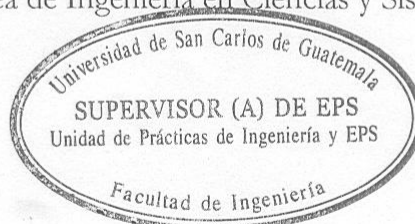
Sin otro particular, me es grato suscribirme.

Atentamente,

*“Id y Enseñad a Todos”*

  
Inga. Floriza Felipa Avila Pesquera de Medinilla  
Supervisora de EPS  
Área de Ingeniería en Ciencias y Sistemas

FFAPdM/RA





UNIDAD DE E.P.S.

Guatemala, 19 de mayo de 2010.  
REF.EPS.D.377.05.10.

Ing. Marlon Antonio Pérez Turck  
Director Escuela de Ingeniería Ciencias y Sistemas  
Facultad de Ingeniería  
Presente

Estimado Ingeniero Perez Turck.

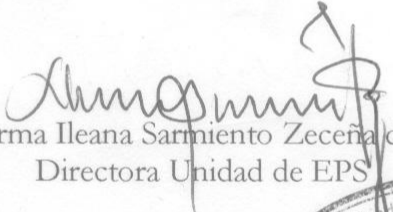
Por este medio atentamente le envío el informe final correspondiente a la práctica del Ejercicio Profesional Supervisado, (E.P.S) titulado **“SISTEMATIZACIÓN DE LA UNIDAD DE INFORMACIÓN DEL ARCHIVO HISTÓRICO DE LA POLICIA NACIONAL, MINISTERIO DE CULTURA Y DEPORTES”**, que fue desarrollado por los estudiantes universitarios **Alvaro Daniel Castillo Carrera** Carné No. **200312531**, **Pedro Luis Domingo Vásquez** Carné No. **200330475** y **Max Alejandro Cerna Flores** Carné No. **200413003** quienes fueron debidamente asesorados por el Ing. Jorge Armin Mazariegos y supervisados por la Inga. Floriza Felipa Ávila Pesquera de Medinilla

Por lo que habiendo cumplido con los objetivos y requisitos de ley del referido trabajo y existiendo la aprobación del mismo por parte del Asesor y de la Supervisora de EPS, en mi calidad de Directora apruebo su contenido solicitándole darle el trámite respectivo.

Sin otro particular, me es grato suscribirme.

Atentamente,

*“Id y Enseñad a Todos”*

  
Inga. Norma Ileana Sarmiento Zecena de Serrano  
Directora Unidad de EPS

NISZ/ra





Guatemala, 19 de mayo de 2010.  
REF.EPS.DOC.655.05.10.

Inga. Norma Ileana Sarmiento Zeceña de Serrano  
Directora Unidad de EPS  
Facultad de Ingeniería  
Presente

Estimada Ingeniera Sarmiento Zeceña.

Por este medio atentamente le informo que como Asesora-Supervisora de la Práctica del Ejercicio Profesional Supervisado, (E.P.S) de los estudiantes universitarios de la Carrera de Ingeniería en Ciencias y Sistemas, **Alvaro Daniel Castillo Carrera** Carné 200312531, **Pedro Luis Domingo Vásquez** Carné 200330475 y **Max Alejandro Cerna Flores** Carné No. 200413003 procedí a revisar el informe final, cuyo título es **“SISTEMATIZACIÓN DE LA UNIDAD DE INFORMACIÓN DEL ARCHIVO HISTÓRICO DE LA POLICIA NACIONAL, MINISTERIO DE CULTURA Y DEPORTES”**.

En tal virtud, **LO DOY POR APROBADO**, solicitándole darle el trámite respectivo.

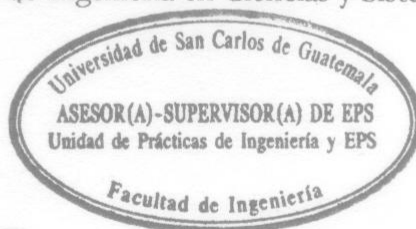
Sin otro particular, me es grato suscribirme.

Atentamente,

“Id y Enseñad a Todos”

Inga. Floriza Felipa Avila Pesquera de Medinilla  
Asesora-Supervisora de EPS  
Área de Ingeniería en Ciencias y Sistemas

FFAPdM/RA





Universidad San Carlos de Guatemala  
Facultad de Ingeniería  
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 07 de Julio de 2010

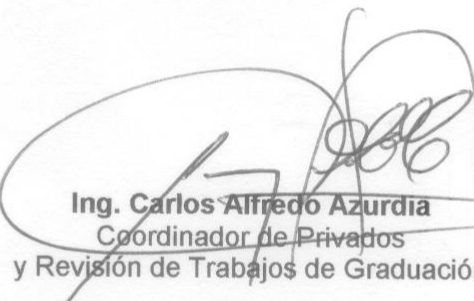
Ingeniero  
**Marlon Antonio Pérez Turk**  
Director de la Escuela de Ingeniería  
En Ciencias y Sistemas

Respetable Ingeniero Pérez:

Por este medio hago de su conocimiento que he revisado el trabajo de graduación-EPS de los estudiantes **ALVARO DANIEL CASTILLO CARRERA** carné 2003-12531, **PEDRO LUIS DOMINGO VASQUEZ** carné 2003-30475 y **MAX ALEJANDRO CERNA FLORES** carné 2004-13003, titulado: "SISTEMATIZACION DE LA UNIDAD DE INFORMACION DEL ARCHIVO HISTORICO DE LA POLICIA NACIONAL, MINISTERIO DE CULTURA Y DEPORTES", y a mi criterio el mismo cumple con los objetivos propuestos para su desarrollo, según el protocolo.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,

  
**Ing. Carlos Alfredo Azurdia**  
Coordinador de Privados  
y Revisión de Trabajos de Graduación





E  
S  
C  
U  
E  
L  
A  
D  
E  
C  
I  
E  
N  
C  
I  
A  
S  
I  
N  
F  
O  
R  
M  
A  
C  
I  
O  
N  
E  
S

UNIVERSIDAD DE SAN CARLOS  
DE GUATEMALA



FACULTAD DE INGENIERÍA  
ESCUELA DE CIENCIAS Y SISTEMAS  
TEL: 24767644

*El Director de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del asesor con el visto bueno del revisor y del Licenciado en Letras, de trabajo de graduación titulado "SISTEMATIZACIÓN DE LA UNIDAD DE INFORMACIÓN DEL ARCHIVO HISTÓRICO DE LA POLICÍA NACIONAL, MINISTERIO DE CULTURA Y DEPORTES", presentado por los estudiantes ALVARO DANIEL CASTILLO CARRERA, PEDRO LUIS DOMINGO VÁSQUEZ Y MAX ALEJANDRO CERNA FLORES, aprueba el presente trabajo y solicita la autorización del mismo.*

"ID Y ENSEÑAD A TODOS"



*[Handwritten Signature]*  
Ing. Marlon Antonio Pérez Turk  
Director, Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 30 de julio 2010



El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería en Ciencias y Sistemas, al trabajo de graduación titulado: **SISTEMATIZACIÓN DE LA UNIDAD DE INFORMACIÓN DEL ARCHIVO HISTÓRICO DE LA POLICÍA NACIONAL, MINISTERIO DE CULTURA Y DEPORTES**, presentado por los estudiantes universitarios **Alvaro Daniel Castillo Carrera, Pedro Luis Domingo Vásquez y Max Alejandro Cerna Flores**, autoriza la impresión del mismo.

IMPRÍMASE.

  
Ing. Murphy Olympo Paiz Recinos  
DECANO

Guatemala, agosto de 2010



/cc  
c.c. archivo.

# ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES .....	V
LISTA DE SÍMBOLOS .....	VII
GLOSARIO .....	IX
RESUMEN .....	XI
OBJETIVOS .....	XIII
INTRODUCCIÓN .....	XV

## FASE DE INVESTIGACIÓN

1. ANTECEDENTES DE LA INSTITUCIÓN	
1.1. Generalidades .....	3
1.2. Cambio de titularidad .....	4
2. BREVE DESCRIPCIÓN DEL PROYECTO	
2.1. Justificación .....	8
2.1.1. Justificación técnica .....	8
2.1.2. Justificación social .....	9
3. PLAN DE CONFIGURACIÓN	
3.1. Propósito del plan de configuración .....	11
3.2. Alcance del plan de configuración .....	12
3.3. Organización .....	12
3.4. Metodología de desarrollo .....	12
3.4.1. Roles .....	12
3.4.2. Reuniones .....	13
3.4.3. Documentos .....	13
3.5. Roles y responsabilidades .....	13
3.6. Herramientas, ambiente e infraestructura .....	14
3.6.1. Control de versiones .....	14
3.6.2. Base de datos .....	15
3.6.3. Lenguaje de programación .....	15
3.6.4. Servidores <i>web</i> .....	15
3.6.5. Reportes .....	16
3.6.6. Entorno de desarrollo integrado .....	16
3.6.7. Sistema de pruebas .....	16
3.6.8. Diseño de interfaces .....	17
3.7. Almacenamiento de versiones .....	17

3.8. Liberación de versiones.....	17
-----------------------------------	----

**FASE TÉCNICO-PROFESIONAL**

**4. ARQUITECTURA DEL SISTEMA**

4.1. Propósito de la arquitectura.....	21
4.2. Alcance de la arquitectura.....	21
4.3. Presentación de la arquitectura.....	22
4.4. Objetivos y restricciones de la arquitectura.....	23
4.5. Vista de casos de uso.....	23
4.5.1. Descripción de casos de uso.....	23
4.5.1.1. Sección “Archivo”.....	23
4.5.1.2. Sección “Departamento”.....	26
4.5.1.3. Sección “Empleado”.....	27
4.5.1.4. Sección “Expediente”.....	29
4.5.1.5. Sección “Usuario”.....	32
4.5.1.6. Sección “Log”.....	35
4.5.1.7. Sección “Reportes”.....	35
4.5.2. Diagramas de casos de uso.....	36
4.5.2.1. Casos de uso del actor “Dirección”.....	36
4.5.2.2. Casos de uso del actor “Coordinación”.....	37
4.5.2.3. Casos de uso del actor “Reprografía”.....	38
4.5.2.4. Casos de uso del actor “Sección de análisis legal”.....	38
4.5.2.5. Casos de uso del actor “Sección de investigación”.....	39
4.5.2.6. Casos de uso del actor “Sección de atención al público”.....	40
4.5.2.7. Casos de uso del actor “Solicitante”.....	41
4.5.2.8. Diagrama general de casos de uso.....	42
4.6. Vista lógica.....	43
4.6.1. Paquete “ <i>util</i> ”.....	44
4.6.2. Paquete “negocio”.....	45
4.6.2.1. Paquete “ <i>pojo</i> ”.....	45
4.6.2.2. Paquete “servicio”.....	46
4.6.3. Paquete “reporte”.....	47
4.6.4. Paquete “persistencia”.....	47
4.6.5. Paquete “seguridad”.....	48
4.6.5. Paquete “presentación”.....	49
4.7. Vista de procesos.....	50
4.7.1. Involucrados.....	50
4.7.2. Descripción del proceso.....	51
4.8. Vista de implementación.....	55
4.8.1. Elección de capas y componentes.....	56
4.8.2. Beneficios de la arquitectura diseñada.....	58
4.8.3. <i>Frameworks</i> empleados.....	58
4.8.3.1. <i>ICEFaces</i> .....	58

4.8.3.2.	<i>Spring</i> .....	59
4.8.3.3.	<i>Hibernate</i> .....	59
4.8.3.4.	<i>JUnit</i> .....	60
4.8.4.	Diagrama de la arquitectura.....	60
4.8.5.	Entornos de desarrollo.....	61
4.9.	Vista de despliegue.....	61
4.9.1.	Servidor de base de datos.....	61
4.9.2.	Servidor de aplicaciones.....	61
4.9.3.	Computadora interna.....	62
5.	<b>ANÁLISIS COSTO-BENEFICIO</b>	
5.1.	Datos generales.....	63
5.2.	Estimación por medio de COCOMO.....	64
5.3.	Estimación con COCOMO básico.....	64
5.4.	Estimación con COCOMO intermedio.....	65
5.5.	Estimación con <i>software</i> de gestión de proyectos.....	67
5.6.	Estimación de tarifas de costos de recursos.....	68
5.7.	Normalización de costos.....	69
5.8.	Resumen de costos.....	71
5.9.	Costo del sistema.....	71
6.	<b>LICENCIAMIENTO</b>	
6.1.	Licencias y tecnologías implementadas.....	76
6.2.	Descripción de licencias.....	77
	<b>FASE DE ENSEÑANZA</b>	
7.	<b>PLAN DE CAPACITACIÓN</b>	
7.1.	La inducción y el entrenamiento en el puesto.....	81
7.2.	Adiestramiento.....	83
7.3.	Capacitación y desarrollo profesional.....	84
7.4.	Determinando la efectividad de la capacitación.....	85
	CONCLUSIONES.....	87
	RECOMENDACIONES.....	89
	REFERENCIAS.....	91



# ÍNDICE DE ILUSTRACIONES

## Figuras

1	Casos de uso del actor “Dirección”	36
2	Casos de uso del actor “Coordinación”	37
3	Casos de uso del actor “Reprografía”	38
4	Casos de uso del actor “Sección de análisis legal”	38
5	Casos de uso del actor “Sección de investigación”	39
6	Casos de uso del actor “Sección de atención al público”	40
7	Casos de uso del actor “Solicitante”	41
8	Diagrama general de casos de uso	42
9	Diagrama de paquetes	43
10	Paquete “ <i>útil</i> ”	44
11	Paquete “ <i>pojo</i> ”	45
12	Paquete “servicio”	46
13	Paquete “reporte”	47
14	Paquete “persistencia”	47
15	Paquete “seguridad”	48
16	Paquete “presentación”	49
17	Esquema modelo-vista-controlador	55
18	Gestión por medio de <i>spring</i>	57
19	Funcionamiento de MVC	60
20	Resultados de costo calculado por <i>software</i>	68
21	Características de licencias de <i>software</i>	74
22	Licencias reconocidas por FSF	75
23	Licencias de herramientas	76

## Tablas

I	Roles y responsabilidades en <i>Scrum</i>	14
II	Descripción de las clases del paquete “ <i>útil</i> ”	44
III	Descripción de las clases del paquete “servicio”	46
IV	Descripción de las clases del paquete “reporte”	47
V	Descripción de las clases del paquete “persistencia”	48
VI	Descripción de las clases del paquete “seguridad”	48
VII	Descripción de los paquetes dentro del paquete “presentación”	49
VIII	Datos generales del código fuente	63
IX	Grados de influencia	66
X	Cantidad de personas y costos	69
XI	Resumen de costos	71
XII	Aprobación de licencias	77



## LISTA DE SÍMBOLOS

AHPN	Archivo histórico de la policía nacional
SEREVIDH	Servicio de referencia sobre violaciones de derechos humanos
PRAHPN	Proyecto de recuperación del archivo histórico de la policía nacional
PDH	Procurador de Derechos Humanos
PN	Policía Nacional
PNC	Policía Nacional Civil
GCS	Gestión de configuración de <i>software</i>
API	Interfaz de programación de aplicaciones
MVC	Modelo vista controlador
FSF	<i>Free software foundation</i>



## GLOSARIO

<i>Scrum</i>	Proceso iterativo e incremental de desarrollo de <i>software</i> , que se utiliza comúnmente con entornos de desarrollo ágil de <i>software</i> .
GNU GPL	Licencia pública general de GNU, es una licencia orientada a la protección de la libre distribución, modificación y utilización del <i>software</i> .
UML	Lenguaje unificado de modelado, es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.
Patente	Según la RAE (Real Academia Española) se define como: “Documento en que oficialmente se le reconoce a alguien una invención y los derechos que de ella se derivan”. Es un título que reconoce el derecho de explotar en exclusiva la invención patentada, impidiendo a otros su fabricación, venta o utilización sin aprobación del inventor. Finalmente, la patente consiste en excluir a otros de la fabricación, utilización o introducción del producto o procedimiento patentado sin el previo consentimiento.
Licencia	Es un contrato que se establece entre el propietario de los derechos de una patente, o derechos de autor y el usuario o los usuarios, para el correcto uso del bien en cuestión, en cuanto a habilitación, limitaciones y restricciones de uso.
<i>Software</i> Comercial	El <i>software</i> comercial es software que está siendo desarrollado por una entidad con la intención de comercializar el uso del <i>software</i> . Hay <i>software</i> libre comercial y hay <i>software</i> no libre no comercial.

