

UNIVERSIDAD NACIONAL DE SAN CRISTÓBAL DE HUAMANGA

FACULTAD DE INGENIERÍA DE MINAS, GEOLOGÍA Y CIVIL

ESCUELA DE FORMACIÓN PROFESIONAL DE INGENIERÍA DE
SISTEMAS



**“SISTEMA DE INFORMACIÓN WEB PARA OPTIMIZAR LA GESTIÓN
DE ARCHIVOS EN LA PROCURADURÍA DE AYACUCHO, 2018”**

Tesis presentada por : Bach. Sosa Carpio, Roger.
Para optar el título profesional de : Ingeniero Informático.
Tipo de investigación : Aplicada.
Área de investigación : Ingeniería de Software.
Asesor : Ing. Hubner Janampa Patilla.

AYACUCHO – PERÚ

2019

DEDICATORIA

A Dios por darme la vida; a mi madre Nina por la confianza, por su apoyo a lo largo de este camino universitario, a sus palabras de aliento y sus oraciones en los momentos difíciles digna de una verdadera creyente de la palabra de Dios; también debo de agradecer a mi hermano Edwin por su paciencia infinita en estos últimos tres años; a mi padre por ser un gran maestro de calma y tolerancia; y al gran amigo y maestro Don Roberto.

AGRADECIMIENTO

A la prestigiosa Universidad Nacional de San Cristóbal de Huamanga, a mis profesores de la escuela de Formación Profesional de Ingeniería de Sistemas a quienes les debo gran parte de mis conocimientos, gracias al Ing. Walter Quispe Jayo por su apoyo y experiencia en la programación web y a mi asesor el Mg. Ing. Hubner Janampa P. por su apoyo en la guía para la construcción de esta tesis.

CONTENIDO

DEDICATORIA	i
AGRADECIMIENTO.....	ii
CONTENIDO	iii
RESUMEN	viii
INTRODUCCIÓN	viii

CAPÍTULO I

PLANTEAMIENTO DE LA INVESTIGACIÓN

1.1. DIAGNÓSTICO Y ENUNCIADO DEL PROBLEMA	1
1.2. DEFINICIÓN DEL PROBLEMA DE INVESTIGACIÓN	2
1.2.1 PROBLEMA GENERAL.....	2
1.2.2 PROBLEMAS SECUNDARIOS	2
1.3. OBJETIVOS DE LA INVESTIGACIÓN	3
1.3.1 OBJETIVO GENERAL.....	3
1.3.2 OBJETIVOS ESPECÍFICOS	3
1.4. HIPÓTESIS DE LA INVESTIGACIÓN.....	3
1.5. JUSTIFICACIÓN Y DELIMITACIÓN DE LA INVESTIGACIÓN.....	4
1.5.1 IMPORTANCIA DEL TEMA.....	4
1.5.2 JUSTIFICACIÓN	4
1.5.3 DELIMITACIÓN	5

CAPITULO II

REVISIÓN DE LITERATURA

2.1 ANTECEDENTES DE LA INVESTIGACIÓN	6
2.1.1 ANTECEDENTE 1	6
2.1.2 ANTECEDENTE 2	6
2.1.3 ANTECEDENTE 3	6
2.2 MARCO TEÓRICO	7
2.2.1 SISTEMA DE INFORMACIÓN.....	7
2.2.1.1 SISTEMA DE INFORMACIÓN AUTOMATIZADO.....	8
2.2.1.2 ESTRUCTURA DE UN SISTEMA DE INFORMACIÓN.....	8
2.2.1.3 CLASIFICACIÓN DE LOS SISTEMAS DE INFORMACIÓN.....	9
2.2.1.4 LOS SISTEMAS DE INFORMACIÓN EN RAZÓN DE CALIDAD SE DIVIDEN EN LOS SIGUIENTES PUNTOS.....	9
2.2.2 COMPONENTES DE UN SISTEMA DE INFORMACIÓN	10
2.2.2.1 EQUIPOS INFORMÁTICOS.....	10
2.2.2.2 PROGRAMAS INFORMÁTICOS.....	10
2.2.2.3 BASES DE DATOS	11
2.2.2.4 TELECOMUNICACIONES	11

2.2.2.5	RECURSOS HUMANOS	12
2.2.2.6	PROCEDIMIENTOS	12
2.2.3	FUNCIONES DE UN SISTEMA DE INFORMACIÓN.....	12
2.2.3.1	RECOLECCION.	12
2.2.3.2	CLASIFICACION.....	12
2.2.3.3	COMPRESION.....	13
2.2.3.4	ALAMCENAMIENTO.	13
2.2.3.5	RECUPERCION.	14
2.2.3.6	PROCESAMIENTO.....	14
2.2.3.7	TRANSMISION	14
2.2.3.8	EXHIBICION.....	15
2.2.4	DEFINICIÓN DE ARCHIVO	16
2.2.5	FUNCIONES DE LOS ARCHIVOS	17
2.2.5.1	GÉNESIS O REUNIÓN DE LOS DOCUMENTOS:.....	18
2.2.5.2	TRATAMIENTO DE LA DOCUMENTACIÓN:.....	18
2.2.5.3	SERVIR LA DOCUMENTACIÓN:	19
2.2.6	LAS TRES EDADES DE LOS DOCUMENTOS Y LOS ARCHIVOS INTERMEDIOS.....	20
2.2.7	GESTION DE DOCUMENTOS	21
2.2.8	METODOLOGÍA ICONIX.....	21
2.2.8.1	CARACTERÍSTICAS DE ICONIX.....	23
2.2.8.1.1	ITERATIVO E INCREMENTAL	23
2.2.8.1.2	TRAZABILIDAD.....	23
2.2.8.1.3	DINÁMICA DE UML.....	23
2.2.8.2	TÉNICAS DE LA METODOLOGÍA ICONIX.....	24
2.2.8.2.1	MODELO DE DOMINIO	24
2.2.8.2.2	MODELO DE CASOS DE USO	25
2.2.8.2.3	REVISION DE REQUISITOS	26
2.2.8.2.4	ANÁLISIS DE ROBUSTEZ	26
2.2.8.2.5	REVISIÓN PRELIMINAR DEL DISEÑO	27
2.2.8.2.6	ARQUITECTURA TÉCNICA	28
2.2.8.2.7	DIAGRAMAS DE SECUENCIA	29
2.2.8.2.8	REVISIÓN CRÍTICA DEL DISEÑO	29
2.2.8.2.9	IMPLEMENTACIÓN: PASO DEL DISEÑO DETALLADO AL CÓDIGO	30
2.2.8.2.10	REVISIÓN DEL CÓDIGO Y ACTUALIZACIÓN DEL MODELO	31
2.2.8.2.11	DISEÑO GUIADO POR PRUEBAS	32
2.2.8.2.12	ATENTIENDO REQUISITOS.....	32
2.2.9	PROGRAMACIÓN ORIENTADA A OBJETOS.....	33
2.2.9.1	CLASES	34
2.2.9.2	OBJETO	34
2.2.10	CARACTERÍSTICAS ESENCIALES DE LA POO.....	35

2.2.11	LENGUAJE DE PROGRAMACIÓN	37
2.2.11.1	PHP	37
2.2.12	LENGUAJE SQL	38
2.2.12.1	POSTGRESQL	39
2.2.13	FRAMEWORK	40
2.2.13.1	CODEIGNITER	40
2.2.14	ARQUITECTURA MODELO VISTA CONTROLADOR	41
2.1.14.1	MODELO	41
2.1.14.2	VISTA	42
2.1.14.3	CONTROLADOR	42
2.2.15	FRONT-END.....	42
2.2.15.1	HTML.....	43
2.2.15.2	CSS.....	43
2.2.15.3	JAVASCRIPT	44
2.2.16	BACK-END.....	44
2.2.17	PROCURADURÍA PÚBLICA REGIONAL AYACUCHO	45
2.2.17.1	MISIÓN	45
2.2.17.2	VISIÓN.....	45
2.2.17.3	ESTRUCTURA ORGANICA FUNCIONAL GOBIERNO REGIONAL DE AYACUCHO	46
2.2.17.4	FUNCIONES GENERALES.....	47
2.2.17.5	PROCURADOR PÚBLICO REGIONAL.....	47
2.2.17.6	ABOGADO IV	49
2.2.17.7	ABOGADO III	50

CAPITULO III

METODOLOGÍA DE LA INVESTIGACIÓN

3.1	TIPO DE INVESTIGACIÓN	52
3.2	NIVEL DE INVESTIGACIÓN	52
3.3	DISEÑO DE INVESTIGACIÓN	53
3.4	POBLACIÓN Y MUESTRA.....	54
3.4.1	POBLACIÓN	54
3.4.2	MUESTRA	54
3.5	VARIABLES E INDICADORES	54
3.5.1	DEFINICIÓN CONCEPTUAL DE LAS VARIABLES	54
3.5.1.1	PRIMERA VARIABLE	54
3.5.1.2	INDICADORES DE LA PRIMERA VARIABLE	54
3.5.1.3	SEGUNDA VARIABLE	55
3.5.1.4	INDICADORES DE LA SEGUNDA VARIABLE.....	55
3.5.2	DEFINICIÓN OPERACIONAL DE LAS VAIABLES	56
3.6	TÉCNICAS E INSTRUMENTOS.....	56
3.6.1	TÉCNICAS PARA RECOLECTAR INFORMACIÓN	56

3.6.2	INSTRUMENTOS PARA RECOLECTAR INFORMACIÓN.....	56
3.6.3	HERRAMIENTAS PARA EL TRATAMIENTO DE DATOS E INFORMACIÓN	56
3.6.4	TÉCNICAS PARA APLICAR ICONIX	57
3.6.4.1	ANÁLISIS DE REQUISITOS.....	58
3.6.4.2	REVISIÓN DE REQUISITOS	59
3.6.4.3	DISEÑO PRELIMINAR	59
3.6.4.4	REVISIÓN DE DISEÑO PRELIMINAR	60
3.6.4.5	DISEÑO	60
3.6.4.6	REVISIÓN CRÍTICA DE DISEÑO.....	61

CAPITULO IV

RESULTADOS DE LA INVESTIGACIÓN

4.1	RESULTADOS Y ARTEFACTOS DEL SOFTWARE APLICANDO EL PROCESO DE ICONIX	62
4.1.1	ETAPA1: ANÁLISIS DE REQUISITOS.....	62
4.1.1.1	REQUISITOS FUNCIONALES	62
4.1.1.2	REQUISITOS NO FUNCIONALES.....	63
4.1.1.3	MODELO DE DOMINIO INICIAL	65
4.1.1.4	PROTOTIPO DE INTERFAZ DE USUARIO DE LOS CASOS DE USO	66
4.1.1.4.1	INTERFAZ DE USUARIO: Registrar Usuario.....	66
4.1.1.4.2	INTERFAZ DE USUARIO: Lista De Usuarios Del Sistema.....	66
4.1.1.4.3	INTERFAZ DE USUARIO: Agregar Expediente /Caso	67
4.1.1.4.4	INTERFAZ DE USUARIO: Lista Expediente/ Caso	67
4.1.1.4.5	INTERFAZ DE USUARIO: Lista de Procesos de un expediente	68
4.1.1.5	IDENTIFICACIÓN DE CASOS DE USO.....	68
4.1.1.6	DESCRIPCIÓN DE CASOS DE USO.....	69
4.1.2	HITO ETAPA 1: REVISIÓN DE REQUISITOS.....	72
4.1.2.1	REQUISITOS FUNCIONALES REVISADOS	72
4.1.2.2	REQUISITOS NO FUNCIONALES REVISADOS	74
4.1.2.3	REVISIÓN DE MODELO DE DOMINIO	77
4.1.2.4	REVISIÓN DE CASOS DE USO	78
4.1.2.5	CASO DE USO POR PAQUETES	88
4.1.2.5.1	PAQUETE “Actores”	89
4.1.2.5.2	PAQUETE “Administrar Usuario”.....	89
4.1.2.5.3	PAQUETE “Administrar Sistema”.....	90
4.1.2.5.4	PAQUETE “Abrir Trámite”	91
4.1.3	ETAPA 2: ANÁLISIS Y DISEÑO PRELIMINAR.	92
4.1.3.1	DIAGRAMA DE ROBUSTEZ	92
4.1.3.1.1	DIAGRAMA DE ROBUSTEZ “Registrar Usuario”.....	92
4.1.3.1.2	DIAGRAMA DE ROBUSTEZ “Listar Usuarios Del Sistema”	93
4.1.3.1.3	DIAGRAMA DE ROBUSTEZ “Agregar Expediente /Caso”	94
4.1.3.1.4	DIAGRAMA DE ROBUSTEZ “Listar Expediente/ Caso”.....	95

4.1.3.1.5	DIAGRAMA DE ROBUSTEZ “Listar Procesos de un expediente”	96
4.1.3.2	MODELO DE DOMINIO ACTUALIZADO	97
4.1.4	HITO ETAPA 2: REVISIÓN DEL DISEÑO PRELIMINAR (PDR).	98
4.1.4.1	REVISION DE LA DESCRIPCION DE LOS CASOS DE USO	98
4.1.4.2	REVISIÓN DEL MODELO DE DOMINIO ACTUALIZADO.....	101
4.1.5	DISEÑO DETALLADO.	102
4.1.5.1	ARQUITECTURA TÉCNICA DEL FRAMEWORK CODEIGNITER.....	102
4.1.5.2	ARQUITECTURA DE LA APLICACIÓN PHP	105
4.1.5.3	ARQUITECTURA TÉCNICA POR CAPAS	108
4.1.5.4	DIAGRAMA DE COMPONENTES.....	109
4.1.5.5	DIAGRAMA DE DESPLIEGUE.....	110
4.1.6	DIAGRAMA DE SECUENCIA.....	111
4.1.6.1	DIAGRAMA DE SECUENCIA “Registrar Usuario”	111
4.1.6.2	DIAGRAMA DE SECUENCIA “Listar Usuarios Del Sistema”.....	112
4.1.6.3	DIAGRAMA DE SECUENCIA “Agregar Expediente/ Caso”	113
4.1.6.4	DIAGRAMA DE SECUENCIA “Listar Expediente/ Caso”	114
4.1.6.5	DIAGRAMA DE SECUENCIA “Listar Procesos de un expediente”	115
4.1.7	HITO ETAPA 3: REVISIÓN DEL DISEÑO DETALLADO.	116
4.1.7.1	DIAGRAMA DE CLASES DE DISEÑO	116
4.1.7.2	DIAGRAMA DEL MODELO LÓGICO DE LA BASE DE DATOS GENERAL	117
4.1.8	IMPLEMENTACIÓN	118
4.1.8.1	CREACIÓN DE LA BASE DE DATOS.....	118
4.1.8.2	ESTRUCTURA DEL PROYECTO	120
4.1.8.3	IMPLEMENTACIÓN DE LAS VISTAS.....	121
4.1.8.4	IMPLEMENTACIÓN DE LOS CASOS DE USOS	129
4.2	REVISIÓN DE CÓDIGO.....	130
4.2.1	PRUEBAS UNITARIAS.....	130
4.2.2	PRUEBAS FUNCIONALES.....	131
4.3	RESULTADO DEL PROCESO DE DESARROLLO DE SISTEMA.	132
4.4	DISCUSIÓN DE RESULTADOS.....	133

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1	CONCLUSIONES	135
5.2	RECOMENDACIONES.....	136
	BIBLIOGRAFÍA.....	137
	ANEXO A	142
	ANEXO B	143

RESUMEN

La gestión de los archivos es tan antigua como las organizaciones sociales, sus orígenes podrían remontarse a la aparición de la escritura, los primeros archivos y la necesidad de gestionarlos surge con los primeros imperios, como, por ejemplo: el uso del archivo en Grecia y Roma, el archivo en la Edad Media, el archivo en el Siglo XIX y en el siglo XX los Archivos Intermedios.

Los archivos en toda institución son un conjunto de documentos que se los puede gestionar, alojar y darle un tratamiento específico; la Procuraduría de Ayacucho no está exento a gestionar sus expedientes, con el uso de la tecnología basado en un sistema de información web se da solución a los problemas descritos; por lo tanto, la gestión de expedientes en la Procuraduría se automatizó.

La presente investigación tiene como objetivo desarrollar un Sistema de información web para optimizar la gestión de archivos en la Procuraduría de Ayacucho, 2018, Mediante investigación y uso de aplicativos y lenguajes de programación web, con el propósito de dar calidad de servicio, con la finalidad de optimizar la gestión de archivos.

Para desarrollar el sistema de información web, se hizo uso de UML, POO, lenguaje de programación PHP, framework CodeIgniter, la base de datos Postgresql y aplicativos (pgModeler de Postgresql, Enterprise Architect), todo lo mencionado fue necesario para aplicar: usabilidad, funcionalidad y fiabilidad; se realizó mediante la investigación, análisis y síntesis.

El resultado que se obtuvo es un sistema de información web que tiene las siguientes cualidades: usabilidad, funcionalidad y fiabilidad.

Palabras Claves: *Sistema de Información, usabilidad, funcionalidad y fiabilidad, Gestión de Archivos.*

INTRODUCCIÓN

El uso de las Tics en un inicio aportaron una ventaja competitiva para quienes lo empezaron a utilizar, en la actualidad todas las entidades privadas o públicas lo utilizan, por lo que dejó de ser un diferenciador, a lo largo de tiempo ese comodín no se puede mantener, por lo tanto el uso de las TI se vuelven en un habilitador, es decir algo común entre empresas privada y públicas, el problema principal de esta investigación no es ¿adoptar las TI?, más bien de como las usamos y que problemas solucionamos para diferenciarnos como entidad pública.

La gestión de expedientes en la Procuraduría de Ayacucho según lo reportado, se realiza de manera manual y en Excel, con tiempos de espera prolongados para su atención, búsquedas en archivos físicos, la falta de automatización de este proceso es un problema que se tiene que solucionar.

Mediante la construcción de un Sistema de Información Web, se espera integrar los criterios de calidad como la usabilidad, funcionalidad y fiabilidad en el resultado de la automatización de la gestión de archivos en la procuraduría de Ayacucho; para tal fin se estará aplicando metodologías de sistemas información y herramientas tecnológicas.

Con la ayuda de este sistema de información web se podrá garantizar la consulta, administración y análisis de grandes cantidades de expedientes concernientes a la representación y defensa jurídica de los derechos e intereses del estado de la región Ayacucho.

CAPÍTULO I

PLANTEAMIENTO DE LA INVESTIGACIÓN

1.1. DIAGNÓSTICO Y ENUNCIADO DEL PROBLEMA

De acuerdo a lo mencionado por los funcionarios de la Procuraduría Regional de Ayacucho, la gestión de expedientes se realiza de forma manual (registro, seguimiento y búsqueda en Excel, hoja física y archivadores), teniendo en cuenta algunas deficiencias en cuanto a lo administrativo, falta de personal (abogados) y una sobrecarga laboral; por otro lado en la actualidad se cuenta con un sin número tecnologías, la utilización de estas tecnologías mediante un Sistema de Información Web minimizarían el impacto negativo a la parte administrativa e de imagen de la Procuraduría de Ayacucho, lo sucinto anteriormente se puede apoyar en los siguientes enunciado.

Periodo	Tramitados	No Atendidos	Total
2003-2019	16414 casos	7586 casos	24000 casos
*Cuatro abogados son los encargados de la atención de los casos.			
*4700 casos nuevos cada año.			

Tabla 1.1: Elaboración propia de acuerdo a lo publicado en el diario "Correo".

Del 2003 hasta la fecha, la Procuraduría Regional de Ayacucho registró en demandas ingresadas un total de 24 mil 113 de procesos judiciales, civiles, contenciosos administrativos, constitucionales y laborales, de los cuales se tiene en trámite un total de 16 mil 414 casos, así lo informó el Procurador Regional, Carlos Paredes Orellana ante el pleno del Consejo Regional de Ayacucho (Correo, 2019).

Gran porcentaje de estos casos corresponden a procesos por deudas sociales de la entidad regional con los trabajadores que ascienden a dos mil 799 expedientes con una deuda de 132 millones 347 mil 300 soles que tienen que ser pagados por el Ministerio de economía y Finanzas (MEF) y sólo en el 2018 se pagó 26 millones (Correo, 2019).

La Procuraduría Pública Regional de Ayacucho tiene a su cargo la defensa de los intereses del Estado en más de tres mil 500 casos y 90 procesos de arbitraje que actualmente se encuentran en curso, así lo informó el procurador encargado, Roberto Oriundo Yaranga (Correo, 2018).

En Ayacucho, las instituciones dedicadas a la defensa de los intereses del Estado presentan una serie de limitaciones para el cumplimiento de sus funciones, principalmente escaso personal, así como falta de logística para la defensa en los procesos legales. Uno de estos casos es la Procuraduría Regional que apenas tiene cuatro abogados para llevar más de siete mil 150 procesos a nivel de toda la región (Correo, 2018).

La sobrecarga laboral es uno de los problemas que aqueja actualmente a la Procuraduría Regional de Ayacucho, el cual empeora con el escaso personal que atiende los más de siete mil procesos legales en curso a nivel regional (Correo, 2018).

1.2. DEFINICIÓN DEL PROBLEMA DE INVESTIGACIÓN

1.2.1 PROBLEMA GENERAL

- ✓ ¿Cómo un **Sistema de información web** optimiza la **gestión de archivos** en la Procuraduría de Ayacucho, 2018?

1.2.2 PROBLEMAS SECUNDARIOS

- ✓ ¿De qué manera la **usabilidad** del sistema de información web ayuda al **registro de documentos**?
- ✓ ¿De qué manera la **funcionalidad** del sistema de información web optimiza la **clasificación de documentos**?
- ✓ ¿De qué manera la **fiabilidad** del sistema de información web mejora la **descripción y recuperación de documentos**?

1.3. OBJETIVOS DE LA INVESTIGACIÓN

1.3.1 OBJETIVO GENERAL

Desarrollar un **Sistema de información web** para optimizar la **gestión de archivos** en la Procuraduría de Ayacucho, 2018, Mediante investigación y uso de aplicativos y lenguajes de programación web, con el propósito de dar calidad de servicio, con la finalidad de optimizar la gestión de archivos.

1.3.2 OBJETIVOS ESPECÍFICOS

- ✓ Aplicar la **Usabilidad** en el sistema de información web ayuda al **registro de documentos**, con el propósito facilitar el uso para del usuario.
- ✓ Utilizar la **Funcionalidad** en el sistema de información web ayuda a la **clasificación de documentos**, con el fin de ser justo y necesario para el usuario.
- ✓ Aplicar la **Fiabilidad** en el sistema de información web mejora la **descripción y recuperación de documentos**, con el fin de brindar integridad y rendimiento para el usuario.

1.4. HIPÓTESIS DE LA INVESTIGACIÓN

“No en todas las investigaciones cuantitativas se planean hipótesis. El hecho de formulemos o no hipótesis depende de un factor esencial: el alcance inicial del estudio. Las investigaciones cuantitativas que formulan hipótesis son aquellas cuyo planteamientos define que su alcance será correlacional o explicativo, o en las que tienen un alcance descriptivo, pero que intentan pronosticar una cifra o un hecho” (Hernández, baptista y Collado, 2014, p.104).

“Las investigaciones de tipo descriptivo no requieren formular hipótesis; es suficiente plantear algunas preguntas de investigación que, como ya se anotó, surgen del planteamiento del problema, de los objetivos y, por supuesto, del

marco teórico que soporta el estudio” (Bernal, 2010, P.136).

La investigación que se desarrolló es de tipo **descriptivo**, por lo que no se pretende pronosticar hallar o verificar lo planteado en los objetivos, se optó por **no plantear hipótesis**.

1.5. JUSTIFICACIÓN Y DELIMITACIÓN DE LA INVESTIGACIÓN

1.5.1 IMPORTANCIA DEL TEMA

Adoptar la TICs ya no son una moda en estos tiempos, es una necesidad primordial, claro está que no te dará la ventaja sobre tus competidores, pero si te alineara y te pondrá al mismo nivel que otras instituciones del estado que ya cuentan con las TI; en la Procuraduría Regional de Ayacucho no se estuvo utilizando las TI para la gestión de expedientes como por ejemplo: registrar, buscar, y dar seguimientos; todas estas tareas se hacía de forma anticuada, manualmente y en hojas de Excel, sumando a este problema se pudo observar: *primero*: la gran cantidad de expedientes o casos no atendidos que suman 7586 casos hasta el 2018 y desde el 2003 hasta el 2018 se atendieron 16414 casos con un promedio de 4700 casos nuevos cada año; *segundo*: los 4 abogados encargados de dar gestión y atención de los expedientes no se dan abasto.

Por lo tanto, la importancia de diseñar un Sistema de Información Web es para optimizar la gestión de archivos en la procuraduría de Ayacucho, los usuarios finales tendrán automatizado el seguimiento de expedientes que minimizara tiempo de atención.

1.5.2 JUSTIFICACIÓN

La presente investigación se desarrolló debido a que hubo una carencia en la gestión de expedientes y archivos en la Procuraduría de Ayacucho, según lo consultado con los encargados de realizar estas tareas, las mismas se hacían de forma ortodoxa y anticuada carente de alguna tecnología. La presente

investigación otorga una herramienta para optimizar puntualmente la gestión de archivos y expedientes, los beneficiarios serán los usuarios que todavía hacen esta tarea de forma manual en la Procuraduría de Ayacucho.

1.5.3 DELIMITACIÓN

La investigación se realizará en la Procuraduría de Ayacucho para el área de gestión expedientes, con información del año 2018.

CAPITULO II

REVISIÓN DE LITERATURA

2.1 ANTECEDENTES DE LA INVESTIGACIÓN

2.1.1 ANTECEDENTE 1

Las aplicaciones propuestas ofrecen una solución potente y de calidad de gestión de contenidos; repositorios, catálogos y plataformas web de trabajo que mejoran el desempeño de actividades rutinarias como son la comunicación entre personal y usuarios, el acceso a Internet o el tratamiento de imágenes entre otros. Se podría decir que el resultado se verá reflejado en los beneficios del software abierto y en el bajo coste de adquisición (Saiz, F. J., Olmeda, R., Olmeda, C. 2013).

2.1.2 ANTECEDENTE 2

El sistema de asignación y control de expedientes (SACE), contribuyó a mejorar la asignación de expedientes en la subdirección de Inspección y Fiscalización, además, ayudó de cierta manera en disminuir el stock de expedientes. El SACE contribuyó en la medición de los tiempos de revisión de los expedientes. También, aportó en la detección de cuello de botella en el proceso de asignación y control de expedientes, luego la Subdirección de Inspección y Fiscalización tomará las decisiones siguiendo los lineamientos de DBR Avalos, (H., Lucio, F., Quispe, P., Henry, G., 2014).

2.1.3 ANTECEDENTE 3

El desarrollo de la aplicación web influye satisfactoriamente en los procesos para la mejora en la gestión de trámite documentario, por que ayudan a los usuarios y las áreas de administrativas, a realizar un proceso de trámite simple, rápido y fácil, cumpliendo y satisfaciendo los indicadores de calidad planteados en los objetivos de este trabajo de tesis (Moscoso, J. 2018).

2.2 MARCO TEÓRICO

2.2.1 SISTEMA DE INFORMACIÓN

De acuerdo a Rafael, Carlos, y Joaquín (2011), todo sistema se puede dividir en subsistemas. Dado que la empresa se comporta como un sistema, es posible fragmentar sus partes en subsistemas. Según la literatura de teoría de la organización, se puede dividir la empresa en los siguientes sistemas: comercial, de operaciones, financiero, de personal, y de información. El sistema de información se relaciona con el resto de sistemas y con el entorno.

Según Meguzzato y Renau (1991), un sistema de información en la empresa debe servir para captar la información que esta necesite y ponerla, con las transformaciones necesarias, en poder de aquellos miembros de la empresa que la requieran, bien sea para la toma de decisiones, bien sea para el control estratégico, o para la puesta en práctica de las decisiones adoptadas.

De ahí que el desempeño de un directivo dependa de su habilidad para explotar las capacidades de los sistemas de información para obtener unos positivos resultados empresariales.

La definición de sistema de información que dan. Andreu, Ricart y Valor (1991). Según estos autores, el sistema de información: Es el conjunto formal de procesos que operando sobre una colección de datos estructurada de acuerdo con las necesidades de una empresa, recopila, elabora y distribuye la información necesaria para la operación de dicha empresa y para las actividades de dirección y control correspondientes, apoyando, al menos en parte, los procesos de toma de decisiones necesarios para desempeñar las funciones de negocio de la empresa de acuerdo con su estrategia.

2.2.1.1 SISTEMA DE INFORMACIÓN AUTOMATIZADO.

Según Pressman (1993), es un “Conjunto de elementos organizados para llevar a cabo algún método, procedimiento o control mediante procesamiento de la información”

Los sistemas de información en razón de calidad se dividen en los siguientes puntos.



Figura nº 2.1: Fuente: Departamento de Informática y Automática Universidad de Salamanca.

2.2.1.2 ESTRUCTURA DE UN SISTEMA DE INFORMACIÓN

Operaciones y transacciones: Nivel de procesamiento de las actividades diarias o rutinarias de una organización (Moreno, Maria., 2019).

Nivel operativo: Análisis de los resultados para tomar decisiones a corto plazo y de consecuencias limitadas (Moreno, Maria., 2019).

Nivel táctico: Análisis de resultados para la toma de decisiones a medio plazo (Moreno, Maria., 2019).

Nivel estratégico: Utilización de la información para decidir las líneas maestras a largo plazo (Moreno, Maria., 2019).



Figura nº 2.2: Fuente: Departamento de Informática y Automática Universidad de Salamanca.

2.2.1.3 CLASIFICACIÓN DE LOS SISTEMAS DE INFORMACIÓN.

- ✓ Sistemas de procesamiento de transacciones
- ✓ Sistemas de automatización de oficina
- ✓ Sistemas de manejo del conocimiento
- ✓ Sistemas de información administrativa o gerencial
- ✓ Sistemas de apoyo a las decisiones
- ✓ Sistemas de apoyo a las decisiones de grupo
- ✓ Sistemas de apoyo a ejecutivos o de planificación estratégica

2.2.1.4 LOS SISTEMAS DE INFORMACIÓN EN RAZÓN DE CALIDAD SE DIVIDEN EN LOS SIGUIENTES PUNTOS.

- ✓ **Funcionalidad:** Las funciones satisfacen necesidades declaradas o implícitas [ISO/IEC 25000: 2014].
- ✓ **Fiabilidad:** Capacidad de un sistema para mantener su nivel de rendimiento [ISO/IEC 25000: 2014].
- ✓ **Usabilidad:** Esfuerzo necesario para el uso y la valoración individual de tal uso, por parte de un conjunto de usuarios. [ISO/IEC 25000: 2014]
- ✓ **Portabilidad:** Es la capacidad de un sistema para ser transferido de un entorno a otro. [ISO/IEC 25000: 2014]
- ✓ **Mantenibilidad:** Es el esfuerzo necesario para realizar modificaciones específicas. [ISO/IEC 25000: 2014]

- ✓ **Eficiencia:** Es la relación entre el nivel de prestaciones de un sistema y el volumen de recursos utilizados en condiciones declaradas. [ISO/IEC 25000: 2014]

2.2.2 COMPONENTES DE UN SISTEMA DE INFORMACIÓN

Según García Bravo (2000), “Los sistemas de información engloban: equipos y programas informáticos, telecomunicaciones, bases de datos, recursos humanos y procedimientos”.

2.2.2.1 EQUIPOS INFORMÁTICOS

Según García Bravo (2000), Actualmente todas las empresas utilizan ordenadores. Por lo general, se utilizan microordenadores, también conocidos como ordenadores personales o pc. Las organizaciones grandes utilizan diversos sistemas computarizados, incluyendo desde grandes ordenadores, que suelen ser denominados mainframes, hasta miniordenadores y los más utilizados, microordenadores. Debemos aclarar que el progreso de las prestaciones técnicas experimentado en los últimos años por los microordenadores hace que puedan realizar más tareas que inicialmente estaban asignadas a los miniordenadores y que cada vez esté menos clara la diferencia entre estas dos categorías de ordenadores. Las tres categorías de ordenadores están organizadas de forma similar. El componente que controla todas las unidades del sistema es el procesador central, que ejecuta las instrucciones de un programa. También hay dispositivos para introducir datos (teclado y ratón) y dispositivos para producir el output del sistema (impresoras).

2.2.2.2 PROGRAMAS INFORMÁTICOS

Según García Bravo (2000), Hay dos tipos de programas informáticos: programas del sistema y aplicaciones. Los programas del sistema administran los recursos del sistema computarizado y simplifican la programación. Las aplicaciones ayudan directamente al usuario final a hacer su trabajo. Ejemplos de aplicaciones: programas de hoja de cálculo o procesadores de texto.

2.2.2.3 BASES DE DATOS

Según García Bravo (2000), podríamos considerar que muchos sistemas de información en las empresas son utilizados como vehículo de entrega de bases de datos. Una base de datos es una colección de datos interrelacionados. Como ejemplo, podríamos mencionar la base de datos de recursos humanos de una organización o la base de datos de productos. Para una empresa, resulta de gran valor la base de datos de clientes, que puede ser explotada para comunicar a estos los nuevos productos o para desarrollar nuevos productos que satisfagan las necesidades percibidas de los mismos. Una base de datos debe estar organizada para que se pueda acceder a ellos por sus atributos. Ej.: «Dame los nombres y direcciones de los clientes a quienes hemos facturado más de 1 millón en el último año». Las bases de datos son administradas por programas de sistemas conocidos como sistemas de administración de bases de datos (dbms) (p.).

2.2.2.4 TELECOMUNICACIONES

Según García Bravo (2000), las telecomunicaciones son el medio de transmisión electrónica de información a largas distancias. En la actualidad, los sistemas computarizados están generalmente conectados en redes de telecomunicaciones. Dependiendo de las necesidades de la empresa se pueden establecer diferentes tipos de conexiones en red. En una empresa pequeña, los ordenadores personales están conectados en redes de área local (LAN), haciendo posible que sus usuarios se comuniquen y compartan datos, trabajo y equipo. Hay redes de área amplia (WAN) que conectan ordenadores ubicados en lugares remotos, tanto dentro de una empresa como fuera de ella. Internet, la red de redes, conecta una gran variedad de redes de distintos ámbitos en todo el mundo. A través de dichas conexiones, los usuarios de ordenadores personales pueden tener acceso a los recursos informáticos de la empresa, como por ejemplo, bases de datos (p.).

2.2.2.5 RECURSOS HUMANOS

Según García Bravo (2000), En cuanto a los recursos humanos, debemos distinguir entre personas especialistas en sistemas de información y usuarios finales. El personal especializado de sistemas de información incluye analistas de sistemas, programadores y operadores. Los usuarios finales son las personas que utilizan los sistemas de información o el output que estos generan, es decir, que se refiere a la mayoría de personas de una organización (p.).

2.2.2.6 PROCEDIMIENTOS

Según García Bravo (2000), Los procedimientos constituyen las políticas y métodos que deben ser seguidos al utilizar, operar y mantener un sistema de información. Por ejemplo, se requiere la utilización de procedimientos para establecer cuándo se debe ejecutar un programa de pago de nóminas, definiendo las veces que se debe ejecutar, quién está autorizado para ejecutarlo, y quién tiene acceso a los informes producidos .

2.2.3 FUNCIONES DE UN SISTEMA DE INFORMACIÓN.

2.2.3.1 RECOLECCION.

De acuerdo a Raúl H. (2002), Esta función implica la captura y el registro de datos. Actúa como el órgano sensorio de la organización. Es una función costosa (con frecuencia es la más cara del sistema de información) y muy expuesta a la generación de errores, aunque este último aspecto está siendo atenuado en grado creciente por la aplicación de nuevas tecnologías de captura de datos, como la lectura de caracteres ópticos o magnéticos y la lectura de código de barras (p. 39).

2.2.3.2 CLASIFICACION.

De acuerdo a Raúl H. (2002), Esta función consiste en identificar los datos, agruparlos en conjuntos homogéneos, y ordenarlos teniendo en cuenta la

manera en que será necesario recuperarlos. Vale decir que los datos se agrupan en estructuras diseñadas conforme a las necesidades del uso que se hará de ellos. El almacenamiento de datos en archivos computadorizados dispone de técnicas que han permitido alcanzar un elevado nivel de refinamiento en este sentido. Sin embargo, ya que el diseño del sistema de clasificación debe hacerse de acuerdo con la forma en que el usuario recuperará la información, tal diseño no puede ser adecuadamente definido si no se posee una clara comprensión de los procesos de decisión (p. 39).