<i>Shareware</i>	Es <i>software</i> que incluye una autorización para redistribuir copias, pero establece que quien continúe haciendo uso de una copia deberá pagar un cargo por licencia. A pesar de ello este <i>software</i> no puede catalogarse como libre o parcialmente libre por dos motivos, 1) el código fuente no está disponible, 2) no incluye una autorización para hacer copiar e instalarlo sin pagar una cantidad por la licencia.
<i>Freeware</i>	Es usada comúnmente para paquetes que permiten la redistribución pero no la modificación (y su código fuente no está disponible).
<i>Software</i> privativo	Es aquel <i>software</i> cuyo uso, redistribución o modificación está prohibida, o requiere que usted solicite autorización o está tan restringida que no pueda hacerla libre de un modo efectivo.
<i>Software</i> parcialmente libre	Es <i>software</i> que no es libre, pero tiene autorización para particulares de usar, copiar, distribuir y modificar (incluyendo la distribución de versiones modificadas) sin fines de lucro. Los programas parcialmente libre tienen restricciones adicionales.
<i>Software</i> libre no protegido con <i>copyleft</i>	El autor autoriza su redistribución y modificación, así como añadirle restricciones adicionales. Si un programa es libre pero no protegido con <i>copyleft</i> , entonces algunas copias o versiones modificadas pueden no ser completamente libres.
<i>Software</i> protegido con <i>copyleft</i>	<i>Software</i> libre cuyos términos de distribución no permiten a los redistribuidores agregar ninguna restricción adicional cuando éstos lo redistribuyen o modifican. Esto significa que cada copia del <i>software</i> , aun si ha sido modificado, debe ser <i>software</i> libre.
<i>Software</i> de dominio público	<i>Software</i> que no está protegido con <i>copyright</i> . Es un caso especial de <i>software</i> libre no protegido con <i>copyleft</i> , es decir, algunas copias o versiones modificadas pueden no ser libre en su totalidad.
<i>Software</i> libre	<i>Software</i> que autoriza su uso, copia y distribución, ya sea literal o con modificaciones, gratis o mediante una gratificación.

## RESUMEN

En el 2005, en la ciudad de Guatemala, fue encontrado el Archivo Histórico de la Policía Nacional (AHPN), formándose así en el 2009 el Serevidh, a cargo de la Procuraduría de Derechos Humanos, con el fin de presentar al público los crímenes y violaciones de los derechos humanos que se cometieron durante el conflicto armado interno que se dio en el país. A mediados del mismo año el Serevidh se trasladó de la PDH al Ministerio de Cultura y Deportes con el nombre de “Unidad de Información del Archivo Histórico de la Policía Nacional”.

A raíz de lo anterior surgió este proyecto de EPS, el cual tuvo como fin cubrir los requerimientos específicos de administración, control y búsqueda de información en los diferentes tipos de medios con los que cuenta la Unidad de Información del Archivo Histórico de la Policía Nacional.

Para el desarrollo del proyecto se utilizó la metodología *Scrum*, una técnica de desarrollo de *software* iterativo e incremental, que busca mantener una buena comunicación entre todos los involucrados, especialmente entre el equipo de desarrollo.

Las herramientas de *software* utilizadas para el ciclo de vida del proyecto fueron: *Subversion*, *TortoiseSVN*, *MySQL*, *MySQL Workbench*, *Java*, *Apache*, *Tomcat*, *JasperReport*, *iReport*, *NetBeans*, *JUnit* y *Java Server Faces*.

Para la arquitectura del sistema a desarrollar se utilizó el patrón MVC (Modelo vista controlado), debido a que permite separar los datos de una aplicación, la interface de usuario, y la lógica de control en tres componentes distintos.

# **OBJETIVOS**

## **General**

Implementar un sistema informático que proporcione un control digitalizado para el proceso de consulta, entrega e investigación de archivos históricos administrados por la Unidad de Información del Archivo Histórico de la Policía Nacional.

## **Específicos**

1. Automatizar el ingreso de solicitudes y control de expedientes, a través de la digitalización de la información.
2. Optimizar los tiempos de respuesta, para el proceso de búsqueda, selección y entrega de documentos del archivo histórico.
3. Garantizar el proceso de acceso a la libre información, y llevar un control y registro sobre el proceso a través de un sistema informático.
4. Crear un sistema informático de ayuda social, que provea un ahorro de al menos Q100 000.00 a Guatemala por medio de la Universidad de San Carlos de Guatemala y el programa de EPS de la Facultad de Ingeniería.





## INTRODUCCIÓN

El día viernes 27 de febrero se instaló el Servicio de Referencia sobre las Violaciones a los Derechos Humanos (Serevidh). El Serevidh se creó como un nuevo mecanismo destinado a consultar los casos contenidos en los documentos de la oficina del Procurador de Derechos Humanos (PHD).

La base sobre la cual inició su funcionamiento el Serevidh fueron los 11 millones de folios de archivos de la antigua Policía Nacional que han sido digitalizados actualmente como parte del proyecto de Recuperación del Archivo Histórico de la Policía Nacional.

En junio de 2009 el Serevidh se trasladó de la Procuraduría de Derechos Humanos al Ministerio de Cultura y Deportes con el nombre de “Unidad de Información del Archivo Histórico de la Policía Nacional”.

Este proyecto tuvo como fin cubrir los requerimientos específicos de administración, control y búsqueda de información en los diferentes tipos de medios con los que cuenta la Unidad de Información del Archivo Histórico de la Policía Nacional. Además, el sistema se encarga del ciclo de vida completo de la información a través de las distintas etapas, que incluyen la generación, digitalización, captura, indexación, control de versiones, almacenamiento, publicación, distribución, automatización y/o preservación de todos los documentos involucrados en el proceso de ingreso de solicitudes, generación de expedientes, búsquedas, entrega de reportes y de documentos.

El sistema actual es una aplicación web que permite la automatización y seguimiento de los documentos, información y tareas que pasan de un participante a otro para la realización de acciones específicas, de acuerdo con ciertas reglas establecidas en la Unidad de Información del Archivo Histórico de la Policía Nacional.

El sistema cuenta con toda la funcionalidad necesaria para cubrir las etapas que atraviesa un expediente, desde su ingreso como una solicitud, el proceso de búsqueda e investigación, hasta la entrega de los documentos encontrados por la Unidad de Información del Archivo Histórico de la Policía Nacional, permitiendo también hacer una comparación de las metas fijadas por esta entidad mediante la utilización de los reportes generados.

Otra tarea del sistema es dar seguimiento a las solicitudes de investigación recibidas en la Unidad de Información del Archivo Histórico de la Policía Nacional, para que se pueda brindar la mayor cantidad de detalle del estado de estas solicitudes entre los distintos departamentos que lo conforman, con el fin de organizar, analizar, interpretar y presentar los datos obtenidos de los archivos a los solicitantes. Entre las funciones que se agregaron en la aplicación, con la finalidad de mejorar el sistema de búsqueda de información, fue la implementación de etiquetas.

El sistema se encarga de que toda la información del Archivo Histórico de la Policía Nacional, que se encuentra dispersa en múltiples documentos electrónicos, papel y en distintos medios de almacenamiento, pueda ser consultada mediante un interfaz web disponible desde cualquier equipo de la red en las instalaciones de la Unidad de Información.

## **FASE DE INVESTIGACIÓN**



# **1. ANTECEDENTES DE LA INSTITUCIÓN**

El día 24 de marzo del año 2009, en la ciudad de Guatemala, tuvo lugar la presentación del informe del Procurador de los Derechos Humanos de Guatemala titulado “El derecho a Saber”. Este informe dio a conocer los datos obtenidos del Archivo Histórico de la Policía Nacional (Ahpn) para evitar la impunidad de crímenes y violaciones de los derechos humanos cometidos durante el conflicto armado.

Ese mismo día se abrió la consulta pública a todos los fondos digitalizados del Ahpn. Dichas consultas se realizan por medio del Servicio de Referencia sobre Violaciones de Derechos Humanos (Serevidh), el cual cuenta con todos los instrumentos de descripción, bases de datos e imágenes digitalizadas por el Proyecto de Recuperación del Archivo Histórico de la Policía Nacional (Prahpn). Todos estos datos están comprendidos, en su mayoría, en los años 1975 a 1985, lo que representa en la actualidad un aproximado del 10% del volumen total de documentos que contiene el Ahpn.

## **1.1. Generalidades**

El Serevidh se instala, organiza y funciona con el recurso humano, físico, material, electrónico y digital del Prahpn y es parte de la estructura física y organizacional del Procurador de Derechos Humanos, de quien dependerá directamente. El servicio en el que se incluyen todos aquellos documentos y datos que, producto de sus investigaciones sobre violaciones a los Derechos Humanos, la institución haya generado y continúe generando como propios.

El Serevidh tiene como función principal facilitar y garantizar al usuario particular o institucional, la disponibilidad y acceso a la información histórica nacional acumulada y contenida en las referencias de los documentos digitalizados por el PDH.

A partir de su creación, instalación y funcionamiento, el Serevidh generará en su funcionamiento diario archivos de gestión producto de su labor administrativa. Dichos archivos se alinearán a los siguientes principios:

- **Libre acceso:** todos los documentos digitales existentes en las bases de datos constituidas por el PDH, y que se encuentren integrados en el Serevidh, serán de acceso libre y gratuito.
- **Máxima publicidad en investigación de violaciones a los Derechos Humanos:** toda la información es pública y no se podrá reservar ni limitar, salvo por disposición constitucional o legal.

## 1.2. Cambio de titularidad

El Archivo Histórico de la Policía Nacional contiene registros que obedecen a la gestión administrativa de la institución. Pueden encontrarse una gran variedad de tipos documentales en los cuales se registró desde capturas, levantamiento de cadáveres y operativos de registro, hasta información de servicio del personal, por lo que el acervo documental refleja el funcionamiento cotidiano de las diferentes estructuras que conformaban la Policía Nacional.

En julio de 2005, personal de la Procuraduría de los Derechos Humanos (PDH) de Guatemala descubrió casualmente la existencia de un voluminoso acervo documental perteneciente a la extinta Policía Nacional (PN) y a la actual Policía Nacional Civil (PNC).

Ante la importancia del hallazgo, dada la magnitud y la calidad de la información que se podía encontrar en dicho Archivo, la PDH inició un proceso de recuperación del mismo con el objeto de realizar una investigación sobre violaciones a los Derechos Humanos.

Para realizar esa labor el PDH elaboró como estrategia: preservar, organizar y respaldar digitalmente la información contenida en el Archivo Histórico de la Policía Nacional, como un aporte a la recuperación de la memoria histórica y política del país. Dicha tarea la realizó el PDH hasta junio de 2009, en el momento que la titularidad del Archivo se traslada del Ministerio de Gobernación al Ministerio de Cultura y Deportes, esta cartera asume las funciones que hasta ese momento cumplía el PDH.

Es así como el Ministerio de Cultura y Deportes, a través del “Proyecto: Archivo Histórico de la Policía Nacional” aprovecha las capacidades humanas, técnicas y logísticas creadas durante el proceso anterior, contribuyendo así a la continuidad y estabilidad de los procesos realizados hasta el momento y reenfocando los esfuerzos en la nueva dirección de trabajo, con el amparo y acompañamiento del Archivo General de Centro América.

Es así como las funciones que desarrollaba el Serevidh para la PDH se trasladan a la Unidad de Información del Archivo Histórico de la Policía Nacional del Ministerio de Cultura y Deportes.





## **2. BREVE DESCRIPCIÓN DEL PROYECTO**

El sistema será una aplicación *web* que permitirá la implementación, automatización y seguimiento del Servicio de Referencia de Violaciones a los Derechos Humanos en donde se encuentran documentos, información y tareas que pasan de un participante a otro para la realización de acciones específicas, de acuerdo a ciertas reglas establecidas.

El sistema contara con toda la funcionalidad necesaria para cubrir las etapas del Serevidh; también permitirá observar el desempeño del servicio y compararlo con las metas que se han fijado.

Otra tarea del sistema será la de controlar y dar seguimiento a las solicitudes de investigación recibidas en el Serevidh, y dar la mayor cantidad de detalle del estado de estas solicitudes para así organizar, analizar, interpretar y presentar los datos obtenidos de los archivos a los solicitantes.

El sistema contará con varios módulos que permitirán cubrir los requerimientos específicos de administración, control y búsqueda de información en los diferentes tipos de medios con los que cuenta el servicio. Además, el sistema administrará el ciclo de vida completo de la información a través de las distintas etapas; estas incluyen: la generación, digitalización, captura, indexación, control de versiones, almacenamiento, publicación, protección, distribución, automatización y/o preservación de todos los documentos involucrados en el proceso.

El sistema se encargará de que toda la información del Archivo Histórico de la Policía Nacional, que se encuentra dispersa en múltiples documentos electrónicos y papel, y en distintos medios de almacenamiento, pueda ser consultada mediante un interfaz *web* disponible desde cualquier equipo de la red en las instalaciones del Serevidh.

Entre los beneficios que se esperan obtener con la implementación del sistema, están:

- Reducción horas-hombre por búsqueda de documentos
- Mejor organización de la información
- Control de los documentos físicos y electrónicos
- Reducción de costos por almacenamiento de documentos en las instalaciones

## **2.1. Justificación**

El Servicio de Referencia sobre Violaciones a los Derechos Humanos (Serevidh) es un nuevo mecanismo destinado a consultar los casos contenidos en los documentos de la oficina del Procurador de Derechos Humanos (PDH), por lo que la justificación de la sistematización de dicho Servicio de Referencia es:

### **2.1.1. Justificación técnica**

El Serevidh basa su funcionamiento en 11 millones de folios de archivos de la antigua PN que han sido digitalizados actualmente como parte del Prahpn. Este proyecto generó y sigue generando un volumen inmenso de datos, lo cual hace necesario que se cree una forma rápida e inteligente de su utilización.

Como el objetivo principal del Serevidh es brindar todos los datos posibles, sobre violaciones a los Derechos Humanos, a personas y entidades interesadas en el tema, es necesario que cuente con un sistema que le permita controlar y dar seguimiento a todas las solicitudes hechas por los interesados. Además, con el tiempo el sistema ayudará a convertir todos estos datos en información de gran importancia para resolver casos de violaciones a los Derechos Humanos.

### **2.1.2. Justificación social**

En Guatemala existen varios archivos similares al Archivo Histórico de la Policía Nacional pero ninguno cuenta con un sistema que se encargue de la administración, seguimiento, control y protección del mismo, por lo que este sistema será el primero en su tipo y servirá como base para la implementación de otros sistemas en los demás archivos del país.



### **3. PLAN DE CONFIGURACIÓN**

El plan de configuración presenta la Gestión de la Configuración de *Software* (GCS), en donde se especifica la organización y responsabilidad que se tiene con respecto a la construcción del *software*, de esta manera se pretende establecer un estándar para la realización del proyecto “Sistematización del Servicio de Referencia sobre Violaciones de los Derechos Humanos”.

La razón de la GCS es mantener la integridad y relación entre todos los componentes a definir en la creación del producto. Más adelante se establecen las políticas que rigen un correcto desarrollo del producto de *software*, además se establecen las nomenclaturas con sus respectivos significados. También se describen las herramientas a utilizar y la forma en que se controlarán las versiones de los componentes del producto de software. Por último se establece el control de cambios.

#### **3.1. Propósito del plan de configuración**

El propósito del Plan de Configuración es especificar un método de trabajo con el cual se verificará e implementarán todos los procesos involucrados en el desarrollo del producto. Con esto se pretende desarrollar el *software* de una forma eficaz y eficiente que permita obtener un producto de alta calidad.

## **3.2. Alcance del plan de configuración**

Se establecen las responsabilidades que los involucrados en el desarrollo del proyecto tendrán a su cargo, dependiendo del rol o función que tengan en determinado momento. También se indica el lugar y forma como se trabajará, obedeciendo los estándares previamente definidos, además de las plataformas a utilizar, lenguajes de programación y el diferente *software* que ayude o sea parte fundamental del desarrollo del producto.

## **3.3. Organización**

Debido a que la metodología utilizada para el desarrollo es *Scrum*, no se tienen demasiados roles o puestos como en otras metodologías. Para *Scrum* únicamente se definen tres roles, los cuales se explican en la sección 3.4.1.

## **3.4. Metodología de desarrollo**

*Scrum* es una metodología de desarrollo de *software* iterativo e incremental, consiste en el mejoramiento iterativo para desarrollar un sistema de programas de manera incremental, permitiendo al desarrollador sacar ventaja de lo que se ha aprendido a lo largo del desarrollo anterior, incrementando versiones entregables del sistema.

### **3.4.1. Roles**

*Scrum* es un modelo de referencia que define un conjunto de prácticas y roles, y que puede tomarse como punto de partida para definir el proceso de desarrollo que se ejecutará durante un proyecto.

### **3.4.2. Reuniones**

*Scrum* busca mantener una buena comunicación entre todos los involucrados, especialmente entre el equipo de desarrollo, por lo que se realizan reuniones periódicamente para tratar los avances, cambios, consultas y lo que se considere necesario para el mejor desarrollo del sistema. Entre las reuniones de *Scrum* se encuentran:

- *Daily Scrum*
- *Scrum de Scrum*
- *Sprint Planning Meeting*
- *Sprint Review Meeting*
- *Sprint Retrospective*

### **3.4.3. Documentos**

*Scrum* maneja varios documentos, para llevar un mejor control de lo que se haga o se decida hacer, tal es el caso de:

- *Product Backlog*
- *Sprint backlog*
- *Burn down*

## **3.5. Roles y Responsabilidades**

La tabla I describe los roles que se manejarán en la metodología *Scrum*, y que son los que se aplicarán durante el desarrollo del proyecto.

Tabla I. Roles y responsabilidades en *Scrum*

Rol	Responsabilidad
Propietario del producto	Representa a todos los interesados en el producto final, sus áreas de responsabilidad son: financiación del proyecto, requisitos del sistema, retorno de la inversión del proyecto y lanzamiento del proyecto
Equipo	Responsables de transformar la pila del sprint ( <i>Sprint Backlog</i> ) en un incremento de funcionalidad de <i>software</i> . El equipo es auto-gestionado, auto-organizado y multifuncional.
<i>Scrum manager</i>	Es el responsable del proceso <i>Scrum</i> , se encarga de la formación y entrenamiento del proceso, incorpora <i>Scrum</i> en la cultura de la empresa y garantiza el cumplimiento de los roles y responsabilidades

### 3.6. Herramientas, ambiente e infraestructura

Las herramientas de software a utilizar durante el desarrollo y ciclo de vida del proyecto son clasificadas según su funcionalidad y descritas a continuación.

#### 3.6.1. Control de versiones

- a. **Subversion (versión 1.6.6):** también conocido como SVN, esta es una herramienta libre, potente, segura y de fácil utilización que permite llevar un control de versiones de archivos.
- b. **Cliente de administración de control de versiones:** se utilizarán los clientes *TortoiseSVN* (versión 1.6.3), cliente SVN con una integración limpia con el explorador del sistema operativo *Microsoft Windows*, y *RapidSVN* (versión 0.12), cliente gráfico para el sistema operativo GNU/Linux.
- c. **Servidor SVN:** como servidor SVN se utilizará el sitio <http://www.assembla.com> el cual proporciona un repositorio SVN tanto de pago como gratuito.



### 3.6.2. Base de datos

- a. **MySQL (versión 5.1.33):** es un sistema de gestión de base de datos relacional, multihilo y multiusuario.
- b. **MySQL Workbench (versión 5.1.15):** es una herramienta que permite diseñar de forma visual bases de datos, ya que proporciona una representación visual de las tablas, vistas, procedimientos, funciones almacenadas y claves foráneas.

### 3.6.3. Lenguaje de programación

Se utilizará el lenguaje de programación *Java*, debido a que es un lenguaje de programación de alto nivel y orientado a objetos, esto permite una mayor comprensión del proyecto a los desarrolladores y, por ende, a todo el equipo de trabajo involucrado, mediante los diagramas de clases, utilizando notación UML; también se utiliza dicho lenguaje de programación debido a que es multiplataforma, lo cual facilita la ejecución de las aplicaciones en cualquier entorno de sistema operativo.

### 3.6.4. Servidores web

- a. **Apache (versión 2.2.13):** el servidor HTTP *Apache* es un servidor *web* HTTP de código abierto para plataformas *Unix* (BSD, GNU/Linux, etc.), *Windows*, *Macintosh* y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.

- b. **Tomcat (versión 6.0.20):** también llamado *Jakarta Tomcat* o *Apache Tomcat* funciona como un contenedor de *servlets* desarrollado bajo el proyecto *Jakarta* en la *Apache Software Foundation*. *Tomcat* implementa las especificaciones de los *servlets* y de *JavaServer Pages* (JSP) de *Sun Microsystems*.

### 3.6.5. Reportes

- a. **JasperReports (versión 3.5.3):** es una herramienta de creación de informes que tiene la habilidad de entregar contenido enriquecido en el monitor, a la impresora o a ficheros PDF, HTML, XLS, CSV y XML.
- b. **iReport (versión 3.5.3):** es un constructor y/o diseñador de informes visual, poderoso, intuitivo y fácil de usar para *JasperReports* escrito en *Java*.

### 3.6.6. Entorno de desarrollo integrado

Se utilizará el IDE de *Java*, *NetBeans* 6.5.1, ya que este provee una plataforma de desarrollo de aplicaciones basadas en *Java*.

### 3.6.7. Sistema de pruebas

Se ha definido el uso de *JUnit* para la realización de las pruebas unitarias a cada una de las clases que se desarrollarán en el proyecto, debido a que *JUnit* es un armazón para facilitar la programación de pruebas automáticas de las aplicaciones, es decir, las pruebas permitirán saber cuándo se ha terminado y cuando dichas pruebas funcionan, se

deberán de escribir pruebas únicamente para aquellos métodos que tengan un alto grado de riesgo a fallar.

### **3.6.8. Diseño de interfaces**

Se utilizarán las *Java Server Faces* debido a que es tecnología que ofrece un ambiente de trabajo estructurado sobre el cual se pueden construir sitios *web* de una manera estructurada.

### **3.7. Almacenamiento de versiones**

El almacenamiento de las versiones se hace con la ayuda de *software* especializado para tal objetivo, *el cual* tiene la capacidad de llevar el control de las diferentes versiones de los artefactos que componen el sistema, llevando la relación cronológica de los cambios, poniendo a disposición de los interesados los artefactos con los que deban trabajar.

### **3.8. Liberación de versiones**

Un desarrollador que modifica un elemento es responsable de cuidar que su código cumpla con los estándares definidos y de probar la funcionalidad y desempeño de su código. Una modificación no debe llegar al servidor de versiones si no compila correctamente en la máquina del desarrollador, en donde debe pasar por un ciclo de pruebas unitarias.

Una versión del *software* no puede ser liberada, conteniendo código que no se integre correctamente; en ese caso se estudia la posibilidad de liberarla sin esa funcionalidad problemática.



## **FASE TÉCNICO-PROFESIONAL**



## 4. ARQUITECTURA DEL SISTEMA

Este capítulo constituye un resumen de la arquitectura de *software* del proyecto. Con la arquitectura se pretende brindar múltiples vistas del sistema, las cuales incluyen los componentes principales del mismo, la conducta de estos componentes en el sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. Además de lo ya mencionado, también se incluyen las herramientas y patrones de diseño que se usarán para el desarrollo e implementación del sistema.<sup>1</sup>

### 4.1. Propósito de la arquitectura

Proveer una vista general de la organización y distribución de la arquitectura del sistema, capturar y describir las decisiones de mayor importancia. Además de presentar de una manera general las herramientas y *frameworks* que se utilizarán para desarrollar el sistema y cómo estas interactúan entre sí, haciendo énfasis en las ventajas que aportan y el porqué se escogieron frente a otras alternativas.

### 4.2. Alcance de la arquitectura

El alcance de la arquitectura se limita a mostrar una visión general de esta y proporcionar una base para la creación del sistema. Así como también solo se presenta una justificación del uso de las herramientas, no un uso concreto de las mismas dentro del desarrollo del sistema.

---

<sup>1</sup> Plantilla *Software Architecture Document*, como guía de redacción y de contenidos a considerar.  
[http://www.ts.mah.se/RUP/RationalUnifiedProcess/webtmpl/templates/a\\_and\\_d/rup\\_sad.htm](http://www.ts.mah.se/RUP/RationalUnifiedProcess/webtmpl/templates/a_and_d/rup_sad.htm)

### 4.3. Presentación de la arquitectura

Con la arquitectura se muestra el diseño y la implementación de la estructura de alto nivel del sistema a desarrollar. Esta arquitectura es el resultado de la unión de varios elementos arquitectónicos, los cuales serán utilizados para satisfacer la funcionalidad y ejecución de todos los requerimientos que debe cumplir el sistema. Con estos requerimientos también se incluyen los requerimientos no funcionales, tales como: fiabilidad, portabilidad, disponibilidad, escalabilidad, etc.<sup>2</sup>

Tratar de mostrar la arquitectura del sistema con un solo modelo o diagrama es bastante complicado debido a la complejidad de esta, por lo que la mejor forma de hacerlo es utilizando distintas vistas en donde cada vista se encarga de representar distintos aspectos y características del sistema. Debido a lo anterior esta arquitectura será mostrada con el modelo de 4 + 1 vista, el cual incluye las siguientes vistas:

- **Vista de casos de uso:** describe los escenarios que satisfacen la arquitectura y que representan los requerimientos funcionales del sistema.
- **Vista de implementación:** se enfoca en los módulos de *software* en el entorno de desarrollo y cómo estos módulos están organizados en capas.
- **Vista lógica:** crea un modelo que describe las estructuras de software para resolver los requerimientos funcionales.
- **Vista de procesos:** describe los objetos instanciados y procesos de ejecución que existen en el sistema.
- **Vista de despliegue:** describe equipo (*hardware*) que será utilizado para el funcionamiento del sistema. Esta vista se centra en los requerimientos no funcionales.

---

<sup>2</sup> Artefacto de la disciplina *Analysis & Design* de RUP.  
[http://www.ts.mah.se/RUP/RationalUnifiedProcess/process/artifact/ar\\_sadoc.htm](http://www.ts.mah.se/RUP/RationalUnifiedProcess/process/artifact/ar_sadoc.htm)



## 4.4. Objetivos y restricciones de la arquitectura

A continuación se muestran los objetivos y restricciones que tienen un impacto significativo sobre el sistema:

- El sistema debe ser *web*.
- Por el momento el sistema solo será accedido desde la *intranet*.
- Se deben permitir varios tipos de acceso según perfiles de usuario.
- La información manejada por el sistema será almacenada en una base de datos, exceptuando los archivos subidos al sistema, que se almacenarán en un directorio específico del servidor.

## 4.5. Vista de casos de uso

Ésta vista proporciona una visión global de la funcionalidad que tienen los casos de uso para el sistema y la manera en que estos están organizados, además representan claramente la forma en que interactúan los distintos actores involucrados con los casos de uso que describen al sistema.

### 4.5.1. Descripción de casos de uso

#### 4.5.1.1. Sección “Archivo”

**Agregar tipo de archivo:** el caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevos tipos de archivo, estos permiten tener una clasificación ordenada y detallada de los diferentes archivos que se manejan en el Serevidh.

**Modificar tipo de archivo:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar tipos de archivo ya existentes; lo que se busca es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de archivo”.

**Borrar tipo de archivo:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los tipos de archivos que no tengan archivos relacionados. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de archivos”.

**Listar tipo de archivo:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los tipos de archivo existentes en el sistema. Esto permite saber, en el momento que se necesite, los tipos de archivo que existen y, de ser necesario, agregar nuevos tipos o modificar los existentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de archivo”.

**Agregar estructura:** el caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevos niveles a la estructura archivística, estos niveles permiten tener una clasificación ordenada y detallada de los diferentes niveles en la estructura que se manejan en el Serevidh. El caso de uso es utilizado por el usuario que contenga los permisos necesarios e inicia cuando el usuario se identifica en el sistema e ingresa en el área “crear estructura”.

**Renombrar estructura:** el caso de uso busca modelar la funcionalidad del sistema que permite renombrar niveles de la estructura archivística ya existentes, con el objetivo de que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el usuario que contenga los permisos necesarios e inicia cuando el usuario se identifica en el sistema y luego ingresa en el área “crear estructura”.

**Modificar estructura:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar niveles de la estructura archivística ya existentes. A diferencia de renombrar la estructura donde solo se actualizan los nombres de los niveles de la estructura, al modificar la estructura se cambian los demás atributos de cada nivel, lo que se busca a través de esto es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el usuario que contenga los permisos necesarios e inicia cuando el usuario se identifica en el sistema y luego ingresa en el área “crear estructura”.

**Borrar estructura:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los niveles de la estructura archivística. El caso de uso es utilizado por el usuario con los permisos necesarios e inicia cuando el usuario se identifica en el sistema y luego ingresa en el área “crear estructura”.

**Agregar tipo de etiqueta:** el caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevos tipos de etiqueta, estos tipos de etiqueta permiten clasificar la información contenida en los archivos que se manejan en el Serevidh. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema e ingresa en el área “gestión de tipos de etiqueta”.

**Modificar tipo de etiqueta:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar tipos de etiqueta ya existentes, con el fin de que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de etiqueta”.

**Borrar tipo de etiqueta:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los tipos de etiquetas que no tengan etiquetas relacionadas. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de etiquetas”.

**Listar tipo de etiqueta:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los tipos de etiqueta existentes en el sistema. Esto permite saber, en el momento que se necesite, los tipos de etiqueta que existen y, de ser necesario, agregar nuevos tipos o modificar los existentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de etiqueta”.

#### **4.5.1.2. Sección “Departamento”**

**Agregar departamento:** el caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevos departamentos, estos departamentos permiten tener una clasificación ordenada y detallada de los diferentes departamentos que se manejan en el Serevidh. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema e ingresa en el área “gestión de departamentos”.

**Modificar departamento:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar departamentos ya existentes, lo que se busca a través de esto es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de departamentos”.

**Borrar departamento:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los departamentos que no tengan aun asignados empleados. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de departamentos”.

**Listar departamento:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los departamentos y entregas existentes en el sistema. Esto permite saber, en el momento que se necesite, los departamentos que existen y, de ser necesario, agregar nuevos departamentos o modificar los existentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de departamentos”.

#### **4.5.1.3. Sección “Empleado”**

**Agregar empleado:** El caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevos empleados, estos empleados permiten tener una clasificación ordenada y detallada de los diferentes empleados que se manejan en el Serevidh. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema e ingresa en el área “gestión de empleados”.

**Modificar empleado:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar empleados ya existentes, con el fin de que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de empleados”.

**Borrar empleado:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los empleados que no tengan aun asignados expedientes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de departamentos”.

**Listar empleado:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los empleados y entregas existentes en el sistema. Esto permite saber, en el momento que se necesite, los empleados existentes y, de ser necesario, agregar nuevos empleados o modificar los existentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de empleados”.

**Cambiar credenciales:** el caso de uso busca modelar la funcionalidad del sistema que permite cambiar las credenciales de un usuario existentes en el sistema (nombre de usuario y/o contraseña), esto permite dar un grado de libertad y seguridad al usuario. El caso de uso es utilizado por todo usuario en la aplicación e inicia cuando este se identifica en el sistema y luego ingresa en el área de empleado, en la sección de “cambiar credenciales”.

**Asignar permisos:** el caso de uso busca modelar la funcionalidad del sistema que permite asignar los permisos de un usuario existente en el sistema, esto permite dar los privilegios respectivos a cada usuario para poder realizar sus funciones. El caso de uso es utilizado por el usuario administrador o aquellos usuarios con este privilegio e inicia cuando el usuario se identifica en el sistema y luego ingresa en el área de empleado en la sección “asignar permisos”.