2.2.3.3 COMPRESION.

De acuerdo a Raúl H. (2002), La compresión es la función por la cual se reduce el volumen de los datos sin disminuir necesariamente la información que suministrarán a su destinatario; muy por el contrario, la compresión generalmente aumenta o hace más expresivo el contenido informativo de los datos. La compresión puede realizarse mediante varios métodos. Uno de ellos es la agregación, por el cual se van acumulando informaciones de detalle para obtener información consolidada de más alto nivel. Teniendo en cuenta que más importante que la existencia de información es la capacidad de procesarla, es fundamental que el sistema de información cuente con componentes que actúen como “compresores”, es decir, proyectados para recibir más información de la que transmiten (p. 40).

2.2.3.4 ALAMCENAMIENTO.

De acuerdo a Raúl H. (2002), Esta función se vincula con la conservación física de los datos y con su adecuada protección. Aunque no todos los datos que procesa un sistema de información se conservan en dispositivos de computación, éstos constituyen el soporte prácticamente obligado del banco de datos de las organizaciones. Aun en las empresas de mayor envergadura en el mundo, la tecnología de computación disponible permite una capacidad virtualmente ilimitada para mantener este banco de datos en condiciones de ser consultado en forma inmediata. En materia de archivos computadorizados, la teoría y la práctica del diseño, la generación, el mantenimiento, la reorganización

y la consulta de las estructuras de datos han alcanzado un alto grado de sofisticación y eficiencia. Como una definición general, puede decirse que se denomina “base de datos” a un conjunto de archivos que responde a la aplicación de herramientas lógicas orientadas específicamente al logro de esa eficiencia (p. 41).

2.2.3.5 RECUPERACION.

De acuerdo a Raúl H. (2002), esta función tiene el propósito de suministrar el acceso a la base de datos. Como se dijo más arriba, depende de un apropiado sistema de clasificación. Cada día están más difundidas las aplicaciones de computación en las que la recuperación de los datos (y, muchas veces, su actualización) debe hacerse en tiempo real, es decir, en el mismo momento en que sucede el hecho que genera la necesidad de la recuperación o la actualización. En estos casos, la computadora interviene en alguna parte de la ejecución de la propia transacción que demanda el uso o actualización de los datos (p. 41).

2.2.3.6 PROCESAMIENTO.

De acuerdo a Raúl H. (2002), el sistema de información (como todo sistema) es un transformador de entradas en salidas a través de un proceso. Esta transformación se realiza mediante cómputos, clasificaciones, cálculos, agregaciones, relaciones, transcripciones y, en general, operaciones que, no importa qué recursos humanos o tecnológicos empleen, persiguen el objetivo de convertir datos en información, es decir, en datos que habrán de tener valor y significado para un usuario. La función de procesamiento implica, principalmente, la modificación de la base de datos para mantenerla actualizada (p. 42).

2.2.3.7 TRANSMISION.

De acuerdo a Raúl H. (2002), esta función comporta la comunicación entre puntos geográficos distantes, sea por el traslado físico del sostén de los

datos (papeles, dispositivos de archivos computadorizados, cintas de audio o video, microfichas, etc.) o por la transmisión de señales (comunicación entre equipos de computación, transmisión de facsímiles, teléfono, etc.) Este aspecto del sistema de información se vincula con la tecnología de comunicaciones, la que se halla tan asociada con la de la computación, e igualmente tan desarrollada, que resulta muy difícil trazar una línea de separación entre ellas. De ahí que suele aplicarse la denominación de telemática a la disciplina o ambiente tecnológico que surge de la combinación de las telecomunicaciones y la informática. Las facilidades disponibles para transmitir datos entre distintos puntos físicos, así como la amplísima gama de capacidades de equipos de computación, permiten descentralizar los recursos de computación y las bases de datos. Esto puede hacerse sin caer en costosas redundancias ni perder la integración de sistemas y archivos, ya que todos los puntos pueden estar interconectados, compartiendo recursos y datos, y manteniendo similares grados de actualización de las bases de datos. Así, se conforman las llamadas redes de procesamiento distribuido, mediante las que se lleva la “inteligencia” de computación al mismo lugar en que se la necesita, sin caer en los costosos aislamientos de la descentralización sin comunicación. Además, las posibilidades de transmisión de datos a través de redes de comunicaciones (desde las limitadas al edificio de una organización hasta las intercontinentales) tienen un impacto fundamental en el planteo estratégico de las empresas y están produciendo cambios trascendentales en la naturaleza y la operación de los negocios (p. 42).

2.2.3.8 EXHIBICION.

De acuerdo a Raúl H. (2002), Mediante esta función, se proporciona una salida de información preparada de modo tal que resulte legible y útil a su destinatario. En un sistema de información basado en el uso de computadoras, esta función es la que implica la interfaz con el ser humano. Todas las funciones descritas hasta aquí realizan diversos tratamientos de la información, pero no producen resultados visibles para el usuario. De ello se encarga esta función de exhibición, la que expone la información en forma impresa, en una pantalla de representación visual o en otros dispositivos La presentación de los resultados

tiene particular importancia para que los mismos revistan el carácter de información, para que aparezcan con significado ante los ojos del usuario, para que reduzcan la ignorancia del mismo, y para que lo induzcan a la acción. En la mayor parte de los sistemas de información ineficientes, el problema central no reside en la ausencia de información, sino en el ocultamiento o enmascaramiento de la misma bajo una maraña de datos en las que el usuario debe “hurgar” para encontrar aquellos que, para él, constituyen información. Esto pone en evidencia la importancia de la función de compresión, por un lado, y la de la precisa determinación de las necesidades informativas de cada puesto de la organización, por el otro (p. 43).

2.2.4 DEFINICIÓN DE ARCHIVO

De una manera muy gráfica y simple se ha dicho que archivo es la suma de tres elementos: Documentos + Organización + Servicio (Antonia H, 1991, p.88). Archivo es uno o más conjuntos de documentos, sea cual sea su fecha, su forma y soporte material, acumulados en un proceso natural por una persona o institución pública o privada en el transcurso de su gestión, conservados, respetando aquel orden, para servir como testimonio e información para la persona o institución que los produce, para los ciudadanos o para servir de fuentes de historia (Antonia H, 1991,p.88).

Un archivo, también denominado fichero, es una unidad de datos o información almacenada en algún medio que puede ser utilizada por programas del ordenador. Cualquier documento de cualquier programa al guardarse recibe el nombre de archivo o fichero, un archivo o fichero son documentos guardados (Santana Bueno, M, 2011, p.1)

Según el Consejo Internacional de Archivos (ICA/CIA), la palabra “archivo” tiene tres acepciones:

1. Conjunto de documentos sean cuales sean su fecha, su forma y su soporte material, producidos o recibidos por toda persona física o moral, y por todo servicios u organismo público o privado, en el ejercicio de su actividad, y

son, ya conservados por sus creadores o por sus sucesores para sus propia necesidades, ya transmitidos a la institución de archivos competente en razón de su valor archivístico (ICA, 2019).

2. Institución responsable de la acogida, tratamiento, inventariado, conservación y servicio de los documentos (ICA, 2019).
3. Edificio o parte de edificio donde los documentos son conservados y servidos (ICA, 2019).

2.2.5 FUNCIONES DE LOS ARCHIVOS

“Tradicionalmente son tres las funciones de un archivo, referidas a los documentos de archivo:” (Antonia H, 1991, p.95).

- Recoger.
- Conservar.
- Servir.

Son funciones muy amplias como ya veremos al hablar de una Administración de Archivos que no se limitan a la materialidad de la recogida y al almacenamiento, sino que exigen un control y canalización de la producción documental, una conservación no sólo física sino que mira a la perdurabilidad y el servicio que conllevará difusión, orientación, ofrecimiento de datos concretos, facilidad de acceso y consulta. La conservación era quizá la única meta del archivero, ampliada hoy por una actitud activa de recogida y superada por la dimensión de servicio a la sociedad (Antonia H, 1991,p.95).

“Las funciones genéricas propias de un archivo se pueden agrupar en tres bloques” (Martin Gavilan, C, 2009, p.3):

2.2.5.1 GÉNESIS O REUNIÓN DE LOS DOCUMENTOS:

Se trata de una reunión “natural” de los materiales, no artificial. Mientras que la formación de una colección bibliográfica es el resultado de una voluntad, de una necesidad o demanda social, el fondo archivístico es el resultado de una actuación espontánea de una institución. El origen de los documentos es fruto de una relación con el organismo productor en el marco de la Administración pública o de la empresa, y no mediante adquisición (compra, intercambio o donación) como en el caso de las bibliotecas. En este sentido, el archivo es consustancial a la actividad de los individuos e instituciones, y de ello se deduce la inevitabilidad de su existencia: el archivo nace espontáneamente como sedimentación documental de una actividad práctica, administrativa o jurídica (Martin Gavilan, C, 2009, p.3).

2.2.5.2 TRATAMIENTO DE LA DOCUMENTACIÓN:

A pesar de la tendencia actual homogeneizadora y estandarizadora de las normas, persisten diferencias muy importantes en el tratamiento de la documentación que se justifica en la distinta génesis y tipología de los materiales. No es posible aplicar sistemas universales de clasificación, a diferencia de lo que ocurre en las bibliotecas: es necesario aplicar sistemas de clasificación propios y específicos, de acuerdo con el tipo de fondo. La selección de la información, en el caso de la archivística, se realiza después de recoger los documentos y la evaluación se lleva a cabo en un contexto complejo de valoración administrativa, fiscal, legal, testimonial e informativa, y con la finalidad de conservar permanentemente los documentos. En las bibliotecas, por lo contrario, la selección se realiza antes, según las necesidades de los usuarios. La descripción en los archivos también presenta algunas peculiaridades: se puede realizar a muchos niveles (fondo, sección, serie, unidad documental), según el tipo de agrupación documental de que se trate. El objeto del tratamiento archivístico es el documento de archivo, que es producto de una acción continuada y forma parte de una secuencia de producción documental, al tiempo que refleja las funciones y

las actividades de los individuos y de las instituciones que lo han generado. Las bibliotecas se ocupan de otras tipologías documentales de concepción individual e independiente, sobre las que asume la responsabilidad de informar. Del carácter único e irreplicable del documento de archivo, y esa originalidad comporta unas obligaciones jurídicas de conservación y acceso que los documentos bibliográficos, de naturaleza múltiple, no tienen (Martin Gavilan, C, 2009, p. 4).

2.2.5.3 SERVIR LA DOCUMENTACIÓN:

El concepto de libre acceso a la información es un derecho constitucional básico que cada día cobra más fuerza en los archivos. La modalidad de acceso en los archivos es también un rasgo distintivo: a los archivos les concierne una fuerte carga jurídica (restricciones, términos, normativa legal). Las funciones de reunir y tratar la documentación están orientadas para dar un servicio a (Martin Gavilan, C, 2009, p.4):

- a.** Las administraciones o personas productoras del archivo para obtener los documentos de forma eficaz;
- b.** Los ciudadanos para ofrecer la información que soliciten, sobre todo en los archivos públicos;
- c.** La investigación y difusión cultural, sobre todo para poner a disposición del solicitante datos del pasado. Junto con la función educativa y cultural, los archivos cada vez cobran mayor importancia en el ámbito de la gestión y tratamiento de los documentos administrativos.

De acuerdo a Martin Gavilán, C (2009), “La función genérica de un archivo, por tanto, es reunir, organizar/conservar y difundir/servir documentos. Estas funciones se resumen en las siguientes tareas llevadas a cabo por el personal de archivo” (p.5):

- Organizar la documentación administrativa.
- Asegurar la transferencia periódica a archivos donde su uso no es periódico.

- Valorar los documentos, tras el tiempo establecido, para su conservación o destrucción.
- Clasificación de fondos.
- Descripción para facilitar el acceso a los documentos utilizando aquellos instrumentos necesarios.
- Adecuación de las instalaciones donde se conservan los documentos.

2.2.6 LAS TRES EDADES DE LOS DOCUMENTOS Y LOS ARCHIVOS INTERMEDIOS.

Wyffels, en 1972, planteó su teoría de las tres edades de los documentos que darían lugar a diferentes categorías de archivos. Dicha teoría está en relación con la idea y realidad de que los documentos no son algo muerto o inactivo sino que tienen una vida propia. Este ciclo vital, como otros, contempla después del nacimiento, el crecimiento, el decrecimiento y la muerte. Esas tres edades del ciclo vital se corresponden respectivamente, una vez producidos, con los denominados documentos corrientes, semicorrientes y no corrientes (Antonia H, 1991, p.73).

✓ **La primera etapa o edad:**

Es de circulación y tramitación por los canales y cauces normales en busca de respuesta o solución para el asunto que se ha iniciado. La documentación forma parte de los archivos sectoriales, corrientes o de gestión. Está cerca del funcionario responsable de su tramitación, en su mesa, en archivadores, en armarios al alcance de su mano para su manejo frecuente (Antonia H, 1991, p.73).

✓ **La segunda etapa o edad:**

En ella, una vez recibida la respuesta o solución al asunto empezado, el documento o expediente que lo testimonia ha de seguir siendo guardado, pudiendo ser objeto de consulta o de antecedente, pero no con la frecuencia que en la primera etapa. Esta fase corresponde a una archivación

intermedia en la que poco a poco va decreciendo el valor primario de los documentos, desarrollándose el valor secundario (Antonia H, 1991, p.73).

✓ **La tercera etapa o edad:**

El documento asume un valor permanente y se ceñirá a ser consultado por su valor cultural e informativo con fines de investigación. Su archivación y conservación serán definitivas (Antonia H, 1991, p.73).

2.2.7 GESTION DE DOCUMENTOS

La gestión de documentos, ligada íntimamente a la valoración de éstos, comprende: la planificación, el control, la dirección, la organización, la promoción y otras actividades de gestión relativas a los documentos antes de llegar a los archivos históricos, incluyendo el manejo de la correspondencia, formularios, microformas, técnicas de la automatización de datos, etc. La eficacia del sistema se traducirá en la obtención de calidad y cantidad de documentos producidos, evitando la acumulación irracional; simplificación de actividades, coordinación entre órganos productores y archivos, eliminación sistemática, información al día; información complementaria, en definitiva fluidez y eficacia. Uno de los mayores problemas con que el profesional de archivo se enfrenta es con la acumulación masiva de la documentación y su eliminación (Antonia H, 1991, p.76).

2.2.8 METODOLOGÍA ICONIX.

ICONIX “es una metodología creada por Doug Rosenberg & Kendall Scott a partir de una síntesis del proceso unificado de los tres amigos” (Booch, Rumbaugh, & Jacobson, 2006) y que ha dado soporte y conocimiento a la metodología ICONIX desde 1993.

Es una metodología semi-ágil, este concepto establece un nuevo enfoque en el desarrollo de sistemas y es de las más reconocidas y usadas por los desarrolladores de e-projects que las metodologías convencionales (ISO-9000, CMM, etc) debido a la simplicidad de sus reglas y prácticas, su orientación a

equipos de desarrollo de pequeño tamaño, su flexibilidad ante los cambios y su ideología de colaboración (Amavizca Valdez, García Ruíz, Jiménez López, Duarte Guerrero, & Vázquez Brindis, 2014).

Proceso de ICONIX es una metodología de análisis y diseño impulsado por casos de uso. El enfoque principal está en cómo conseguir fiabilidad desde los casos de uso hasta el código en el menor número de pasos posible (Rosenberg, Collins-Cope, & Stephens, 2007).

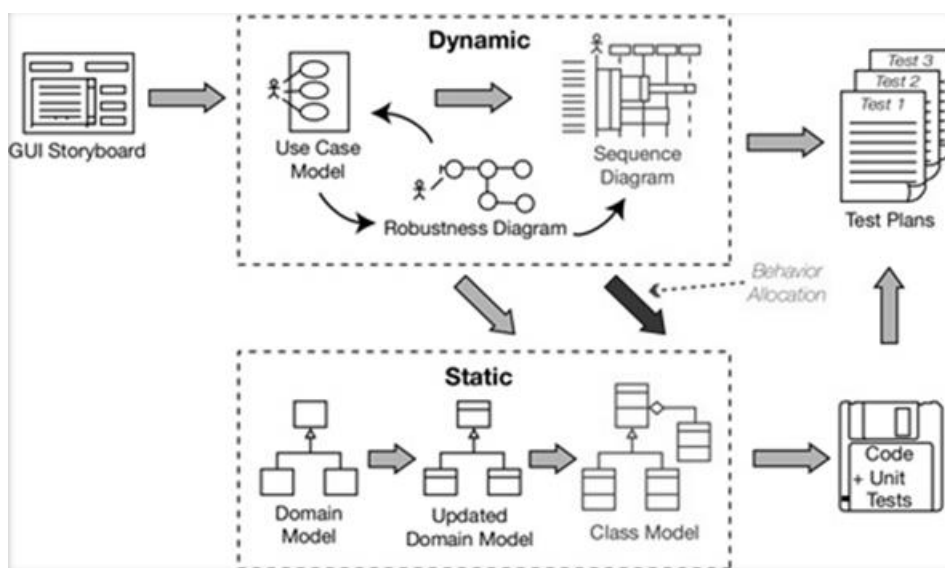


Figura n° 2.3: Fuente: (Rosenberg & Stephens, Use Case Driven Object Modeling with UMLTheory And Practice: Theory and Practice, 2007).

El proceso de ICONIX es proceso de modelado de objetos abierto y libre de usar. Es simple, impulsado por casos de uso, y ágil. El proceso se enfoca en el área que se encuentra entre los casos de uso y el código. Su énfasis está en qué se necesita para pasar un punto en el ciclo de vida desde donde estás iniciando: tienes un punto de partido sobre algunos casos de uso, y ahora necesitas hacer un buen análisis y diseño. El Proceso de ICONIX se divide en flujos de trabajo dinámicos y estáticos, que son altamente iterativos: se puede ir a través de una iteración de todo el proceso para un pequeño lote de casos de uso (uno par de casos de uso de valor, que no sean muy extensos dada que cada caso de uso es sólo un par de párrafos), todo el camino hasta el código fuente y pruebas unitarias. Por esta razón, el proceso de ICONIX se adapta bien a los proyectos ágiles, donde se

necesita la regeneración rápida de factores tales como los requisitos, el diseño y las estimaciones (Rosenberg & Stephens, 2007).

2.2.8.1 CARACTERÍSTICAS DE ICONIX

2.2.8.1.1 ITERATIVO E INCREMENTAL

Las iteraciones múltiples ocurren entre el desarrollo del modelo de dominio e identificar y analizar los casos de uso. Otras iteraciones existen también, como los procesos del equipo a través del ciclo de vida. El modelo estático se refina incrementalmente durante las iteraciones sucesivas a través del modelo dinámico (compuesto de los casos de uso, análisis de robustez y el diagrama de secuencia) (Rosenberg y Stephens, 2007).

2.2.8.1.2 TRAZABILIDAD

El enfoque ofrece un alto grado de seguimiento. Por el contrario, a cada paso usted consultara de alguna manera los requisitos anteriores. Nunca hay un punto en que el proceso le permita desviarse lejos de las necesidades del usuario. Seguimiento se refiere también, al hecho que usted puede seguir los objetos paso a paso como el análisis dentro del diseño (Rosenberg y Stephens, 2007).

2.2.8.1.3 DINÁMICA DE UML

“El enfoque ofrece uso aerodinámico del UML” (Fernández, 2004).

Rosenberg y Stephens (2007), destacan un análisis de requisitos, un análisis y diseño preliminar, un diseño y una implementación como las principales tareas.

2.2.8.2 TÉCNICAS DE LA METODOLOGÍA ICONIX

Según Doug Rosenberg y Matt Stephens (2007), las técnicas de la metodología ICONIX son:

2.2.8.2.1 MODELO DE DOMINIO

Rosenberg y Scott (1999), Rosenberg y Stephens (2007) definen el modelado de dominio es la tarea de descubrir (clases), que representan cosas y conceptos del "mundo real". El modelo de dominio para un proyecto define el alcance y la base para construir los casos de uso. El modelo de dominio sirve como un glosario de términos, que es utilizado al inicio para escribir los casos de uso. El objetivo del modelado de dominio es realizar un primer levantamiento de las clases (objetos) que forman parte del problema.

“El Modelo de Dominio es un artefacto colaborativo vivo. Es refinado y actualizado en cada parte del proyecto, de modo que refleja siempre la comprensión actual del espacio del problema.” (Rosenberg y Stephens, 2007).

“El modelo del dominio no es más que un diagrama de clases sin ningún tipo de detalle (sin atributos, sin métodos,...)” (Gutiérrez, 2008)

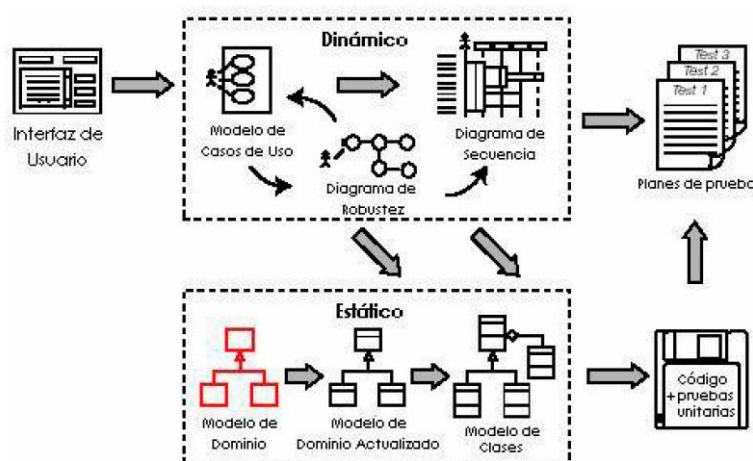


Figura n° 2.4: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.2 MODELO DE CASOS DE USO

Con un primer modelo de dominio inicial en su lugar, es hora de comenzar a escribir los casos de uso. Los casos de uso dan un modo estructurado de capturar los requisitos de comportamiento de un sistema, de modo que puede razonablemente crear un diseño desde ellos. Le ayudan a responder ciertas preguntas fundamentales: ¿Qué están tratando de hacer los usuarios del sistema? ¿Cuál es la experiencia del usuario? Una cantidad sorprendente de lo que su software debe hacer se dicta por el modo en que los usuarios deben interactuar con él (Rosenberg y Stephens, 2007).

Rosenberg y Scott (1999), muestran elementos clave, donde; los casos de uso son desarrollados a partir del modelo de dominio, el análisis de robustez identifica un conjunto de objetos que satisfacen cada caso de uso, el diagrama de secuencia traza el flujo de los mensajes entre los objetos conforme se especificó en los casos de uso, los requisitos de usuario son asignados a los casos de uso y clases, los casos de uso se usan para las pruebas durante la implementación.

Un caso de uso describe la interacción entre el usuario y el sistema para alcanzar un objetivo (Rosenberg y Stephens, 2007).

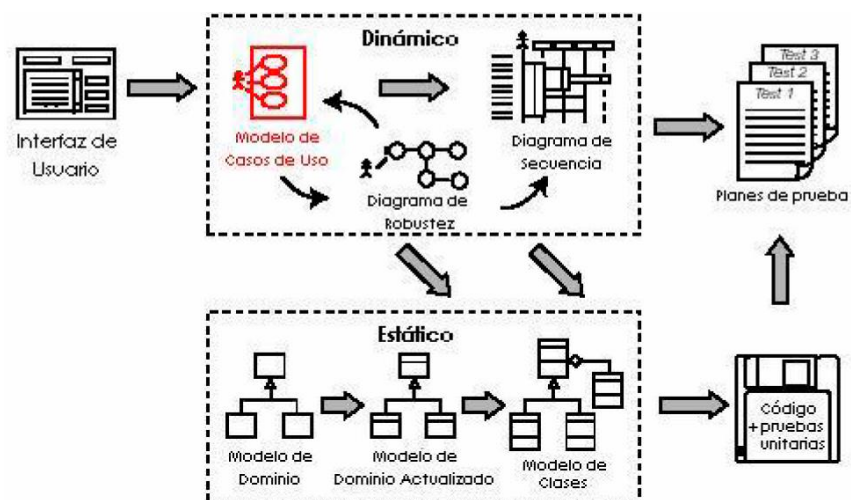


Figura n° 2.5: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.3 REVISION DE REQUISITOS

La sesión de revisión de requisitos garantiza que el sistema tal y como se describe coincide con los requisitos. Se trata de un período de sesiones de colaboración que impliquen al representante(s) del cliente, los usuarios finales (es decir, las personas que realmente van a utilizar el sistema, o quien está usando el sistema actual que se sustituirá), y las personas de marketing básicamente, todos los stakeholders que tienen un interés en asegurar que los requisitos encajen con su punto de vista del sistema (Rosenberg y Stephens, 2007).

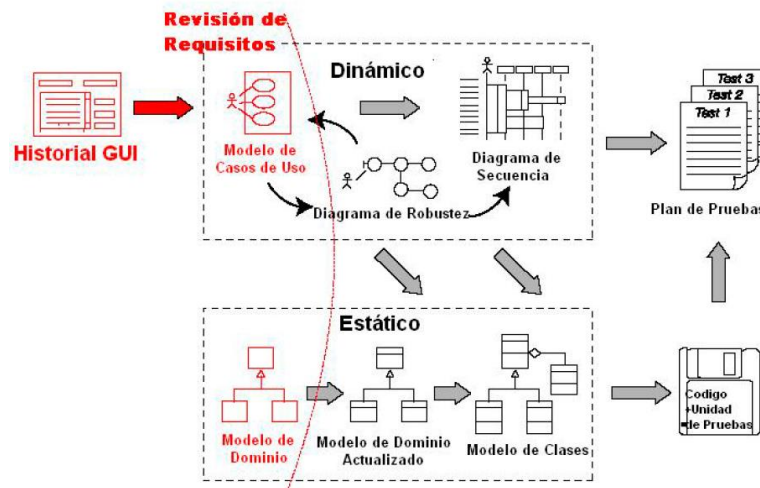


Figura n° 2.6: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.4 ANÁLISIS DE ROBUSTEZ

Para obtener a partir de los casos de uso un diseño detallado (y luego el código), lo que se necesita es enlazar los casos de uso a los objetos. La técnica que se describe en este capítulo, el análisis de robustez, le ayuda a superar la brecha que existe del análisis al diseño. En pocas palabras, es una manera de analizar sus casos de uso e identificar un primer conjunto de objetos para cada caso de uso. Estos se clasifican en objetos interfaz, objetos entidad, y controladores (que son a menudo más como funciones que como objetos) (Rosenberg y Stephens, 2007).

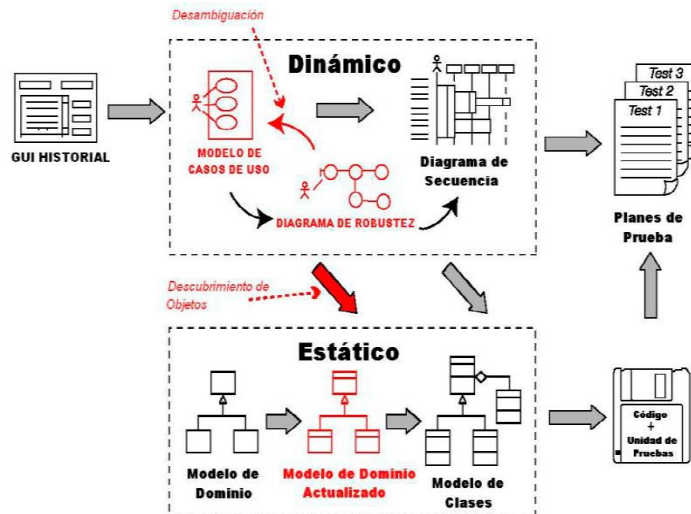


Figura n° 2.7: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.5 REVISIÓN PRELIMINAR DEL DISEÑO

Las sesiones de Revisión del Diseño Preliminar (RDP) ayudan a asegurarse que los diagramas de robustez, el modelo de dominio, y la descripción de casos de uso coincidan entre sí. Esta revisión es el "puente" entre el diseño preliminar y en las etapas del diseño detallado, para cada paquete de casos de uso. (Rosenberg y Stephens, 2007).

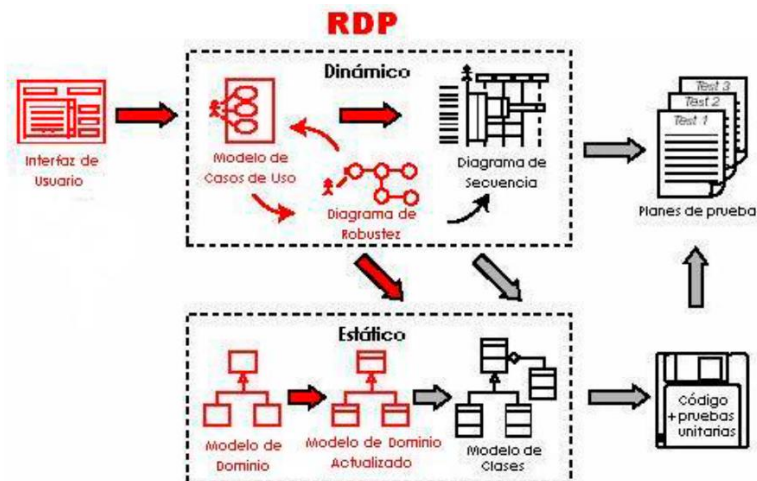


Figura n° 2.8: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.6 ARQUITECTURA TÉCNICA

El objetivo de arquitectura técnica (AT) es obtener un sentido general del sistema que vas a desarrollar. ¿Será un sistema basado en Internet o un sistema en Php, VB NET o Java Swing, para un cliente muy rico? Es necesario utilizar un framework de aplicación específico (por ejemplo, un framework de una compañía estándar). No hay una notación estándar o un formato para documentar la AT, la profundidad y el formato de la arquitectura técnica-y los convenios para crearla-varían mucho de empresa a empresa, por lo que no insistiremos en esta área demasiado tiempo (Rosenberg y Stephens, 2007).

Según Rosenberg y Stephens (2007), la AT abarca tres áreas: (1) modelo de despliegue (servidores, comunicaciones, topología del sistema, navegadores Web, etc.), (2) modelo de paquetes/componentes, (3) modelo de datos. Diseñar una arquitectura técnica (arquitectura del sistema y arquitectura del software) implica tener una visión general del software a desarrollar, a fin seleccionar la tecnología para la implementación en relación a los requisitos. La AT describe el sistema en términos de estructura, incluye la topología del sistema (servidores, ubicación física de la red, etc.). Para diseñar una buena AT considerar: accesos simultáneos al software, horas pico de uso, número de transacciones por minuto, posibles fallas, escalabilidad, seguridad, disponibilidad, localización, auditoría, portabilidad e implementación.

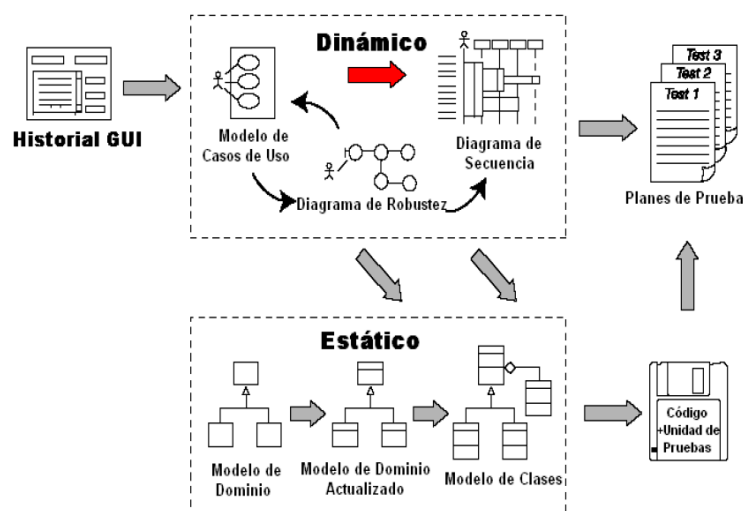


Figura n° 2.9: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.7 DIAGRAMAS DE SECUENCIA

Una vez que se ha finalizado el análisis de robustez, y la Revisión del Diseño Preliminar, es tiempo de iniciar el Diseño Detallado. En este momento, la descripción de sus casos de uso debe ser completa, correcta, detallada y explícita. En resumen, los casos su uso deben estar en un estado del que se pueda crear un diseño detallado, tener descubiertas casi todas las clases de dominio y, contar con la arquitectura técnica. (Rosenberg y Stephens, 2007).

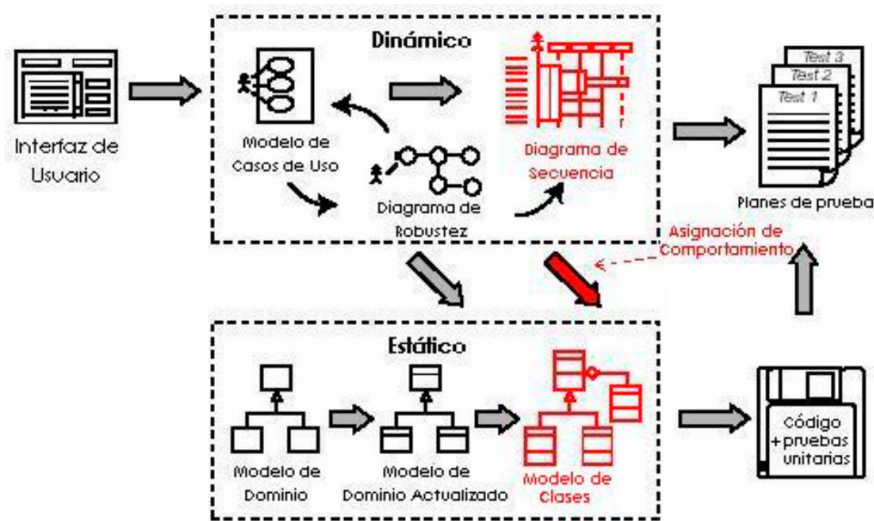


Figura n° 2.10: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.8 REVISIÓN CRÍTICA DEL DISEÑO

Tu proyecto debería estar ahora en mucha mejor condición que muchos otros proyectos que están en esta etapa. Por ahora, has utilizado el análisis de robustez para desambiguar la descripción de los casos de uso y descubrir las clases de dominio faltantes, has mantenido una revisión de diseño preliminar (PDR) para asegurarte que los casos de uso coinciden con lo que el cliente realmente quiere, y has elaborado cuidadosamente un diseño detallado de los casos de uso que has implementado para esta versión.

Por tanto, estás casi listo para comenzar la codificación-hay sólo una etapa rápida (pero vital) para comprobar el primero de la lista: Revisión del Diseño Critico (CDR) (Rosenberg y Stephens, 2007).

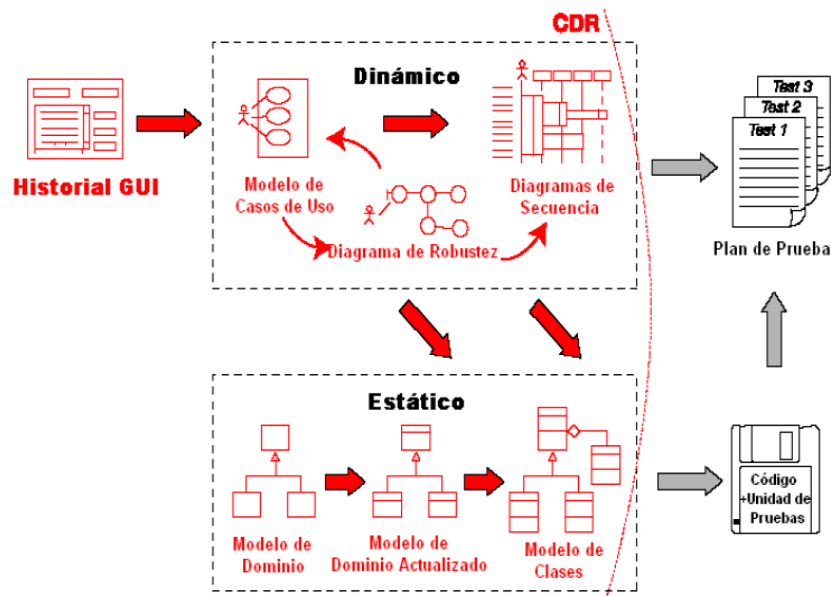


Figura n° 2.11: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.9 IMPLEMENTACIÓN: PASO DEL DISEÑO DETALLADO AL CÓDIGO

“Si ha pasado por todo el esfuerzo para crear un diseño agradable y detallado, vale la pena tener una buena idea de cómo traducir ese diseño detallado en el código fuente (y las pruebas unitarias, por supuesto” (Rosenberg y Stephens, 2007).

El objetivo de la implementación según ICONIX, es cómo hacer el último salto del diseño detallado al código fuente y, mostrar cómo encaja todo para comenzar a codificar. Antes de comenzar la codificación del diseño, definir la arquitectura técnica, contar con los diagramas de secuencia para cada caso de uso y, haber completado la revisión crítica de diseño (Porras, 2010).