**Desasignar permisos:** el caso de uso busca modelar la funcionalidad del sistema que permite desasignar los permisos de un usuario existente en el sistema. Esto permite quitar los privilegios respectivos a cada usuario para poder realizar sus funciones. El caso de uso es utilizado por el usuario administrador o aquellos usuarios con este privilegio e inicia cuando el usuario se identifica en el sistema y luego ingresa en el área “empleado” en la sección “asignar permisos”.

**Listar actividades empleado:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de las actividades realizadas por empleado en cada expediente y sus fases existentes en el sistema. Esto permite saber, en el momento que se necesite, las acciones dentro de la aplicación tales como: errores, sesiones, entre otros, así como los usuarios que intervinieron. El caso de uso es utilizado por el usuario que tenga los permisos necesarios e inicia cuando el usuario se identifica en el sistema y luego ingresa en el área “empleado”.

#### **4.5.1.4. Sección “Expediente”**

**Agregar bitácora propuesta:** el caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevas propuestas de búsqueda a los hechos de un expediente.

El caso de uso es utilizado por los usuarios que posean los permisos necesarios para acceder e inicia cuando el usuario se identifica en el sistema e ingresa en el área “bitácora propuesta” cuando se selecciona un expediente en “listar expediente”.

**Modificar bitácora propuesta:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar la propuesta de los hechos de un expediente ya existentes. Lo que se busca a través de esto es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el usuario que contenga los permisos necesarios e inicia cuando el usuario se identifica en el sistema y luego ingresa en el área “bitácora propuesta” en la sección “listar expediente”.

**Agregar bitácora investigación:** el caso de uso busca modelar la funcionalidad del sistema que permite agregar datos de la investigación a las propuestas de los hechos de un expediente. Estos datos encontrados permiten tener una clasificación ordenada y detallada de la información que se administra en un expediente. El caso de uso es utilizado por los usuarios que posean los permisos necesarios para acceder e inicia cuando el usuario se identifica en el sistema e ingresa en el área “bitácora de investigación” cuando se selecciona un expediente en “listar expediente”.

**Modificar bitácora investigación:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar los datos de la propuesta de los hechos en expedientes ya existentes. Lo que se busca a través de esto es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el usuario que contenga los permisos necesarios e inicia cuando el usuario se identifica en el sistema y luego ingresa en el área “bitácora de investigación” en la sección “listar expediente”.



**Agregar expediente:** el caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevos expediente, estos expedientes permiten tener información ordenada y detallada de cada caso dado por un usuario, con sus hechos e involucrados relacionados a dicho expediente. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema e ingresa en el área “agregar usuario o listar usuario”.

**Listar expediente:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los expedientes existentes en el sistema. Esto permite, en el momento que se necesite, asignar o verificar la fecha de prórroga de un expediente. El caso de uso es utilizado por el servicio de atención al público e inicia cuando el servicio de atención al público se identifica en el sistema y luego ingresa en el área “gestión de expediente”.

**Agregar tipo de soporte entrega:** busca modelar la funcionalidad del sistema que permite agregar nuevos tipos de soporte entrega. Estos tipos de soporte entrega permiten tener una clasificación ordenada y detallada de sus diferentes variantes presentes en el Serevidh.

**Modificar tipo de soporte entrega:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar tipos de soportes y entregas ya existentes. Lo que se busca a través de esto es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de soporte entrega”.

**Borrar tipo de soporte entrega:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los tipos de soportes y entrega que no tengan detalles de entregas relacionadas. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de soporte y entrega”.

**Listar tipo de soporte entrega:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los tipos de soporte y entrega existentes en el sistema. Esto permite saber, en el momento que se necesite, los tipos de soportes y entregas que existen y de ser necesario agregar nuevos tipos o modificar los existentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de soporte entrega”.

#### **4.5.1.5. Sección “Usuario”**

**Agregar tipo de documento de identificación:** busca modelar la funcionalidad del sistema que permite agregar nuevos tipos de documento de identificación, estos tipos de documento de identificación clasifican a los usuarios en un expediente. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema e ingresa en el área “gestión de tipos de documento de identificación”.

**Modificar tipo de documento de identificación:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar tipos de documento de identificación ya existentes. Lo que se busca a través de esto es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de documento de identificación”.

**Borrar tipo de documento de identificación:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los tipos de documentos de identificación que no tengan usuarios relacionados. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de documentos de identificación”.

**Listar tipo de documento de identificación:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los tipos de documento de identificación existentes en el sistema. Esto permite saber, en el momento que se necesite, los tipos de documento de identificación que existen y, de ser necesario, agregar nuevos tipos o modificar los existentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de documento de identificación”.

**Agregar tipo de usuario:** El caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevos tipos de usuario, lo cual permite contar con una clasificación ordenada y detallada de los diferentes usuarios que se manejan en el Serevidh.

**Modificar tipo de usuario:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar tipos de usuario ya existentes. Lo que se busca aquí es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de usuario”.

**Borrar tipo de usuario:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los tipos de usuarios que no tengan expedientes relacionados. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de usuario”.

**Listar tipo de usuario:** busca modelar la funcionalidad del sistema que permite crear un reporte de los tipos de usuario existentes en el sistema. Esto permite saber, en el momento que se necesite, los tipos de usuario que existen y, de ser necesario, agregar nuevos tipos o modificar los existentes. El caso de uso es utilizado por el coordinador e inicia cuando el coordinador se identifica en el sistema y luego ingresa en el área “gestión de tipos de usuario”.

**Agregar usuario:** el caso de uso busca modelar la funcionalidad del sistema que permite agregar nuevos usuarios, los cuales permiten habilitar una clasificación ordenada y detallada de los diferentes usuarios que se administran en el Serevidh. El caso de uso es utilizado por la sección de atención al público (SAP) e inicia cuando SAP se identifica en el sistema e ingresa en el área “gestión de usuarios”.

**Modificar usuario:** el caso de uso busca modelar la funcionalidad del sistema que permite modificar departamentos ya existentes. Lo que se busca aquí es que los cambios que se realicen sobre estos datos sean ordenados y coherentes. El caso de uso es utilizado por el servicio de atención al público (SAP) e inicia cuando SAP se identifica en el sistema y luego ingresa en el área “gestión de usuarios”.

**Listar usuario:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los usuarios y entregas existentes en el sistema. Esto permite saber, en el momento que se necesite, los usuarios que existen y, de ser necesario modificar los existentes. El caso de uso es utilizado por el servicio de atención al público (SAP) e inicia cuando SAP se identifica en el sistema y luego ingresa en el área “gestión de usuarios”.

**Borrar usuario:** el caso de uso busca modelar la funcionalidad del sistema que permite eliminar el/los usuarios que todavía no tengan asignados empleados. El caso de uso es utilizado por servicio de atención al público (SAP) inicia cuando SAP se identifica en el sistema y luego ingresa en el área “gestión de usuarios”.

#### **4.5.1.6. Sección “Log”**

**Listar log:** el caso de uso busca modelar la funcionalidad del sistema que permite crear un reporte de los *logs* existentes en el sistema. Esto permite saber, en el momento que se necesite, las acciones dentro de la aplicación tales como: errores, sesiones, entre otros, así como los usuarios que intervinieron. El caso de uso es utilizado por el administrador e inicia cuando el administrador se identifica en el sistema y luego ingresa en el área “log”.

#### **4.5.1.7. Sección “Reportes”**

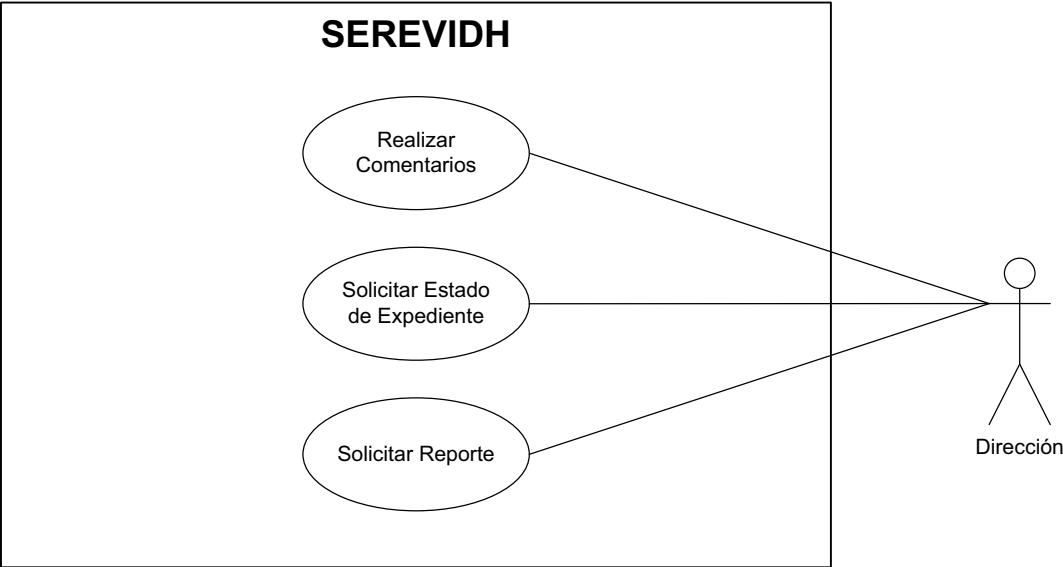
**Generar reportes:** el caso de uso busca modelar la funcionalidad del sistema que permite crear reportes de información almacenada en el sistema con el fin de mostrar estadísticas para contribuir con la toma de decisiones o como una simple extracción de datos.

### 4.5.2. Diagramas de casos de uso

En este apartado se muestran los diagramas de casos de uso, los diagramas están clasificados por actores.

#### 4.5.2.1. Casos de uso del actor “Dirección”

Figura 1. Casos de uso del actor “Dirección”



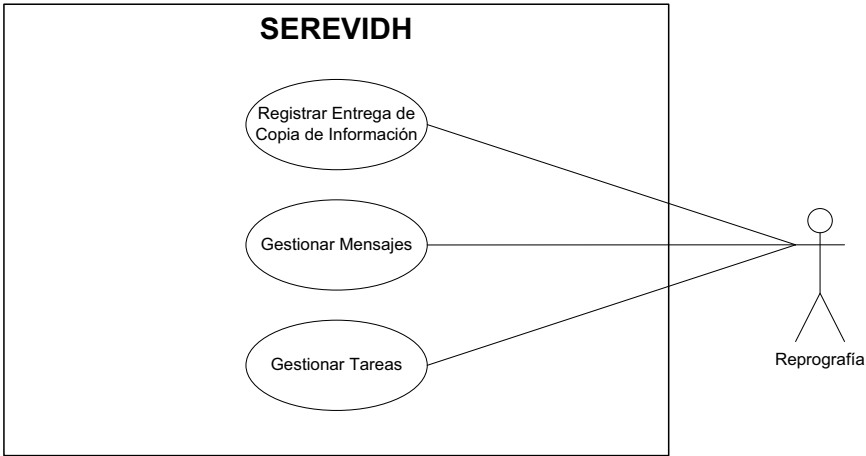
### 4.5.2.2. Casos de uso del actor “Coordinación”

Figura 2. Casos de uso del actor “Coordinación”



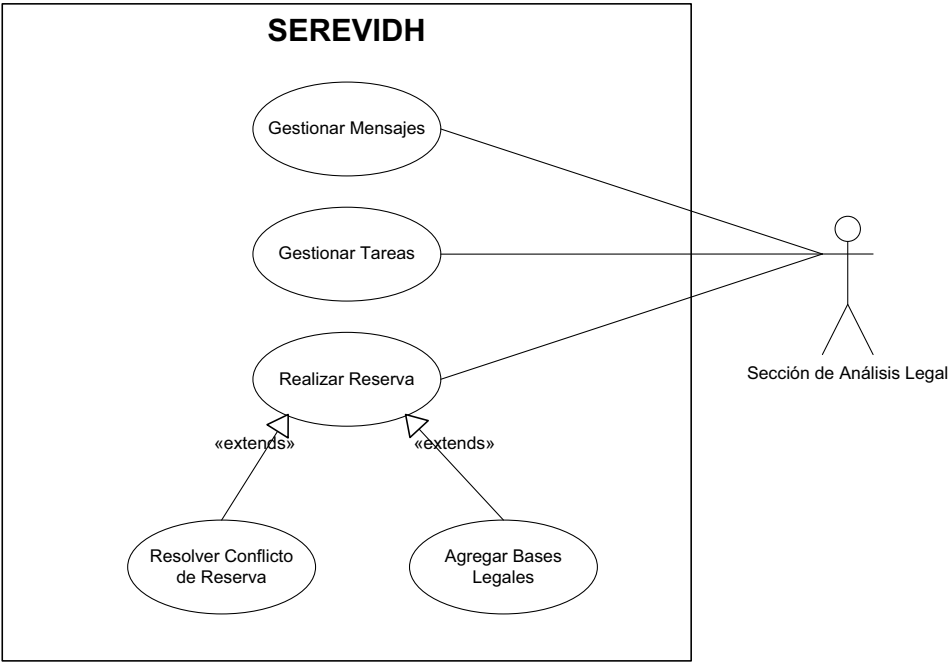
### 4.5.2.3. Casos de uso del actor “Reprografía”

Figura 3. Casos de uso del actor “Reprografía”



### 4.5.2.4. Casos de uso del actor “Sección de análisis legal”

Figura 4. Casos de uso del actor “Sección de análisis legal”





### 4.5.2.5. Casos de uso del actor “Sección de investigación”

Figura 5. Casos de uso del actor “Sección de investigación”



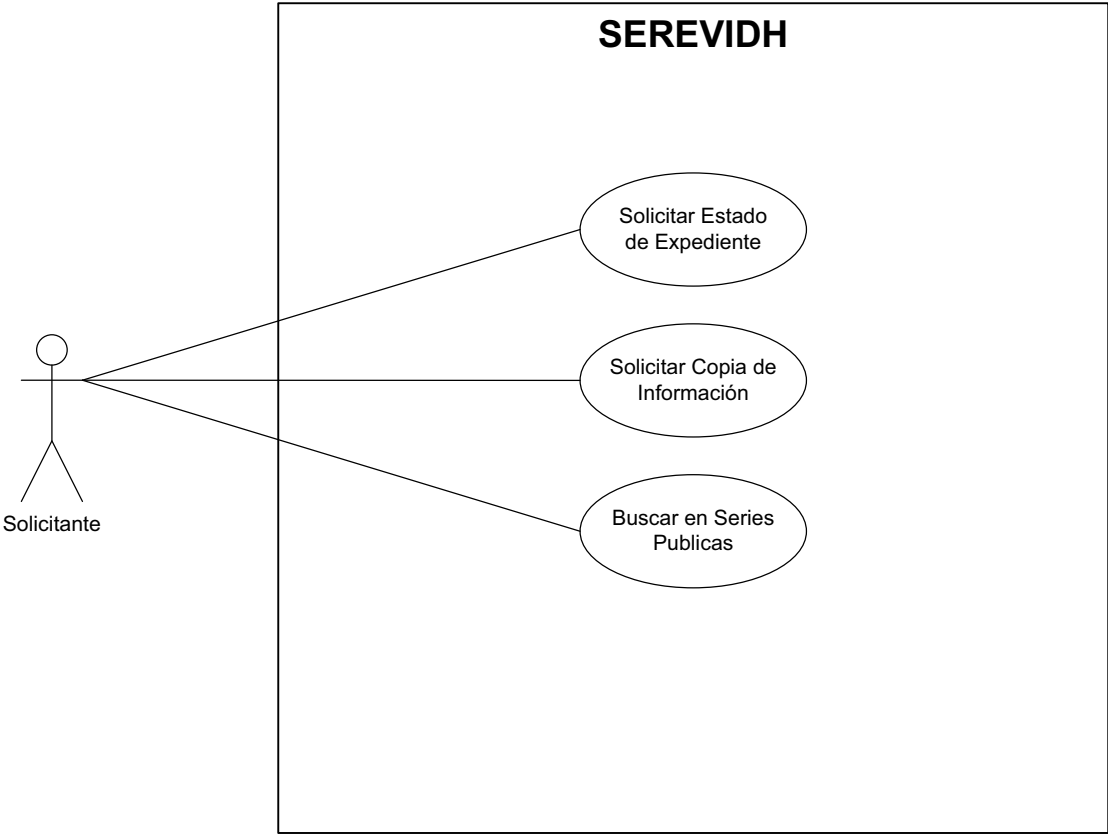
### 4.5.2.6. Casos de uso del actor “Sección de atención al público”

Figura 6. Casos de uso del actor “Sección de atención al público”



**4.5.2.7. Casos de uso del actor “Solicitante”**

Figura 7. Casos de uso del actor “Solicitante”



### 4.5.2.8. Diagrama general de casos de uso

Figura 8. Diagrama general de casos de uso

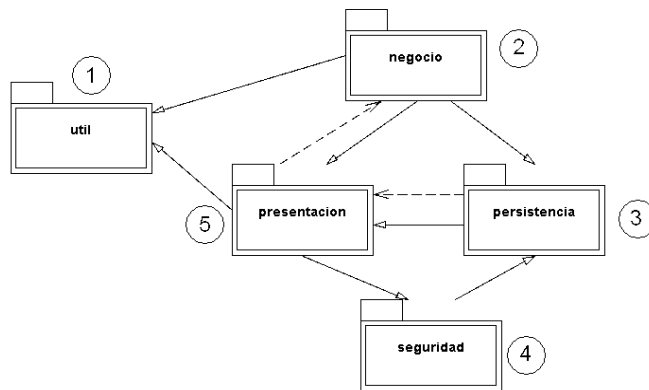


## 4.6. Vista lógica

La vista lógica describe las clases más importantes, tomando en cuenta que estas se agrupan en paquetes para proporcionar servicios; como los paquetes, forman subsistemas y, como estos subsistemas, se organizan en capas. También se describen los casos de uso de mayor importancia. Se muestran diagramas de clases para ilustrar las relaciones entre las clases, paquetes, subsistemas y capas de mayor importancia dentro de la arquitectura de la aplicación.