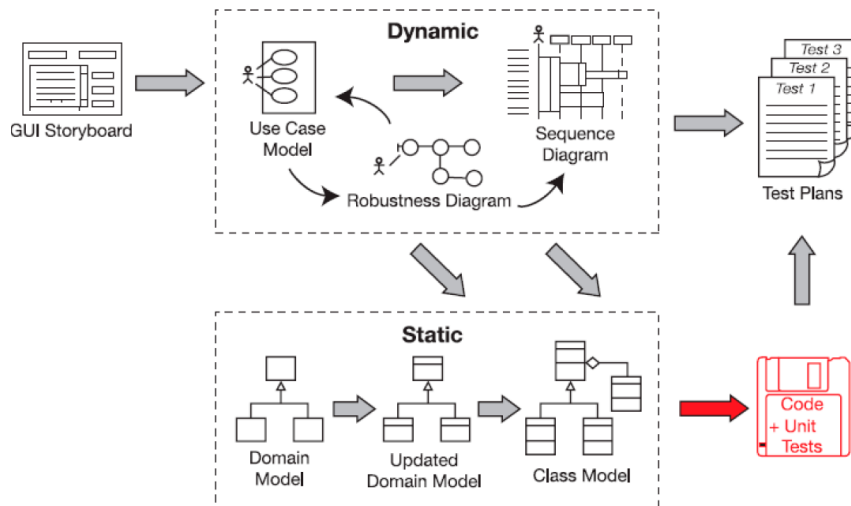


Figura n° 2.12: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.10 REVISIÓN DEL CÓDIGO Y ACTUALIZACIÓN DEL MODELO

Durante la codificación, lo más probable es que te han hecho algunos cambios en el diseño, por lo que el código ahora está ligeramente fuera de sincronía con los diagramas de diseño. Una reacción tristemente común en esta etapa consiste en considerar que la documentación de diseño obsoleto, tirar a la basura, y seguir todo el posterior desarrollo del trabajo sin hacer ningún trabajo de diseño más (Rosenberg y Stephens, 2007).

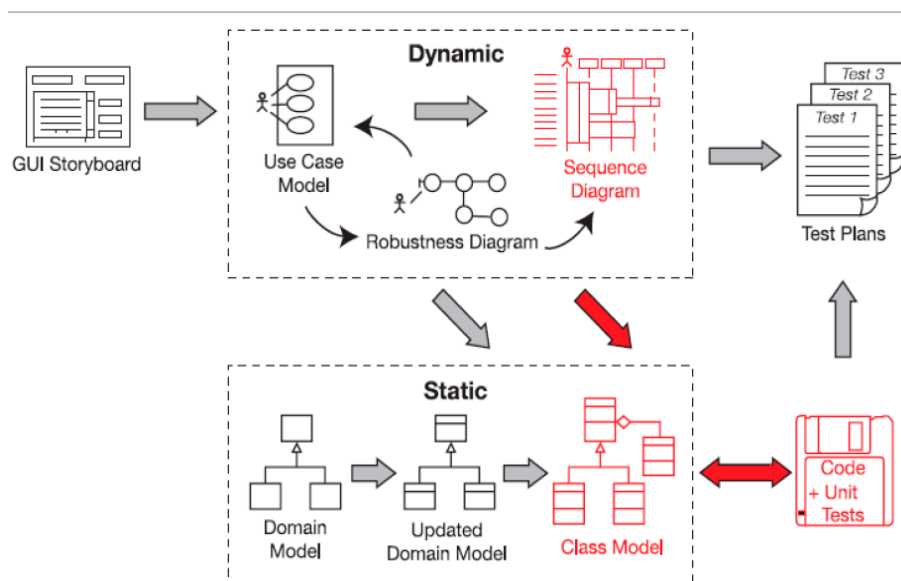


Figura n° 2.13: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.11 DISEÑO GUIADO POR PRUEBAS

(Rosenberg y Stephens, 2007) señaló que al finalizar la codificación no se puede estar seguro de que el código reúne todos los escenarios de casos de uso. Las Prueba basadas en el diseño (DDT, por sus siglas en inglés) provee un método eficaz para producir casos de prueba y verificar que todos los escenarios específicos están completos. Se puede además usar este proceso para escribir pruebas unitarias ejecutables de estos casos de prueba.

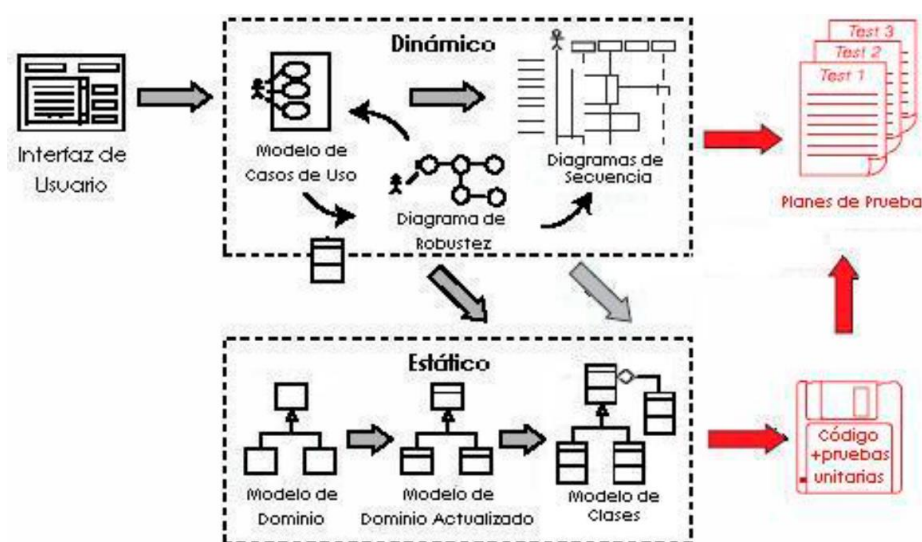


Figura n° 2.14: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.8.2.12 ATENDIENDO REQUISITOS

“No es una parte fundamental del proceso ICONIX, porque las diferentes organizaciones tienen diferentes estrategias para el manejo de los requisitos” (Rosenberg y Stephens, 2007).

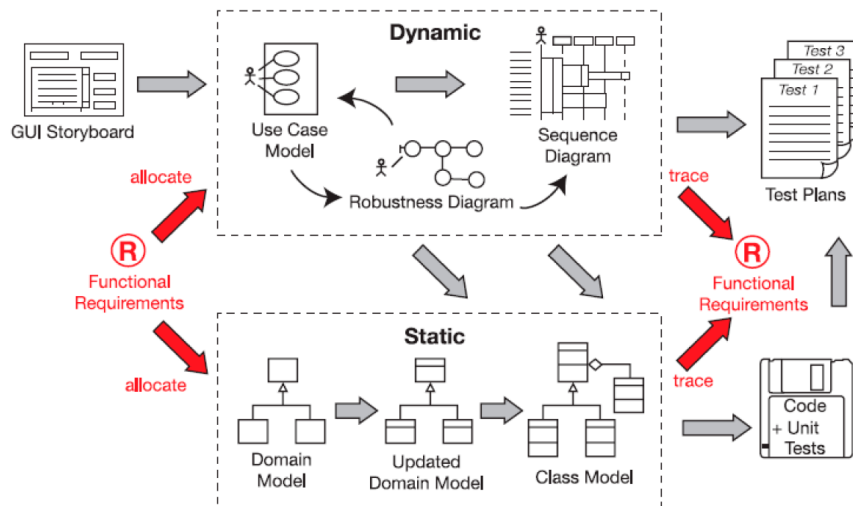


Figura n° 2.15: Fuente: (Doug Rosenberg y Matt Stephens, 2007).

2.2.9 PROGRAMACIÓN ORIENTADA A OBJETOS.

De acuerdo a Mazón B., Catuche J., Chimarro V. y Rivas W (2015), Programación Orientada a Objetos. Es una técnica o estilo de programación que utiliza objetos como bloque esencial de construcción. Se parte de la creación de clases que son en realidad tipos abstractos de datos donde se definen datos y métodos. Es decir, un objeto es una instancia de una clase que posee una copia de la definición de datos y métodos, considerándose estado a los datos con sus valores (atributos o características), y comportamiento a los métodos que trabajan con estos datos. Entonces la poo enfatiza en los tipos de datos y las operaciones intrínsecas que pueden desarrollarse en aquellos tipos de datos, los datos no fluyen libremente por el sistema, ya que están protegidos de alguna modificación accidental; son los “mensajes” los que pueden mover en el sistema, pues se envía un mensaje a un objeto para que ejecute una acción (pp.17-18).

De acuerdo a, Joyanes (2007), es una filosofía, un modelo de programación, con su teoría y su metodología, que conviene conocer y estudiar, antes de nada. Un lenguaje orientado a objetos es un lenguaje de programación que permite el diseño de aplicaciones orientadas a objetos. Dicho esto, lo normal es que toda persona que vaya a desarrollar aplicaciones orientadas a objetos aprenda primero la “filosofía” (o adquiera la forma de pensar) y después el lenguaje, porque “filosofía” sólo hay una y lenguajes muchos. En este documento veremos brevemente los conceptos básicos de la programación orientada a objetos desde un

punto de vista global, sin particularizar para ningún lenguaje de programación específico.

Larman (1999), el paradigma Orientado a Objetos se basa en el concepto de objeto, un objeto es aquello que tiene estado (propiedades más valores), comportamiento (acciones y reacciones a mensajes) e identidad (propiedad que lo distingue de los demás objetos). La estructura y comportamiento de objetos similares están definidos en una clase común, los términos instancia y objeto son intercambiables. Una clase es un conjunto de objetos que comparten una estructura y comportamiento común, la diferencia entre un objeto y una clase es que un objeto es una entidad concreta que existe en tiempo y espacio, mientras que una clase representa una abstracción, la "esencia" de un objeto, tal como son, de aquí que un objeto no es una clase, sin embargo, una clase puede ser un objeto.

2.2.9.1 CLASES

Según Moreno, F. (2000), una clase, es simplemente una abstracción que hacemos de nuestra experiencia sensible. El ser humano tiende a agrupar seres o cosas -objetos- con características similares en grupos -clases-. Así, aun cuando existen por ejemplo multitud de vasos diferentes, podemos reconocer un vaso en cuanto lo vemos, incluso aun cuando ese modelo concreto de vaso no lo hayamos visto nunca. El concepto de vaso es una abstracción de nuestra experiencia sensible. Quizás el ejemplo más claro para exponer esto lo tengamos en las taxonomías; los biólogos han dividido a todo ser (vivo o inerte) sobre la tierra en distintas clases.

2.2.9.2 OBJETO

“Los objetos son entes, entidades, sujetos o cosas que encontramos en el dominio del problema de nuestra realidad. Entiéndase; en situaciones o problemas de nuestro mundo cotidiano empresarial, organizacional o institucional, que se requiere manejar o solucionar mediante la computadora” (López, 2006, p.243).

“Un objeto es una encapsulación abstracta de información, junto con los métodos o procedimientos para manipularla. Un objeto contiene operaciones que definen su comportamiento y variables que definen su estado entre las llamadas a las operaciones.” (Florez, 2005, p. 49).

2.2.10 CARACTERÍSTICAS ESENCIALES DE LA POO

2.2.10.1 ABSTRACCIÓN

De acuerdo a Mazón B., Catuche J., Chimarro V. y Rivas W (2015), el proceso de abstracción consiste en identificar o seleccionar entidades u objetos reales con similares características (atributos) y comportamiento; luego, se crean las clases que vienen a ser los modelos o plantillas donde se definen datos y métodos.

2.2.10.2 INSTANCIACIÓN

De acuerdo a Mazón B., Catuche J., Chimarro V. y Rivas W (2015), el proceso de instanciación implica crear objetos a partir de clases. Para instanciar objetos en la aplicación informática, es necesario contar con el modelo de clases. Los objetos informáticos son instancias (copia exacta) de clases con espacio en memoria para leer o escribir sus datos a través de sus métodos. Una vez que el objeto está creado, éste posee identidad, estado (datos con valores o atributos) y comportamiento (métodos que actúan sobre los propios datos).

2.2.10.3 ENCAPSULACIÓN

La Encapsulación es el proceso de ocultar todos los detalles de un objeto que no contribuyen a sus características esenciales. Esencialmente, significa que aquello que está en el interior de la clase está oculto; Solo las interfaces externas se pueden ver por otros objetos. El usuario de un objeto nunca necesitará conocer el interior de una clase. Una de las ventajas del encapsulamiento es que proporciona al programador libertad en la implementación de los detalles de un sistema. La única restricción que tendrá el

programador es mantener la interfaz abstracta que ven los usuarios externos (Joyanes, 2007).

2.2.10.4 HERENCIA

Según los autores Cervantes, J., Gómez, M., González P. y García A. (2016), la generalización es una relación entre clases en las que hay una clase padre, llamada superclase, y una o más clases hijas especializadas, a las que se les denomina subclases. La herencia es el mecanismo mediante el cual se implementa la relación de generalización. En la práctica, cuando se codifica un sistema, se habla de herencia en lugar de generalización. Cuando hay herencia, la clase hija o subclase adquiere los atributos y métodos de la clase padre.

Según el autor, Joyanes (2008), él entiende que generalización es la propiedad que permite compartir información entre dos entidades evitando la redundancia. En el comportamiento de objetos existen con frecuencia propiedades que son comunes en diferentes objetos y esta propiedad se denomina generalización. En orientación a objetos, el mecanismo que implementa la propiedad de generalización se denomina herencia. La herencia permite definir nuevas clases a partir de otras clases ya existentes, de modo que presentan las mismas características y comportamiento de éstas, así como otras adicionales (p.61).

2.2.10.5 POLIMORFISMO

De acuerdo a, Joyanes (2008), el polimorfismo es la propiedad que permite a una operación (función) tener el mismo nombre en clases diferentes y que actúe de modo diferente en cada una de ellas. Esta propiedad es intrínseca a la vida ordinaria ya que una misma operación puede realizar diferentes acciones dependiendo del objeto sobre el que se aplique. Así, por ejemplo, se puede abrir una puerta, abrir una ventana, abrir un libro, abrir un periódico, abrir una cuenta corriente en un banco, abrir una conversación, abrir un congreso, etc. En cada caso se realiza una operación diferente. En orientación a objetos, cada clase “conoce” cómo realizar esa operación. En la práctica, el polimorfismo significa la

capacidad de una operación de ser interpretada sólo por el propio objeto que lo invoca. Desde un punto de vista práctico de ejecución del programa, el polimorfismo se realiza en tiempo de ejecución ya que durante la compilación no se conoce qué tipo de objeto y por consiguiente qué operación ha sido invocada.

Según los autores Cervantes, J., Gómez, M., González P. y García A. (2016), el polimorfismo, en la POO, es la propiedad que tienen los objetos de las subclases de mostrarse como de la superclase. Por ejemplo, si tenemos dos clases X y Z que tienen el mismo ancestro W, entonces podemos usar una variable de clase W que pueda tomar el valor de un objeto de clase X o también el valor de un objeto de clase Z.

2.2.11 LENGUAJE DE PROGRAMACIÓN.

Define, Joyanes (2008), cuando el procesador es una computadora, el algoritmo se ha de expresar en un formato que se denomina programa, ya que el pseudocódigo o el diagrama de flujo no son comprensibles por la computadora, aunque pueda entenderlos cualquier programador. Un programa se escribe en un lenguaje de programación y las operaciones que conducen a expresar un algoritmo en forma de programa se llaman programación. Así pues, los lenguajes utilizados para escribir programas de computadoras son los lenguajes de programación y programadores son los escritores y diseñadores de programas. El proceso de traducir un algoritmo en pseudocódigo a un lenguaje de programación se denomina codificación, y el algoritmo escrito en un lenguaje de programación se denomina código fuente. En la realidad la computadora no entiende directamente los lenguajes de programación sino que se requiere un programa que traduzca el código fuente a otro lenguaje que sí entiende la máquina directamente, pero muy complejo para las personas (p.36)

2.2.11.1 PHP

De acuerdo a, Gallego, A., (2003), Que es PHP?; PHP (acrónimo de Hipertexto Preprocessor) es un lenguaje “del lado del servidor” (esto significa que PHP funciona en un servidor remoto que procesa la página Web antes de que

sea abierta por el navegador del usuario) especialmente creado para el desarrollo de páginas Web dinámicas. Puede ser incluido con facilidad dentro del código HTML, y permite una serie de funcionalidades tan extraordinarias que se ha convertido en el favorito de millones de programadores en todo el mundo. Combinado con la base de datos MySQL, es el lenguaje estándar a la hora de crear sitios de comercio electrónico o páginas Web dinámicas. Entre sus características fundamentales son: *Gratuito, Gran popularidad, Enorme eficiencia y Sencilla integración con múltiples bases de datos* (p.26).

2.2.12 LENGUAJE SQL.

De acuerdo a la referencia de los autores Silberschatz, A., Korth, H. y Sudarshan, S. (2006), IBM desarrolló la versión original de SQL, originalmente denominado Sequel, como parte del proyecto System R a principios de 1970. El lenguaje Sequel ha evolucionado desde entonces y su nombre ha pasado a ser SQL (Structured Query Language, lenguaje estructurado de consultas). Hoy en día, numerosos productos son compatibles con el lenguaje SQL y se ha establecido como el lenguaje estándar para las bases de datos relacionales. En 1986, ANSI (American National Standards Institute, Instituto nacional americano de normalización) e ISO (International Standards Organization, Organización internacional de normalización) publicaron una norma SQL, denominada SQL-86. En 1989 ANSI publicó una extensión de la norma para SQL denominada SQL-89. La siguiente versión de la norma fue SQL-92 seguida de SQL: 1999; la versión más reciente es SQL: 2003. Las notas bibliográficas proporcionan referencias a esas normas (pp.61-62).

El lenguaje SQL tiene varios componentes:

- ✓ **Lenguaje de definición de datos (LDD).** El LDD de SQL proporciona comandos para la definición de esquemas de relación, borrado de relaciones y modificación de los esquemas de relación.
- ✓ **Lenguaje interactivo de manipulación de datos (LMD).** El LMD de SQL incluye un lenguaje de consultas basado tanto en el álgebra relacional como en el

cálculo relacional de tuplas. También contiene comandos para insertar, borrar y modificar tuplas.

- ✓ **Integridad.** El LDD de SQL incluye comandos para especificar las restricciones de integridad que deben cumplir los datos almacenados en la base de datos. Las actualizaciones que violan las restricciones de integridad se rechazan.
- ✓ **Definición de vistas.** El LDD de SQL incluye comandos para la definición de vistas. Control de transacciones. SQL incluye comandos para especificar el comienzo y el final de las transacciones.
- ✓ **SQL incorporado y SQL dinámico.** SQL incorporado y SQL dinámico definen cómo se pueden incorporar instrucciones de SQL en lenguajes de programación de propósito general como C, C++, Java, PL/I, Cobol, Pascal y Fortran.
- ✓ **Autorización.** El LDD de SQL incluye comandos para especificar los derechos de acceso a las relaciones y a las vistas.

2.2.12.1 POSTGRESQL

De acuerdo a la referencia de los autores Silberschatz, A., Korth, H. y Sudarshan, S. (2006), PostgreSQL es un sistema gestor de bases de datos relacionales de objetos de código abierto. Es un descendiente de uno de los primeros sistemas de este tipo, el sistema Postgres desarrollado bajo la dirección del profesor Michael Stonebraker en la Universidad de California en Berkeley. El nombre "Postgres" proviene del nombre de un sistema pionero de bases de datos relacionales, Ingres, también desarrollado bajo la dirección de Stonebraker en Berkeley. Actualmente, PostgreSQL soporta SQL92 y SQL: 1999 y ofrece características como las consultas complejas, las claves externas, los disparadores, las vistas, la integridad transaccional y el control de concurrencia de varias versiones. Además, los usuarios pueden ampliar PostgreSQL con tipos de datos, funciones, operadores y métodos de indexación nuevos. PostgreSQL trabaja con gran variedad de lenguajes de programación (incluidos C, C++, Java, Perl, Tcl y Python). Quizás, el punto fuerte de PostgreSQL sea que, junto con MySQL, son los dos sistemas de bases de datos relacionales de código abierto más utilizados.

La licencia de PostgreSQL es la licencia BSD, que concede libre de cargo permiso para el uso, modificación y distribución del código y de la documentación de PostgreSQL con cualquier propósito. (p. 807).

2.2.13 FRAMEWORK.

De acuerdo a la publicación de, Galindo, J. y Camps, J. (2008), se define como un conjunto de clases cooperativas que construyen un diseño reutilizable para un tipo específico de software. Un Framework proporciona la arquitectura partiendo el diseño en clases abstractas y definiendo sus responsabilidades y colaboraciones. Un desarrollador realiza una aplicación haciendo subclases y componiendo instancias a partir de las clases definidas por el Framework. Un framework es una aplicación semi-completa, de control invertido, así un Framework difiere de una librería de clases: -En una librería de clases, el control del flujo se encuentra en el código de la aplicación que realiza llamadas a los métodos de la librería de clases -En un framework, el control del flujo está en código del framework que realiza llamadas al código de la aplicación (control invertido)

2.2.13.1 CODEIGNITER

De acuerdo a los sustraído de, CodeIgniter, (2019), es un Framework, un marco de desarrollo de aplicaciones, un conjunto de herramientas, para crear sitios web utilizando PHP. Su objetivo es permitirle desarrollar proyectos mucho más rápido de lo que podría hacerlo si estuviera escribiendo código desde cero, proporciona un amplio conjunto de bibliotecas para las tareas más comunes, así como una interfaz simple y una estructura lógica para acceder a estas bibliotecas CodeIgniter le permite centrarse creativamente en su proyecto al minimizar la cantidad de código necesario para una tarea determinada.

CodeIgniter es adecuado si:

- ✓ Quiere un framework ligero.
- ✓ Necesita un rendimiento excepcional.

- ✓ Quiere compatibilidad con cuentas de hosting estándar que ejecutan una variedad de versiones y configuraciones de PHP.
- ✓ Necesita un framework que requiera una configuración mínima.
- ✓ Quiere un framework que no requiera el uso de la línea de comando.
- ✓ Desea un framework que no requiera reglas de codificación restrictivas.
- ✓ No está interesado en bibliotecas monolíticas de gran tamaño como PEAR.
- ✓ No quiere verse forzado a aprender un lenguaje de plantillas (aunque, si lo desea, dispone de un analizador de plantillas propio).
- ✓ Busca soluciones simples, lejos de la complejidad.
- ✓ Quiere una documentación clara y completa.

2.2.14 ARQUITECTURA MODELO VISTA CONTROLADOR.

De acuerdo a, Galindo, J. y Camps, J. (2008), En la arquitectura MVC existe un proceso de abstracción que permite dividir la aplicación entre los componentes involucrados en mantener los datos y los que presentan los datos. De ésta forma minimizamos la dependencia entre ellos y permitimos que diferentes desarrolladores o equipos de desarrolladores puedan cumplir perfiles diferenciados. Lo más importante de ésta arquitectura es que permite que futuros cambios técnicos o de aparición de estándares tengan el mínimo impacto sobre lo ya desarrollado. De ésta forma se consigue construir aplicaciones compuestas por componentes débilmente acoplados. Esta circunstancia añade una enorme cantidad de ventajas.

La arquitectura MVC divide los objetos o componentes implicados en una aplicación en 3 tipos: *modelo, vista y controlador* (Galindo, J. y Camps, J., 2008).

2.1.14.1 MODELO

Representa los datos de la aplicación y las reglas de negocio (que en algunos casos gobiernan el acceso y la modificación a estos datos). El modelo notifica a las vistas de sus cambios y proporciona los métodos necesarios para que

las vistas puedan consultar su modelo asociado y conocer la información que contienen.

2.1.14.2 VISTA

Muestra el contenido del modelo. Accede a los datos del modelo y especifica de qué manera se muestran estos datos. La Vista también se encarga de redireccionar todos los eventos/acciones del usuario hacia el Controlador.

2.1.14.3 CONTROLADOR

Define el comportamiento de la aplicación, interpreta los eventos del usuario y mapea las acciones a realizar sobre el modelo. Las acciones pueden ser realizar un cambio al estado del modelo o ejecutar funciones de la lógica de negocio del modelo. Dependiendo de la acción realizada por el usuario, el Controlador puede seleccionar una nueva vista a mostrar como respuesta a la petición realizada por el usuario.

2.2.15 FRONT-END

Trabaja del lado Cliente, en el navegador, en el lado de lo que se ve. Principalmente se ocupa de los componentes externos del sitio web o de la aplicación web. Como consecuencia, deben dominar obligatoriamente; Html, Css y JavaScript. En general se asocia a los desarrolladores front-end con los principios de diseño y de estructura de páginas. Sin embargo, un desarrollador web va más allá que un diseñador. Obviamente tiene que tener en cuenta la usabilidad y la legibilidad de la página o de la aplicación web, pero como buen programador es consciente de que su trabajo se ejecutará en el lado Cliente, en la mayoría de los casos, en el navegador. Y la información no se almacena en el lado cliente (CampusMvp, 2019).

2.2.15.1 HTML

HTML, que significa Lenguaje de Marcado para Hipertextos (HyperText Markup Language) es el elemento de construcción más básico de una página web y se usa para crear y representar visualmente una página web. Determina el contenido de la página web, pero no su funcionalidad. Otras tecnologías distintas de HTML son usadas generalmente para describir la apariencia/presentación de una página web (CSS) o su funcionalidad (JavaScript). HTML le da "valor añadido" a un texto estándar en español. Hiper Texto se refiere a enlaces que conectan una página Web con otra, ya sea dentro de una página web o entre diferentes sitios web. los vínculos son un aspecto fundamental de la Web. Al subir contenido a Internet y vincularlo a páginas de otras personas, te haces participante activo de esta Red Mundial.

HTML usa "markup" o marcado para anotar textos, imágenes, y otros contenidos que se muestran en el Navegador Web. El lenguaje de marcado HTML incluye "elementos" especiales tales como <head>, <title>, <body>, <header>, <article>, <section>, <p>, <div>, , , y muchos otros más (MDN web docs, 2019).

2.2.15.2 CSS

Hojas de Estilo en Cascada (del inglés *Cascading Style Sheets*) o CSS es el lenguaje utilizado para describir la presentación de documentos HTML o XML, esto incluye varios lenguajes basados en XML como son XHTML o SVG. CSS describe como debe ser renderizado el elemento estructurado en pantalla, en papel, hablado o en otros medios. CSS es uno de los lenguajes base de la *Open Web* y posee una especificación estandarizada por parte del W3C. Desarrollado en niveles, CSS1 es ahora obsoleto, CSS2.1 es una recomendación y CSS3, ahora dividido en módulos más pequeños, está progresando en camino al estándar (MDN web docs, 2019).

2.2.15.3 JAVASCRIPT

Es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C. Al igual que Java, JavaScript es un lenguaje orientado a objetos propiamente dicho, ya que dispone de Herencia, si bien ésta se realiza siguiendo el paradigma de programación basada en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad. Todos los navegadores modernos interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del DOM. (Morrison, 2008).

2.2.16 BACK-END

El desarrollador back-end trabaja del lado Servidor, detrás del escenario, permitiendo con su trabajo que el usuario disfrute de su experiencia. Sin él, el desarrollo llevado a cabo por su anterior compañero no se sostendría. Para ser programador del lado Servidor, son numerosos los lenguajes y *frameworks* entre los que elegir, todo dependerá de la empresa en la que caigas. A día de hoy, los más comunes son: ASP.NET, PHP, RUBY, PYTHON, NODE.JS Y JAVA (CampusMvp, 2019).

Sin embargo, no es suficiente con dominar un lenguaje y un *framework*. Toda aplicación web debe almacenar datos de alguna manera. Por lo tanto, un desarrollador back-end también debe estar familiarizado con las bases de datos. Entre las más comunes destacan: SQL SERVER, MYSQL, ORACLE, POSTGRESQL y MONGODB (CampusMvp, 2019).

2.2.17 PROCURADURÍA PÚBLICA REGIONAL AYACUCHO.

2.2.17.1 MISIÓN

Fortalecer, unificar y modernizar la defensa jurídica del Estado en el ámbito local, regional, nacional, supranacional e internacional en sede *judicial, militar, arbitral, tribunal constitucional, órganos administrativos e instancias de similar naturaleza*, optimizando la utilización de los recursos disponibles, innovando, dinamizando los procesos y procedimientos sin vulnerar el ordenamiento jurídico y con respeto irrestricto al debido proceso, con ética, probidad y honestidad.

2.2.17.2 VISIÓN

Una Procuraduría Pública Regional: Consolidada, independiente, comprensible y oportuna en la defensa jurídica del Estado, donde se planteen selectivamente acciones que estén debidamente alineadas a fortalecer, unificar y modernizar la defensa jurídica del Estado a nivel regional, extensiva al ámbito nacional, supranacional e internacional, con énfasis en las acciones de garantías constitucionales, lucha contra la criminalidad organizada y la corrupción en la administración pública regional; consolidando la normatividad, coadyuvando en forma eficiente y eficaz al sistema de defensa judicial del Estado.

2.2.17.3 ESTRUCTURA ORGANICA FUNCIONAL GOBIERNO REGIONAL DE AYACUCHO

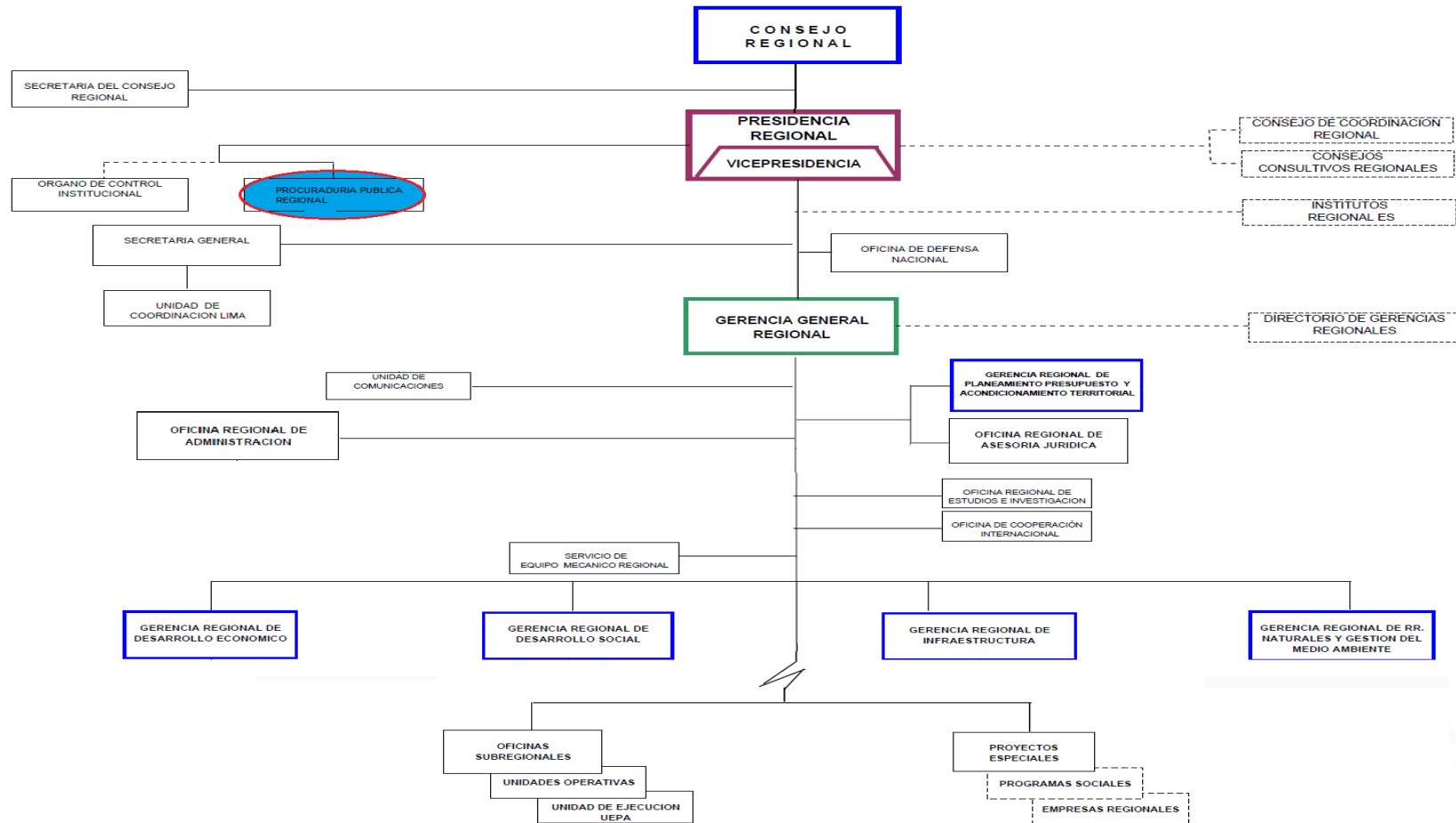


Figura n° 2.16: Fuente: (Estructura Orgánica Funcional Gobierno Regional de Ayacucho).

2.2.17.4 FUNCIONES GENERALES.

- ✓ Ejercitar en los juicios, todas las acciones y recursos legales que sean necesarios en defensa de los derechos e interés del Gobierno Regional Ayacucho.
- ✓ Solicitar informes, antecedentes y el apoyo necesario de cualquier entidad pública para el ejercicio de su función.
- ✓ Informar permanente al Consejo Regional así como al Consejo de Defensa Judicial del Estado, sobre los procesos judiciales iniciados y tramitados en defensa de los derechos e intereses del Gobierno Regional.
- ✓ Formular anualmente su Memoria de Gestión, a más tardar hasta el mes de febrero del año siguiente al informado, conforme a las disposiciones que el Consejo de Defensa Judicial del Estado establezca.
- ✓ Las demás señaladas en el Decreto Legislativo N° 1068 y su reglamento Por Decreto Supremo Nro. 017-98-jus, sus modificatorias y ampliatorias sobre Representación y Defensa del Estado en juicio.

2.2.17.5 PROCURADOR PÚBLICO REGIONAL

FUNCIONES

- ✓ EL Procurador Público Regional es independiente en el ejercicio de sus funciones, las que desempeñara según su criterio y en la forma que estime más arreglada a la ley.
- ✓ Dirigir, programar. Ejecutar, evaluar y controlar las actividades del Procurador Público Regional.
- ✓ Ejercitar la Defensa de los Derechos e intereses del Gobierno Regional de Ayacucho, a nivel judicial en todas las instancias, en los procesos en el que él sea el demandante, demandado, denunciante o parte ante cualquier Tribunal o Juzgado de los diferentes Distritos Judiciales de la República.
- ✓ Representar y defender jurídicamente al Estado en temas concernientes al Gobierno Regional y dependencias administrativas o en aquellos procesos

que por su especialidad asuman los que de manera específica le asignen por delegación.

- ✓ Formular demandas y/o denuncias en representación del Estado a nivel del Gobierno Regional de Ayacucho.
- ✓ Coordinar con el titular de la entidad el cumplimiento y ejecución de las sentencias contrarias a los intereses del estado.
- ✓ Desistirse, allanarse, conciliar o transigir, en los procesos en el que interviene, previa autorización con Resolución Ejecutiva Regional.
- ✓ Prestar declaración de parte o preventiva en juicio.
- ✓ Delegar sus facultades a los abogados de la Procuraduría Pública Regional y otros órganos y Organismos del Gobierno Regional, para intervenir en representación del Estado en los procesos judiciales en defensa de los derechos e intereses del estado a nivel judicial.
- ✓ Solicitar a toda dependencia pública y privada la remisión de información, en el ejercicio de sus funciones.
- ✓ Participar en las investigaciones preliminares o complementarias llevadas a cabo por el Ministerio Público o la Policía bajo la conducción de aquél, para lo que deberá ser debidamente notificado; el procurador Público puede ofrecer pruebas y solicitar la realización de actos de investigación, así como intervenir en las de testigos y en los demás diligencias de investigación, todo ello sin menoscabo de las funciones y acciones que correspondan al Ministerio Público como titular de la acción penal.
- ✓ Interponer recurso de queja de derecho contra resolución del Fiscal que deniega la formalización de denuncia penal e intervenir en el procedimiento recursal ante el Fiscal Superior, debiendo notificarse a de toda decisión como resultado de este procedimiento en su domicilio fiscal.
- ✓ Requerir al órgano jurisdiccional que las resoluciones y actuaciones judiciales que no le fueran puestas en su conocimiento oportunamente sean notificadas en el domicilio procesal.
- ✓ Tener acceso a los documentos administrativos y legales de todas las dependencias, órganos y organismos estructurados del Gobierno Regional de

Ayacucho para Defensa de los intereses del estado a Nivel del Gobierno regional.