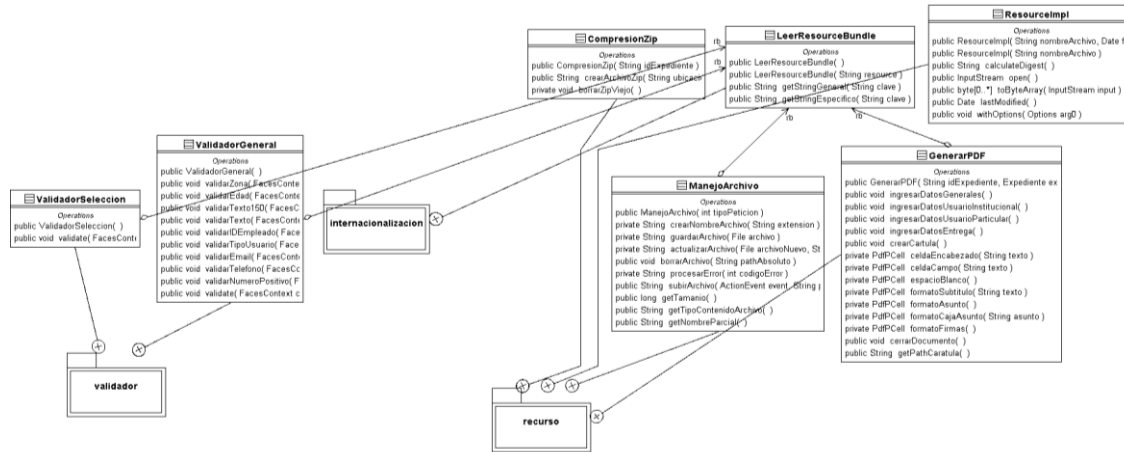
La figura 9 muestra el diagrama de paquetes y las relaciones de asociación directa e indirecta que existen entre estos.

Figura 9. Diagrama de paquetes



#### 4.6.1. Paquete “util”

Figura 10. Paquete “util”



En la figura 10 se observa el paquete *util* y las relaciones entre sus clases. Las clases de este paquete cuentan con métodos de propósito general, que funcionan como apoyo a los métodos de las clases principales. En este paquete se encuentran clases como: validadores, internacionalización, gestión de archivos y exportación de archivos, los cuales son descritos en la tabla II.

Tabla II. Descripción de las clases del paquete “util”

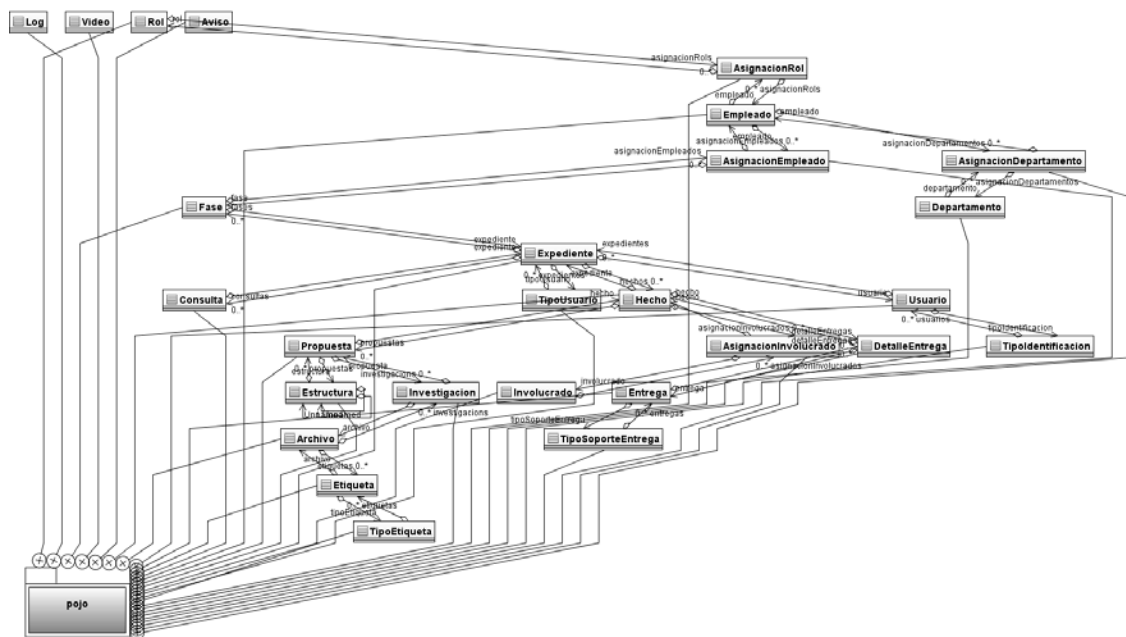
Clase	Descripción
<i>ValidadorSelección</i>	Esta clase es un validador personalizado, que se encarga de validar la selección de un valor en los <i>comboBox</i> .
<i>ValidadorGeneral</i>	Esta clase se encarga de validar datos más generales, como: rangos de fechas, el ingreso números positivos en edad, etc.
<i>CompresiónZip</i>	Se encarga de generar archivos <i>zip</i> , donde se le envía el <i>path</i> o ruta de los archivos que serán empaquetados.
<i>LeerResourceBundle</i>	Se encarga de leer los archivos de internacionalización, donde se encuentra el soporte para distintos idiomas.
<i>ResourceImpl</i>	Se encarga de cargar llamar a los archivos de internacionalización correspondientes a las configuraciones del navegador <i>web</i> .
<i>ManejoArchivo</i>	Se encarga de abrir, crear, borrar, modificar archivos que se encuentran en el disco duro.
<i>GenerarPDF</i>	Se encarga de exportar a formato PDF los archivos.

## 4.6.2. Paquete “negocio”

Este paquete se compone de otros como los pojo (acrónimo de *Plain Old Java Object*), reporte y servicio. Estos paquetes ayudan a implementar la lógica del negocio.

### 4.6.2.1. Paquete “pojo”

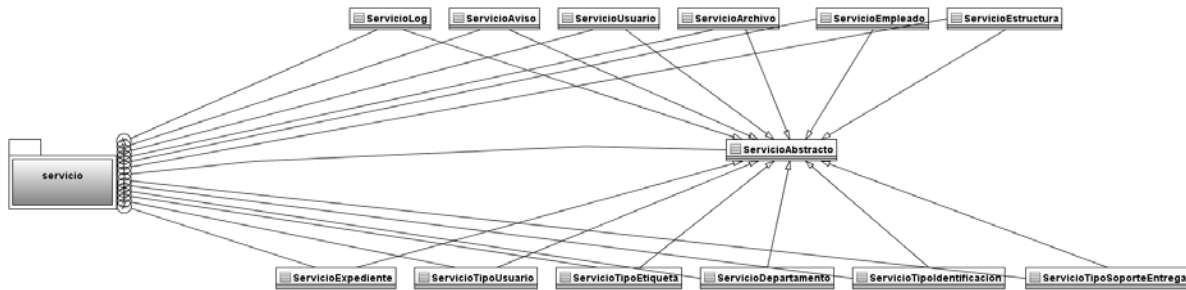
Figura 11. Paquete “pojo”



En la figura 11 se muestra el paquete “pojo”: contiene las clases que están asociadas a las entidades de la base de datos relacional, por lo que para entenderlas mejor, ver el capítulo “Mapeo de Datos”.

#### 4.6.2.2. Paquete “servicio”

Figura 12. Paquete “servicio”



En la figura 12 se muestran las relaciones entre las clases del paquete “servicio”. Este paquete contiene la lógica para manejar los diferentes accesos a la base de datos, y hacer los respectivos ingresos, eliminaciones, modificaciones o consultas.

Todas las clases de este paquete trabajan de forma similar, creando sesiones a la base de datos y accediendo a esta por medio de *Hibernate*. Las clases contienen métodos para devolver listas de datos asociadas a tablas en la base de datos, así como métodos para borrar registros concretos, o búsquedas por parámetros. En la tabla III se detalla la descripción de las clases de este paquete.

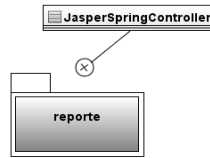
Tabla III. Descripción de las clases del paquete “servicio”

Clase	Descripción
<i>ServicioAbstracto</i>	Clase abstracta que encapsula lógica común a todos los servicios. Todos los servicios se extienden de esta clase para incorporar la lógica de agregar, actualizar y borrar, la cual es la misma para todos los servicios, la única lógica diferente entre servicios es la lógica de búsquedas ya que esta depende el tipo de información que se necesite.
<i>ServicioExpediente</i>	Se encarga de proporcionar la lógica de negocio necesaria para las operaciones relacionadas de forma directa con la información de un expediente.
<i>Servicio</i>	Al igual que el servicio “expediente”, las demás clases manejan la lógica necesaria para las operaciones relacionadas de forma directa de los <i>pojo</i> que tienen asociados.



### 4.6.3. Paquete “reporte”

Figura 13. Paquete “reporte”



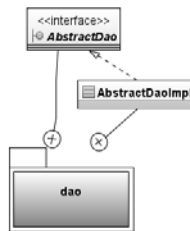
En la figura 13 se muestra la única clase de este paquete llamada “*JasperSpringController*”, responsable de la lógica para generar los reportes. En la tabla IV se detalla el funcionamiento de esta clase.

Tabla IV. Descripción de las clases del paquete “reporte”

Clase	Descripción
<i>JasperSpringController</i>	Clase que se encarga de generar los reportes, esto lo hace por medio del acceso a la base de datos para la obtención de los resultados por medio de una consulta, posteriormente generando un reporte en formato PDF por medio de una plantilla gestionada por <i>JasperReport</i> .

### 4.6.4. Paquete “persistencia”

Figura 14. Paquete “persistencia”



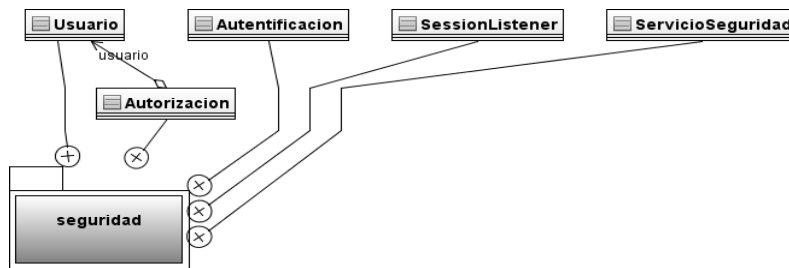
En la figura 14 se muestran las clases del paquete “persistencia” y sus relaciones. Estas clases ofrecen métodos para acceder a la base de datos, tales como: *select*, *insert*, etc. El detalle de estas clases se puede ver en la tabla V.

Tabla V. Descripción de las clases del paquete “persistencia”

Clase	Descripción
<i>AbstractDao</i>	Clase abstracta que contiene métodos que realizan operaciones básicas con la base de datos: <i>select</i> , <i>insert</i> , <i>update</i> y <i>delete</i> .
<i>AbstractDaoImpl</i>	Esta clase es la implementación de la clase <i>AbstractDao</i> , además de manejar las operaciones proporcionadas por la clase <i>HibernateDaoSupport</i> incluida en el <i>framework Hibernate</i> .

#### 4.6.5. Paquete “seguridad”

Figura 15. Paquete “seguridad”



En la figura 15 se demuestra que este paquete de seguridad provee de clases y métodos que utilizan las clases de persistencia para luego ser usadas por las clases de presentación. Las clases y métodos de este paquete garantizan que los usuarios solo puedan acceder a las páginas con permisos habilitados, según los roles que manejen. El detalle de estas clases se puede apreciar en la tabla VI.

Tabla VI. Descripción de las clases del paquete “seguridad”

Clase	Descripción
<i>Autorización</i>	Clase encargada de garantizar si un usuario tiene roles asociados.
<i>Usuario</i>	Clase que lleva el control de los empleados y las acciones que estos realicen.
<i>Autenticación</i>	Clase que valida si los usuarios pueden manejar solamente su información, y protegiendo la ajena.
<i>SessionListener</i>	Clase encargada de crear o cerrar la sesión de un usuario, según sea el caso.
<i>ServicioSeguridad</i>	Clase encargada de acceder a la base de datos para obtener la información de los empleados, y los roles asociados a estos.

#### 4.6.5. Paquete “presentación”

Figura 16. Paquete “presentación”



En la figura 16 se visualizan los paquetes contenidos en “presentación”. Este paquete maneja la interfaz grafica del usuario, para ello utiliza varios paquetes donde las clases de estos manejan métodos que proporcionan funcionalidades específicas de comunicación y presentación para cada pantalla. El detalle de los paquetes se puede apreciar en la tabla VII.

Tabla VII. Descripción de los paquetes dentro del paquete “presentación”

<b>Paquete</b>	<b>Descripción</b>
<i>Index</i>	Contiene las clases que manejan el encabezado general, y el menú de índices entre páginas.
<i>Reportes</i>	Contiene las clases respectivas de las páginas en donde se seleccionan los diferentes reportes, así como el ingreso de parámetros a los mismos.
<i>Departamento</i>	Maneja las páginas y clases relacionadas con el departamento como: crear nuevos departamentos, o agregar empleados a los departamentos.
<i>Errores</i>	Administra los errores que ocurren dentro de la aplicación, y los proporciona de manera legible al usuario.
<i>Archivo</i>	Controla las diferentes opciones de los archivos como: crear estructura archivística, cargar archivos, compartir archivos, etc.
<i>Empleado</i>	Contiene las clases y páginas que controlan la información de los empleados, así como crear nuevos empleados, modificar, y asignar permisos, entre otros.
<i>Usuario</i>	Maneja las páginas y clases de los usuarios, así como los listados de estos.
<i>Log</i>	Controla las modificaciones, o avisos que el sistema genera, proveyendo una página para la gestión de estos.
<i>Expediente</i>	Paquete que contiene las clases y páginas encargadas de manejar los expedientes, así como de crear un expediente e iniciar un proceso, hasta crear bitácoras y llevar el seguimiento de los mismos, hasta la finalización y entrega de información de los mismos.