- ✓ Atención, seguimiento, supervisión, control y evaluación de los procesos judiciales.
- ✓ Aprobar normas y procedimientos para la defensa del Estado a Nivel del Gobierno Regional.
- ✓ Requerir el auxilio de la fuerza pública a través de la autoridad pública para el mejor desempeño de su función.
- ✓ Participar en las sesiones ordinarias y extraordinarias convocadas por el Consejo de Defensa Judicial del Estado con derecho a participación con voz y voto.
- ✓ Informar al Consejo Regional y al Consejo de Defensa Judicial del Estado sobre el estado situacional de los procesos judiciales.
- ✓ Elaborar La Memoria Gestión Anual.
- ✓ Divulgación, capacitación y actualización de los fines u objetivos de la Procuraduría.
- ✓ Participar en eventos nacionales e internacionales, asumiendo por delegación la representación del Estado.
- ✓ Las demás establezca la Ley.

2.2.17.6 ABOGADO IV

FUNCIONES

- ✓ Colaborar con el Procurador Público Regional en asuntos legales y judiciales de toda índole, así como, emitir pronunciamiento técnico legal requerido en asuntos de interés para la defensa judicial del Estado, relacionado con su área.
- ✓ Asumir defensa judicial en los derechos e intereses del estado a nivel Regional, en el que el Procurador Público regional de Ayacucho le hubiere delegado representación.
- ✓ Ejercitar las acciones y recursos legales necesarios ante los Órganos Jurisdiccionales para una eficaz defensa judicial del Estado, previo estudio y análisis del Expediente Judicial, observando los plazos procesales

establecidos por Ley, asimismo, dar impulso necesario a los procesos judiciales en los que interviene la Procuraduría Pública regional.

- ✓ Dar cuenta de las actividades jurídicas realizadas en los procesos en los que hubiere intervenido.
- ✓ Coordinar con el sector público privado, a efectos de que coadyuven en proporcionar, antecedentes y otros; que requiera la Procuraduría Pública Regional para ejercitar la Defensa Judicial del Estado.
- ✓ Participar en eventos nacionales e internacionales, asumiendo por delegación la representación del Estado.

2.2.17.7 ABOGADO III

FUNCIONES

- ✓ Proporcionar asesoramiento jurídico legal a la Procuraduría Pública Regional en asuntos que le sean consultados, en materia de competencia.
- ✓ Analiza, proyectos de normas, dispositivos y procedimientos jurídicos.
- ✓ Evaluar la documentación técnico-administrativa y los expedientes y emitir informes correspondientes.
- ✓ Brindar asesoramiento especializado jurídico legal a la Procuraduría Pública Regional en asuntos relacionado a la defensa del Estado y/o Regional.
- ✓ Solicitar y coordinar con el sector público y privado, a efectos de proporcionar antecedentes que requiera la Procuraduría Pública Regional.
- ✓ Apoyar a la Procuraduría Pública Regional en asuntos legales y judiciales del Estado.
- ✓ Intervenir en las diligencias judiciales para respaldar los intereses del Estado a Nivel Gobierno Regional.
- ✓ Participar en la formulación de políticas de carácter jurídico.
- ✓ Conformar equipos de trabajo especializado para entender asuntos de Defensa judicial.
- ✓ Supervisar las actividades de procesamiento de expedientes de carácter técnico legal.

- ✓ Las demás atribuciones y responsabilidades que se deriven del cumplimiento de sus funciones y otras que le sean asignadas en materia de su competencia.

CAPITULO III

METODOLOGÍA DE LA INVESTIGACIÓN

3.1 TIPO DE INVESTIGACIÓN

Se desarrolla una sistema web “Sistema de Información Web para optimizar la gestión de archivos en la procuraduría de Ayacucho, 2018”, usando la metodología ICONIX.

De acuerdo al autor Carrasco (2006) define que, la "investigación aplicada se distingue por tener propósitos prácticos inmediatos definidos es decir, se investiga para actuar, transformar, modificar o producir cambios en un determinado sector de la realidad. Para realizar investigaciones aplicadas es muy importante contar con el aporte de las teorías científicas, que son producidas por la investigación básica y sustantiva" (p.44). Por esta consideración el tipo de investigación es **aplicada**.

3.2 NIVEL DE INVESTIGACIÓN

De acuerdo al autor Hemández Sampieri Et. Al (2010), las investigaciones descriptivas vienen a ser "los estudios descriptivos buscan especificar las propiedades, las características y los perfiles importantes de personas, grupos, comunidades o cualquier otro fenómeno que se someta a análisis."

De acuerdo al autor (2006), una de las funciones principales de la investigación descriptiva es la capacidad para seleccionar las características fundamentales del objeto de estudio y su descripción detallada de las partes, categorías o clases de dicho objeto”; y agrega “La investigación descriptiva es uno de los tipos o procedimientos investigativos más populares y utilizados por los principiantes en la actividad investigativa. Los trabajos de grado, en los pregrados y en muchas maestrías, son estudios de carácter eminentemente descriptivo. En tales estudios se muestran, narran, reseñan o identifican hechos, situaciones, rasgos características de un objeto de estudio, o se diseñan productos, modelos, prototipos, guías, etcétera (p.112).

De acuerdo al autor Carrasco (2006), señala que, la investigación descriptiva se soporta principalmente en técnicas como la encuesta, la entrevista, la observación y revisión documental. Este tipo de investigación estudia, analiza, describe y especifica situaciones y propiedades de personas, grupos, comunidades o cualquier otro fenómeno u objeto que sea sometido al análisis.

Por esta consideración el nivel de investigación es **descriptivo**.

3.3 DISEÑO DE INVESTIGACIÓN

De acuerdo con el autor Hernández Sampieri Et. Al (2010), se puede definir la investigación no experimental, “como aquella investigación que se realiza sin manipular deliberadamente las variables, se trata de estudios donde no hacemos variar en forma intencional las variables independientes para ver su efecto sobre otras variables. Lo que hacemos en la investigación no experimental es observar fenómenos tal como se dan en su contexto natural, para posteriormente analizarlos” (p.191).

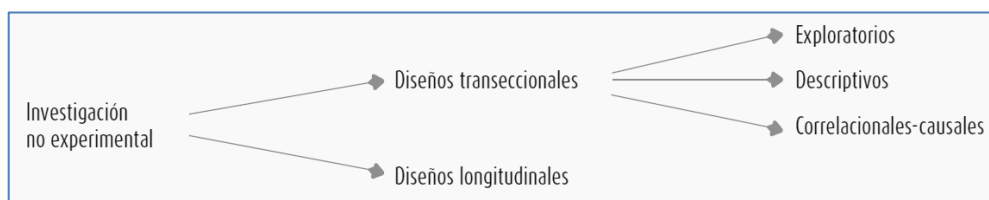


Figura n° 3.17: Diseño de investigación (Hernández, 2010).

Según Carrasco (2008), el diseño de investigación transversal descriptivo, se emplea para analizar y conocer las características, cualidades internas y externas, propiedades y rasgos esenciales de los hechos y fenómenos de un hecho realizado a momento determinado del tiempo.

De acuerdo a Hernández et al 2010, una investigación de diseño transversal es “cuando se recolectan datos en un solo momento o en un tiempo único y su propósito es describir variables y analizar los hechos tal como se dan”. Los instrumentos de recolección de datos, son usados durante el proceso de forma única.

En esta investigación se tenía que analizar procesos, conocer características, estudiar rasgos y entender funcionalidades para encontrar datos necesarios para la construcción del aplicativo; por lo tanto el diseño de investigación es **no experimental de tipo transversal descriptivo**.

3.4 POBLACIÓN Y MUESTRA

3.4.1 POBLACIÓN

La población está compuesta por todas las gestiones internas y externas de los trámites y archivos en la Procuraduría de Ayacucho, 2018.

3.4.2 MUESTRA.

Se tomó una muestra no probabilística por juicio de experto y muestra de casos tipo, conformada por cada tipo de gestión interna y externa de los trámites y archivos en la Procuraduría de Ayacucho, 2018.

3.5 VARIABLES E INDICADORES

3.5.1 DEFINICIÓN CONCEPTUAL DE LAS VARIABLES

3.5.1.1 PRIMERA VARIABLE

X: Sistema de información web.

Medio automatizado para realizar reportes, transacciones, operaciones y procesos que sean los más funcionales para la organización; se debe tener en cuenta un enlace a internet o una intranet, el hardware y los periféricos y también el software de sustento a lado del usuario y servidor.

3.5.1.2 INDICADORES DE LA PRIMERA VARIABLE

X1: Usabilidad.

Capacidad del producto software para ser entendido, aprendido, usado y resultar atractivo para el usuario, cuando se usa bajo determinadas condiciones.

X2: Funcionalidad.

Representa la capacidad del producto software para proporcionar funciones que satisfacen las necesidades declaradas e implícitas, cuando el producto se usa en las condiciones especificadas.

X3: Fiabilidad.

Capacidad de un sistema o componente para desempeñar las funciones especificadas, cuando se usa bajo unas condiciones y periodo de tiempo determinados.

3.5.1.3 SEGUNDA VARIABLE**Y: Gestión de archivos**

Son archivos de las oficinas que conservan la documentación todavía en trámite y aquella otra que, aun después de finalizado el procedimiento administrativo, está sometida a continua utilización y consulta administrativa por las mismas oficinas, por lo que es conveniente conservar en su ámbito.

3.5.1.4 INDICADORES DE LA SEGUNDA VARIABLE**Y1: Registro de documentos.**

Capacidad de registrar detalles y características de un documento en un momento dado.

Y2: Clasificación de documentos.

Es la característica organizar y distribuir los documentos digitalizados para una posterior búsqueda y seguimiento.

Y3: Descripción y recuperación de documentos.

Es la cualidad de obtener un resumen total de todos los documentos registrados y ver el estado que lleva cada documento en su trámite.

3.5.2 DEFINICIÓN OPERACIONAL DE LAS VAIABLES

3.5.2.1 VARIABLE INDEPENDIENTE.

Variable	Indicadores
X: Sistema de información web.	X1: Usabilidad.
	X2: Funcionalidad.
	X3: Fiabilidad.

Cuadro n° 3.2: Variable Independiente

3.5.2.2 VARIABLE DEPENDIENTE.

Variable	Indicadores
Y: Gestión de archivos	Y1: Registro de documentos.
	Y2: Clasificación de documentos.
	Y3: Descripción y recuperación de documentos.

Cuadro n° 3.3: Variable Dependiente

3.6 TÉCNICAS E INSTRUMENTOS

3.6.1 TÉCNICAS PARA RECOLECTAR INFORMACIÓN

Se usaron la técnica de **Entrevista** que se realizó: Al Procurador Público Regional, a los Abogados IV y III y al personal involucrado en la Procuraduría; por otro lado **El Análisis Documental** se realizará al MOF y ROF del Gobierno Regional de Ayacucho y también al Manual del sistema peruano de justicia.

3.6.2 INSTRUMENTOS PARA RECOLECTAR INFORMACIÓN

Se Levantó información usando **Guía de entrevista** y a trabajadores de la Procuraduría de Ayacucho y **Análisis Documental**, la cual permite obtener información para dar respuesta a las preguntas, el logro de los objetivos y las demostración de la hipótesis, instrumento que se muestra en el anexo A.

3.6.3 HERRAMIENTAS PARA EL TRATAMIENTO DE DATOS E INFORMACIÓN

Las herramientas que se usa para transformar todo lo recolectado y convertirlo en un sistemas web es el siguiente; se detalla según el cuadro n°.3.4.

SOFTWARE	FABRICANTE	SERVICIO
WINDOWS 10	Creada por Microsoft Corporation.	Es la última versión más reciente de Microsoft Windows.
BOOTSTRAP	Desarrollado por Twitter.	Bootstrap es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web.
PHP	Rasmus Lerdorf, arquitecto de infraestructuras y creador del lenguaje PHP.	PHP es un lenguaje de código abierto muy popular, adecuado para desarrollo web y que puede ser incrustado en HTML
MYSQL	MySQL fue inicialmente desarrollado por MySQL AB (empresa fundada por David Axmark, Allan Larsson y Michael Widenius). MySQL AB fue adquirida por Sun Microsystems en 2008, y ésta a su vez fue comprada por Oracle Corporation en 2010	MySQL es muy utilizado en aplicaciones web, como Joomla, Wordpress, Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python).
JAVASCRIPT	Desarrollada por Netscape Communications Corp, Mozilla Foundation.	Está diseñado para interactuar con el cliente y dar más dinamismos al interfaz de usuario.
NETBEANS 8.2	Desarrollada por Sun Microsystems.	La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos.
HTML5	Desarrollado por World Wide Web Consortium & WHATWG	Es un conjunto de símbolos o palabras que definen varios componentes de un documento Web.
CSS3	Desarrollado por W3C	Es un lenguaje artificial usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML).
JQUERY	Creado por The jQuery Project	Es un framework de desarrollo UI, que nos permite escribir una aplicación web para móviles, completamente funcional, sin necesidad de escribir una sola línea de código JavaScript.
JSON	Creado por Javascript bajo el estándar RFC	JSON, acrónimo de JavaScript Object Notation, es un formato ligero para el Intercambio de datos. JSON es un Subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.

Cuadro n° 3.4: Herramientas tecnológicas para el tratamiento de datos.

3.6.4 TÉCNICAS PARA APLICAR ICONIX

De acuerdo al marco teórico del capítulo interior, las fases para el desarrollo de la aplicación web con la metodología ICONIX se presentan en los cuadros siguientes.

3.6.4.1 ANÁLISIS DE REQUISITOS

TAREA	ARTEFACTO	TÉCNICA	RESPONSABLES
Identificar requisitos	<ul style="list-style-type: none"> Requisito funcionales y no funcionales Casos de prueba 	<ul style="list-style-type: none"> Entrevistas Definir lo que el sistema debe hacer Escribir al menos un caso de prueba para cada requisito 	<ul style="list-style-type: none"> Usuario Cliente Analista
Identificar objetos del mundo real y dibujar modelo de dominio	<ul style="list-style-type: none"> Modelo de dominio 	<ul style="list-style-type: none"> Identificar clases clave del negocio Identificar sustantivos y depurar Identificar objetos en requisitos funcionales y asignar al modelo de dominio Utilizar agregación y generalización 	<ul style="list-style-type: none"> Analista
Realizar prototipo de interfaz gráfica	<ul style="list-style-type: none"> Prototipo GUI 	<ul style="list-style-type: none"> Utilizar historia de eventos del usuario Utilizar los requisitos funcionales Diseñar interfaz gráfica básica 	<ul style="list-style-type: none"> Programador analista
Descubrir casos de uso	<ul style="list-style-type: none"> Lista de casos de uso 	<ul style="list-style-type: none"> Utilizar requisitos funcionales Entrevistas 	<ul style="list-style-type: none"> Usuario Cliente Analista
Dibujar y empaquetar casos de uso	<ul style="list-style-type: none"> Diagrama de casos de uso Paquete entre requisitos funcionales y casos de uso 	<ul style="list-style-type: none"> Identificar roles y responsabilidades de actores. Asociar actores con casos de uso. Relacionar casos de uso Agrupar lógicamente casos de uso. 	<ul style="list-style-type: none"> Analista
Asignar requisitos funcionales a los casos de uso	<ul style="list-style-type: none"> Relación entre requisitos funcionales y casos de uso 	<ul style="list-style-type: none"> Asignar requisitos funcionales a los casos de uso. 	<ul style="list-style-type: none"> Analista
Escribir el primer borrador de casos de uso	<ul style="list-style-type: none"> Primer borrador de casos de uso 	<ul style="list-style-type: none"> Utilizar glosario de objetos del modelo de dominio. Utilizar la regla de dos párrafos Escribir el caso de uso como flujos de evento/respuesta. Escribir el caso de uso con estructura sustantivo-verbo-sustantivo. Escribir caso de uso en voz activa. Referenciar por su nombre a las pantallas. 	<ul style="list-style-type: none"> Analista

Cuadro n° 3.5: Análisis de requisitos (Porras, 2011)

3.6.4.2 REVISIÓN DE REQUISITOS

TAREA	ARTEFACTO	TÉCNICA	RESPONSABLES
Revisar el modelo de dominio	<ul style="list-style-type: none"> Modelo de dominio 	<ul style="list-style-type: none"> Identificar al menos el 80% de clases clave de dominio del problema 	<ul style="list-style-type: none"> Usuario Cliente Analista Programador
Revisar el prototipo GUI	<ul style="list-style-type: none"> Prototipo GUI 	<ul style="list-style-type: none"> Diseñar con precisión la GUI relacionada al caso de uso 	<ul style="list-style-type: none"> Usuario Cliente Analista Programador
Revisar el modelo de casos de uso	<ul style="list-style-type: none"> Casos de uso revisado 	<ul style="list-style-type: none"> Eliminar clases fuera del dominio del problema Cambiar descripción de voz pasiva a activa Describir todos los cursos alternos Asociar todos los requisitos a los casos de uso Describir que intenta hacer el usuario para cada caso de uso. 	<ul style="list-style-type: none"> Usuario Cliente Analista Programador

Cuadro n° 3.6: Revisión de requisitos (Porras, 2011)

3.6.4.3 DISEÑO PRELIMINAR

TAREA	ARTEFACTO	TÉCNICA	RESPONSABLES
Reescribir el primer borrador para cada caso de uso	<ul style="list-style-type: none"> Casos de uso desambiguado 	<ul style="list-style-type: none"> Reescribir el caso de uso durante el análisis de robustez 	<ul style="list-style-type: none"> Analista
Identificar el primer corte de objetos que completan escenarios para cada caso de uso	<ul style="list-style-type: none"> Diagrama de robustez 	<ul style="list-style-type: none"> Copiar la descripción del caso de uso en el diagrama de robustez Usar clases del modelo de dominio Crear un objeto interfaz por cada GUI y nombrarlo Transformar verbos del caso de uso en controladores Relacionar un caso de uso al diagrama de robustez cuando es invocado Utilizar las reglas para construir el diagrama de robustez 	<ul style="list-style-type: none"> Analista
Actualizar el modelo de dominio	<ul style="list-style-type: none"> Modelo de dominio actualizado 	<ul style="list-style-type: none"> Actualizar el modelo de dominio con nuevas clases atributos durante el análisis de robustez 	<ul style="list-style-type: none"> Analista
Actualizar el diagrama de clases de análisis	<ul style="list-style-type: none"> Modelo de dominio actualizado 	<ul style="list-style-type: none"> Actualizar el diagrama de clases de análisis al finalizar el análisis de robustez Asignar atributos a las clases entidad 	<ul style="list-style-type: none"> Analista

Cuadro n° 3.7: Diseño preliminar (Porras, 2011).

3.6.4.4 REVISIÓN DE DISEÑO PRELIMINAR

TAREA	ARTEFACTO	TÉCNICA	RESPONSABLES
Revisar descripción de casos de uso	<ul style="list-style-type: none"> Caso de uso 	<ul style="list-style-type: none"> Coincidir la descripción del caso de uso con el diagrama de robustez. 	<ul style="list-style-type: none"> Usuario Cliente Analista Programador
Revisar diagrama de robustez	<ul style="list-style-type: none"> Diagrama de robustez 	<ul style="list-style-type: none"> Coincidir el diagrama de robustez con descripción de caso de uso. Verificar que el diagrama de robustez cumple con las reglas. Verificar que el diagrama de robustez tener todos los cursos alternos. 	<ul style="list-style-type: none"> Usuario Cliente Analista Programador
Revisar modelo de dominio actualizado	<ul style="list-style-type: none"> Modelo de dominio actualizado 	<ul style="list-style-type: none"> Coincidir los objetos entidad del diagrama de robustez con el modelo de dominio actualizado. 	<ul style="list-style-type: none"> Usuario Cliente Analista Programador

Cuadro n° 3.8 Revisión de diseño preliminar (Porras, 2011).

3.6.4.5 DISEÑO

TAREA	ARTEFACTO	TÉCNICA	RESPONSABLE
Dividir modelo de dominio actualizado para cada caso de uso	<ul style="list-style-type: none"> Parte de modelo de dominio actualizado 	<ul style="list-style-type: none"> Coincidir las clases entidad del diagrama de robustez con parte del modelo de dominio actualizado y dibujado 	<ul style="list-style-type: none"> Diseñador
Dibujar un diagrama de secuencia para cada caso de uso	<ul style="list-style-type: none"> Diagrama de secuencia 	<ul style="list-style-type: none"> Copiar la descripción del caso de uso Copiar objetos entidad, interfaz y actores del diagrama de robustez Verificar que un mensaje del diagrama de secuencia es verbo en el caso de uso Hacer refactoring al diagrama de secuencia antes de codificar. 	<ul style="list-style-type: none"> Programador Diseñador
Actualizar el diagrama de un caso de uso	<ul style="list-style-type: none"> Diagrama de clase 	<ul style="list-style-type: none"> Asignar operaciones a las clases a partir de los mensajes del diagrama de secuencia Establecer multiplicidad en las clases Depurar la clases, operaciones y atributos del diagrama de clases 	<ul style="list-style-type: none"> Programador Diseñador
Extraer controladores para pruebas unitarias	<ul style="list-style-type: none"> Lista de controladores. 	<ul style="list-style-type: none"> Identificar controladores para la lógica del negocio desde un diagrama de robustez. 	<ul style="list-style-type: none"> Programador Diseñador

Cuadro n° 3.9. Diseño (Porras, 2011)

3.6.4.6 REVISIÓN CRÍTICA DE DISEÑO

TAREA	ARTEFACTO	TÉCNICA	RESPONSABLE
Revisar diagrama de secuencia	<ul style="list-style-type: none"> Diagrama de secuencia. 	<ul style="list-style-type: none"> Verificar que el diagrama de secuencia coincide con la descripción del caso de uso Verificar que el diagrama de secuencia representa los cursos básicos y alterno Verificar en los mensajes que los atributos y valores de retorno son correctos 	
Revisar diagrama de clases	<ul style="list-style-type: none"> Diagrama de clases 	<ul style="list-style-type: none"> Verificar que el nombre, atributos y operaciones que asignaron correctamente a las clases. Asignar requisitos no funcionales a los casos de uso y clases 	<ul style="list-style-type: none"> Diseñador BD Programador
Revisar modelo de dominio actualizado	<ul style="list-style-type: none"> Modelo de dominio actualizado 	<ul style="list-style-type: none"> Verificar nombres y atributos del modelo dominio actualizado 	
Revisar lista de pruebas unitarias	<ul style="list-style-type: none"> Lista de controladores 	<ul style="list-style-type: none"> Actualizar la lista de controladores 	

Cuadro n° 3.10. Revisión crítica de diseño (Porras, 2011).

CAPITULO IV

RESULTADOS DE LA INVESTIGACIÓN

4.1 RESULTADOS Y ARTEFACTOS DEL SOFTWARE APLICANDO EL PROCESO DE ICONIX

Realizando un estudio a la metodología ICONIX, obtenemos el procedimiento y los pasos a seguir para la ejecución de esta tarea, los cuales están resumidos en el capítulo III; los artefactos se utilizarán para el modelado e implementación del Sistema de Información Web para optimizar la gestión de archivos en la procuraduría de Ayacucho, 2018.

4.1.1 ETAPA1: ANÁLISIS DE REQUISITOS.

La solución de este problema está enfocada a mejorar la gestión de documentos exclusivos de la Procuraduría de Ayacucho; por lo que toda la información que se recopila solo de esta área; obteniendo los siguientes requisitos:

4.1.1.1 REQUISITOS FUNCIONALES

N°	REQUISITOS FUNCIONALES
ReqF01.	El software debe tener la capacidad de personalizar el acceso a los módulos y tareas, mediante claves encriptados, según las funciones de los usuarios responsables del sistema.
ReqF02.	El software ser capaz de emitir consultas y reportes de los datos de usuarios del sistema, considerando los siguientes datos: Número de identificación, apellidos, nombres, cuenta de usuario y tipo de usuario.
ReqF03.	El software debe permitir mantener los datos de persona: tipo de documento de identificación, número de identificación, apellidos, nombres, tipo de persona (natural o jurídica), en caso de persona jurídica su razón social.
ReqF04.	El software debe ser capaz de emitir consultas y reportes de los datos de las personas registradas en el sistema.
ReqF05.	El software debe permitir mantener los datos de un profesional jurídico (Juez, fiscal, etc.): número de documento de identificación, tipo de colegio profesional, número de registro de colegio profesional, apellidos y nombres.
ReqF06.	El software debe ser capaz de emitir consultas y reportes de datos de los profesionales jurídicos registrados en el sistema.
ReqF07.	El software debe permitir mantener los datos de las entidades públicas: Gobiernos regionales, dirección regional de educación, Unidad de Gestión Educativa Local, etc. Considerando los siguientes datos: Denominación, ubicación geográfica (Ubigeo), dirección, teléfono y representante.
ReqF08.	El software debe permitir mantener los datos de los órganos

	jurisdiccionales del poder judicial y del ministerio público. Considerando los siguientes datos: ubicación geográfica (Ubigeo), denominación, teléfono y dirección.
ReqF09.	El software debe ser capaz de emitir consultas y reportes de los datos de los <u>órganos jurisdiccionales del poder judicial y del ministerio público.</u>
ReqF10.	El software debe ser capaz de emitir consultas y reportes de los datos de las entidades públicas registradas.
ReqF11.	El software debe permitir mantener los datos de los expedientes provenientes de los <u>órganos jurisdiccionales del Poder Judicial y/o del Ministerio Público (Fiscalía)</u> , considerando los siguientes datos: número de expediente, número de folder, pago monetario en cuestión, demandantes, entidad demandado, representante de la entidad demandado, tipo de trámite jurídico (Poder Judicial o Ministerio Público), cargos y responsabilidades de los profesionales del <u>órganos jurisdiccionales del poder judicial y/o del ministerio público.</u>
ReqF12.	El software debe permitir mantener documentos adjuntos del expediente registrado, considerando los siguientes datos: resumen de documento adjunto, descripción del documento adjunto.
ReqF13.	El software debe permitir mantener los procesos de un expediente. Desde el momento de ingreso hasta su conclusión, actualizando permanentemente los siguientes datos: número de expediente, número de proceso cautelar, descripción de proceso cautelar, programación de diligencias, profesionales jurídicos (juez, fiscal, etc.), documentos adjuntos y estado del expediente.
ReqF14.	El software debe permitir distribuir carga laboral de los expedientes ingresantes a los trabajadores (Procuraduría).
ReqF15.	El software debe ser capaz de emitir consultas y reportes de los trabajos realizados por los <u>profesionales de la procuraduría.</u>
ReqF16.	El software debe ser capaz de emitir consultas y reportes de los expedientes y procesos.
ReqF17.	El software debe ser capaz de emitir la estadística para una materia y para todas las materias.

Cuadro n° 4.11. Requisitos Funcionales.

4.1.1.2 REQUISITOS NO FUNCIONALES

N°	REQUISITOS NO FUNCIONALES
ReqNF01.	El software debe garantizar la autenticidad, la cual asegura que los usuarios son quienes dicen que son y que se puede verificar su autenticidad
ReqNF02.	El software debe garantizar la confidencialidad, implementando un mecanismo para asegurar la confidencialidad de datos y transacciones, mediante el acceso restringido a la base de datos y las operaciones
ReqNF03.	El software debe garantizar la integridad, para asegurar la integridad de datos y de los cálculos
ReqNF04.	Las interfaces deben cumplir con requisitos mínimos de usabilidad

Cuadro n° 4.12. Requisitos No Funcionales.

Observando los requisitos se obtiene una lista tentativa de objetivos:

- Usuario
- Persona
- Entidad pública
- Ubigeo
- Profesional Jurídico
- Órgano jurisdiccional
- Expediente
- Demandante
- Demandado
- Entidad
- Representante
- Trámite
- Documento adjunto
- Proceso de expediente
- Proceso Cautelar
- Diligencia
- Trabajador

4.1.1.3 MODELO DE DOMINIO INICIAL

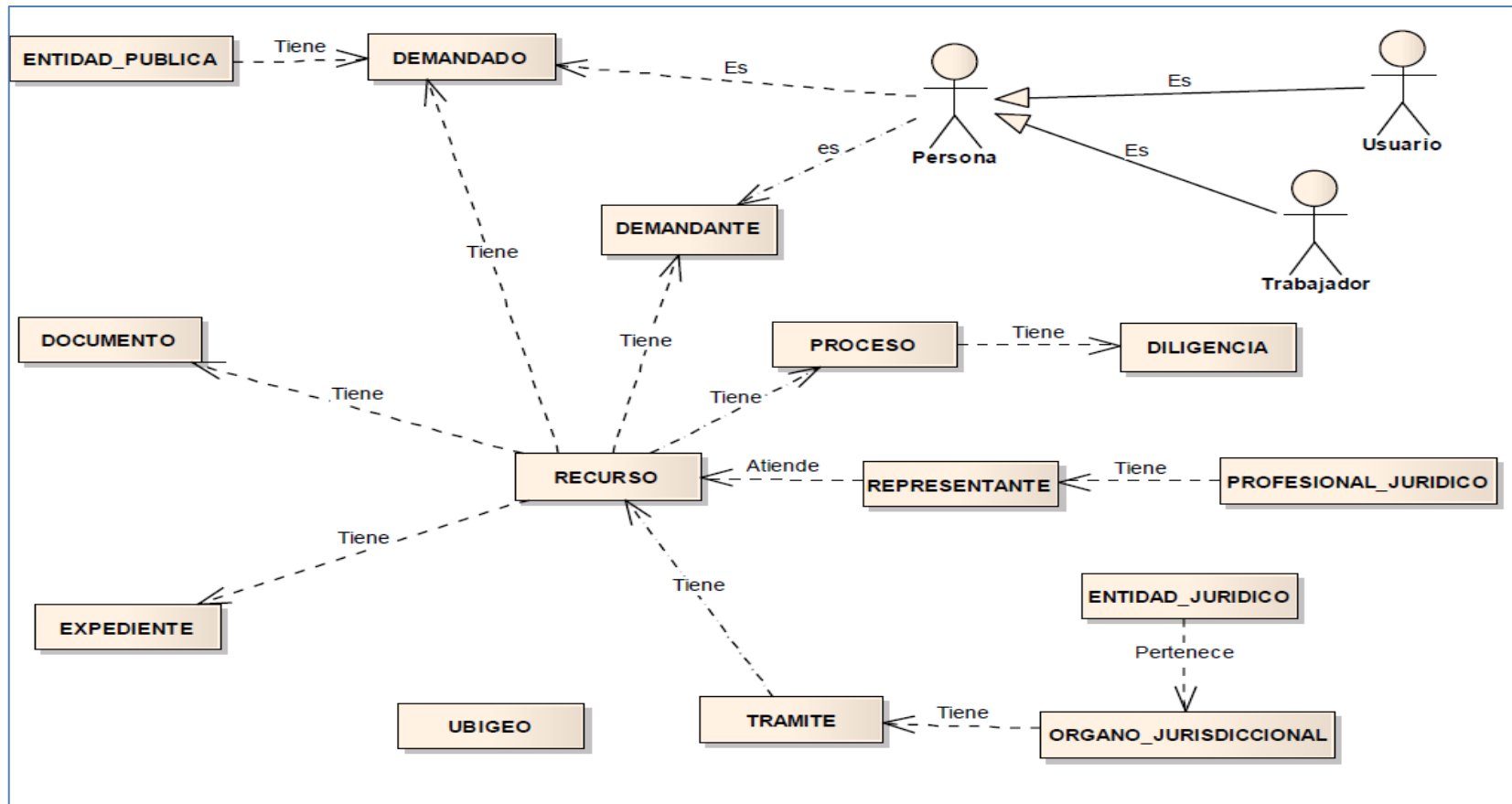


Figura n° 4.18: Primer Intento de Modelo de Dominio.

4.1.1.4 PROTOTIPO DE INTERFAZ DE USUARIO DE LOS CASOS DE USO

4.1.1.4.1 INTERFAZ DE USUARIO: Registrar Usuario

Registrar Usuario

N° Documento

Tipo de Usuario

Ap. Paterno

Ap. Materno

Nombres

Login

Password

Figura n° 4.19: Interfaz De Usuario: Registrar Usuario

4.1.1.4.2 INTERFAZ DE USUARIO: Lista De Usuarios Del Sistema.

Lista de Usuarios del Sistema

Usuario 🔍

N° Documento	Tipo Función	Apellidos y Nombres	Login	Estado	Acción
Texto	Texto	Texto	Texto	Texto	Texto
Texto	Texto	Texto	Texto	Texto	Texto
Texto	Texto	Texto	Texto	Texto	Texto

Figura n° 4.20: Interfaz De Usuario: Lista De Usuarios Del Sistema.

4.1.1.4.3 INTERFAZ DE USUARIO: Agregar Expediente /Caso

Registrar Expediente/Caso

N° Folder: Instancia:

Tipo Trámite: Especialidad:

N° de Expediente: Doc. Adjunto:

Monto Pecuniario: Resumen Doc. Adj.:

Responsables Jurídicos (Poder Judicial/Ministerio Público)

N° Documento	Colegio Prof.	N° Registro	Ap. y Nombres	Cargo	Dependencia	Órgano Jurisdiccional	Acción
Texto	Texto	Texto	Texto	Texto	Texto	Texto	X
Texto	Texto	Texto	Texto	Texto	Texto	Texto	X

Demandante

Tipo Persona	Tipo Documento	N° Documento	Nombres o Razón Social	Dirección Procesal	Acción
Texto	Texto	Texto	Texto	Texto	X
Texto	Texto	Texto	Texto	Texto	X

Demandado

Región	Provincia	Distrito	Entidad Demandado	Representante	Vigencia Cargo	Dirección Procesal	Acción
Texto	Texto	Texto	Texto	Texto	Texto	Texto	X
Texto	Texto	Texto	Texto	Texto	Texto	Texto	X

Figura n° 4.22: Interfaz De Usuario: Agregar Expediente /Caso

4.1.1.4.4 INTERFAZ DE USUARIO: Lista Expediente/ Caso

Lista de Expedientes / Casos

Tipo Trámite: N° Folder: N° Expediente / Caso: Demandante: Especialidad:

N° Exp./Caso	N° Folder	Tipo	Distrito	Órgano Jurisdic.	Juez/Fiscal	Instancia	Materia	Doc. Adjunto	Demandante	Demandado	Acción
Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	
Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	
Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	Texto	

Figura n° 4.23: Interfaz De Usuario: Lista Expediente/ Caso

4.1.1.4.5 INTERFAZ DE USUARIO: Lista de Procesos de un expediente

Lista de Procesos de un Expediente

N° Recurso: Especialidad:

N° Folder: Demandante:

Fecha de Ingreso: Demandado:

Estado: Juez/Fiscal:

Especialista Legal:

N° Proceso	Descripción	F. de Registro	N° Notificación	Doc. Adjunto	Tipo Trámite	Acción
3	Texto	Texto	Texto		Poder Judicial	
2	Texto	Texto	Texto		Poder Judicial	
1	Texto	Texto	Texto		Fiscalía	

Figura n° 4.24: Interfaz De Usuario: Lista de Procesos de un expediente

4.1.1.5 IDENTIFICACIÓN DE CASOS DE USO

CASOS DE USO	REQUISITOS FUNCIONALES
1. Editar datos de usuario	1. El software debe tener la capacidad de personalizar el acceso a los módulos y tareas, mediante claves encriptados, según las funciones de los usuarios responsables del sistema.
2. Listar Usuario.	2. El software ser capaz de emitir consultas y reportes de los datos de usuarios del sistema, considerando los siguientes datos: Número de identificación, apellidos, nombres, cuenta de usuario y tipo de usuario.
3. Registra Persona	3. El software debe permitir mantener los datos de persona: tipo de documento de identificación, número de identificación, apellidos, nombres, tipo de persona (natural o jurídica), en caso de persona jurídica su razón social.
4. Listar Persona	4. El software debe ser capaz de emitir consultas y reportes de los datos de las personas registradas en el sistema.
5. Registrar Administrador de justicia	5. El software debe permitir mantener los datos de un profesional jurídico (Juez, fiscal, etc.): número de documento de identificación, tipo de colegio profesional, número de registro de colegio profesional, apellidos y nombres.
6. Listar Administradores de justicia	6. El software debe ser capaz de emitir consultas y reportes de datos de los profesionales jurídicos registrados en el sistema.
7. Registrar Entidad	7. El software debe permitir mantener los datos de las entidades públicas: Gobiernos regionales, dirección regional de educación, Unidad de Gestión Educativa Local, etc. Considerando los siguientes datos: Denominación, ubicación geográfica (Ubigeo), dirección, teléfono y representante.
8. Registrar Entidad Jurídico	8. El software debe permitir mantener los datos de los órganos jurisdiccionales del poder judicial y del

	ministerio público. Considerando los siguientes datos: ubicación geográfica (Ubigeo), denominación, teléfono y dirección.
9. Listar Órganos Jurisdiccionales	9. El software debe ser capaz de emitir consultas y reportes de los datos de los órganos jurisdiccionales del poder judicial y del ministerio público.
10. Listar Entidades	10. El software debe ser capaz de emitir consultas y reportes de los datos de las entidades públicas registradas.
11. Agregar Expediente	11. El software debe permitir mantener los datos de los expedientes provenientes de los órganos jurisdiccionales del Poder Judicial y/o del Ministerio Público (Fiscalía), considerando los siguientes datos: número de expediente, número de folder, pago monetario en cuestión, demandantes, entidad demandado, representante de la entidad demandado, tipo de trámite jurídico (Poder Judicial o Ministerio Público), cargos y responsabilidades de los profesionales del órganos jurisdiccionales del poder judicial y/o del ministerio público.
12. Registrar Incidencia / Medida Cautelar	12. El software debe permitir mantener documentos adjuntos del expediente registrado, considerando los siguientes datos: resumen de documento adjunto, descripción del documento adjunto.
13. Registrar Proceso	13. El software debe permitir mantener los procesos de un expediente. Desde el momento de ingreso hasta su conclusión, actualizando permanentemente los siguientes datos: número de expediente, número de proceso cautelar, descripción de proceso cautelar, programación de diligencias, profesionales jurídicos (juez, fiscal, etc.), documentos adjuntos y estado del expediente.
14. Distribuir Expedientes	14. El software debe permitir distribuir carga laboral de los expedientes ingresantes a los trabajadores (Procuraduría).
15. Listar trabajos de profesionales de la Procuraduría	15. El software debe ser capaz de emitir consultas y reportes de los trabajos realizados por los profesionales de la procuraduría.
16. Listar expedientes	16. El software debe ser capaz de emitir consultas y reportes de los expedientes y procesos.
17. Resumen Estadístico por materia(s).	17. El software debe ser capaz de emitir la estadística para una materia y para todas las materias.