## 4.7. Vista de procesos

La vista de procesos dentro de la arquitectura de *software* proporciona un grado de detalle bastante amplio con respecto al orden en el que los procesos son ejecutados, así como también describe los procedimientos asociados dentro del sistema y la manera en que son ejecutados.

El modelo de procesos que se muestra a continuación, ilustra el camino registrado de las clases a utilizar en el proyecto, vistas como un conjunto de procesos ejecutables. El proceso global describe la forma en que el sistema funcionará, es decir, abstraerá por completo la lógica del funcionamiento de los mismos, tratando de llegar a un nivel en el que sea absolutamente comprensible la secuencia de ejecución y la dependencia de los mismos.

### 4.7.1. Involucrados

El proceso del sistema interno del Serevidh está compuesto por los siguientes involucrados o participantes:

**El usuario o solicitante:** entidad, investigador o persona cualquiera que desea obtener información, presenta la información necesaria para empezar el proceso de búsqueda.

**Sección de atención al público:** departamento del Serevidh encargado de interactuar con el usuario, validar la información brindada por el mismo, y generar la solicitud necesaria para la creación de un expediente para el usuario, el cual posteriormente pasará a una siguiente fase de búsqueda.

**Sección de investigación:** departamento del Serevidh que, según la información brindada en el expediente para un usuario determinado, procederá a realizar la búsqueda necesaria dentro del archivo histórico, identificando aquellos hechos relacionados.

**Sección de control legal:** una vez identificada la información necesaria para un expediente, este departamento se encarga de validar que la información brindada esté legalmente disponible para el usuario.

**Coordinación:** encargada de verificar la información a entregar, resolver aquellos expedientes que tengan algún tipo de conflicto y autorizar las respuestas.

**Sección de reprografía:** sección encargada de generar las copias digitales o impresas que serán entregadas al usuario mediante la sección de atención al público.

#### **4.7.2. Descripción del proceso**

Al iniciar el proceso el usuario que realiza la solicitud, puede hacerlo por diversos medios tales como: teléfono, correo, *fax email* o de forma presencial; en tal caso atención al público recibirá dicha solicitud.

**Caso 1: Solicitud presencial:** el usuario deberá llenar un formulario de solicitud con la información necesaria y esperar la respuesta por parte del SEREVIDH, dicho tiempo de respuesta está definido por el reglamento interno, en caso de no existir la información se notifica al usuario y se finaliza el proceso, de lo contrario, el usuario puede optar en verificar la información en las instalaciones utilizando una maquina interna para la revisión.

**Caso 2: Solicitud telefónica:** en caso el usuario solicite información por vía telefónica este deberá proporcionar la información necesaria para la solicitud, donde posteriormente atención al público se encargará de verificar en la información pública si existe dicha información; si este fuera el caso, se notifica al usuario para que proceda a presentarse en las instalaciones y llenar el formulario necesario para iniciar una búsqueda más profunda.

**Caso 3: Solicitud por *email*, *fax* o correo:** el usuario puede optar por llenar el formulario de solicitud y enviarla por cualquiera de estos medios, para que la sección de atención al público se encargue de verificar la existencia de la información, debido a que, según las restricciones de la institución, la entrega de documentos debe ser personal y presencial, por lo cual solo se responderá si existe información; en caso de existir, el usuario debe presentarse para realizar el trámite presencial.

Prosiguiendo el proceso en la sección “atención al público”, se valida que la solicitud sea nueva. En caso de no serlo, se comprueba la identificación del solicitante y su capacidad para acceder a la información solicitada; en caso de cumplir con la identificación se puede asignar una máquina para la revisión de la misma y, posteriormente, emitir una copia digital o impresa. De lo contrario, si no cumple con la identificación para el acceso a la información, se toman las restricciones necesarias según el reglamento del Serevid.

En caso de ser una solicitud nueva, esta puede presentarse mediante cualquiera de las vías anteriores: encargados de la sección de atención al cliente deberán:

**Caso 1: Solicitud telefónica:** llenar el formulario con los datos del usuario mediante el mensaje entregado por este medio.

**Caso 2: Solicitud por *fax*, *email* o correo:** se asigna una solicitud mediante la información recibida en el mensaje enviado por este medio.

**Caso 3: Solicitud presencial:** se recibe el formulario llenado por el usuario.

En cualquiera de los tres casos anteriores, si se obtiene la información de la forma correcta mediante el formulario de solicitud, se registra dicha información y se crea un expediente, luego se identifica el usuario por tipo de solicitud.

Se procede a sugerir series donde puede existir información relacionada al expediente. Si en la propuesta de series no existe información pública, se genera una respuesta inmediata para el usuario, indicando que no existe información que pueda entregarse.

En caso de existir información pública se asigna directamente al usuario las series propuestas, brindándole una computadora para la consulta, en caso de ser necesario y/o permitiendo a este obtener una copia digital o impresa de la información.

Si la información no es pública y la solicitud no fue presencial se indica al usuario que se recomienda realizar la solicitud presencial. Si existe dentro de los archivos del Serevidh algún tipo de información relacionada con el expediente, se verifica si el usuario es de tipo investigador externo; en este caso se asigna una computadora especial para él o ella, registrando su uso, para luego poder brindarle una copia digital o impresa, según el investigador externo lo requiera.

Si el usuario no es un investigador externo, la propuesta de series pasa a una revisión en la sección de investigación para buscar más información relacionada con las series propuestas, generándose una bitácora de búsquedas. En caso de sobrepasar el límite de tiempo para búsquedas, según el reglamento del Serevidh, se le notifica al usuario de la necesidad de una prórroga en la entrega mediante la sección de atención al público.

En caso de no encontrar información relacionada, incluso con la prórroga, se procede a notificar al usuario que no existe información, mediante la sección de atención al público. De lo contrario, se verifica si la información necesita algún tipo de reserva legal, sino se entrega al coordinador para que revise dicha información y este autorice la respuesta, para que atención al público notifique al usuario la respuesta y, al presentarse este se entregue la propuesta de series para que consulte de manera interna, y/o solicite la entrega impresa o digital.

Si la información encontrada por sección de investigación necesita algún tipo de reserva legal, se genera una solicitud de revisión de propuesta con reserva y la sección de análisis legal verifican si se aplican las bases legales; si son necesarias, se procede a enviar el expediente al coordinador para su revisión final y este da el visto bueno para que atención al público entregue al usuario la información a través de los medios antes mencionados.

De existir en un desacuerdo interno con la reserva legal, se entrega el expediente al coordinador para que decida como resolver el conflicto, para luego autorizar la respuesta y que atención al cliente se encargue de notificar al usuario, y se brinde la información por los medios antes descritos.

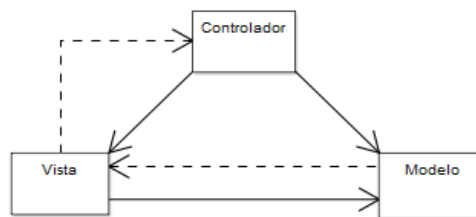


Al crear un expediente, se realiza el siguiente proceso, se crea una hoja de control para el mismo, se verifica si la solicitud fue vía telefónica: en caso de no serlo se escanea el documento de identificación, siendo este un documento necesario solicitado por el Serevidh, se almacena en la forma de archivo físico y digital. Si la solicitud fue vía telefónica, únicamente se archiva de forma física y digital la solicitud, sin documento de identificación solicitado.

#### 4.8. Vista de implementación

Con base al problema y la solución que planteamos, que consiste en el uso de varios *frameworks*, utilizaremos el patrón MVC (Modelo Vista controlado) que se muestra en la figura 17, debido a que nos permite separar la los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Figura 17. Esquema modelo-vista-controlador



- **Modelo:** contiene la información y la lógica con la que el sistema funciona. Esta capa también es llamada “persistencia” y es la que contiene los datos a los que el usuario quiere acceder.
- **Vista:** esta es la representación de la información en un formato entendible para el usuario. Esta capa es llamada también “presentación”, y la única que utiliza el usuario, por la cual solicita y envía información.

- **Controlador:** es el encargo de responder a las acciones que solicite el usuario por medio de la vista, estas acciones la mayor parte del tiempo son cambios al modelo. Esta capa contiene la lógica del negocio, y tiene que garantizar la comunicación entre la vista y el modelo, logrando resolver las solicitudes del usuario, sin comprometer la integridad de la capa de persistencia.

#### 4.8.1. Elección de capas y componentes

El sistema que se creará tiene que funcionar en el entorno de una *intranet*, pero a la vez tiene que poder acoplarse fácilmente para que en el futuro pueda ser utilizado por medio de *Internet*, por lo que la arquitectura elegida nos debe permitir la implementación de lo que hemos creado, en un ambiente de Internet, además de que nos deje agregar fácilmente nuevas funcionalidades al sistema, sin necesidad de cambiar las ya existentes.

La elección del patrón MVC fue porque nos permite la separación del sistema en tres capas, dejando la vista o presentación al usuario, separada de la lógica y de los datos en cuestión. Este tipo de arquitectura, se ajusta a nuestras necesidades ya que podemos reutilizar el código y no se necesitan hacer grandes cambios para que la aplicación funcione en *Internet* o *intranet*.

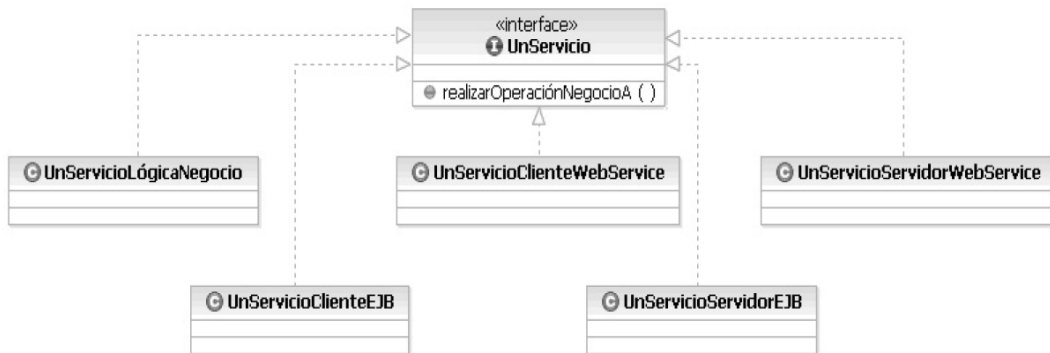
Para la capa de presentación (la vista) se buscaba un *framework* que nos permitiera una mayor facilidad en la elaboración de pantallas, la toma de datos de los usuarios por medio de formularios y sus clases en el servidor, la validación, conversión, gestión de errores, que pudiéramos trasladar la aplicación a varios idiomas de manera sencilla y, de ser posible, que facilitase también incluir componentes complejos (menús, árboles, AJAX, etc.) de una forma sencilla y fácil de mantener.

En la capa de negocio y persistencia, se utiliza una solución basada en servicios, porque no es flexible y el mantenimiento no es tan difícil, además de que permite la integración con *Web Services* de una manera limpia. La persistencia de las clases se sustenta en DAO (Objetos de Acceso a Datos), manteniendo aislada la capa de persistencia de la capa de negocio. Tanto los servicios como los DAO, así como el propio modelo son realmente POJO (clases simples de Java), con la simplicidad que conllevan y sin dependencias reales con ningún *framework* concreto. Para realizar esta integración se ha elegido *Spring*.

Para la capa de persistencia se buscó una herramienta que nos permitiera trabajar con el mapeo relacional, además de que pudiéramos cambiar de base de datos sin mayor dificultad, debido a que utilizamos como RDBMS *MySQL*, pero por el crecimiento de la información con la que cuenta el sistema, puede que sea necesario migrar a un RDMBS más robusto y potente. Para solucionar estos problemas utilizaremos *Hibernate*, que es justamente lo que se ajusta a nuestras necesidades, ya que nos permite un mapeo objeto/relacional y es independiente del RDBMS a utilizar.

Las jerarquías son sencillas y limpias. *Spring* gestiona todos estos objetos y sus dependencias tal y como se muestra en la figura 18.

Figura 18. Gestión por medio de *Spring*



## 4.8.2. Beneficios de la arquitectura diseñada

La arquitectura a utilizar permite un diseño modular e independiente por capas, lo que hace que se simplifique su desarrollo y que los cambios o mejoras sean más fáciles de hacer, así como encontrar y corregir errores. Esto es de suma utilidad ya que grandes cambios no requieren grandes modificaciones al desarrollo total, sino simplemente al modulo que se está desarrollando.

Además de modular, el hecho de que el sistema esté separado por capas da la facilidad de poder cambiar herramientas si fuese necesario; por ejemplo, para la capa de presentación se utilizará *ICEFaces*, ya que permite una integración más limpia si quisieras el sistema en *Internet*, pero si los requerimientos cambiaran, y solo van a requerir una aplicación que funcione en la *intranet*, bien se podría dejar por un lado *ICEFaces* y utilizar *Swing*, sin afectar la capa de persistencia, ni la de la lógica.

Entre los beneficios de la arquitectura a utilizar, también se puede mencionar la persistencia, esta se vuelve más sencilla de manejar, ya que abstrae los accesos a la base de datos con *Hibernate*, lo cual permite separar los accesos de los datos, de los datos en sí, lo cual nos da la facilidad de cambiar de RDBMS con facilidad, sin necesidad de cambiar los accesos a los datos previamente creados.

## 4.8.3. Frameworks empleados

### 4.8.3.1. *ICEFaces*

*ICEFaces* es un *framework* para aplicaciones Java desarrolladas para un ambiente web. *ICEFaces* nos ayuda a implementar el patrón MVC, ya que nos sirve para crear interfaces de usuario en aplicaciones de *Java EE* (capa de presentación).

*ICEFaces* proporciona un modelo de *JavaBeans* para enviar eventos (peticiones) desde la interface de usuario, a la capa de la lógica del negocio, que en el contexto del patrón MVC sería el controlador.

#### **4.8.3.2. Spring**

El *framework Spring* es un conjunto de librerías de las cuales se escogen las que faciliten el desarrollo de la aplicación. Entre las características más potentes de *Spring* se halla el contenedor de inversión de control, que consiste en configurar las clases en un archivo XML y definir en él las dependencias. Esto permite que la aplicación se vuelva muy modular, y la ventaja de que el archivo sea un XML es que la aplicación no se vuelve dependiente de *Spring*.

El *framework Spring* trabaja en la capa de la lógica de negocio (controlador), por lo que tiene que ser capaz de poderse comunicar con otros componentes y servir de puente para que estos se comuniquen, lo cual se hace de una manera muy limpia con tecnologías como JSF e *Hibernate*.

#### **4.8.3.3. Hibernate**

*Hibernate* es una herramienta de mapeo objeto-relacional, es de mucha utilidad ya que permite el mapeo de nuestros atributos, que están en una base de datos relacional y a la vez los enlaza con el modelo de objetos de nuestra aplicación. Esto lo hace por medio de relaciones establecidas en un archivo XML.

Esta solución es parte de nuestra capa de persistencia (Modelo), y es de gran utilidad debido a que genera las sentencias SQL, haciendo que el desarrollador no maneje manualmente los datos.

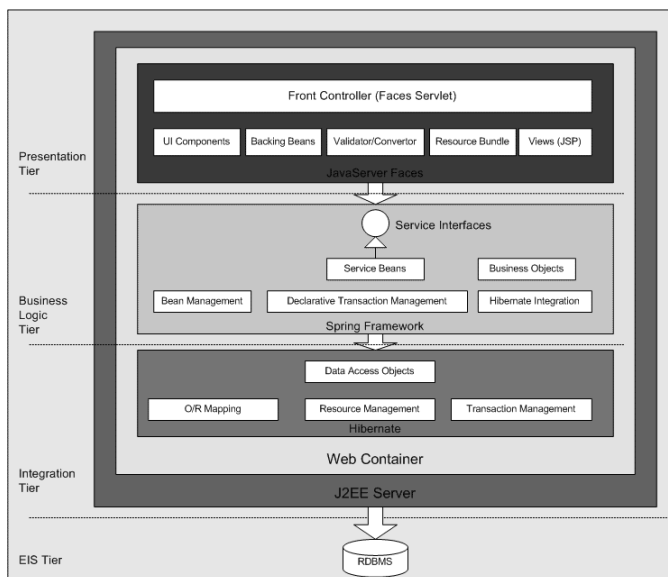
#### 4.8.3.4. *JUnit*

*JUnit* es un conjunto de bibliotecas que permiten hacer pruebas unitarias de aplicaciones Java. Este *framework* sirve principalmente para probar los métodos y asegurar de que su comportamiento sea el que se espera.