Cuadro n° 4.12. Casos de uso identificados según requisitos.

4.1.1.6 DESCRIPCIÓN DE CASOS DE USO

Se describirá los casos de uso de los primeros prototipos de interfaz de usuario descritos en el capítulo 4.1.1.4.

4.1.1.6.1 CASO DE USO: *Registrar Usuario.*

CASO DE USO: Registrar Usuario.
ACTOR: Usuario Administrador
DESCRIPCIÓN: Este caso de uso permite al Usuario Administrador, registrar nuevos

usuarios administradores del sistema y trabajadores.
ESCENARIO NORMAL
El Usuario Administrador hace clic en el botón agregar, el sistema muestra un formulario donde se registrará la información del nuevo usuario, luego hacer clic en el botón guardar, el sistema muestra un mensaje de confirmación indicando que los datos fueron guardados correctamente.
ESCENARIO ALTERNO
El Usuario Administrador no ingresó todos los datos requeridos: El sistema muestra un mensaje indicando campo que falta ingresar. El Usuario Administrador ingresó datos incorrectos: El sistema muestra un mensaje indicando la característica de los datos a ingresar. El Usuario Administrador hace clic en el botón salir: El sistema regresa a una vista de gestión de usuario.
PRECONDICIONES: El Usuario Administrador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.
POSTCONDICIONES: Se gestionó adecuadamente el registro de usuario.

Cuadro n° 4.13. CASO DE USO: Registrar Usuario.

4.1.1.6.2 CASO DE USO: *Listar Usuarios Del Sistema.*

CASO DE USO: Lista De Usuarios Del Sistema.
ACTOR: Usuario Administrador
DESCRIPCIÓN: En este caso de uso permite al Usuario Administrador, desplegar todos los usuarios habilitados para el uso del sistema.
ESCENARIO NORMAL
El Usuario Administrador hace clic en la opción del menú “Administración->Usuario del Sistema”; el sistema muestra una vista con el listado de usuarios y con opciones de editar, agregar y salir de las opciones.
ESCENARIO ALTERNO
El Usuario Administrador hace clic en la opción Editar de un usuario: El sistema muestra un formulario con los campos del usuario para poder actualizarlos. El Usuario Administrador hace clic en la opción Agregar: El sistema muestra un formulario con los campos vacíos para poder ingresar los datos del nuevo usuario. El Usuario Administrador hace clic en la opción Salir: El sistema regresa al estado por defecto donde las opciones de menú no están seleccionadas.
PRECONDICIONES: El Usuario Administrador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.
POSTCONDICIONES: Se generó correctamente el listado de usuarios.

Cuadro n° 4.14. CASO DE USO: Listar Usuarios Del Sistema.

4.1.1.6.3 CASO DE USO: *Registrar Recurso.*

CASO DE USO: Registrar Recurso.
ACTOR: Trabajador
DESCRIPCIÓN: Este caso de uso permite al Trabajador, registrar nuevos recursos en el sistema.
ESCENARIO NORMAL
El Trabajador hace clic en el botón agregar, el sistema muestra un formulario donde se registrará la información del nuevo recurso, luego hacer clic en el botón guardar, el sistema muestra un mensaje de confirmación indicando que los datos fueron guardados correctamente.

ESCENARIO ALTERNO
<p>El Trabajador no ingresó todos los datos requeridos: El sistema muestra un mensaje indicando campo que falta ingresar.</p> <p>El Trabajador ingresó datos incorrectos: El sistema muestra un mensaje indicando la característica de los datos a ingresar.</p> <p>El Trabajador hace clic en el botón salir: El sistema regresa a una vista de gestión de recurso.</p>
PRECONDICIONES: El Trabajador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.
POSTCONDICIONES: Se gestionó adecuadamente el registro de recurso.

Cuadro n° 4.15. CASO DE USO: Registrar Recurso.

4.1.1.6.4 CASO DE USO: *Agregar Expediente /Caso.*

CASO DE USO: Agregar Expediente /Caso.
ACTOR: Trabajador
DESCRIPCIÓN: Este caso de uso permite al Trabajador, registrar nuevos expedientes /caso en el sistema.
ESCENARIO NORMAL
El Trabajador hace clic en el botón agregar, el sistema muestra un formulario donde se registrará la información del nuevo recurso, luego hacer clic en el botón guardar, el sistema muestra un mensaje de confirmación indicando que los datos fueron guardados correctamente.
ESCENARIO ALTERNO
<p>El Trabajador no ingresó todos los datos requeridos: El sistema muestra un mensaje indicando campo que falta ingresar.</p> <p>El Trabajador ingresó datos incorrectos: El sistema muestra un mensaje indicando la característica de los datos a ingresar.</p> <p>El Trabajador hace clic en el botón salir: El sistema regresa a una vista de gestión de recurso.</p>
PRECONDICIONES: El Trabajador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.
POSTCONDICIONES: Se gestionó adecuadamente el registro de Expediente.

Cuadro n° 4.16. CASO DE USO: Agregar Expediente /Caso.

4.1.1.6.5 CASO DE USO: *Listar Expediente/ Caso.*

CASO DE USO: Listar Expediente/ Caso.
ACTOR: Trabajador
DESCRIPCIÓN: En este caso de uso permite al Trabajador, desplegar una Lista de Expedientes Caso guardados en el sistema.
ESCENARIO NORMAL
El trabajador hace clic en la opción del menú "Expediente"; el sistema muestra una vista con el listado de Expedientes/ Caso, nuevo recurso y salir de las opciones.
ESCENARIO ALTERNO
<p>El Trabajador hace clic en la opción Acción de un expediente: El sistema realiza una acción de activación para cada expediente.</p> <p>El Trabajador hace clic en la opción Nuevo Recurso: El sistema muestra un formulario con los campos vacíos para poder ingresar los datos del nuevo recurso.</p> <p>El Trabajador hace clic en la opción Salir: El sistema regresa al estado por defecto donde las opciones de menú no están seleccionadas.</p>

PRECONDICIONES: El Trabajador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.
POSTCONDICIONES: Se generó correctamente el listado de Expediente /Caso.

Cuadro n° 4.17. CASO DE USO: Lista Expediente/ Caso.

4.1.1.6.6 CASO DE USO: *Listar de Procesos de un expediente.*

CASO DE USO: Lista de Procesos de un expediente.
ACTOR: Trabajador
DESCRIPCIÓN: En este caso de uso permite al Trabajador, desplegar una Lista Procesos De Un Expediente guardados en el sistema.
ESCENARIO NORMAL
El trabajador hace clic en el nombre del expediente y de inmediato se despliega un formulario con descripción del expediente y una lista de procesos, una opción para agregar nuevo proceso y salir.
ESCENARIO ALTERNO
El Trabajador hace clic en la opción nuevo proceso: El sistema muestra un formulario donde se pueda agregar un nuevo proceso para el expediente en elección. El Trabajador hace clic en la opción Salir: El sistema regresa al estado por defecto donde las opciones de menú no están seleccionadas.
PRECONDICIONES: El Trabajador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.
POSTCONDICIONES: Se generó correctamente el listado de procesos de un expediente.

Cuadro n° 4.18. CASO DE USO: Lista de Procesos de un expediente.

4.1.2 HITO ETAPA 1: REVISIÓN DE REQUISITOS.

En esta etapa se revisan los requisitos funcionales, revisión requisitos no funcionales, revisión del modelo de dominio y revisión de casos de uso.

4.1.2.1 REQUISITOS FUNCIONALES REVISADOS

REVISIÓN DE REQUISITOS FUNCIONALES
ReqF01. El sistema debe ser accedido mediante un nombre de usuario y contraseña, según el rol que cumple cada empleado, dependiendo de su área de trabajo.
ReqF02. El sistema debe ser accedido, mediante un nombre de administrador y contraseña.
ReqF03. El sistema mediante el administrador debe permitir registrar una cuenta de usuario para los empleados, según el perfil y facultades de usuario.
ReqF04. El sistema debe permitir modificar una cuenta de usuario; el Administrador puede modificar múltiples cuentas, mientras el usuario solo la de su sesión.
ReqF05. El sistema mediante el administrador debe permitir eliminar las cuentas de usuario.
ReqF06. El sistema mediante el administrador debe permitir buscar las cuentas de usuario.
ReqF07. El sistema debe permitir listar las cuentas de usuario.
ReqF08. El sistema debe permitir agregar una entidad pública.
ReqF09. El sistema debe permitir habilitar la entidad pública en el sistema.
ReqF10. El sistema debe de permitir realizar búsqueda de la entidad pública, por descripción.
ReqF11. El sistema debe de permitir agregar órgano jurisdiccional por distrito y dependencia
ReqF12. El sistema debe de permitir habilitar órgano jurisdiccional por distrito y dependencia en el sistema.

ReqF13. El sistema debe de listar los órganos jurisdiccionales.
ReqF14. El sistema debe permitir agregar materia por Especialidades según su naturaleza.
ReqF15. El sistema debe permitir actualizar materia por Especialidades según su naturaleza.
ReqF16. El sistema debe permitir listar las materias por Especialidades según su naturaleza.
ReqF17. El sistema debe permitir agregar abogado procurador.
ReqF18. El sistema debe de permitir la búsqueda de personas registradas.
ReqF19. El sistema debe de permitir asignar el cargo de abogado procurador.
ReqF20. El sistema muestra una lista de abogados procurador.
ReqF21. El sistema tiene la capacidad de buscar por nombre o núm. de DNI en la lista de abogados procuradores.
ReqF22. El sistema debe permitir eliminar a los abogados procuradores de la lista.
ReqF23. El sistema debe permitir Registrar el tramite (Abogado Procurador, Tipo de Tramite, N° Expediente, N° Folder, Fecha de Registro y Observación)
ReqF24. El sistema debe permitir buscar por Tramite(N° Expediente y N° Folder)
ReqF25. El sistema debe permitir buscar por Demandante (Documento, Ap. Paterno, Ap. Materno y Raz. Social)
ReqF26. El sistema debe permitir buscar por Demandado(Persona y Entidad)
ReqF27. Una vez registrado el trámite en el sistema debe permitir ejecutar la acción de abrir el trámite.
ReqF28. El sistema en la ejecución de trámite debe permitir agregar a los abogados procuradores.
ReqF28. El sistema en la ejecución de trámite debe permitir dar de baja a los abogados procuradores.
ReqF29. El sistema en la ejecución de trámite debe permitir agregar expedientes(Tipo de trámite, N° Expediente, fecha de registro y observación)
ReqF30. El sistema en la ejecución de trámite debe permitir editar expedientes(Tipo de trámite, N° Expediente, fecha de registro y observación)
ReqF31. El sistema en la ejecución de trámite debe permitir eliminar expedientes(Tipo de trámite, N° Expediente, fecha de registro y observación)
ReqF32. El sistema en la ejecución de trámite debe permitir actualizar el estado del trámite(Estado de trámite y observación)
ReqF33. El sistema en la ejecución de trámite debe permitir agregar actos procesales a cada expediente del estado de tramite
ReqF34. El sistema en la ejecución de trámite debe permitir editar los actos procesales de cada expediente.
ReqF35. El sistema en la ejecución de trámite debe permitir eliminar los actos procesales de cada expediente
ReqF36. El sistema en la ejecución de trámite debe permitir agregar administradores de justicia
ReqF37. El sistema en la ejecución de trámite debe permitir editar administradores de justicia
ReqF38. El sistema en la ejecución de trámite debe permitir eliminar administradores de justicia
ReqF39. El sistema en la ejecución de trámite debe permitir agregar documentos/archivos digitales pdf de cada expediente
ReqF40. El sistema en la ejecución de trámite debe permitir eliminar documentos/archivo digitales pdf de cada expediente.
ReqF41. El sistema en la ejecución de trámite debe permitir editar los documentos/archivo digitales pdf de cada expediente.
ReqF42. El sistema en la ejecución de trámite debe permitir agregar a los demandantes (P. Natural o P. Jurídico)
ReqF43. El sistema en la ejecución de trámite debe permitir eliminar a los demandantes.
ReqF44. El sistema en la ejecución de trámite debe permitir agregar a los demandados (P. Natural o P. Jurídica)
ReqF45. El sistema en la ejecución de trámite debe permitir eliminar a los demandados.
ReqF46. El sistema en la ejecución de trámite debe permitir agregar Incidencias/ Medida Cautelar

ReqF47. El sistema en la ejecución de trámite debe permitir editar Incidencias/ Medida Cautelar
ReqF48. El sistema en la ejecución de trámite debe permitir eliminar Incidencias/ Medida Cautelar
ReqF49. El sistema en la ejecución de trámite debe permitir agregar Materia/Especialidades
ReqF50. El sistema en la ejecución de trámite debe permitir eliminar Materia/Especialidades
ReqF51. El sistema en la ejecución de trámite debe permitir agregar Órgano Jurisdiccional del trámite(Distrito Judicial/Fiscal y Órgano Jurisdiccional)
ReqF52. El sistema en la ejecución de trámite debe permitir agregar la lista de pagos(Monto pedido, Monto a pagar y Monto Acumulado)
ReqF53. El sistema en la ejecución de trámite debe permitir editar lista de pagos(Monto pedido, Monto a pagar y Monto Acumulado)
ReqF54. El sistema en la ejecución de trámite debe permitir eliminar lista de pagos(Monto pedido, Monto a pagar y Monto Acumulado)
ReqF55. El sistema debe permitir listar las diligencias pendientes.
ReqF56. El sistema debe permitir listar las diligencias sin concluir.
ReqF57. El sistema debe permitir listar a los demandantes.
ReqF58. El sistema debe permitir buscar a demandantes de la lista.
ReqF59. El sistema debe permitir visualizar un cuadro sumaria estadística por cada mes.
ReqF60. El sistema debe permitir visualizar un gráfico en pastel por tipo de trámite.
ReqF61. El sistema debe permitir visualizar un gráfico en pastel por tipo de materia.
ReqF62. El sistema debe permitir visualizar un gráfico en barras por estado de trámite.
ReqF63. El sistema en la ejecución de trámite debe permitir listar Procuradores asignados al trámite.
ReqF64. El sistema en la ejecución de trámite debe permitir listar Expedientes del trámite.
ReqF65. El sistema en la ejecución de trámite debe permitir listar actos procesales del trámite.
ReqF66. El sistema en la ejecución de trámite debe permitir listar a los administradores de justicia.
ReqF67. El sistema en la ejecución de trámite debe permitir listar los documentos adjuntos del trámite.
ReqF68. El sistema en la ejecución de trámite debe permitir listar a los demandantes.
ReqF69. El sistema en la ejecución de trámite debe permitir listar a los demandados.
ReqF70. El sistema en la ejecución de trámite debe permitir listar las diligencias del trámite.
ReqF71. El sistema en la ejecución de trámite debe permitir Agregar las diligencias para el trámite.
ReqF72. El sistema en la ejecución de trámite debe permitir Dar de baja las diligencias para el trámite.
ReqF73. El sistema en la ejecución de trámite debe permitir listar las Incidencias/ Medida/ Cautelar.
ReqF74. El sistema en la ejecución de trámite debe permitir listar Especialidad/Materia
ReqF75. El sistema en la ejecución de trámite debe permitir listar Órganos Jurisdiccionales.

Cuadro n° 4.19. Requisitos funcionales revisados.

4.1.2.2 REQUISITOS NO FUNCIONALES REVISADOS

REVISIÓN DE REQUISITOS NO FUNCIONALES
ReqNF01. La aplicación debe ser de interfaz amigable, moderno e intuitivo para con los usuarios, mostrando ayudas que guiarán al usuario en su correcto uso del mismo.
ReqNF02. El sistema debe ser robusto que contemple la distribución de carga a nivel de componentes siendo altamente escalable.
ReqNF03. En el desarrollo de la aplicación se usarán estándares de nomenclatura para las clases, objetos, vistas, controladores, etc.; que permitan una adecuada mantenibilidad de la aplicación en las etapas de pruebas unitarias y post-producción.
ReqNF04. Para el desarrollo de la aplicación web se empleará la tecnología Php para proveer

mayor agilidad en el desarrollo de aplicaciones.
ReqNF05. En la fase de desarrollo de la aplicación se usará el patrón de arquitectura “n capas” para separar cada fase de ejecución de la aplicación.
ReqNF06. El sistema debe contener niveles de seguridad adecuados y acceso a la información por roles y grupos de usuarios como administrador, digitador.
ReqNF07. La aplicación estará basada en entorno web, debe tener una arquitectura técnica y codificación usando estándares que permita realizar el mantenimiento de la aplicación periódicamente.
ReqNF08. El software debe garantizar la autenticidad, la cual asegura que los usuarios son quienes dicen que son y que se puede verificar su autenticidad
ReqNF09. El software debe garantizar la confidencialidad, implementando un mecanismo para asegurar la confidencialidad de datos y transacciones, mediante el acceso restringido a la base de datos y las operaciones
ReqNF10. El software debe garantizar la integridad, para asegurar la integridad de datos y de los cálculos
ReqNF11. Las interfaces deben cumplir con requisitos mínimos de usabilidad.

Cuadro n° 4.20. Requisitos no funcionales revisados.

De los requisitos funcionales y no funcionales revisados detalladamente se resumió la siguiente lista para las clases del **modelo de dominio**.

- Entidad publica
- Demandado
- Persona
- Registro pago
- Proceso cautelar
- Documento adjunto
- Expediente
- Demandante
- Recurso
- Recurso especialidad
- Especialidad
- Diligencia
- Proceso
- Recurso financiero
- Recurso instancia
- Tablas código
- Usuario
- Funcionario

- Instancia
- Tablas
- Ubigeos
- Profesional Jurídico
- Entidad Jurídico
- Órgano Jurisdiccional

Desambiguación de términos:

- El término **Entidad** y se resumirá en un solo término **Entidad Jurídico**.
- El término **Representante** será representado con el término **Profesional Jurídico**.
- El término **Proceso De Expediente** se convierte en los términos **Proceso** y **Expediente**.
- El término **Trabajador** se representara con el término **Persona**.

4.1.2.3 REVISIÓN DE MODELO DE DOMINIO

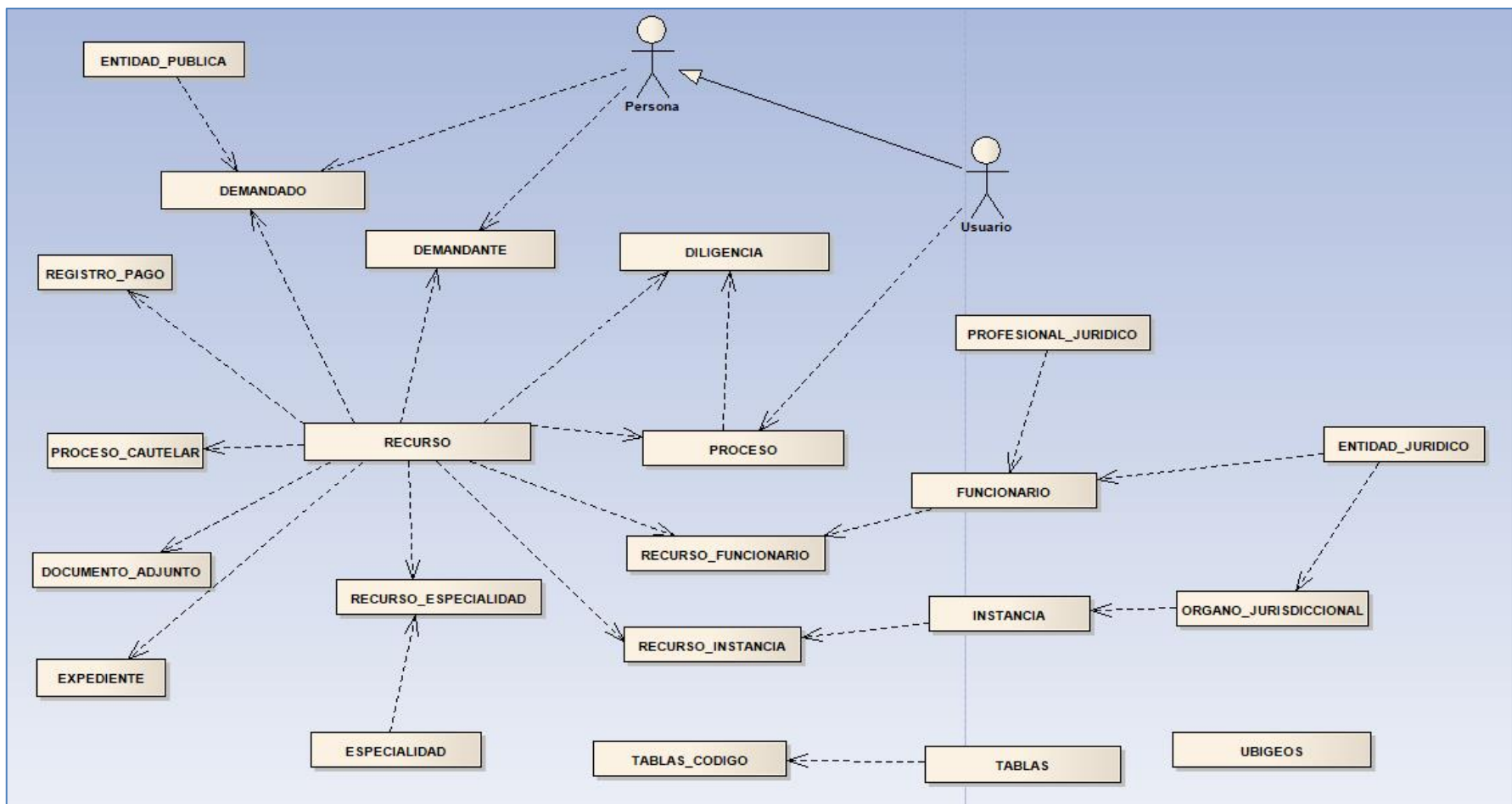


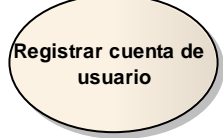
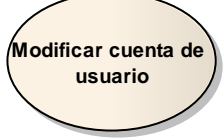
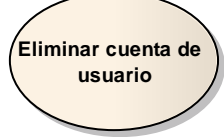
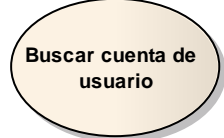
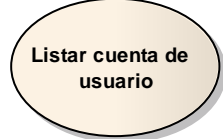








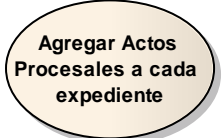


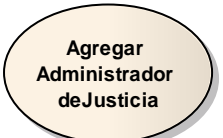
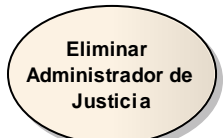
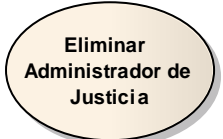
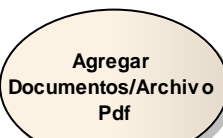
Figura n° 4.24: Modelo de dominio general revisado

4.1.2.4 REVISIÓN DE CASOS DE USO

CASO DE USO	REQUISITOS FUNCIONALES
	<p>ReqF01. El sistema debe ser accedido mediante un nombre de usuario y contraseña, según el rol que cumple cada empleado, dependiendo de su área de trabajo.</p>
	<p>ReqF02. El sistema debe ser accedido, mediante un nombre de administrador y contraseña.</p>
	<p>ReqF03. El sistema mediante el administrador debe permitir registrar una cuenta de usuario para los empleados, según el perfil y facultades de usuario.</p>
	<p>ReqF04. El sistema debe permitir modificar una cuenta de usuario; el Administrador puede modificar múltiples cuentas, mientras el usuario solo la de su sesión.</p>
	<p>ReqF05. El sistema mediante el administrador debe permitir eliminar las cuentas de usuario.</p>
	<p>ReqF06. El sistema mediante el administrador debe permitir buscar las cuentas de usuario.</p>
	<p>ReqF07. El sistema debe permitir listar las cuentas de usuario.</p>
	<p>ReqF08. El sistema debe permitir agregar una entidad pública.</p>
	<p>ReqF09. El sistema debe permitir habilitar la entidad pública en el sistema.</p>
	<p>ReqF10. El sistema debe de permitir realizar búsqueda de la entidad pública, por descripción.</p>

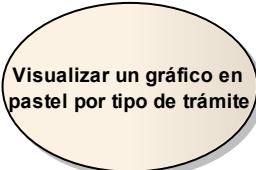
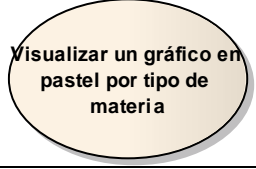
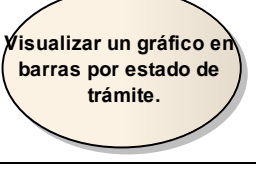
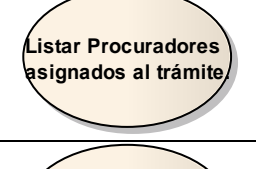


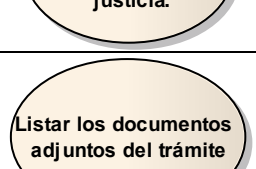



<p style="text-align: center;">Agregar Organo Jurisdiccional por Distrito y Dependencia.</p>	<p>ReqF11. El sistema debe de permitir agregar órgano jurisdiccional por distrito y dependencia</p>
<p style="text-align: center;">Habilitar Organo Jurisdiccional</p>	<p>ReqF12. El sistema debe de permitir habilitar órgano jurisdiccional por distrito y dependencia en el sistema.</p>
<p style="text-align: center;">Listar Organos Jurisdiccionales</p>	<p>ReqF13. El sistema debe de listar los órganos jurisdiccionales.</p>
<p style="text-align: center;">Agregar Materia por Especialidades según Natualeza</p>	<p>ReqF14. El sistema debe permitir agregar materia por Especialidades según su naturaleza.</p>
<p style="text-align: center;">Actualizar Materia por Especialidades según Natualeza</p>	<p>ReqF15. El sistema debe permitir actualizar materia por Especialidades según su naturaleza.</p>
<p style="text-align: center;">Listar Materia por Especialidades según Naturaleza</p>	<p>ReqF16. El sistema debe permitir listar las materias por Especialidades según su naturaleza.</p>
<p style="text-align: center;">Agregar abogado Procurador</p>	<p>ReqF17. El sistema debe permitir agregar abogado procurador.</p>
<p style="text-align: center;">Buscar Personas Registradas</p>	<p>ReqF18. El sistema debe de permitir la búsqueda de personas registradas.</p>
<p style="text-align: center;">Asignar El cargo de abogado Procurador</p>	<p>ReqF19. El sistema debe de permitir asignar el cargo de abogado procurador.</p>
<p style="text-align: center;">Listar a los Abogador Procurador</p>	<p>ReqF20. El sistema muestra una lista de abogados procurador.</p>

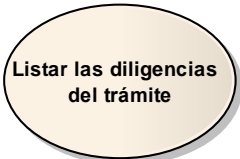
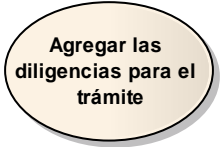
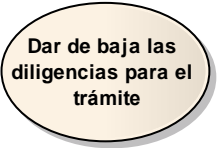



<p>Buscar por nombre o núm. de DNI en la lista de abogados procuradores</p>	<p>ReqF21. El sistema tiene la capacidad de buscar por nombre o núm. de DNI en la lista de abogados procuradores.</p>
<p>Eliminar de la lista a los abogados procuradores</p>	<p>ReqF22. El sistema debe permitir eliminar a los abogados procuradores de la lista.</p>
<p>Registrar Trámite</p>	<p>ReqF23. El sistema debe permitir Registrar el trámite (Abogado Procurador, Tipo de Trámite, N° Expediente, N° Folder, Fecha de Registro y Observación)</p>
<p>Buscar trámite</p>	<p>ReqF24. El sistema debe permitir buscar por Trámite(N° Expediente y N° Folder)</p>
<p>Buscar Trámite por Demandante</p>	<p>ReqF25. El sistema debe permitir buscar por Demandante (Documento, Ap. Paterno, Ap. Materno y Raz. Social)</p>
<p>Buscar Trámite por demandado</p>	<p>ReqF26. El sistema debe permitir buscar por Demandado(Persona y Entidad)</p>
<p>Ejecutar acción de Trámite</p>	<p>ReqF27. Una vez registrado el trámite en el sistema debe permitir ejecutar la acción de abrir el trámite.</p>
<p>Agregar al Trámite un Abogado procurador</p>	<p>ReqF28. El sistema en la ejecución de trámite debe permitir agregar a los abogados procuradores.</p>
<p>Dar de baja del Trámite al abogado procurador</p>	<p>ReqF28. El sistema en la ejecución de trámite debe permitir dar de baja a los abogados procuradores.</p>
<p>Agregar Expediente</p>	<p>ReqF29. El sistema en la ejecución de trámite debe permitir agregar expedientes(Tipo de trámite, N° Expediente, fecha de registro y observación)</p>

	<p>ReqF30. El sistema en la ejecución de trámite debe permitir editar expedientes(Tipo de trámite, N° Expediente, fecha de registro y observación)</p>
	<p>ReqF31. El sistema en la ejecución de trámite debe permitir eliminar expedientes(Tipo de trámite, N° Expediente, fecha de registro y observación)</p>
	<p>ReqF32. El sistema en la ejecución de trámite debe permitir actualizar el estado del trámite(Estado de trámite y observación)</p>
	<p>ReqF33. El sistema en la ejecución de trámite debe permitir agregar actos procesales a cada expediente del estado de tramite</p>
	<p>ReqF34. El sistema en la ejecución de trámite debe permitir editar los actos procesales de cada expediente.</p>
	<p>ReqF35. El sistema en la ejecución de trámite debe permitir eliminar los actos procesales de cada expediente</p>
	<p>ReqF36. El sistema en la ejecución de trámite debe permitir agregar administradores de justicia</p>
	<p>ReqF37. El sistema en la ejecución de trámite debe permitir editar administradores de justicia</p>
	<p>ReqF38. El sistema en la ejecución de trámite debe permitir eliminar administradores de justicia</p>
	<p>ReqF39. El sistema en la ejecución de trámite debe permitir agregar documentos/archivos digitales pdf de cada expediente</p>

<p style="text-align: center;">Eliminar Documentos/Archivos Pdf</p>	<p>ReqF40. El sistema en la ejecución de trámite debe permitir eliminar documentos/archivo digitales pdf de cada expediente.</p>
<p style="text-align: center;">Editar Documentos/Archivos Pdf</p>	<p>ReqF41. El sistema en la ejecución de trámite debe permitir editar los documentos/archivo digitales pdf de cada expediente.</p>
<p style="text-align: center;">Agregar Demandante</p>	<p>ReqF42. El sistema en la ejecución de trámite debe permitir agregar a los demandantes (P. Natural o P. Jurídico)</p>
<p style="text-align: center;">Eliminar Demandante</p>	<p>ReqF43. El sistema en la ejecución de trámite debe permitir eliminar a los demandantes.</p>
<p style="text-align: center;">Agregar Demandado</p>	<p>ReqF44. El sistema en la ejecución de trámite debe permitir agregar a los demandados (P. Natural o P. Jurídica)</p>
<p style="text-align: center;">Eliminar Demandado</p>	<p>ReqF45. El sistema en la ejecución de trámite debe permitir eliminar a los demandados.</p>
<p style="text-align: center;">Agregar Incidencias/ Medida Cautelar</p>	<p>ReqF46. El sistema en la ejecución de trámite debe permitir agregar Incidencias/ Medida Cautelar</p>
<p style="text-align: center;">Editar Incidencias/ Medida Cautelar</p>	<p>ReqF47. El sistema en la ejecución de trámite debe permitir editar Incidencias/ Medida Cautelar</p>
<p style="text-align: center;">Eliminar Incidencias/ Medida Cautelar</p>	<p>ReqF48. El sistema en la ejecución de trámite debe permitir eliminar Incidencias/ Medida Cautelar</p>
<p style="text-align: center;">Agregar Materia/Especialidades</p>	<p>ReqF49. El sistema en la ejecución de trámite debe permitir agregar Materia/Especialidades</p>

<p style="text-align: center;">Eliminar Materia/Especialidades</p>	<p>ReqF50. El sistema en la ejecución de trámite debe permitir eliminar Materia/Especialidades</p>
<p style="text-align: center;">Agregar Órgano Jurisdiccional del trámite</p>	<p>ReqF51. El sistema en la ejecución de trámite debe permitir agregar Órgano Jurisdiccional del trámite(Distrito Judicial/Fiscal y Órgano Jurisdiccional)</p>
<p style="text-align: center;">Agregar la lista de pagos</p>	<p>ReqF52. El sistema en la ejecución de trámite debe permitir agregar la lista de pagos(Monto pedido, Monto a pagar y Monto Acumulado)</p>
<p style="text-align: center;">Editar lista de pagos</p>	<p>ReqF53. El sistema en la ejecución de trámite debe permitir editar lista de pagos(Monto pedido, Monto a pagar y Monto Acumulado)</p>
<p style="text-align: center;">Eliminar lista de pagos</p>	<p>ReqF54. El sistema en la ejecución de trámite debe permitir eliminar lista de pagos(Monto pedido, Monto a pagar y Monto Acumulado)</p>
<p style="text-align: center;">Listar Diligencias Pendientes</p>	<p>ReqF55. El sistema debe permitir listar las diligencias pendientes.</p>
<p style="text-align: center;">Listar Diligencias sin Concluir</p>	<p>ReqF56. El sistema debe permitir listar las diligencias sin concluir.</p>
<p style="text-align: center;">Listar Demandantes</p>	<p>ReqF57. El sistema debe permitir listar a los demandantes.</p>
<p style="text-align: center;">Buscar Demandantes</p>	<p>ReqF58. El sistema debe permitir buscar a demandantes de la lista.</p>
<p style="text-align: center;">Visualizar un cuadro sumaria estadística por cada mes</p>	<p>ReqF59. El sistema debe permitir visualizar un cuadro sumaria estadística por cada mes.</p>

 <p>Visualizar un gráfico en pastel por tipo de trámite</p>	<p>ReqF60. El sistema debe permitir visualizar un gráfico en pastel por tipo de trámite.</p>
 <p>Visualizar un gráfico en pastel por tipo de materia</p>	<p>ReqF61. El sistema debe permitir visualizar un gráfico en pastel por tipo de materia.</p>
 <p>Visualizar un gráfico en barras por estado de trámite.</p>	<p>ReqF62. El sistema debe permitir visualizar un gráfico en barras por estado de trámite.</p>
 <p>Listar Procuradores asignados al trámite</p>	<p>ReqF63. El sistema en la ejecución de trámite debe permitir listar Procuradores asignados al trámite.</p>
 <p>Listar Expedientes del trámite</p>	<p>ReqF64. El sistema en la ejecución de trámite debe permitir listar Expedientes del trámite.</p>
 <p>Listar actos procesales del trámite</p>	<p>ReqF65. El sistema en la ejecución de trámite debe permitir listar actos procesales del trámite.</p>
 <p>Listar a los administradores de justicia.</p>	<p>ReqF66. El sistema en la ejecución de trámite debe permitir listar a los administradores de justicia.</p>
 <p>Listar los documentos adjuntos del trámite</p>	<p>ReqF67. El sistema en la ejecución de trámite debe permitir listar los documentos adjuntos del trámite.</p>
 <p>Listar Demandantes</p>	<p>ReqF68. El sistema en la ejecución de trámite debe permitir listar a los demandantes.</p>
 <p>Listar Demandados</p>	<p>ReqF69. El sistema en la ejecución de trámite debe permitir listar a los demandados.</p>

	ReqF70. El sistema en la ejecución de trámite debe permitir listar las diligencias del trámite.
	ReqF71. El sistema en la ejecución de trámite debe permitir Agregar las diligencias para el trámite.
	ReqF72. El sistema en la ejecución de trámite debe permitir Dar de baja las diligencias para el trámite.
	ReqF73. El sistema en la ejecución de trámite debe permitir listar las Incidencias/ Medida/ Cautelar.
	ReqF74. El sistema en la ejecución de trámite debe permitir listar Especialidad/Materia
	ReqF75. El sistema en la ejecución de trámite debe permitir listar Órganos Jurisdiccionales.