#### 4.8.4. Diagrama de la arquitectura

La figura 19 fue tomada de *JavaWorld*, en donde, separando por capas (MVC), muestra el comportamiento de los diferentes *frameworks* que va a implementar (JSF, *Spring*, *Hibernate*). La URL de la imagen es la siguiente: <http://www.javaworld.com/javaworld/jw-07-2004/images/jw-0719-jsf4.gif>

Figura 19. Funcionamiento de MVC



#### **4.8.5. Entornos de desarrollo**

Como entorno de desarrollo se utilizará *NetBeans* en su versión 6.5, este IDE fue desarrollado y es mantenido por *Sun*, por lo que le da peso en el mercado. El uso principal de este IDE es debido a la facilidad del desarrollo que ofrece, ya que permite realizar código automáticamente, además de que cuenta con varios *plug-ins* entre estos el de *JUnit*.

#### **4.9. Vista de despliegue**

En esta sección se describe la configuración del *hardware* sobre el cual será desplegada y ejecutada la aplicación. Se muestra la configuración de los nodos físicos y cómo las tareas establecidas en la vista de procesos se reparten en los nodos.

##### **4.9.1. Servidor de base de datos**

Servidor principal de base de datos. En este servidor se almacena toda la información sobre empleados, usuarios, expedientes, archivos y otra información de vital importancia.

##### **4.9.2. Servidor de aplicaciones**

Servidor principal de aplicaciones, en este servidor se encuentran todos los servicios de la aplicación y servicios de acceso a datos.

### **4.9.3. Computadora interna**

Computadora conectada a la aplicación por medio de la red interna (*intranet*), en esta computadora se carga la interface de usuario para que los usuarios administrativos puedan realizar tareas de mantenimiento y administración de la aplicación.



## 5. ANÁLISIS COSTO-BENEFICIO

El costo del *software* muchas veces tiende a ser subjetivo, por lo que existen diferentes métodos que toman como base diversos parámetros, como: las líneas de código escritas, el número de clases, o el número de métodos, el número de funcionalidades o requerimientos; también por medio de la complejidad del mismo, tomando en cuenta las entradas de datos, salidas o mensajes, accesos a la base de datos, manejo y/o modificación de archivos. Por último, pero en menor medida, se puede determinar el costo del *software* por la ausencia de fallos que esté presente, pero para esto el *software* tiene que estar en ejecución cierto tiempo, para determinar si presenta fallos.

Para determinar el costo del sistema Serevidh, de la manera más objetiva posible, se va a determinar el costo por diferentes métodos y luego a normalizar los resultados de estos costos para determinar un costo estándar.

### 5.1. Datos generales

La tabla VIII muestra datos generales sobre el código fuente del sistema.

Tabla VIII. Datos generales del código fuente

Número de archivos	157
Líneas codificadas	23974
Líneas en blanco	6528
Líneas comentadas	12401
<b>Total de líneas</b>	<b>42903</b>

## 5.2. Estimación por medio de Cocomo

El modelo constructivo de costes (Cocomo, por su acrónimo del inglés *Constructive Cost Model*) es un modelo matemático de base empírica utilizado para estimación de costes de *software*. Incluye tres sub-modelos, cada uno ofrece un nivel de detalle y aproximación, cada vez mayor, a medida que avanza el proceso de desarrollo del software: básico, intermedio y detallado<sup>3</sup>. A la vez, cada sub-modelo también se divide en modos que representan el tipo de proyecto, y puede ser:

- **Modo orgánico:** un pequeño grupo de programadores experimentados desarrollan software en un entorno familiar. El tamaño del *software* varía desde unos pocos miles de líneas (tamaño pequeño) a unas decenas de miles (medio).
- **Modo semi-libre o semi-encajado:** corresponde a un esquema intermedio entre el orgánico y el rígido; el grupo de desarrollo puede incluir una mezcla de personas experimentadas y no experimentadas.
- **Modo rígido o empotrado:** el proyecto tiene fuertes restricciones, que pueden estar relacionadas con la funcionalidad y/o pueden ser técnicas. El problema a resolver es único y es difícil basarse en la experiencia, puesto que puede no haberla.

## 5.3. Estimación con Cocomo básico

El COCOMO básico se utiliza para obtener una primera aproximación rápida del esfuerzo. Para ello es necesario introducir en el primer campo el tamaño de la aplicación en miles de líneas de código (KLDC).<sup>4</sup>

---

<sup>3</sup> Cocomo <http://es.wikipedia.org/wiki/COCOMO>

<sup>4</sup> Medir COCOMO básico <http://www.oei.eui.upm.es/Asignaturas/PInformaticos/ficheros/software/opcion3/CBasico.html>

**Parámetros:**

Miles de líneas de código: **23.974 KLDC**

Tipo de aplicación: **Modo orgánico**

Personas involucradas: **3**

Pago mensual por persona: **Q9 000.00**

**Cálculos:**

$$\text{Esfuerzo obtenido } \left( \frac{\text{personas}}{\text{mes}} \right) = 67.44 \frac{\text{personas}}{\text{mes}}$$

$$\text{Tiempo estimado (meses)} = \frac{\text{esfuerzo obtenido}}{\text{personas involucradas}} = 67.44 \frac{44}{3} = 22.48 \text{ meses}$$

*Costo del sistema = costo por persona \* personas involucradas \* tiempo estimado*

*Costo del sistema = 9000.00 \* 3 \* 22.4*

*Costo del sistema = Q606,960.00*

**5.4. Estimación con Cocomo intermedio**

El esfuerzo se calcula como función del tamaño del producto, modificado por la valoración de los atributos directores del coste, los cuales incluyen una valoración subjetiva del producto, del *hardware*, del personal, etc. Los valores de los diferentes atributos se consideran como términos de impacto agregado al coste total del proyecto. Para ello es necesario introducir en el primer campo el tamaño de la aplicación en miles de líneas de código (KLDC).<sup>5</sup>

---

<sup>5</sup> Medir Cocomo intermedio

<http://www.oei.upm.es/Asignaturas/PInformaticos/ficheros/software/opcion3/CIntermedio.html>

**Parámetros:**

- Miles de líneas de código: **23.974 KLDC**
- Tipo de aplicación: **Modo orgánico.**
- Personas involucradas: **3**
- Pago mensual por persona: **Q9 000.00**
- Selección de grados de influencia:

Tabla IX. Grados de influencia

Atributos de coste	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extremadamente alto
Fiabilidad requerida	X					
Tamaño de BD			X			
Complejidad			X			
Restricción de t. de ejecución				X		
Restricción de memoria			X			
Volatilidad de la máquina virtual		X				
Tiempo de espera		X				
Habilidad del analista			X			
Experiencia en la aplicación		X				
Habilidad del programador				X		
Experiencia en la MV			X			
Experiencia en el lenguaje				X		
Uso de modernas practicas de programación					X	
Utilización de herramientas					X	
Plan de desarrollo requerido			X			

**Cálculos:**

$$\text{Esfuerzo obtenido } \left( \frac{\text{personas}}{\text{mes}} \right) = 35.97 \frac{\text{personas}}{\text{mes}}$$

$$\text{Tiempo estimado (meses)} = \frac{\text{esfuerzo obtenido}}{\text{personas involucradas}} = 35. \frac{97}{3} = 11.99 \text{ meses}$$

Costo del sistema = costo por persona\* número de personas \* tiempo estimado

Costo del sistema = 9000.00\*3\*11.99

Costo del sistema = Q323,730.00

## 5.5. Estimación con *software* de gestión de proyectos

Este tipo de estimación consiste en poder utilizar diferentes recursos como: las aplicaciones de software de estimación de costes, las hojas de cálculo computarizadas, y las herramientas de simulación y estadísticas. Es ampliamente utilizado para asistir en el proceso de estimación de costes. Dichas herramientas pueden simplificar el uso de algunas de las técnicas de estimación de costes y, por consiguiente, facilitar la consideración rápida de las diversas alternativas de estimación de costes.

El *software* de estimación de costes a utilizar será “*Construx Estimate*”, en el que se usarán los siguientes parámetros para la estimación:

- Cantidad máxima para el programa (meses): 10 meses
- Cantidad máxima del personal: 3 personas
- Líneas de código del programa: 23,974
- Cantidad de meses para arreglar errores: 2 meses

A partir de los anteriores parámetros *Construx Estimate*, da como resultado los datos que se muestran en la figura 20 <sup>6</sup>.

De los datos mostrados en la figura 20 se estima que el sistema S se desarrolle en 10 meses, y trabajando 9 personas por mes para ello, con un costo mensual de \$11,513 equivalente en quetzales a Q92104.00 (\$1.00 = Q8.00).

Costo del sistema = 92,104.00\*10

Costo del sistema = Q921,040.00

---

<sup>6</sup> *Software* para la gestión de costos <http://www.construx.com/Page.aspx?nid=68>

Figura 20. Resultados de costo calculado por software

### Proyecto de EPS

Software Accepted

Calibration Type: Project Type (from industry data)

### Scope (Lines of Code)

Expected: 23,816  
 Std Dev: 525 ( $\pm 2\%$ )  
 Min (5th Percentile): 22,916  
 Max (95th Percentile): 24,716

### Nominal Plan

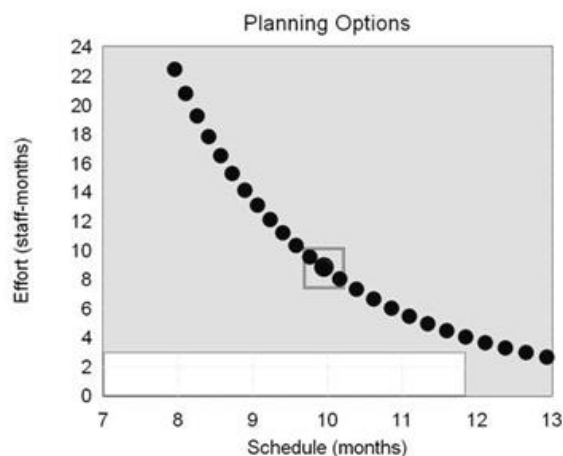
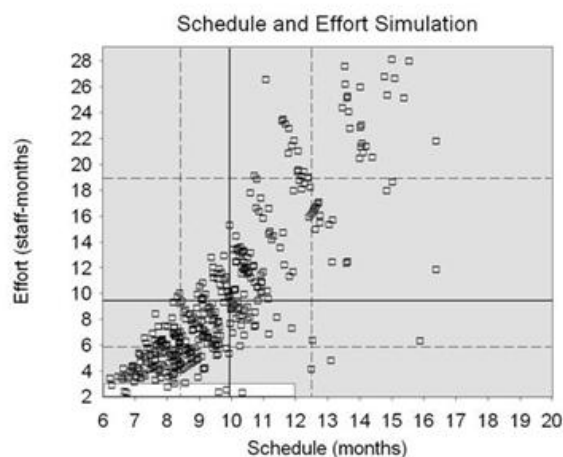
(All priorities equal weighted)

Effort: 9 staff-months  
 Schedule: 10.0 months  
 Peak Staff: 1.4 staff  
 Cost: \$11,513

### Optimum Plan

(priorities set by estimator)

Effort: n/a  
 Schedule: n/a  
 Peak Staff: n/a  
 Cost: n/a  
 Project planning is currently  
 overconstrained by the effort constraint.



## 5.6. Estimación de tarifas de costos de recursos

Para este tipo de estimación la persona que determina las tarifas o el grupo que prepara las estimaciones debe conocer las tarifas de costes unitarios, tales como el coste del personal por hora y el coste del material a granel por yarda o metro cúbico, correspondientes a cada recurso para estimar los costes de la actividad del cronograma.

Reunir cotizaciones es un método de obtener las tarifas. Para los productos, servicios o resultados que deben obtenerse por contrato, se pueden incluir las tarifas estándar con factores de escalamiento en el contrato<sup>7</sup>.

Es importante señalar que este tipo de estimación refleja el costo de las personas en los meses trabajados, sin tomar en cuenta el tiempo que se dedicó por día, ni el tamaño o la complejidad del sistema en sí.

Tabla X. **Cantidad de personas y costos**

Cantidad de personas	3
Meses de producción (análisis del proceso, toma de requerimientos, análisis, diseño y codificación)	8
Meses de implementación y pruebas	2
Costo mensual por persona	Q9,000.00

***Costo del sistema = costo mensual \* numero de meses \* cantidad de personas***

Costo del sistema = 9000.00\*10\*3

Costo del sistema = Q270,000.00

## **5.7. Normalización de costos**

La principal razón para normalizar los costos es debido a que los métodos representan algunas deficiencias y no toman en cuenta muchos factores indispensables en el desarrollo de sistemas de información y comunicación. Entre las deficiencias o factores que no toman en cuenta los métodos podemos listar:

---

<sup>7</sup> Estimación de tiempos, costos y recursos  
<http://www.mitecnologico.com/Main/EstimacionDeTiemposCostosYRecursos>

**Cocomo básico:**

- Como único factor toma las líneas de código escritas, dejando por un lado las líneas documentadas.
- No toma en cuenta las diferentes etapas del desarrollo del *software*, únicamente se centra en la fase de codificación.
- No mide la complejidad del sistema.

**Cocomo intermedio:**

- Esta métrica toma muchos factores en el desarrollo del sistema, así que tiene una objetividad mayor, por lo que se tomará como costo base, para normalizar el costo.

***Software* de gestión:**

- Al igual que Cocomo básico solo toma en cuenta las líneas de código finales, por lo que su representatividad no es muy objetiva, ya que excluye a muchos factores importantes para realizar su análisis.
- Lo interesante de esta herramienta es que muestra en un gráfico la variación entre personas que desarrollan el sistema y los meses que se tardaría en producirlo. Como se pudo observar en la figura 20, se estableció que para cumplir con la entrega del sistema en 10 meses, era necesario tener a nueve personas; pero el sistema fue desarrollado en 10 meses por tres personas, por lo que se va a descartar este método.

**Estimación de Costes de Recursos:**

- La principal deficiencia de este método, es que solo toma en cuenta el tiempo de entrega del *software*, y el costo de cada persona por mes, excluyendo la complejidad del sistema y el dominio/aprendizaje de las herramientas.



- Otro aspecto que no toma en cuenta es el tiempo invertido por día, por lo que días de extenso trabajo (más de 8 horas al día) no son reflejados en el costo final del sistema.

## 5.8. Resumen de costos

La Tabla XI muestra un resumen de todos los costos obtenidos con los distintos métodos para cálculos de costos de proyectos de software.

Tabla XI. **Resumen de costos**

Método	Costo
Cocoma Básico	Q 606,960.00
Cocoma Intermedio	Q 323,730.00
Software de Gestión	Q 921,040.00
Estimación de Costes de Recursos	Q 270,000.00

## 5.9. Costo del sistema

Para normalizar el costo, utilizaremos la siguiente fórmula:

$$\text{Costo del sistema} = \frac{\text{COCOMO Básico} + 4(\text{COCOMO Intermedio}) + \text{Tarifas de costos}}{6}$$

$$\text{Costo del sistema} = \frac{606,960.00 + 4(323,730) + 270,000}{6}$$

$$\text{Costo del sistema} = \mathbf{Q\ 361.980.00}$$



## 6. LICENCIAMIENTO

En el mundo actual, de una u otra forma se ha escuchado hablar sobre lo que son las patentes y licencias en relación con una gran diversidad de productos. Con el paso del último siglo y el surgimiento del *software* como un producto más dentro del mercado, no tardó en ser sometido a las mismas reglas del juego, pero la estructura del *software* añadió una complejidad enorme a la interpretación de las leyes, que de una forma u otra, intentan proteger los derechos de los usuarios, así como a sus propietarios (en caso de haberlos).