Cuadro n° 4.21. Revisión de casos de uso.

4.1.2.4.1 CASO DE USO: Registrar Usuario.

CASO DE USO: Registrar Usuario.
ACTOR: Usuario Administrador
DESCRIPCIÓN: Este caso de uso permite al Usuario Administrador, registrar nuevos usuarios administradores del sistema y usuario trabajadores.
ESCENARIO NORMAL
<ol style="list-style-type: none"> 1). En la página principal el usuario deberá hacer clic en el menú Administración y luego Usuario del Sistema. 2). El sistema muestra una vista con opciones para la administración de usuario. 3). El usuario hace clic en la opción Agregar. 4). El sistema muestra una vista con los campos de usuarios, para ser llenado posteriormente. 5). El usuario tiene la opción de buscar los datos de usuarios ya guardados en el sistema o ingresar nuevos datos; El usuario tiene la opción seleccionar el perfil de usuario dentro de un select; el usuario deberá ingresar el login de usuario el mismo que será el password; una vez completado la digitación en los campos de usuario, se procede con dar clic al botón guardar. 6). El sistema guarda la información de usuario en la base de datos.
ESCENARIO ALTERNO
El Usuario Administrador no ingresó todos los datos requeridos: El sistema muestra un

<p>mensaje indicando campo que falta ingresar.</p> <p>El Usuario Administrador ingresó datos incorrectos: El sistema muestra un mensaje indicando la característica de los datos a ingresar.</p> <p>El Usuario Administrador hace clic en el botón salir: El sistema regresa a una vista de gestión de usuario.</p>
<p>PRECONDICIONES: El Usuario Administrador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.</p>
<p>POSTCONDICIONES: Se gestionó adecuadamente el registro de usuario.</p>

Cuadro n° 4.22. Revisión del caso de uso Registrar Usuario.

4.1.2.4.2 CASO DE USO: Listar Usuarios Del Sistema.

<p>CASO DE USO: Listar De Usuarios Del Sistema.</p>
<p>ACTOR: Usuario Administrador</p>
<p>DESCRIPCIÓN: En este caso de uso permite al Usuario Administrador, desplegar todos los usuarios habilitados para el uso del sistema.</p>
<p>ESCENARIO NORMAL</p>
<p>En la página principal el usuario deberá hacer clic en el menú Administración y luego Usuario del Sistema.</p> <p>El sistema muestra una vista con opciones para la administración de usuario (agregar, editar y salir) y también muestran una lista de usuarios registrados los mismos que son administrables.</p>
<p>ESCENARIO ALTERNO</p>
<p>El Usuario Administrador hace clic en la opción Editar de un usuario: El sistema muestra un formulario con los campos del usuario para poder actualizarlos.</p> <p>El Usuario Administrador hace clic en la opción Agregar: El sistema muestra un formulario con los campos vacíos para poder ingresar los datos del nuevo usuario.</p> <p>El Usuario Administrador hace clic en la opción Salir: El sistema regresa al estado por defecto donde las opciones de menú no están seleccionadas.</p>
<p>PRECONDICIONES: El Usuario Administrador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.</p>
<p>POSTCONDICIONES: Se generó correctamente el listado de usuarios.</p>

Cuadro n° 4.23. Revisión del caso de uso Listar Usuarios Del Sistema.

4.1.2.4.3 CASO DE USO: Agregar Expediente /Caso.

<p>CASO DE USO: Agregar Expediente /Caso.</p>
<p>ACTOR: Usuario Trabajador</p>
<p>DESCRIPCIÓN: Este caso de uso permite al Trabajador, registrar nuevos expedientes /caso en el sistema.</p>
<p>ESCENARIO NORMAL</p>
<ol style="list-style-type: none"> 1). En la página principal el Usuario Trabajador deberá hacer clic en el menú Trámite Documentario y luego Administrar Trámite Sistema. 2). El sistema muestra una vista con opciones para la Administración de Trámite. 3). El Usuario Trabajador hace clic en la opción Registrar. 4). El sistema muestra una vista con los campos de Trámite, posteriormente para ser llenados. 5). El Usuario Trabajador puede elegir al Abogado Procurador el mismo que fue guardado por el caso de uso "Agregar Abogado Procurador", se mostrará un select para elegir; se podrá elegir el tipo de tramite mediante un select el cual desplegara una lista; podrá registrar el N° de Expediente, N° de Folder, fecha de registro y alguna observación si es el caso; seguidamente se hace clic en el botón Guardar. 6). El sistema guarda la información del Trámite en la base de datos.

ESCENARIO ALTERNO
<p>El Trabajador no ingresó todos los datos requeridos: El sistema muestra un mensaje indicando campo que falta ingresar.</p> <p>El Trabajador ingresó datos incorrectos: El sistema muestra un mensaje indicando la característica de los datos a ingresar.</p> <p>El Trabajador hace clic en el botón salir: El sistema regresa a una vista de gestión de recurso.</p>
PRECONDICIONES: El Trabajador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.
POSTCONDICIONES: Se gestionó adecuadamente el registro de Expediente.

Cuadro n° 4.24. Revisión del caso de uso Agregar Expediente/ Caso.

4.1.2.4.4 CASO DE USO: Listar Expediente/ Caso.

CASO DE USO: Listar Expediente/ Caso.
ACTOR: Trabajador
DESCRIPCIÓN: En este caso de uso permite al Trabajador, desplegar una Lista de Expedientes Caso guardados en el sistema.
ESCENARIO NORMAL
<ol style="list-style-type: none"> 1). En la página principal el Usuario Trabajador deberá hacer clic en el menú Trámite Documentario y luego Administrar Trámite Sistema. 2). El sistema muestra una vista con opciones para la Administración de Trámite (Buscar por Trámite, Buscar por Demandante, Buscar por Demandado, Registrar y Salir) y una lista de Expedientes/ Caso, el mismo tiene la opción (Abrir Tramite).
ESCENARIO ALTERNO
<p>El Trabajador hace clic en la opción Acción Abrir Tramite: El sistema realiza una acción de Abrir Tramite para cada expediente.</p> <p>El Trabajador hace clic en la opción Registrar: El sistema muestra un formulario con los campos vacíos para poder ingresar los datos del nuevo Trámite.</p> <p>El Trabajador hace clic en la opción Salir: El sistema regresa al estado por defecto donde las opciones de menú no están seleccionadas.</p>
PRECONDICIONES: El Trabajador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.
POSTCONDICIONES: Se generó correctamente el listado de Expediente /Caso.

Cuadro n° 4.25. Revisión del caso de uso Listar Expediente/ Caso.

4.1.2.4.5 CASO DE USO: Listar Procesos de un expediente.

CASO DE USO: Lista de Procesos de un expediente.
ACTOR: Trabajador
DESCRIPCIÓN: En este caso de uso permite al Trabajador, desplegar una Lista Procesos De Un Expediente guardados en el sistema.
ESCENARIO NORMAL
<ol style="list-style-type: none"> 1). En la página principal el Usuario Trabajador deberá hacer clic en el menú Trámite Documentario y luego Administrar Trámite y por ultimo Abrir Tramite (Es una opción). 2). El sistema muestra una vista con un menú lateral y un detalle de trámite en la parte inferior. 3). En el menú lateral se encuentra una opción “Expedientes”, hacemos clic en la descripción. 4). El sistema muestra una vista con una lista de todos los procesos por expediente.

ESCENARIO ALTERNO
<p>El Trabajador hace clic en la opción Agregar: El sistema muestra un formulario donde se pueda agregar un nuevo proceso para el expediente en elección.</p> <p>El Trabajador hace clic en la opción Salir: El sistema regresa al estado por defecto donde las opciones de menú no están seleccionadas.</p>
<p>PRECONDICIONES: El Trabajador tiene que estar autenticado y haber iniciado una sesión la cual tiene estar activa.</p>
<p>POSTCONDICIONES: Se generó correctamente el listado de procesos de un expediente.</p>

Cuadro n° 4.26. Revisión del caso de uso Listar Procesos de un Expediente.

4.1.2.5 CASO DE USO POR PAQUETES

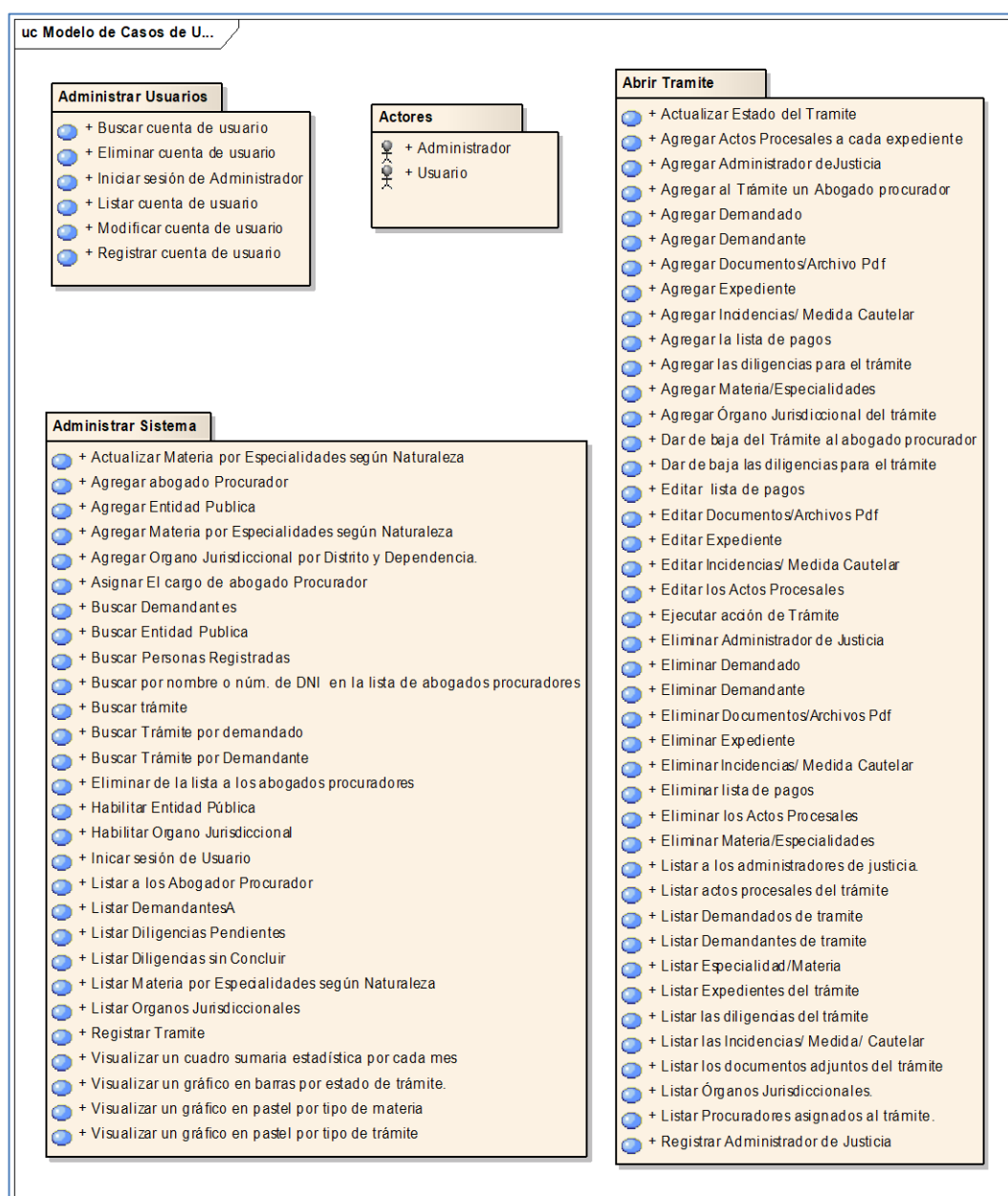


Figura n° 4.25: Empaquetado General de Casos de Uso.

4.1.2.5.1 PAQUETE “Actores”

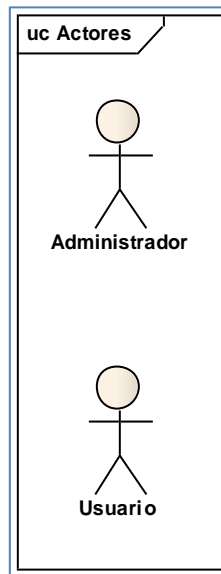


Figura n° 4.26: Empaquetado General de Casos de Uso.

4.1.2.5.2 PAQUETE “Administrar Usuario”

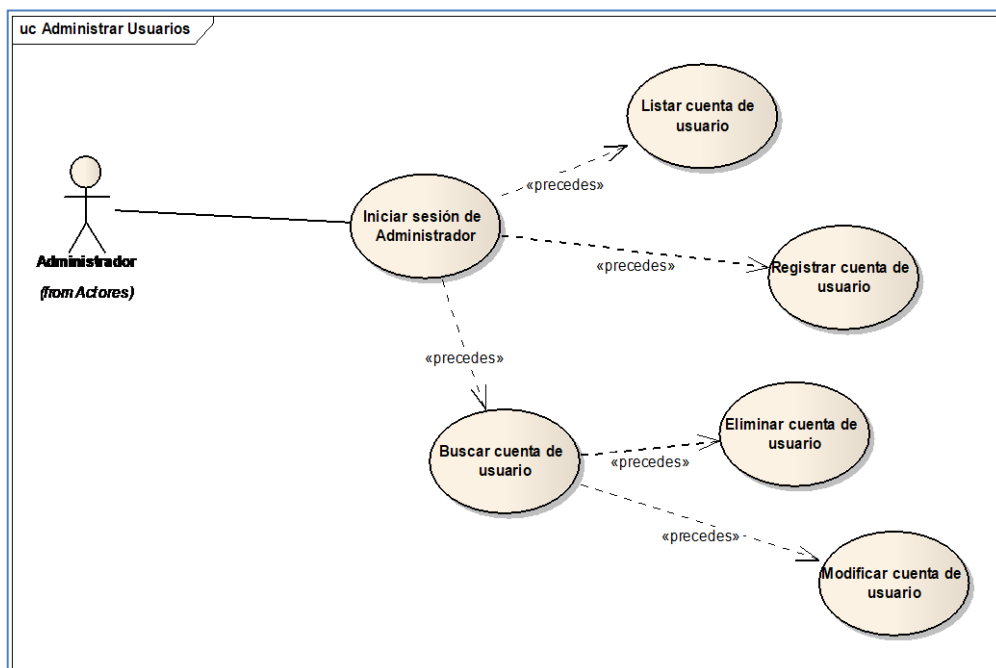


Figura n° 4.27: Empaquetado General de Casos de Uso.

4.1.2.5.3 PAQUETE “Administrar Sistema”

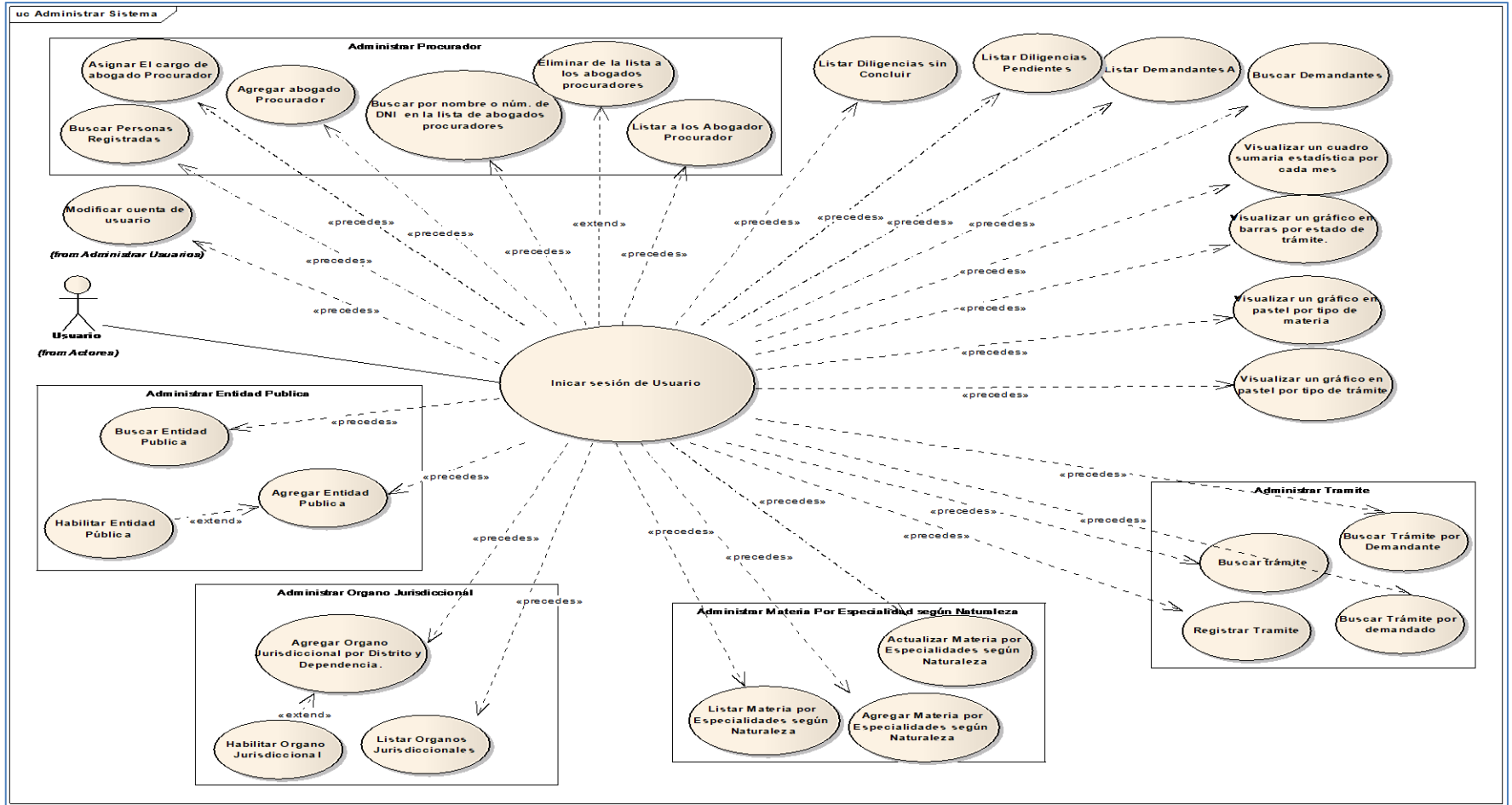


Figura n° 4.28: Empaquetado General de Casos de Uso.

4.1.2.5.4 PAQUETE “Abrir Trámite”

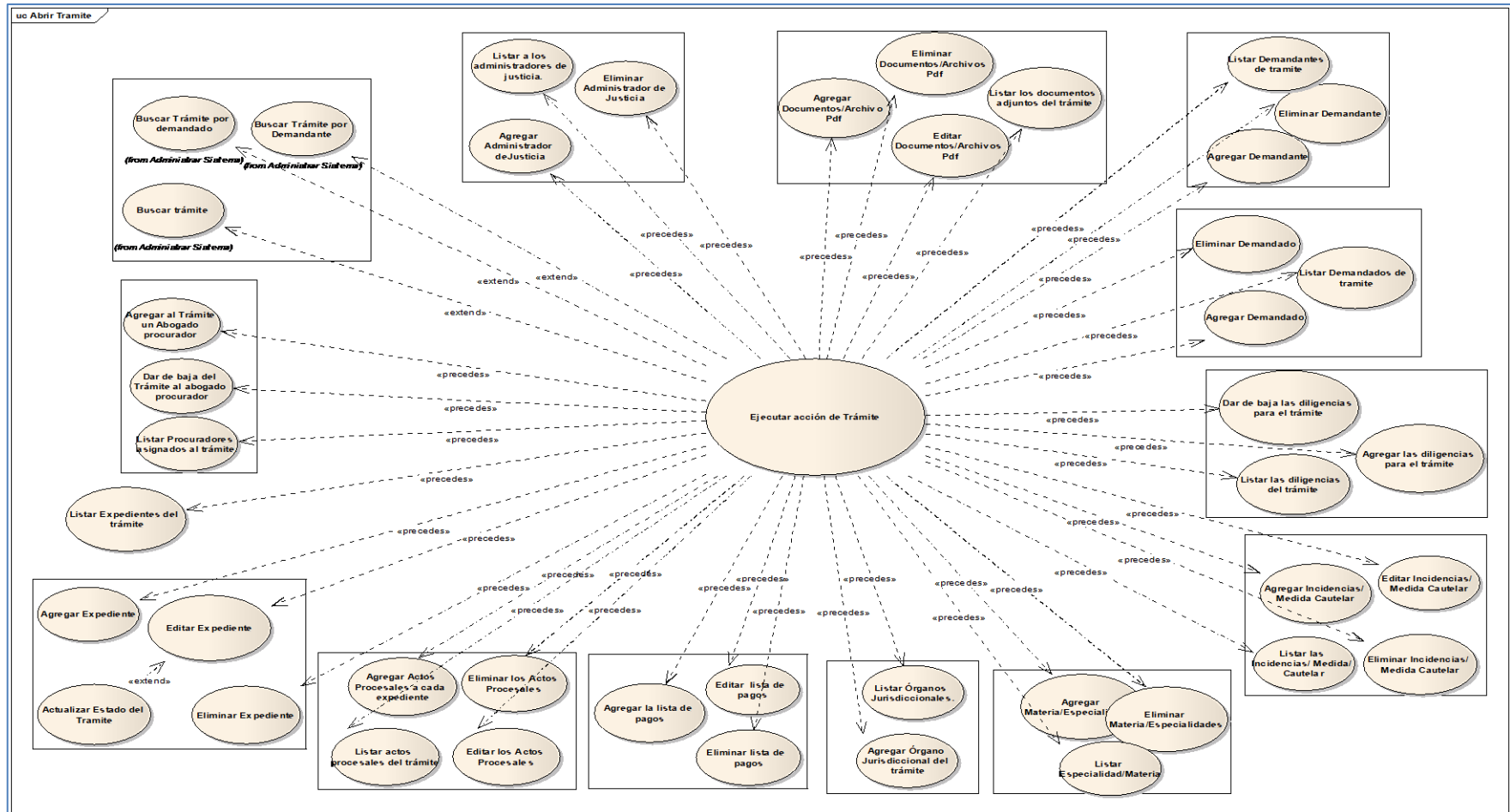


Figura n° 4.29: Empaquetado General de Casos de Uso.

4.1.3 ETAPA 2: ANÁLISIS Y DISEÑO PRELIMINAR.

4.1.3.1 DIAGRAMA DE ROBUSTEZ

Se elaborara los diagramas de robustez par los casos de uso elegidos.

4.1.3.1.1 DIAGRAMA DE ROBUSTEZ “Registrar Usuario”

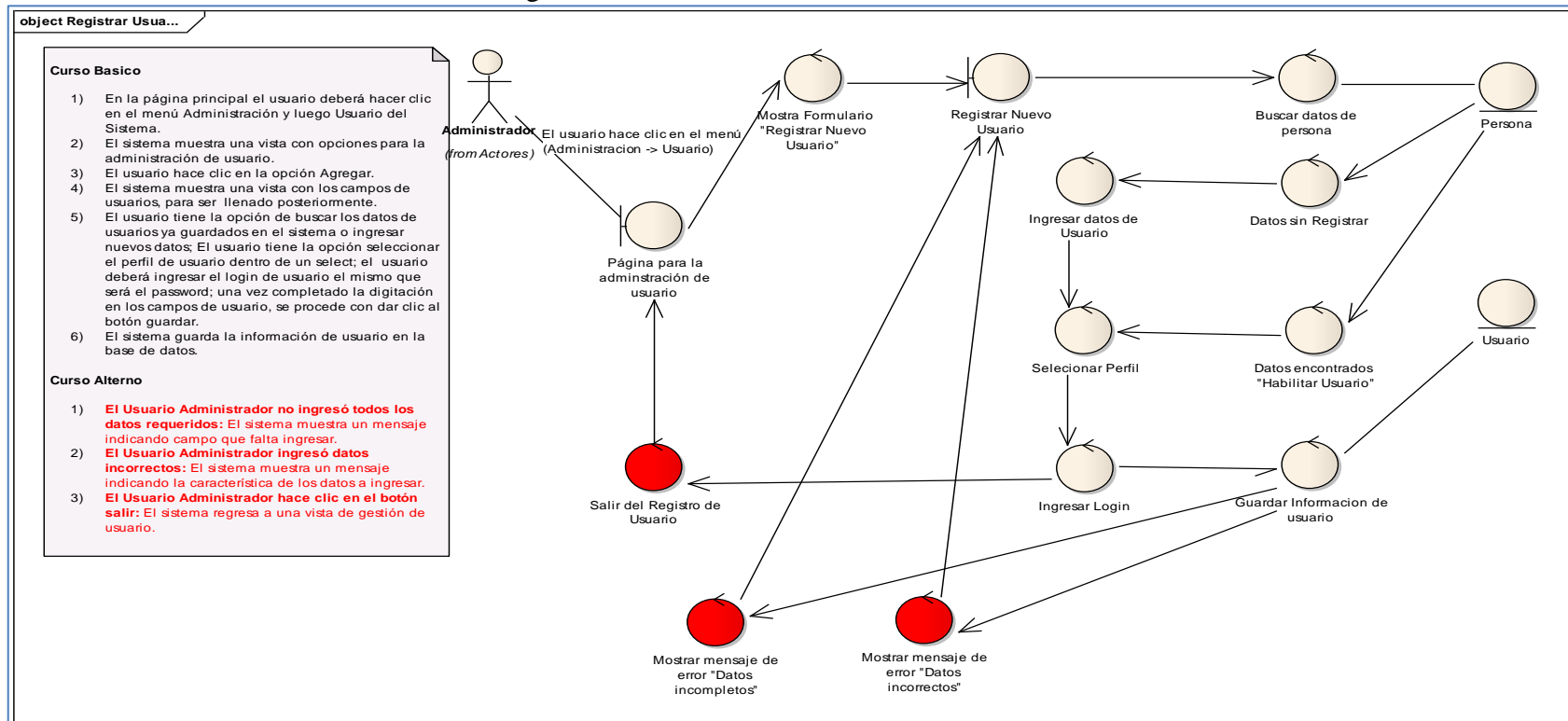


Figura n° 4.30: Diagrama de Robustez: Registrar Usuario.

4.1.3.1.2 DIAGRAMA DE ROBUSTEZ “Listar Usuarios Del Sistema”

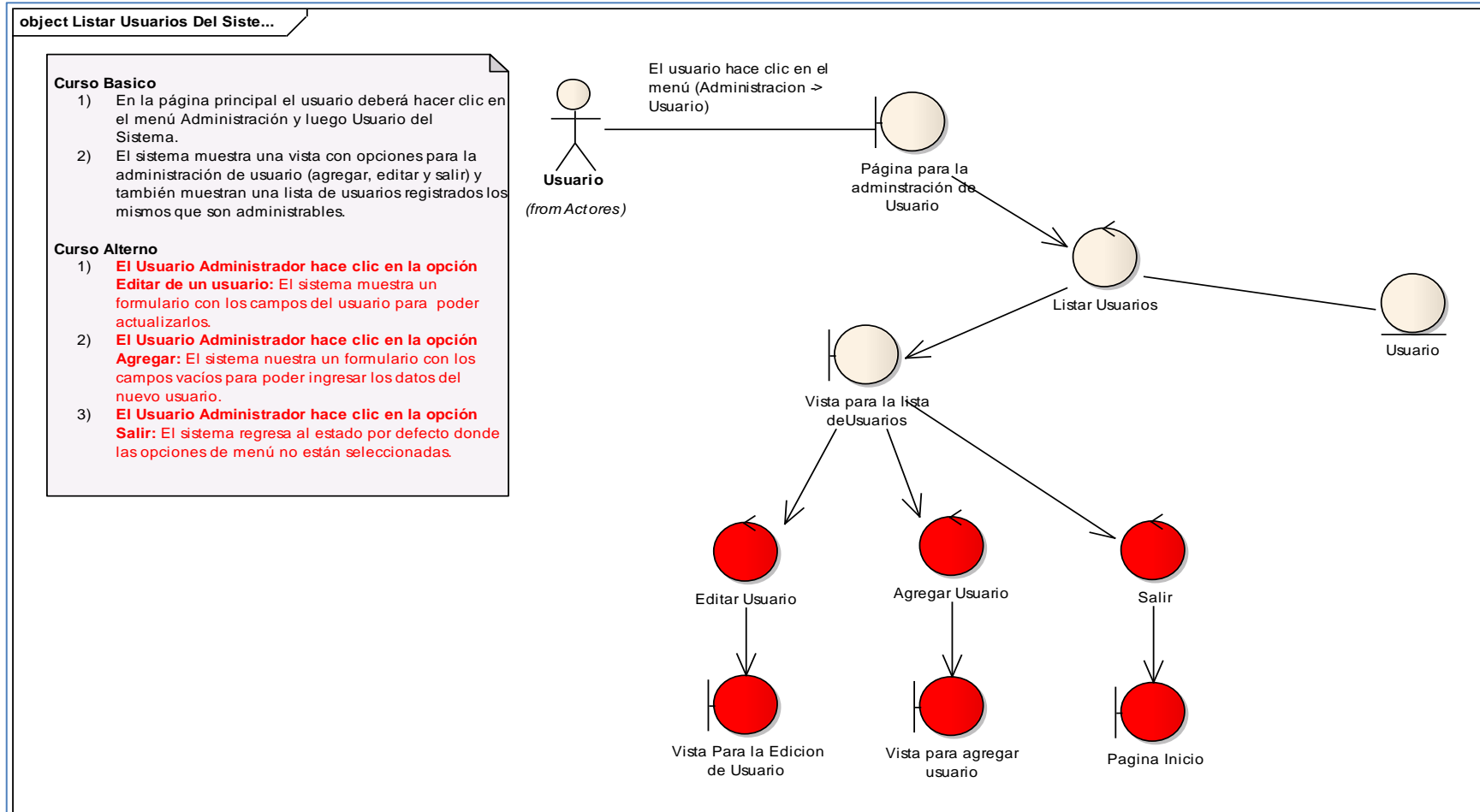


Figura n° 4.31: Diagrama de Robustez: Listar Usuarios Del Sistema.

4.1.3.1.3 DIAGRAMA DE ROBUSTEZ “Agregar Expediente /Caso”

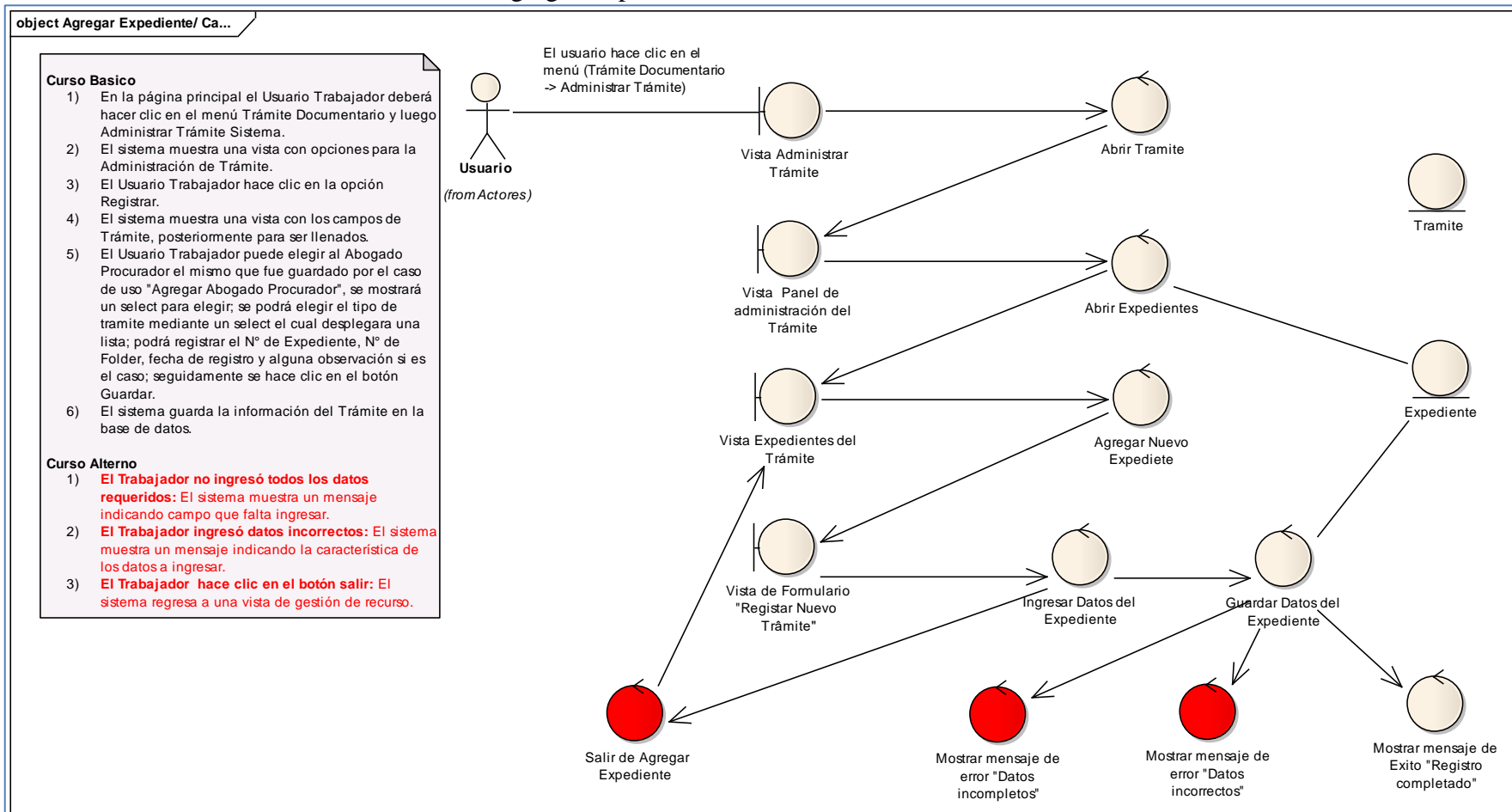


Figura n° 4.32: Diagrama de Robustez: Agregar Expediente /Caso.