La "propiedad" sobre un invento, o descubrimiento, ha acompañado en la historia al desarrollo industrial prácticamente desde sus inicios como uno de los pilares del factor financiero. Esto ha suscitado un vivo debate, durante todos los períodos por los que ha pasado el registro de estas actividades, generalmente debido a cuestiones técnicas en la naturaleza de los inventos o descubrimientos.









Dado que el *software* ha tenido una amplia aceptación y difusión en las últimas dos décadas, aprovechando que la comunicación se ha ampliado debido a la implementación de redes masivas de distribución como el Internet, ahora los sectores industriales como el de la música han entrado en la discusión acerca de la naturaleza de las patentes y licencias, como parte específica de su desarrollo.

No se olvida que para difundir un fragmento musical o una fotografía, hace falta un programa informático que lo haga posible, y dentro de éste unas implementaciones en cuanto a estándar de almacenamiento, compresión, empaquetado, que a su vez son otros productos informáticos, que pueden estar protegidos por cláusulas restrictivas, que pararían el proceso de libre acceso, de no atenderse a la legalidad de su uso.

Antes de definir un tipo de licencia a utilizar es necesario entender algunas definiciones de ciertos términos que en muchos casos tienden a confundirse o mal interpretarse, primeramente se debe entender que las patentes y las licencias son términos totalmente distintos.

Existen diferentes tipos de licencias que cuentan con características cuyas definiciones fueron definidas por una entidad gubernamental o empresa.<sup>8</sup> Para entender mejor los alcances de cada tipo de licencia se definen las características mostradas en la figura 21.

Figura 21. Características de licencias de *software*

-  Garantiza al usuario las cuatro libertades del Software Libre.
-  Garantiza el acceso al código fuente.
-  Los cambios realizados deben ser publicados.
-  El software derivado debe tener la misma licencia.
-  Utilizar el *software* a través de la red cuenta como distribución.
-  Contiene garantía contra patentes.
-  Contiene garantía contra tivoización.
-  Contiene condiciones poco usuales que deben estudiarse.

NOTA: Por ser un proyecto definido para un ente público sin fines de lucro y elaborado bajo la representación de la Universidad de San Carlos para el pueblo de Guatemala, se descarta la utilización de una licencia privativa para la definición del proyecto.

---

<sup>8</sup> Patentes y tipos de licencia de *software*  
[http://doc.ubuntu-es.org/Patentes\\_y\\_tipos\\_de\\_Licencias\\_de\\_Software](http://doc.ubuntu-es.org/Patentes_y_tipos_de_Licencias_de_Software)

En la figura 22 se muestran algunas de las licencias más reconocidas por la *Free Software Foundation* (FSF).

Figura 22. Licencias reconocidas por FSF<sup>9</sup>

































Name	Icons
BSD 2-Clause	fs src C C -> pat !
zlib	fs src C C -> pat !
MIT	fs src C C -> pat !
ruby	fs src C C -> pat !
LGPLv2	fs src C C -> pat !
GPLv2	fs src C C -> pat !
MS-PL	fs src C C -> pat !
AFL	fs src C C -> pat !
CPL	fs src C C -> pat !
EPL	fs src C C -> pat !
APL	fs src C C -> pat !
OSL	fs src C C -> pat !
MPL1.0	fs src C C -> pat !
CDDL	fs src C C -> pat !
MPL1.1	fs src C C -> pat !
LGPLv3	fs src C C -> pat !
GPLv3	fs src C C -> pat !

<sup>9</sup> Tipos de Licencias  
<http://bitelia.com/2009/12/las-licencias-de-software-libre-explicadas-con-icenos>

## 6.1. Licencias y tecnologías implementadas

En el proyecto desarrollado para el Serevidh se utilizaron un conjunto de tecnologías para su implementación, cada tecnología está desarrollada bajo alguna licencia de *software*, por lo cual se debe tomar en cuenta dichas licencias para tomar una decisión sobre cuál de ellas se ajusta más para las necesidades del proyecto sin afectar las limitaciones de las otras. La figura 23 muestra las licencias de las herramientas utilizadas.

Figura 23. Licencias de Herramientas

▪ <b>Hibernate:</b> GNU Lesser General Public License (LGPL) v2.1	
LGPLv2	       
▪ <b>IceFaces:</b> Open Source MPL	
MPL1.0	       
▪ <b>Jasypt:</b> Apache Software License v2	
▪ <b>Spring:</b> Apache Software License v2	
▪ <b>Log4j:</b> Apache Software License v2	
▪ <b>JasperReports:</b> LGPL	
LGPLv2	       
▪ <b>Apache Tomcat:</b> Apache Software License v2	
▪ <b>JUnit:</b> Common Public License (CPL)	
CPL	       

## 6.2. Descripción de licencias

Tabla XII. Aprobación de licencias

	<i>Software libre</i>	Aprobado por OSI	Compatible GPL	<i>Copyleft</i>	Posibilidad de diferentes licencias
CPL	sí	sí	no	sí	sí
<i>Apache License</i> versión 2	sí	sí	sí	no	sí
CDDL	sí	sí	no	limitado	sí
GPL versión 2.0	sí	sí	sí	sí	no
LGPL versión 2.0	sí	sí	sí	sí	sí
MPL versión 1.0	sí	sí	no	limitado	sí
BSD	sí	no	no	no	sí

Según la tabla XII muchas de las licencias comparten la aprobación de OSI y son de tipo *software* libre, pero varias tiene como limitante la compatibilidad con la GPL por lo que se limita la utilización a la licencia: *Apache License*, CDDL, MPL y BSD (dicha licencia es propuesta por ser de mayor adaptabilidad incluso a licencias privativas).

La capacidad limitada de CDDL y MPL no coinciden con *Apache License* en *Copyleft*, causando un conflicto, por lo cual únicamente se puede optar entre: *Apache License* versión 2 y BSD.

Tomando en cuenta las licencias restantes, BSD es capaz de adaptarse a cualquiera de las licencias de las tecnologías utilizadas no importando las restricciones, la *Apache License* también se adapta prácticamente a todas, pero en caso de que en un futuro fuera necesario cambiar las tecnologías puede que las limitaciones que la *Apache License* versión 2 restrinjan el uso de esas herramientas, por lo que finalmente es recomendable utilizar en el proyecto la licencia BSD.





## **FASE DE ENSEÑANZA**



## 7. PLAN DE CAPACITACIÓN

La implementación de sistemas de *software* en una organización es de vital importancia hoy en día para que estos puedan gestionar la información, acelerar los procesos, hacerlos más seguros y medibles. Sin embargo, el trabajo no se limita a la implementación del *software*, se debe incluir al equipo que utilizará el sistema, ya que el éxito o mejora de la organización depende en gran medida del equipo de personas y el desenvolvimiento de estas con el sistema de *software* implementado, por lo que es indispensable asegurarse de que el equipo tenga todas las herramientas y conocimientos necesarios para desempeñar correctamente su labor.<sup>10</sup>

La capacitación es una actividad sistemática, planificada y permanente cuyo propósito general es: preparar, desarrollar e integrar a los recursos humanos al proceso productivo, mediante la entrega de conocimientos, desarrollo de habilidades y actitudes necesarias, para el mejor desempeño de todos los trabajadores en sus actuales y futuros cargos y adaptarlos a las exigencias cambiantes del entorno.

### 7.1. La inducción y el entrenamiento en el puesto

Una vez que se ha reclutado y seleccionado al colaborador deseado, es necesario orientarlo y capacitarlo, proporcionándole la información y los conocimientos necesarios para que tenga éxito en su nueva posición, aunque ya cuente con experiencia en el uso de sistemas de *software* similares.

---

<sup>10</sup> La capacitación empresarial puede acelerar la productividad de tus recursos humanos.  
<http://blogorama.com.mx/Relaciones/la-capacitacion-empresarial-puede-acelerar-la-productividad-de-tus-recursos-humanos/>

- **La inducción.** La inducción es el proceso inicial por medio del cual se proporcionará al individuo la información básica que le permita integrarse rápidamente a las funcionalidades del *software*. Es común que la inducción incluya: los accesos, las pantallas, los permisos, los alcances y limitaciones del sistema, etcétera.
- **Entrenamiento en el puesto.** Una vez terminado el proceso de inducción, el empleado requiere entrenamiento específico sobre el puesto que va a desempeñar o permisos que tendrá sobre el sistema. Para preparar esta información es necesario saber cuáles van a ser sus responsabilidades, quién va a ser su jefe directo y el organigrama de la compañía. Con este proceso, se le dará a conocer de una manera muy clara qué es exactamente lo que se espera de él. Una herramienta necesaria para el proceso antes mencionado es la “descripción del puesto”, la cual debe contener la siguiente información:
  - Título del puesto
  - Departamento al que pertenece
  - Fecha de elaboración
  - Descripción general del trabajo que realizará el ocupante del puesto.
  - Descripción específica detallando punto por punto cada una de las actividades que realizará el ocupante del puesto, de manera muy clara y definida.
  - Si la persona va a tener funciones de Jefatura, enuncie los puestos que va a tener a su cargo.
  - Describa la relación directa e indirecta con otras posiciones similares o superiores dentro de la compañía (organigrama).

## 7.2. Adiestramiento

El adiestramiento va a auxiliar para que una persona aprenda a desempeñar sus labores involucrándose en situaciones reales. El adiestramiento se torna esencial cuando el trabajador ha tenido poca experiencia o en caso de que use el sistema de *software* por primera vez. Para ello existen varias técnicas, pero una que se aplica con mayor éxito es el método de los cuatro pasos:

- Prepare al trabajador.
- Muéstrole el trabajo.
- Póngalo a prueba.
- Sígalo en la práctica.

El siguiente es un método alternativo:

- El instructor dice y hace.
- El instructor dice y el alumno hace.
- El alumno dice y el instructor hace.
- El alumno hace y dice.

No se debe perder de vista que no todos los empleados aprenden a la misma velocidad y que hay algunos que necesitan más tiempo que otros para aprender; es necesario ayudarlos a que esto suceda por el bien de la organización, y el éxito del sistema en sí.

### 7.3. Capacitación y desarrollo profesional

Cuando se habla de capacitación y desarrollo profesional se refiere a la educación que recibe una persona con el fin de estimular su efectividad en la posición que desempeña dentro de la compañía.

Normalmente, la capacitación tiene objetivos a corto o mediano plazo y busca desarrollar una capacidad específica, como por ejemplo: un curso de *MS Excel*. En contraste, el desarrollo profesional busca formar a mediano o largo plazo, líderes y ejecutivos con conocimientos y talentos específicos, por ejemplo: un posgrado en finanzas.

Para tomar las decisiones correctas en cuanto a qué programas de capacitación requieren nuestros colaboradores y, con la finalidad de no convertir a la capacitación en un gasto sino en una inversión, se debe realizar previamente las siguientes actividades:

- Elabore una descripción de todos los puestos de su compañía.
- Realice una “detección de necesidades de capacitación”. Observe cómo se desempeñan sus empleados, como tratan a los clientes, o simplemente conteste lo siguiente: ¿Qué debería tener esta persona para poder ser gerente del área? A través de la observación, realizando cuestionarios a los empleados sobre sus intereses y evaluando su desempeño, podemos formarnos una idea sobre las necesidades de capacitación.
- Determine cuál o cuáles cursos-entrenamientos son necesarios para mejorar el desempeño de su empresa en general, y después seleccione qué empleados son los adecuados para adquirir esa capacitación.
- Establezca los objetivos que quiere alcanzar con la capacitación y determine de qué forma recuperará el dinero que invierta (retorno sobre inversión).

#### **7.4. Determinar la efectividad de la capacitación**

Una vez que los conceptos aprendidos fueron puestos en práctica y la medición de los avances reflejan resultados positivos, se puede determinar la efectividad de la capacitación impartida. Cuando un curso no tuvo el impacto esperado, puede atribuirse a que este no fue bien canalizado o no se detectaron adecuadamente las necesidades de capacitación.

Si la capacitación fue efectiva, se podrá observar:

- Cambio de conducta en el personal
- Impacto positivo en la productividad de la organización
- Mejoría en el desempeño después de la capacitación





## **CONCLUSIONES**

1. El sistema implementado en la unidad de información garantiza el ingreso de solicitudes en formato digital, aplicando validadores para garantizar la congruencia de la información, así como la administración de expedientes y su evolución durante las diferentes fases del proceso.
2. La aplicación cuenta con módulos que facilitan las búsquedas por medio de diferentes parámetros, así como la integración de los resultados encontrados a los respectivos expedientes, permitiendo así agilizar las búsquedas y entregas de archivos históricos.
3. El sistema, al manejar toda la información digitalizada en su base de datos, puede proporcionar la consulta del proceso de los expedientes solicitados, así como la búsqueda de información sobre los archivos históricos, garantizando así que toda persona pueda solicitar información en cualquier momento.
4. El proyecto tiene un costo normalizado de Q361 980.00, superando las expectativas del ahorro inicialmente estimado.



## **RECOMENDACIONES**

1. El ingreso de solicitudes actualmente se hace de manera presencial. Esto implica que muchas personas que no puedan llegar presencialmente, debido a circunstancias de horarios o distancias, no puedan solicitar información, por lo que implementar la solución para ser accedida vía Internet, podría llegar a tener un mayor alcance, además de agilizar los procesos de ingreso de solicitudes, ya que estos se harían automáticamente y esto acortaría los tiempos de entrega.
2. Actualmente, el proceso de búsqueda consiste en recorrer todos los archivos, por cada expediente que se solicite, aunque hayan expedientes que soliciten información similar, esto hace que el proceso de búsqueda y selección sea lento, por lo que se podrían mejorar estas búsquedas mediante la integración del modulo de etiquetas.
3. Debido a que la información tiende a crecer considerablemente, se recomienda el mantenimiento a través de políticas de respaldo periódico de información.
4. Con el presupuesto actual del proyecto, que brinda un ahorro a Guatemala, puede invertirse dicho presupuesto, para generar investigación o desarrollo para nuevos proyectos de beneficio social, guiados por la Universidad de San Carlos de Guatemala.



## REFERENCIAS

- [1] Artefacto de la disciplina *Analysis & Design* de RUP.  
[http://www.ts.mah.se/RUP/RationalUnifiedProcess/process/artifact/ar\\_sadoc.htm](http://www.ts.mah.se/RUP/RationalUnifiedProcess/process/artifact/ar_sadoc.htm)
- [2] Plantilla *Software Architecture Document*, como guía de redacción y de contenidos a considerar.  
[http://www.ts.mah.se/RUP/RationalUnifiedProcess/webtmpl/templates/a\\_and\\_d/rup\\_sad.htm](http://www.ts.mah.se/RUP/RationalUnifiedProcess/webtmpl/templates/a_and_d/rup_sad.htm)
- [3] Patentes y tipos de licencia de *software*  
[http://doc.ubuntu-es.org/Patentes\\_y\\_tipos\\_de\\_Licencias\\_de\\_Software](http://doc.ubuntu-es.org/Patentes_y_tipos_de_Licencias_de_Software)
- [4] Tipos de licencia  
<http://www.bionicismutton.org/ade/licenses/>
- [5] Cocomo  
<http://es.wikipedia.org/wiki/COCOMO>
- [6] Medir Cocomo básico  
<http://www.oei.eui.upm.es/Asignaturas/PInformaticos/ficheros/software/opcion3/CBasico.html>
- [7] Medir Cocomo intermedio  
<http://www.oei.eui.upm.es/Asignaturas/PInformaticos/ficheros/software/opcion3/CIntermedio.html>
- [8] *Software* para la gestión de costos  
<http://www.construx.com/Page.aspx?nid=68>
- [9] Estimación de tiempos, costos y recursos  
<http://www.mitecnologico.com/Main/EstimacionDeTiemposCostosYRecursos>
- [10] La capacitación empresarial puede acelerar la productividad de tus recursos humanos.  
<http://blogorama.com.mx/Relaciones/la-capacitacion-empresarial-puede-acelerar-la-productividad-de-tus-recursos-humanos/>
- [11] Ejemplo citado en documentación oficial, como una ilustración de la forma correcta de construir un documento de arquitectura  
[http://www.ts.mah.se/RUP/RationalUnifiedProcess/examples/csports/ex\\_sad.htm](http://www.ts.mah.se/RUP/RationalUnifiedProcess/examples/csports/ex_sad.htm)