4.1.3.1.4 DIAGRAMA DE ROBUSTEZ “Listar Expediente/ Caso”

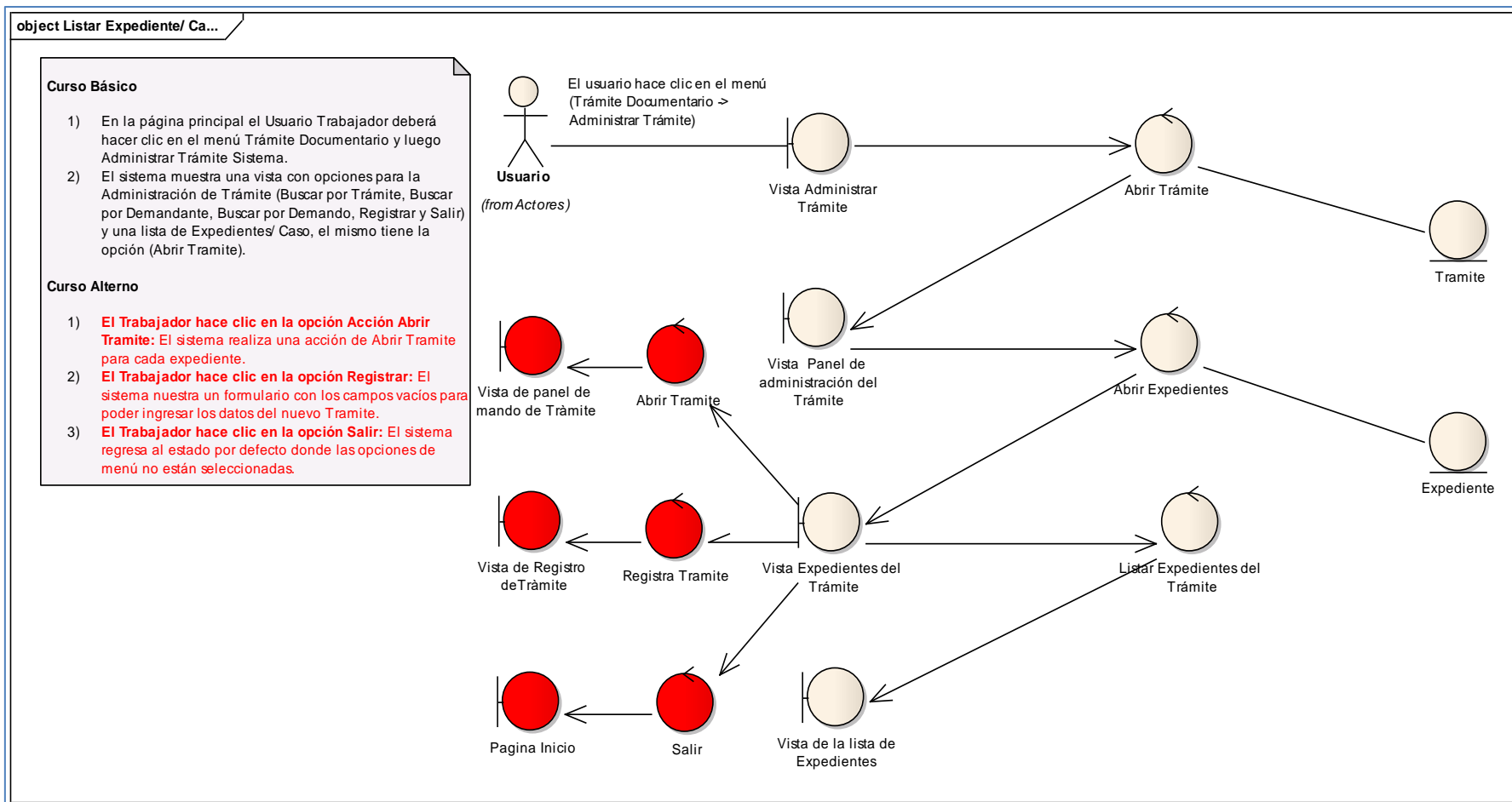


Figura n° 4.33: Diagrama de Robustez: Listar Expediente/ Caso.

4.1.3.1.5 DIAGRAMA DE ROBUSTEZ “Listar Procesos de un expediente”

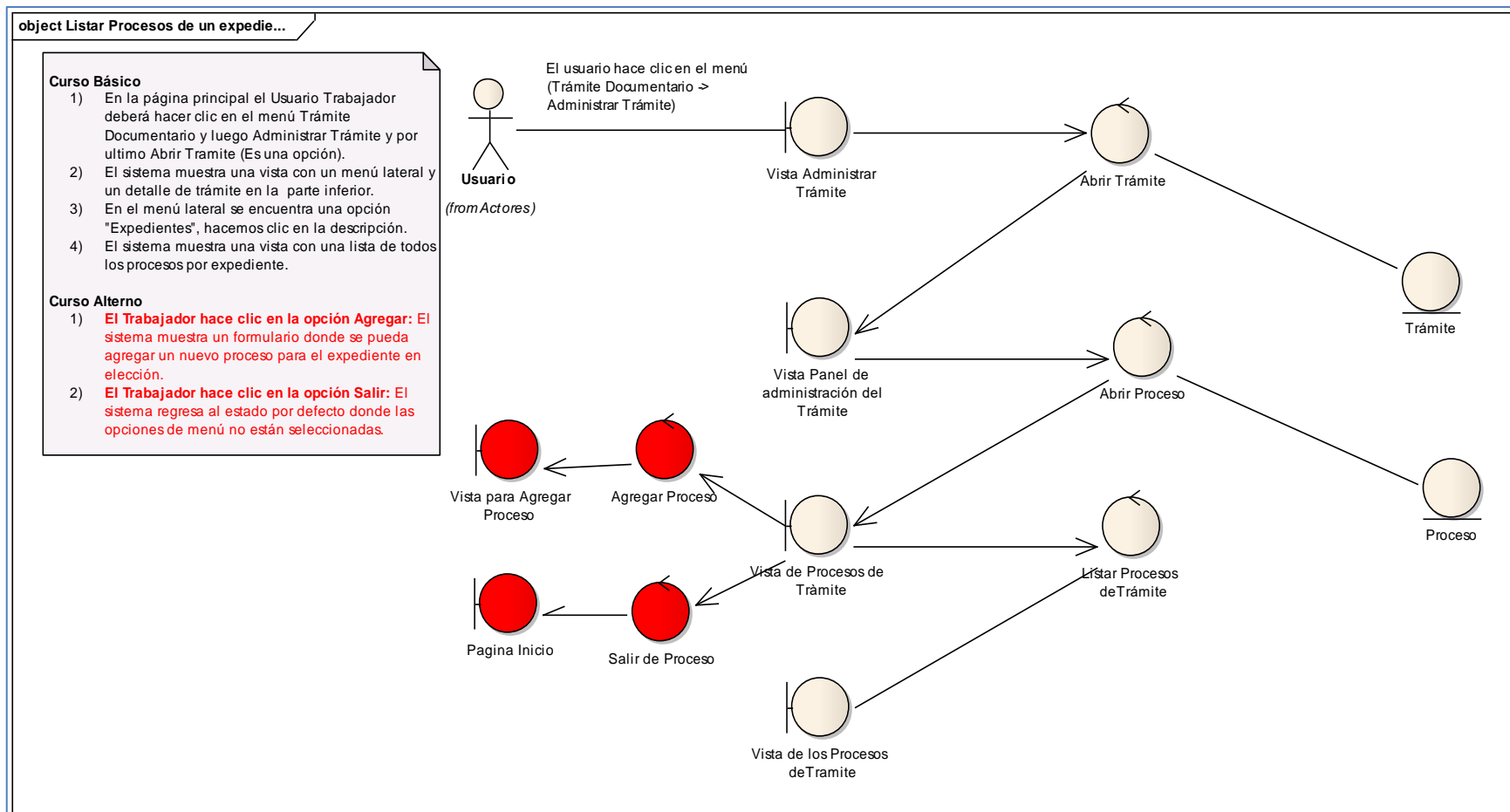


Figura n° 4.34: Diagrama de Robustez: Listar Procesos de un Trámite.

4.1.3.2 MODELO DE DOMINIO ACTUALIZADO

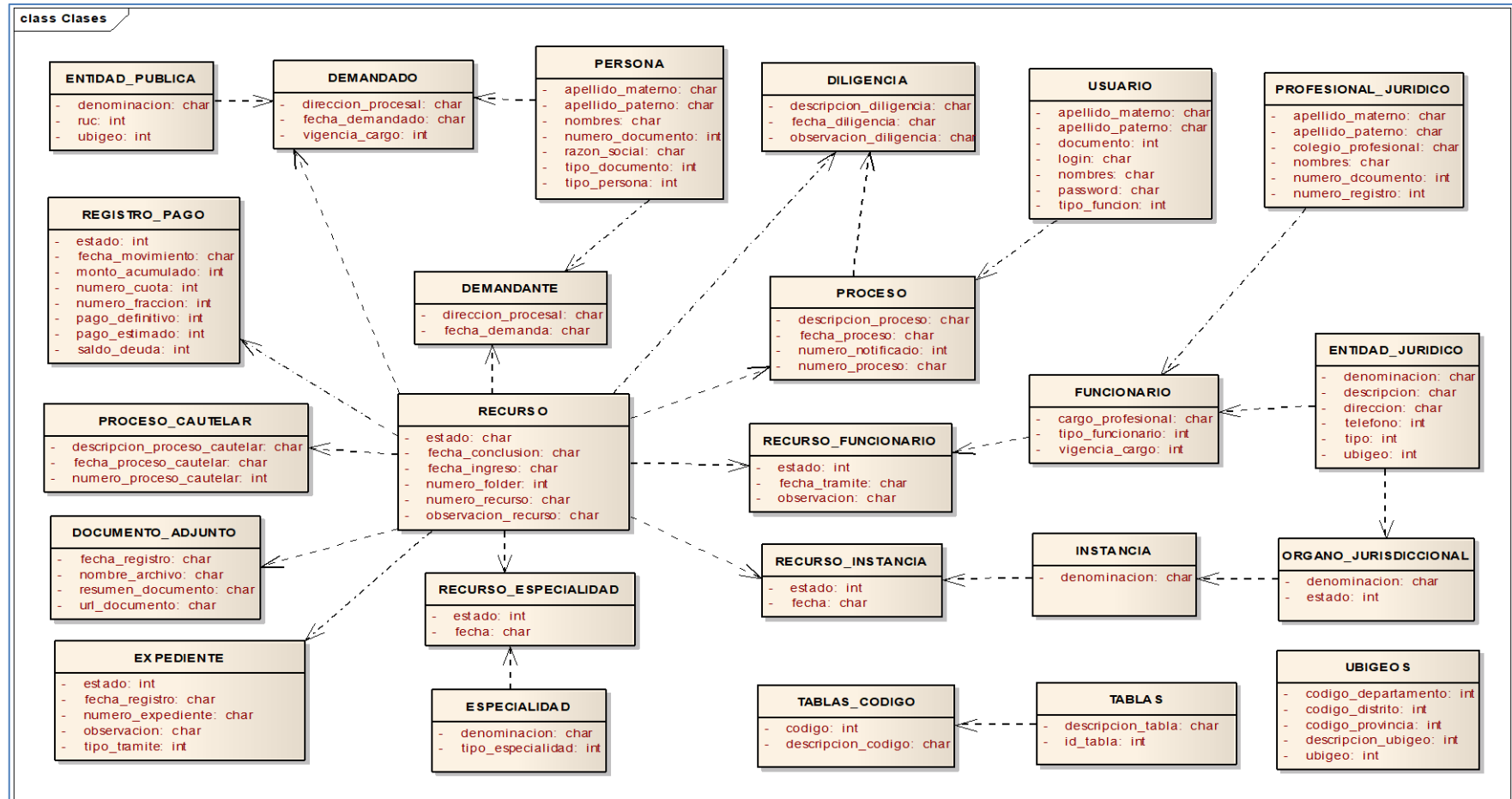


Figura n° 4.35: Diagrama de Robustez: Listar Procesos de un Trámite

4.1.4 HITO ETAPA 2: REVISIÓN DEL DISEÑO PRELIMINAR (PDR).

4.1.4.1 REVISION DE LA DESCRIPCION DE LOS CASOS DE USO

CASO DE USO	DESCRIPCIÓN
Listar Usuarios Del Sistema	<p>Curso Básico</p> <ol style="list-style-type: none"> 1) En la página principal del sistema el Usuario deberá hacer clic en el menú Administración y luego Usuario del Sistema. 2) El sistema muestra una Página para la Administración de Usuario, con opción de agregar usuario y el sistema hace una consulta a <u>entidad Usuario</u> mediante la <u>acción Listar Usuario</u> y lo muestra en una Vista de lista de Usuarios registrados los mismos que son administrables (editar y eliminar). <p>Curso Alterno</p> <ol style="list-style-type: none"> 1) El Usuario hace clic en la opción Editar usuario: El sistema muestra un formulario con los campos del usuario para poder actualizarlos. 2) El Usuario hace clic en la opción Agregar usuario: El sistema muestra un formulario para poder ingresar los datos del nuevo usuario. 3) El Usuario hace clic en la opción Salir: El sistema regresa a la página inicio del sistema
Registrar Usuario	<p>Curso Básico</p> <ol style="list-style-type: none"> 1) En la página principal el Usuario deberá hacer clic en el menú Administración y luego Usuario del Sistema. 2) El sistema muestra una página con opciones para la administración de usuario. 3) El Usuario hace clic en la opción Agregar. 4) El sistema muestra la vista Registrar Usuario, que posteriormente serán llenados. 5) El Usuario tiene la opción de buscar los datos de usuarios ya guardados en el sistema en la <u>entidad Persona</u> o ingresar datos del nuevo Usuario; El Usuario tiene la opción seleccionar el perfil desde de un select; el Usuario deberá ingresar el login que será igual al password; una vez completado la digitación en los campos de usuario, se procede con dar clic al botón guardar. 6) El sistema guarda la información en la <u>entidad Usuario</u>. <p>Curso Alterno</p> <ol style="list-style-type: none"> 1) El Usuario Administrador no ingresó todos los datos requeridos: El sistema muestra un mensaje indicando los campos que falta ingresar, se mantendrá la acción en la vista Registrar Usuario. 2) El Usuario Administrador ingresó datos incorrectos: El sistema muestra un mensaje indicando la característica de los datos a ingresar, se mantendrá la acción en la vista Registrar Usuario. 3) El Usuario Administrador hace clic en el botón salir: El sistema regresa a la Página de

	Administración de Usuario.
Agregar Expediente /Caso	<p>Curso Básico</p> <ol style="list-style-type: none"> 1) En la página principal el Usuario Trabajador deberá hacer clic en el menú Trámite Documentario y luego Administrar Trámite Sistema. 2) El Sistema muestra una vista (Administrar Trámite) con opciones para la Administración de Trámite. 3) El Usuario Trabajador hace clic en la opción Abrir Tramite. 4) El Sistema muestra una vista (Panel de administración del Trámite). 5) El Usuario Trabajador hace clic en la opción Abrir Expediente. 6) El Sistema muestra una vista(Expedientes del Trámite) 7) El Usuario Trabajador hace clic en la opción Agregar Nuevo expediente. 8) El Sistema muestra la vista (Registrar Nuevo Trámite). 9) El Usuario Trabajador ingresa los Datos Del nuevo Expediente, luego hace clic en Guardar Datos del Expediente. 10) El Sistema guarda la información del Trámite en la base de datos de forma correcta, se muestra un mensaje de éxito "Registro Completado" <p>Curso Alterno</p> <ol style="list-style-type: none"> 1) El Trabajador no ingresó todos los datos requeridos: El sistema muestra un mensaje "Indicando campo que falta ingresar". 2) El Trabajador ingresó datos incorrectos: El sistema muestra un mensaje indicando "Datos Incorrectos". 3) El Trabajador hace clic en el botón salir: El sistema regresa a una vista (Expedientes del Trámite).
Listar Expediente/ Caso	<p>Curso Básico</p> <ol style="list-style-type: none"> 1) En la página principal el Usuario Trabajador deberá hacer clic en el menú Trámite Documentario y luego Administrar Trámite Sistema. 2) El Sistema muestra una vista (Administrar Trámite) con opciones para la Administración de Trámite. 3) El Usuario Trabajador hace clic en la opción Abrir Tramite. 4) El Sistema muestra una vista (Panel de administración del Trámite). 5) El Usuario Trabajador hace clic en la opción Abrir Expediente. 6) El Sistema muestra una vista (Expedientes del Trámite) y a su vez una Lista Expedientes del Trámite. <p>Curso Alterno</p> <ol style="list-style-type: none"> 1). El Trabajador hace clic en la opción Acción Abrir

	<p>Trámite: El sistema realiza una acción de Abrir Trámite para cada expediente y se muestra una vista (Panel de mando de trámite).</p> <p>2). El Trabajador hace clic en la opción Registrar: El sistema muestra una vista de Registro de trámite</p> <p>3). El Trabajador hace clic en la opción Salir: El sistema regresa al estado por defecto donde las opciones de menú no están seleccionadas.</p>
<p>Listar Procesos de un expediente</p>	<p>Curso Básico</p> <ol style="list-style-type: none"> 1) En la página principal el Usuario Trabajador deberá hacer clic en el menú Trámite Documentario y luego Administrar Trámite Sistema. 2) El Sistema muestra una vista (Administrar Trámite) con opciones para la Administración de Trámite. 3) El Usuario Trabajador hace clic en la opción Abrir Trámite. 4) El Sistema muestra una vista (Panel de administración del Trámite). 5) El Usuario Trabajador hace clic en la opción Acto Procesal. 6) El Sistema muestra una vista (Procesos de Trámite) y a su vez una Lista De Los Procesos Del Trámite. <p>Curso Alterno</p> <ol style="list-style-type: none"> 1) El Trabajador hace clic en la opción Agregar: El sistema muestra una vista (Agregar un nuevo Proceso) para el expediente en elección. 2) El Trabajador hace clic en la opción Salir: El sistema regresa al estado por defecto donde las opciones de menú no están seleccionadas.

Cuadro n° 4.27. Revisión del caso de uso.

4.1.4.2 REVISIÓN DEL MODELO DE DOMINIO ACTUALIZADO

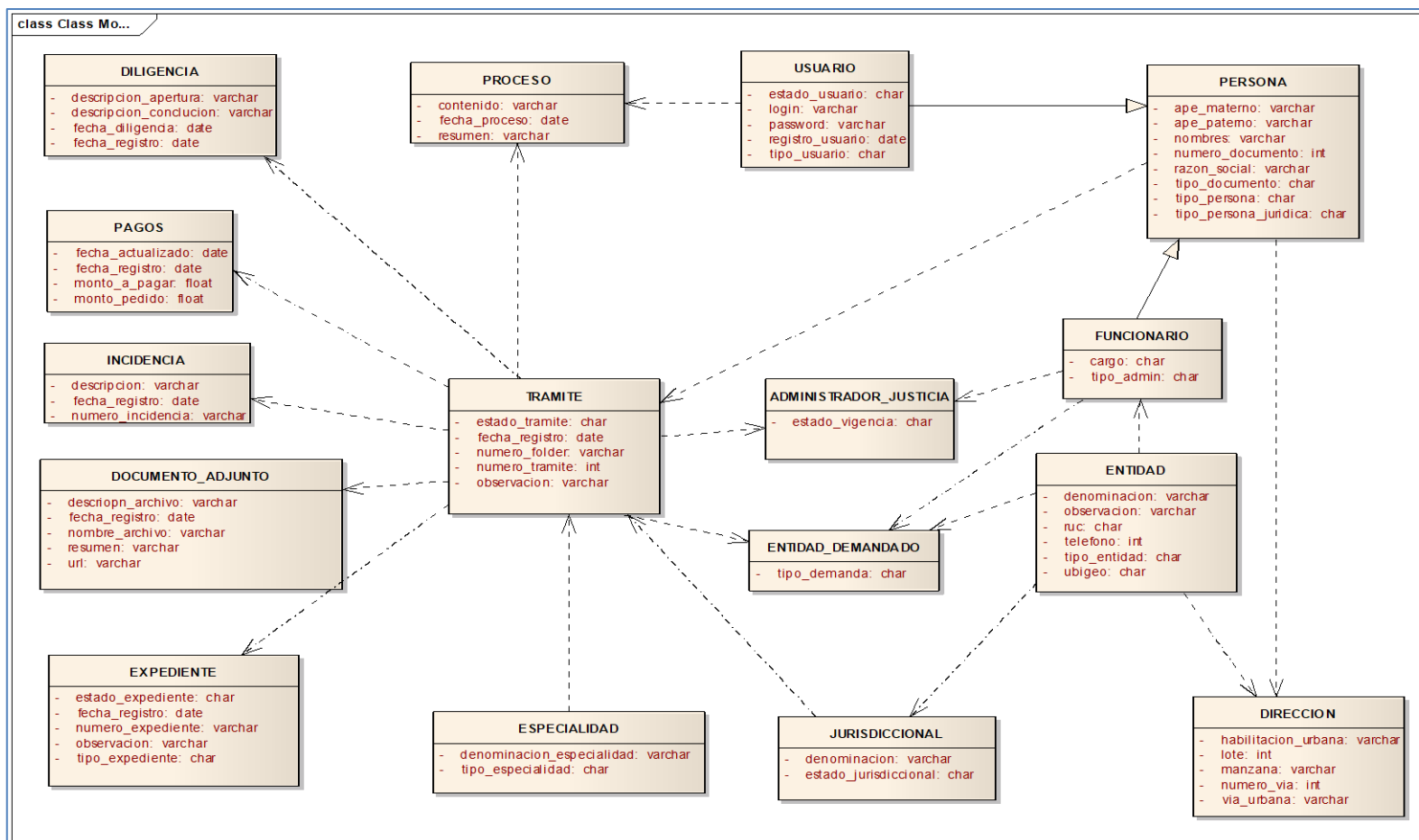


Figura n° 4.36: Modelo de dominio revisado

4.1.5 DISEÑO DETALLADO.

4.1.5.1 ARQUITECTURA TÉCNICA DEL FRAMEWORK CODEIGNITER

- **LA ESTRUCTURA DE CODEIGNITER**

El diseño orientado al rendimiento de este framework de desarrollo web se revela en su parca arquitectura, pues se basa en el patrón Modelo-Vista-Controlador (MVC). El principio fundamental que sustenta a la arquitectura de desarrollo MVC es la estricta separación entre el código y la presentación, gracias a una estructura modular de software y a la externalización del código PHP. Esta separación se realiza en estos tres grupos: el modelo (model), la vista (view) y el controlador (controller), que explicamos a continuación (Digital Guide, 2017):

El modelo representa la estructura de datos de una aplicación web desarrollada con CodeIgniter. Para ello, en el código fuente se definen las denominadas clases (“model classes”), que contienen funciones especiales con las cuales se puede recibir, insertar o actualizar la información de la base de datos (Digital Guide, 2017).

La vista es aquello que se le presenta al usuario final. Por lo general, se trata de un documento HTML en el cual se ha insertado contenido de forma dinámica con PHP, convirtiéndose en una especie de plantilla. CodeIgniter también permite definir fragmentos de una página web como la cabecera y el pie de página o páginas RSS como vista. Normalmente las aplicaciones web utilizan varias vistas, que toman su contenido desde el mismo modelo, de tal forma que es posible presentar diversas características del programa en vistas diferentes (Digital Guide, 2017).

El controlador media entre el modelo, la vista y cualquier otro recurso necesario para procesar una petición HTTP o generar una página web de forma dinámica. Este componente recibe las peticiones entrantes, valida la entrada, selecciona la vista deseada y le entrega el contenido que el modelo ha cargado desde una base de datos (Digital Guide, 2017).

En este esquema se aprecia claramente cómo se produce la interacción en la arquitectura MVC:

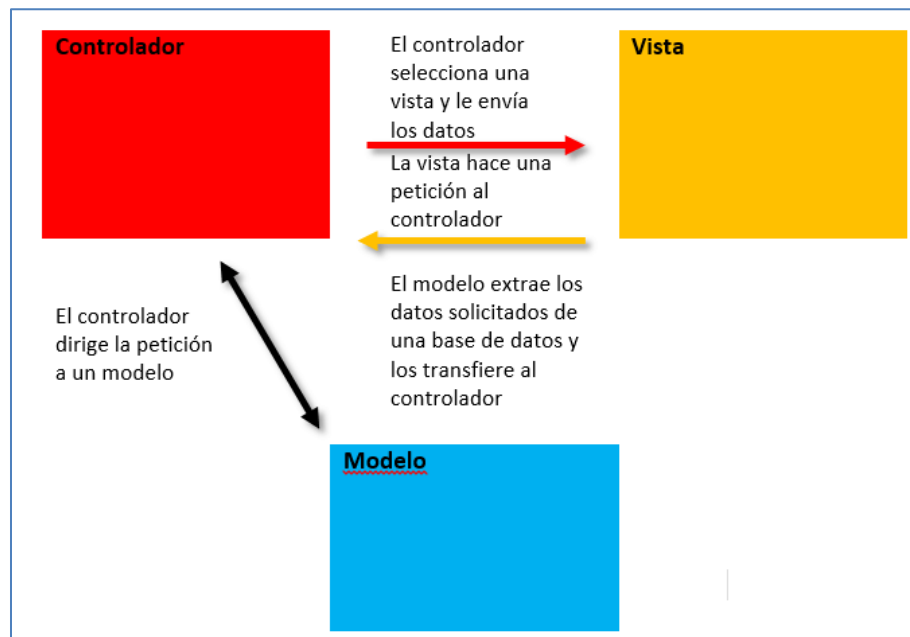


Figura n° 4.37: Interacción de los componentes modelo, vista y controlador de la arquitectura MVC

; Fuente: (Digital Guide, 2017)

La estructura MVC permite diseñar software de forma flexible, ya que se pueden substituir, editar y reutilizar los módulos individuales de programación muy fácilmente. Los cambios que se realizan en un componente no suelen tener ningún efecto en el código fuente de otros componentes, siempre y cuando estos cambios no tengan lugar en los puntos de contacto entre unos y otros (Digital Guide, 2017).

La mencionada división entre la lógica del programa y la presentación resulta en un código claro y bien estructurado. Esto es lo que hace que las aplicaciones web diseñadas según un patrón MVC estén consideradas como fáciles de mantener, puesto que, en caso de error o fallo, la fuente suele encontrarse en uno solo de los componentes. Esta separación también permite desarrollar la lógica y el diseño de forma independiente. Si los desarrolladores del backend y del frontend trabajan en paralelo, las aplicaciones se pueden terminar mucho más rápidamente (Digital Guide, 2017).

Aun cuando CodeIgniter utiliza el patrón MVC, los usuarios no están obligados a hacerlo por completo. Si el controlador y la vista son componentes obligatorios, no sucede lo mismo con el modelo, cuyo uso como vínculo con la base de datos es meramente opcional. Una aplicación realizada con CodeIgniter también puede realizarse, además, con una arquitectura MVC jerárquica (HMVC), que ampliaría el patrón MVC clásico con una lógica jerárquica (Digital Guide, 2017).

▪ EL FLUJO DE APLICACIÓN EN CODEIGNITER: SÍNTESIS FINAL

La siguiente ilustración resume el flujo de aplicación de una aplicación basada en CodeIgniter en siete pasos:

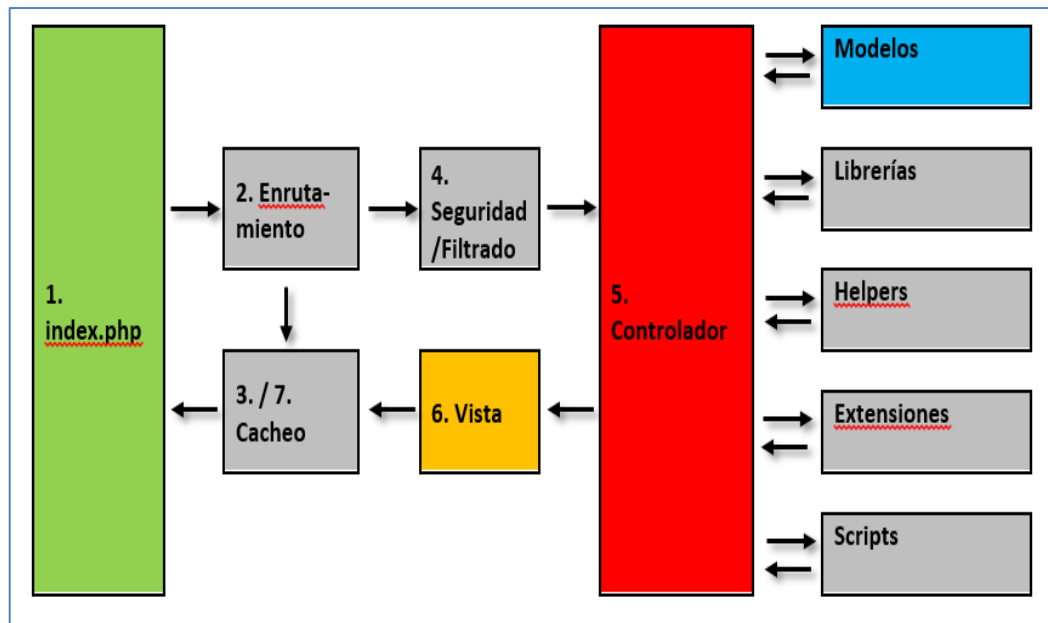


Figura n° 4.38: Ilustración esquemática del flujo de aplicación del framework PHP CodeIgniter;

Fuente (Digital Guide, 2017)

Y ahora, paso a paso:

- ✓ El archivo **index.php** sirve a CodeIgniter como primer controlador de las solicitudes HTTP entrantes. Es en este punto donde se inicializan todos los recursos básicos necesarios para ejecutar la aplicación (Digital Guide, 2017).
- ✓ En el **enrutamiento**, CodeIgniter comprueba qué acción se debe ejecutar. Para ello, la aplicación compara la dirección de la solicitud con las reglas de

enrutamiento comprendidas en el **routes.php** (Digital Guide, 2017).

- ✓ Al enrutamiento le sigue el **caching**. Si en la memoria caché se encuentra una respuesta adecuada a la petición, esta se envía directamente al navegador que la ha realizado. En caso contrario, se ejecuta el controlador indicado durante el enrutamiento tras activar la función de filtrado (Digital Guide, 2017).
- ✓ CodeIgniter contiene un filtro integrado que capta peticiones maliciosas. Antes de que la aplicación cargue un controlador ajustado a la solicitud, cada petición HTTP se somete a una prueba de seguridad (Digital Guide, 2017).
- ✓ Si la petición supera el filtrado se ejecuta el controlador. Este selecciona la vista adecuada y carga el modelo, así como todas las librerías, clases helper, plugins y scripts necesarios para poder responder la petición (Digital Guide, 2017).
- ✓ Tan pronto la **vista** ha recibido todos los datos relevantes, puede entregarlos al navegador (Digital Guide, 2017).
- ✓ Si se ha activado el **cacheo**, CodeIgniter conserva temporalmente los datos salientes para responder peticiones repetitivas directamente (Digital Guide, 2017).

4.1.5.2 ARQUITECTURA DE LA APLICACIÓN PHP

PHP (acrónimo de "PHP: Hypertext Preprocessor") es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. Hay que entender primero como funciona la solicitud de páginas en un navegador para comenzar a programar en PHP (Tutoriales Programación Ya. 2019).

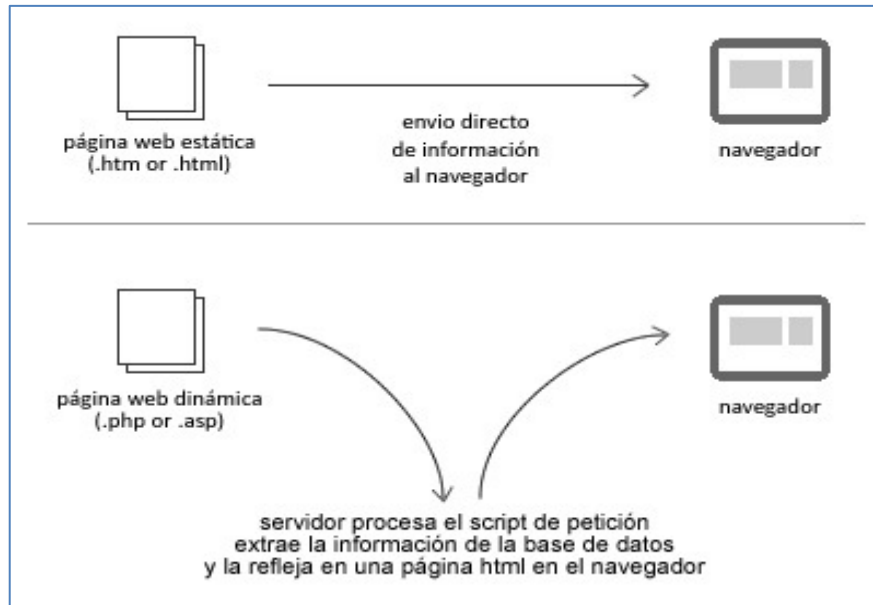


Figura n° 4.39: Páginas estáticas vs páginas dinámicas; Fuente: (Tutoriales Programación Ya. 2019).

✓ COMUNICACIÓN ENTRE EL CLIENTE Y EL SERVIDOR SIN PHP:

1. Escribimos en la barra del navegador la dirección y el archivo a solicitar (Tutoriales Programación Ya. 2019).
2. El web browser (navegador) envía el mensaje a través de Internet a la computadora, por ejemplo `www.lanacion.com/pagina1.htm` solicitando la página (archivo) `pagina1.htm` (Tutoriales Programación Ya. 2019).
3. El web server (servidor web, que puede ser el Apache, IIS, etc.) que es un programa que se ejecuta en la máquina `www.lanacion.com`, recibe el mensaje y lee el archivo solicitado desde el disco duro (Tutoriales Programación Ya. 2019).
4. El servidor web envía el archivo solicitado por el navegador tal cual está en el disco duro (Tutoriales Programación Ya. 2019).
5. El navegador muestra en pantalla el archivo que envió el servidor web (Tutoriales Programación Ya. 2019).

Este proceso siempre es el mismo cuando hablamos de páginas estáticas (páginas que no cambian), cualquiera sea el cliente que solicita la página el contenido siempre será el mismo.

La única forma que el contenido del archivo cambie es que el administrador de ese sitio web edite el contenido del archivo pagina1.htm y haga modificaciones (Tutoriales Programación Ya. 2019).

✓ **COMUNICACIÓN ENTRE EL CLIENTE Y EL SERVIDOR CON PHP:**

1. Escribimos en la barra del navegador la dirección y el archivo a solicitar (Tutoriales Programación Ya. 2019).
2. El web browser (navegador) envía el mensaje a través de Internet a la computadora llamada www.lanacion.com solicitando la página (archivo) pagina1.php (Tutoriales Programación Ya. 2019).
3. El web server (servidor web, que puede ser el Apache, IIS, etc.), recibe el mensaje y al ver que la extensión es "php" solicita al intérprete de PHP (que es otro programa que se ejecuta en el servidor web) que le envíe el archivo (Tutoriales Programación Ya. 2019).
4. El intérprete PHP lee desde el disco el archivo pagina1.php (Tutoriales Programación Ya. 2019).
5. El intérprete PHP ejecuta los comandos contenidos en el archivo y eventualmente se comunica con un gestor de base de datos (ejemplos de ellos pueden ser MySql, Oracle, Informix, SQL Server, etc.) (Tutoriales Programación Ya. 2019).
6. Luego de ejecutar el programa contenido en el archivo envía éste al servidor web (Tutoriales Programación Ya. 2019).
7. El servidor web envía la página al cliente que la había solicitado (Tutoriales Programación Ya. 2019).
8. El navegador muestra en pantalla el archivo que envió el servidor web (Tutoriales Programación Ya. 2019).

Para aquellas personas que conocen otros lenguajes de programación (C - Java - C# - Pascal etc.) la salida de los resultados normalmente es la pantalla, en PHP la salida es la página HTML (luego ésta viaja por internet y llega al cliente que la solicitó) (Tutoriales Programación Ya. 2019).

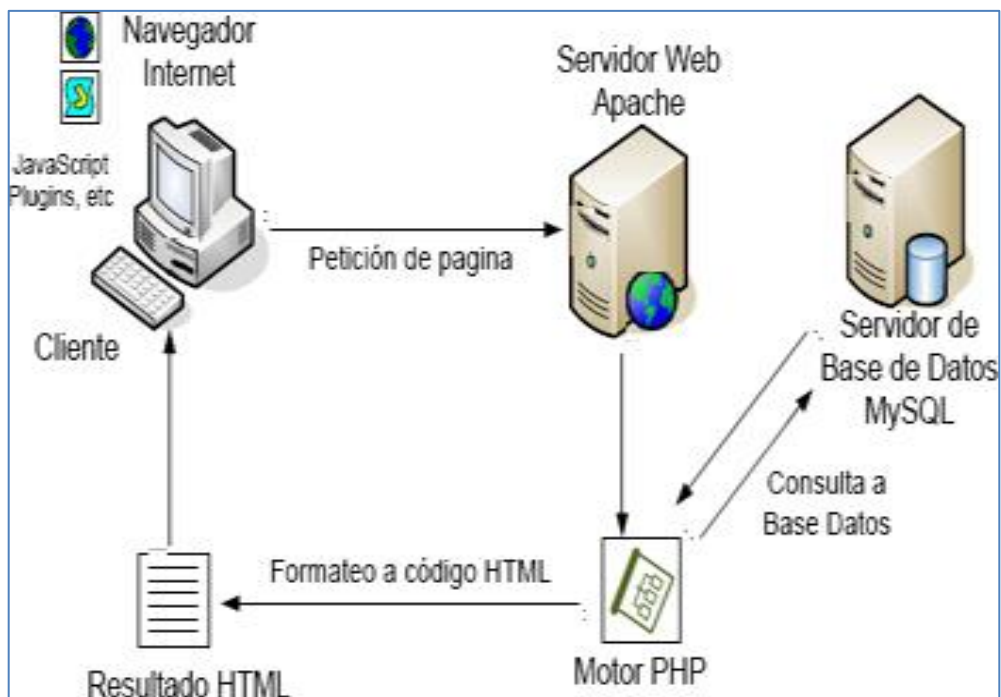


Figura n° 4.40: Arquitectura web en PHP; Fuente: (Tutoriales Programación Ya. 2019).

4.1.5.3 ARQUITECTURA TÉCNICA POR CAPAS

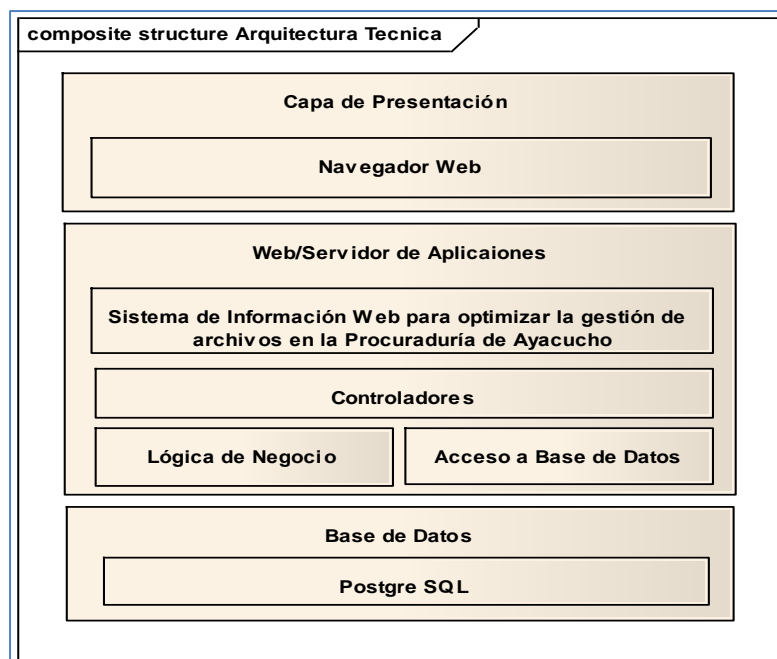


Figura n° 4.41: Arquitectura técnica por capas

4.1.5.4 DIAGRAMA DE COMPONENTES

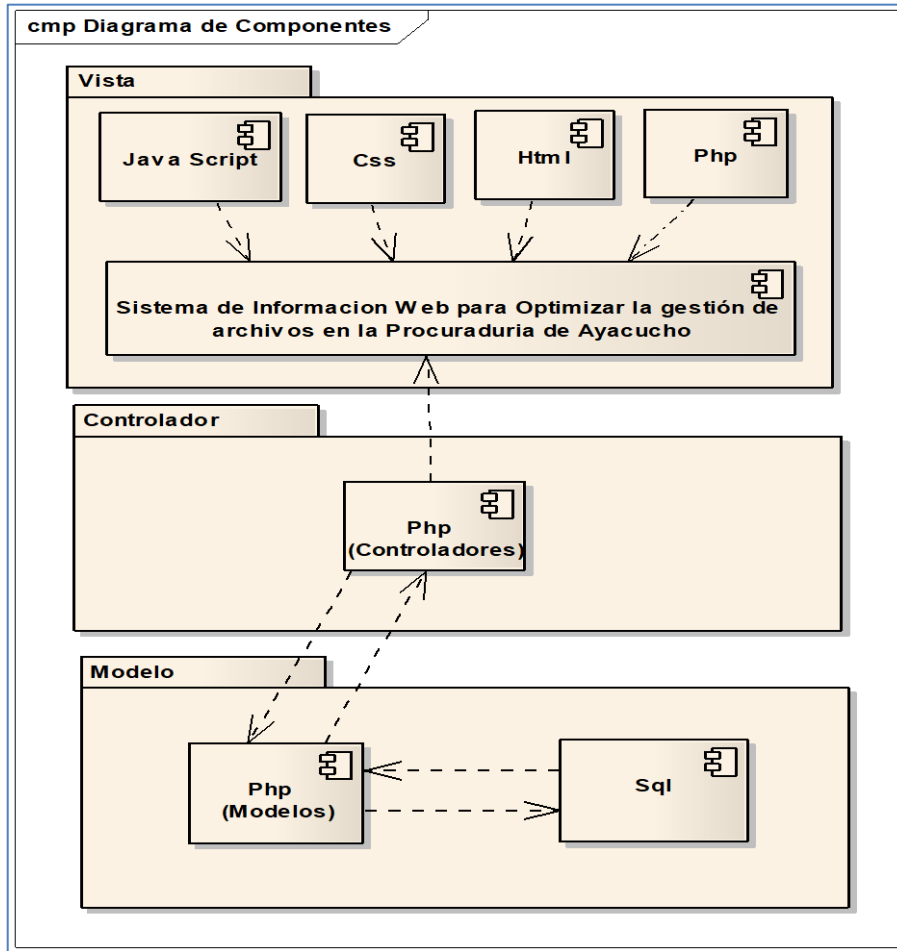


Figura n° 4.42: Diagrama de Componentes

4.1.5.5 DIAGRAMA DE DESPLIEGUE

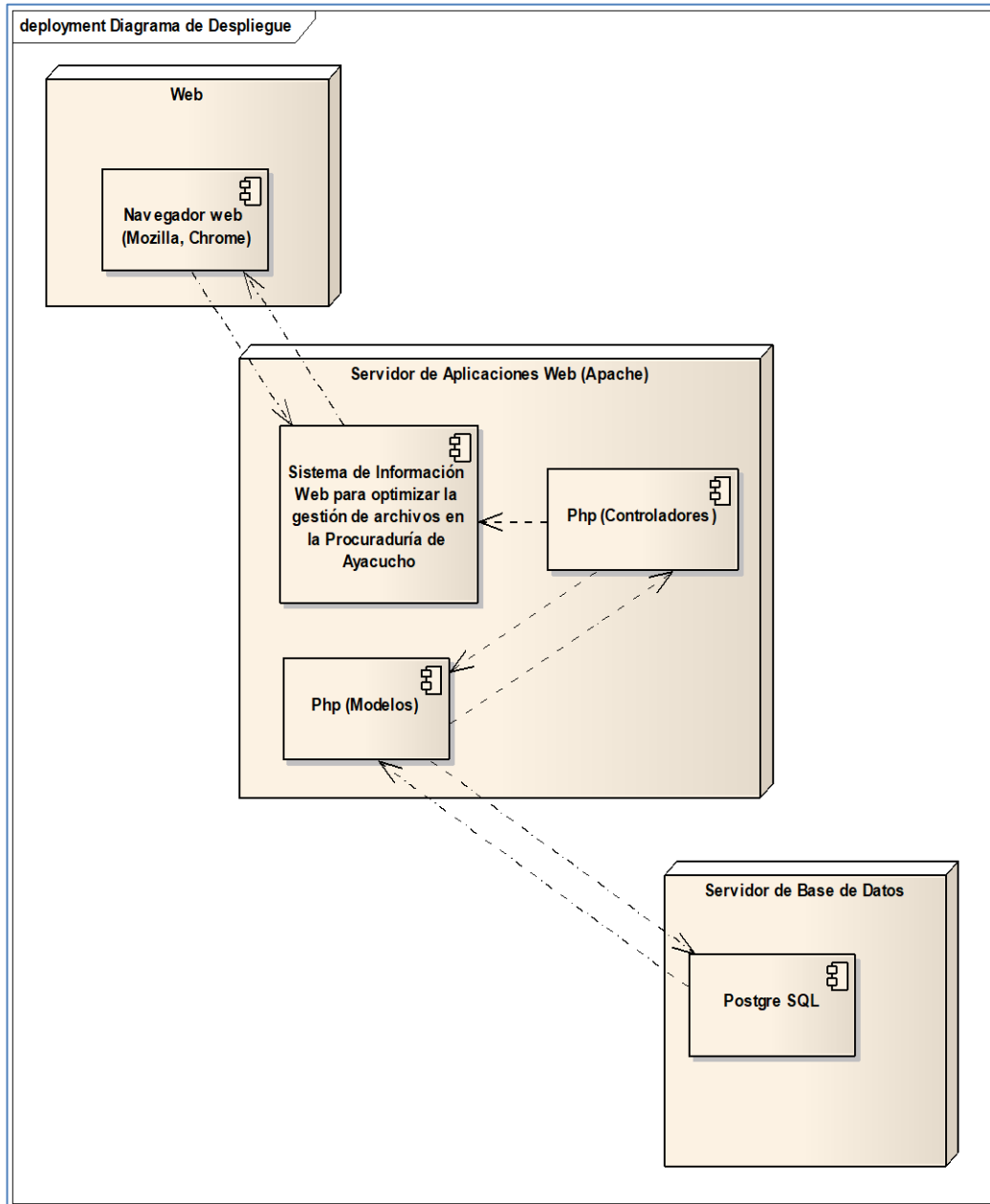


Figura n° 4.43: Diagrama de Despliegue

4.1.6 DIAGRAMA DE SECUENCIA

4.1.6.1 DIAGRAMA DE SECUENCIA “Registrar Usuario”

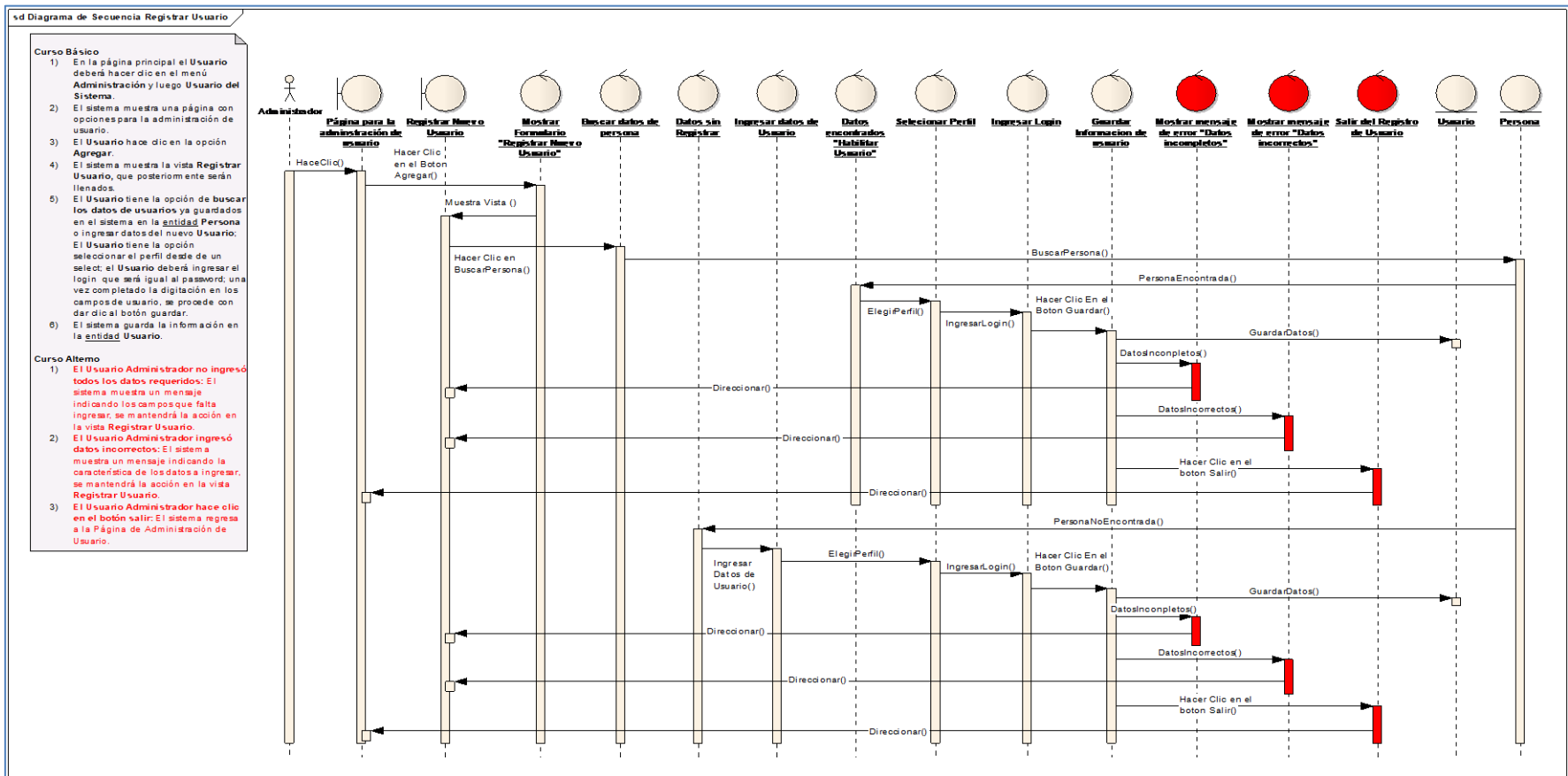


Figura n° 4.44: Diagrama De Secuencia “Registrar Usuario”

4.1.6.2 DIAGRAMA DE SECUENCIA “Listar Usuarios Del Sistema”

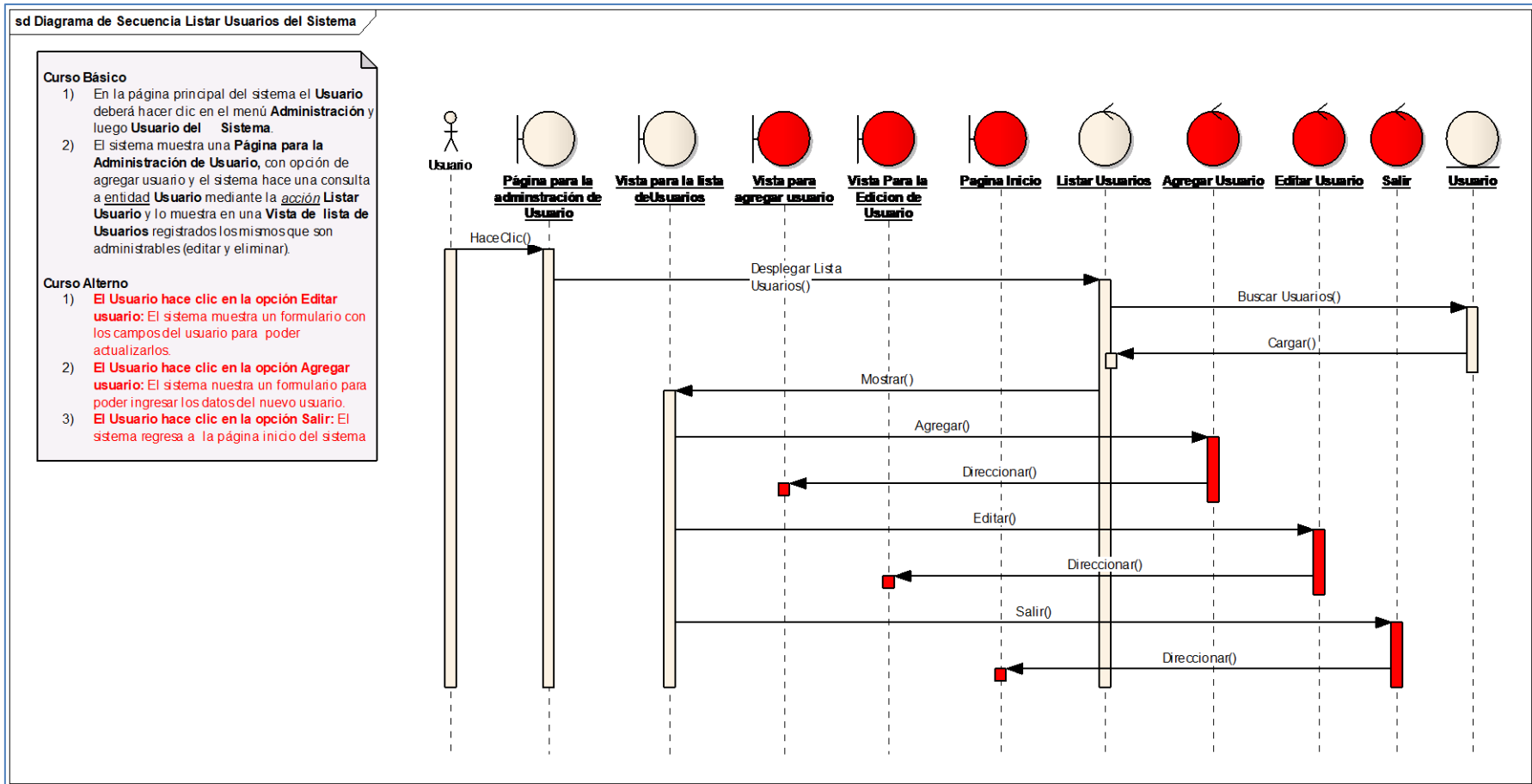


Figura n° 4.45: Diagrama de Secuencia “Listar Usuarios Del Sistema”

4.1.6.3 DIAGRAMA DE SECUENCIA “Agregar Expediente/ Caso”

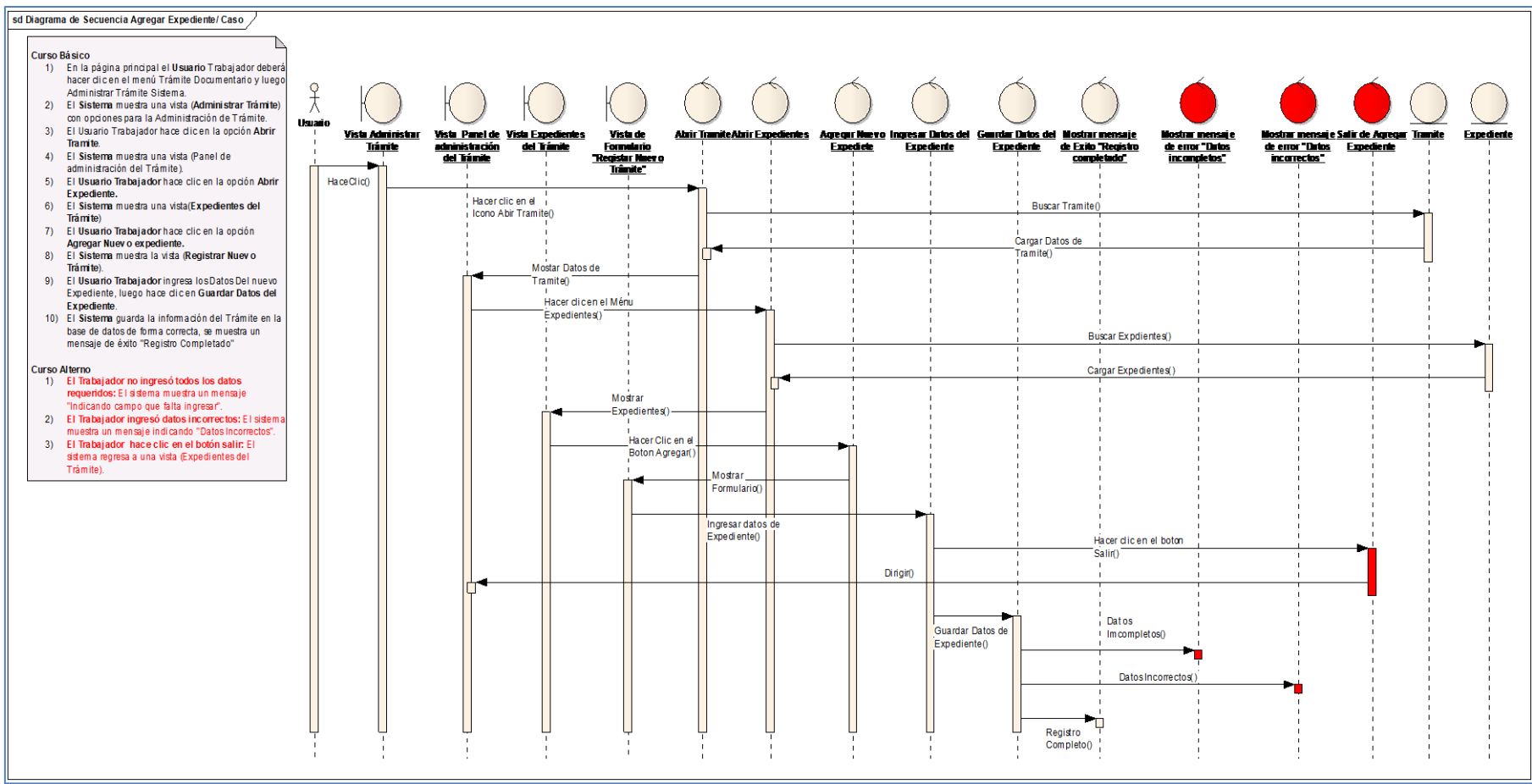


Figura n° 4.46: Diagrama De Secuencia “Agregar Expediente/ Caso”

4.1.6.4 DIAGRAMA DE SECUENCIA “Listar Expediente/ Caso”

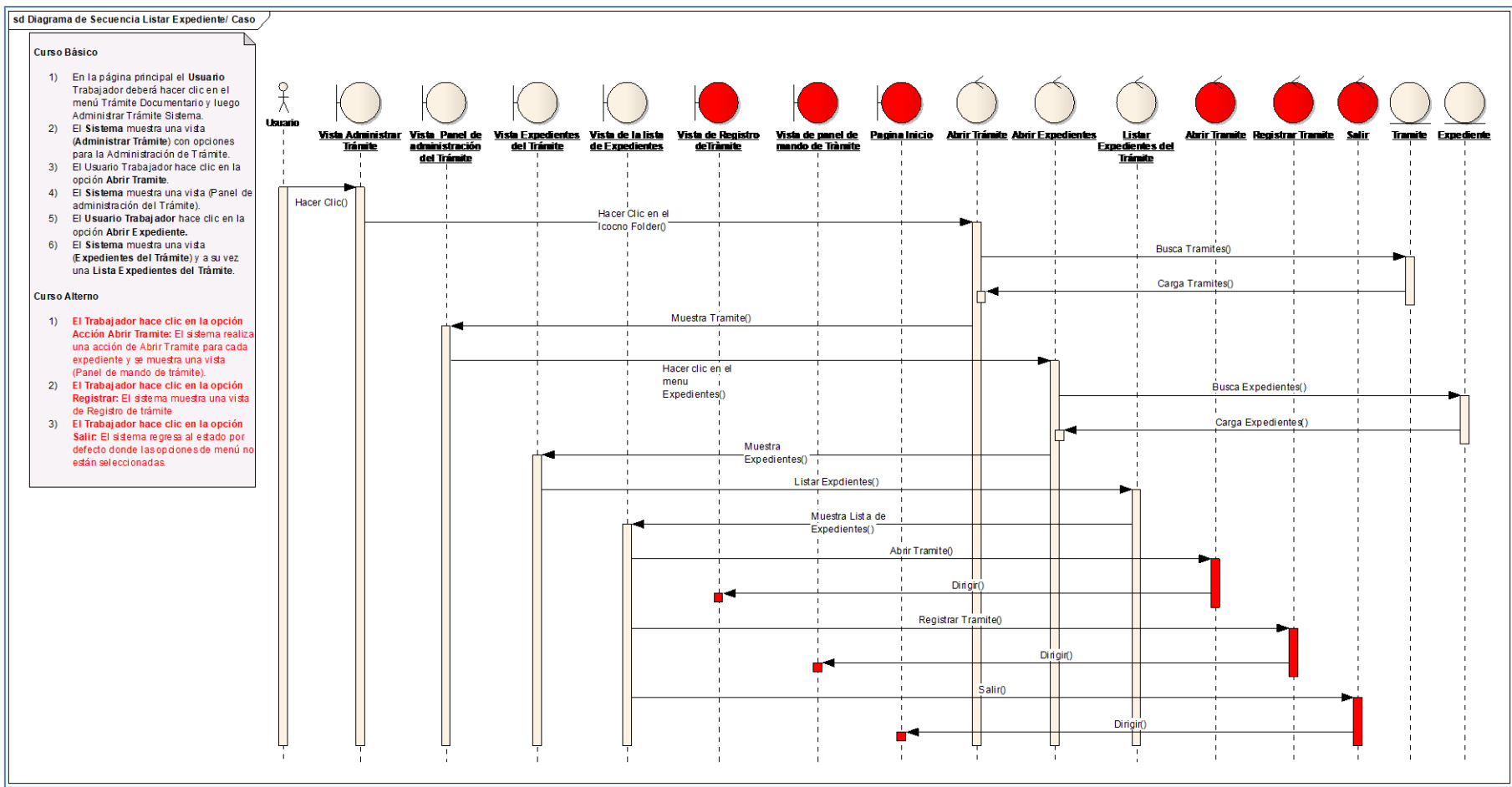


Figura n° 4.47: Diagrama De Secuencia “Listar Expediente/ Caso”

4.1.6.5 DIAGRAMA DE SECUENCIA “Listar Procesos de un expediente”

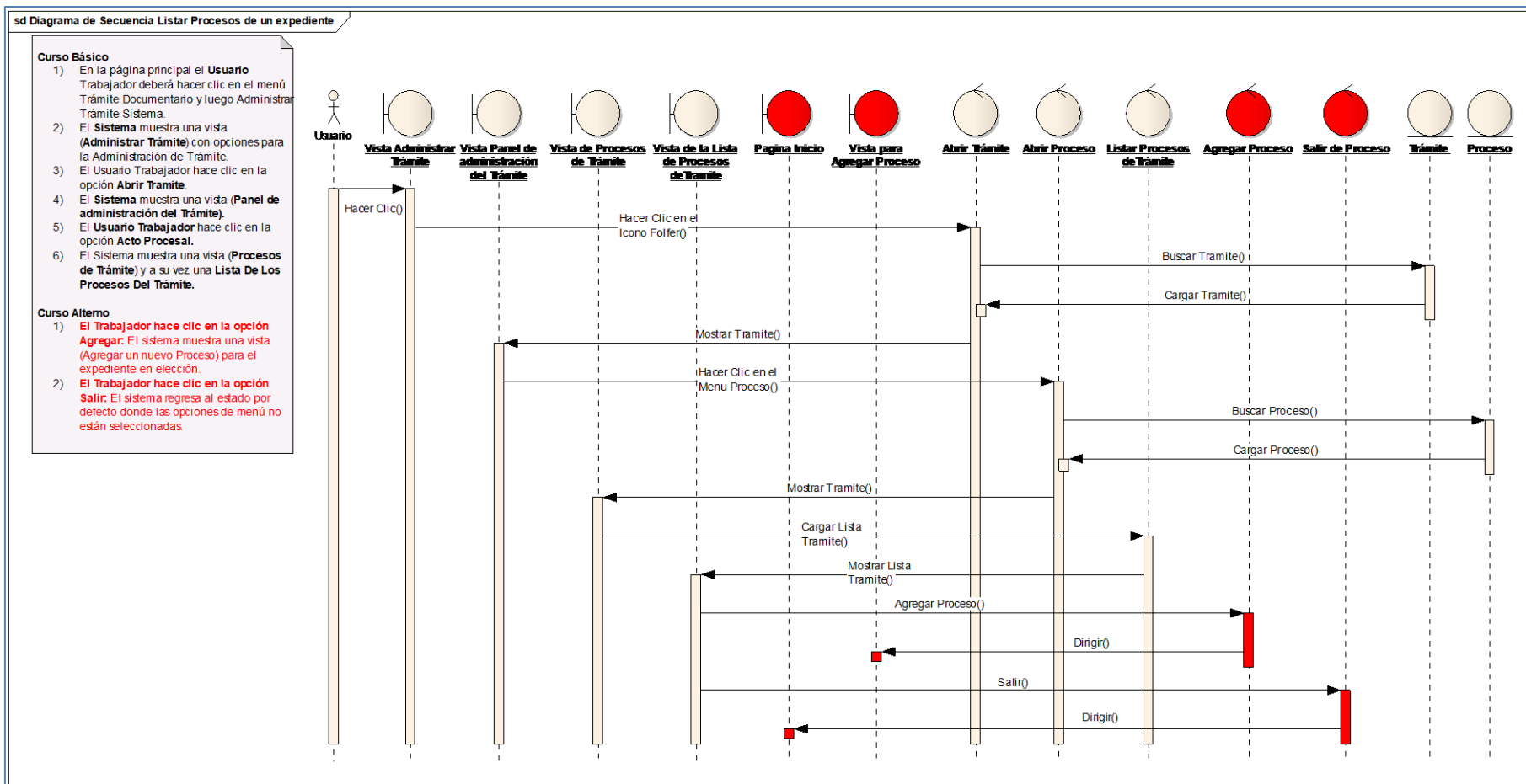


Figura n° 4.48: Diagrama De Secuencia “Listar Procesos de un expediente”

4.1.7 HITO ETAPA 3: REVISIÓN DEL DISEÑO DETALLADO.

No se encontraron cambios en los diagramas de secuencia, y se pasa a diseñar el diagrama de clases

4.1.7.1 DIAGRAMA DE CLASES DE DISEÑO

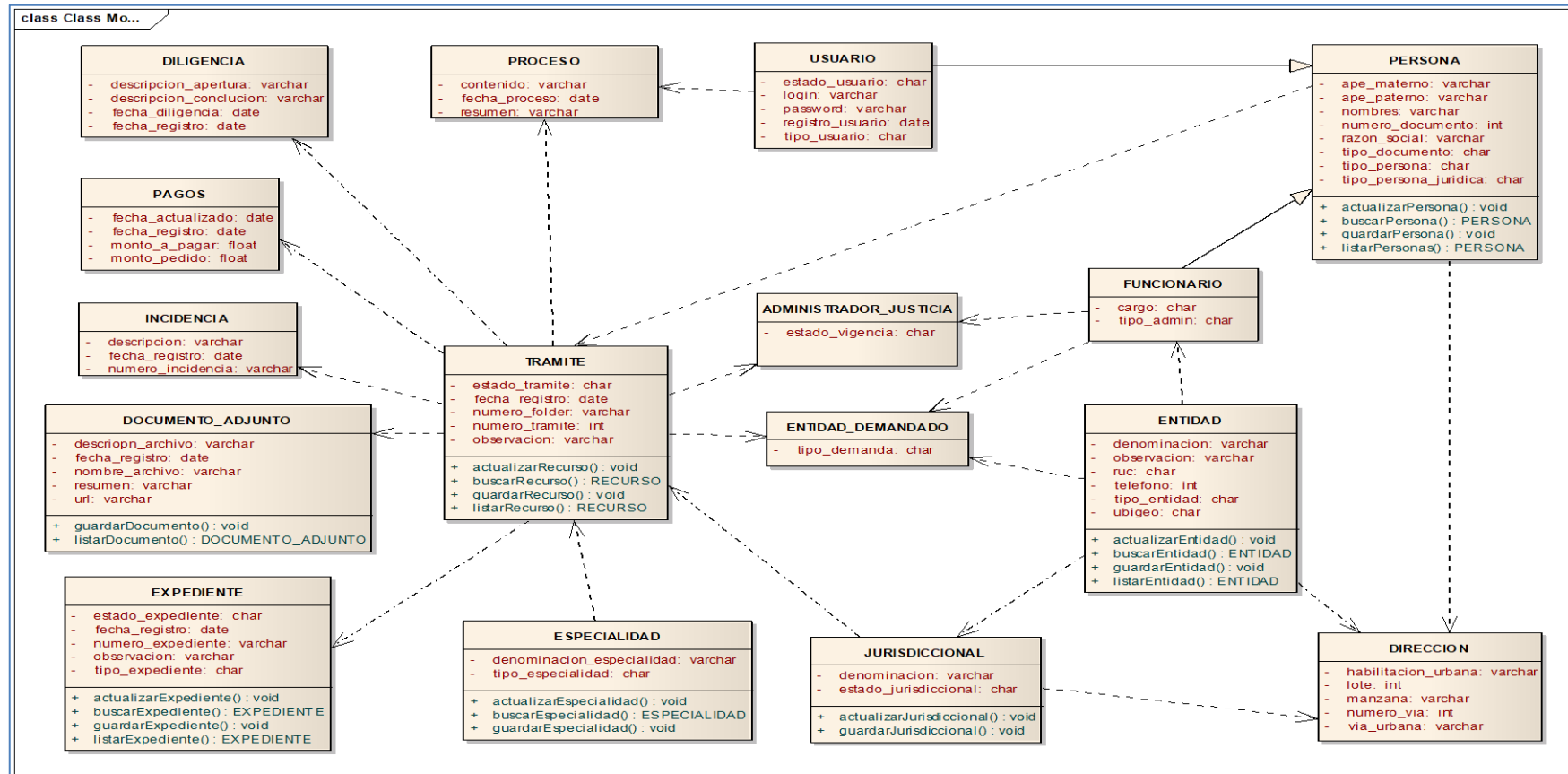


Figura n° 4.49 Diagrama de clases.

4.1.7.2 DIAGRAMA DEL MODELO LÓGICO DE LA BASE DE DATOS GENERAL

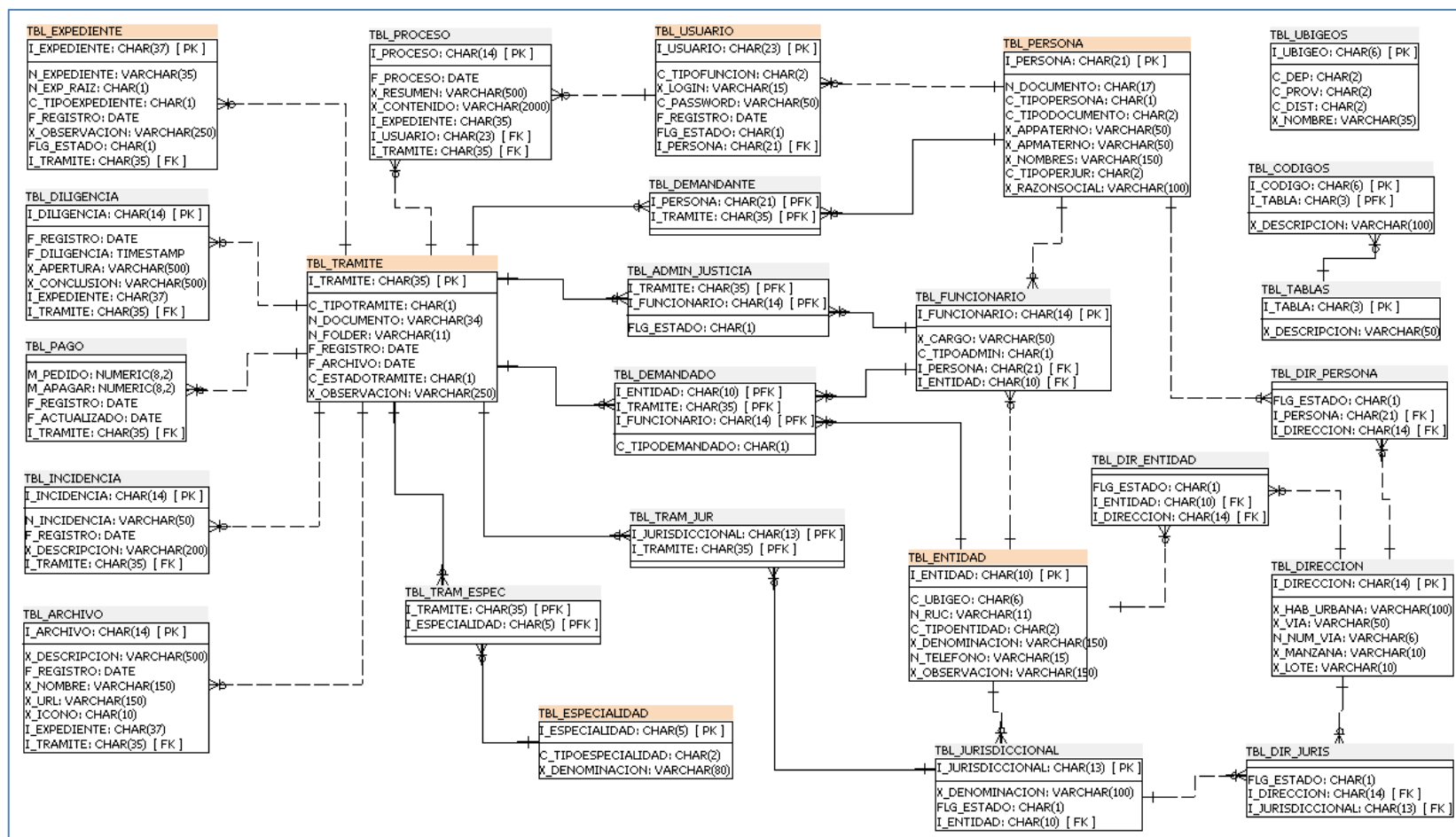


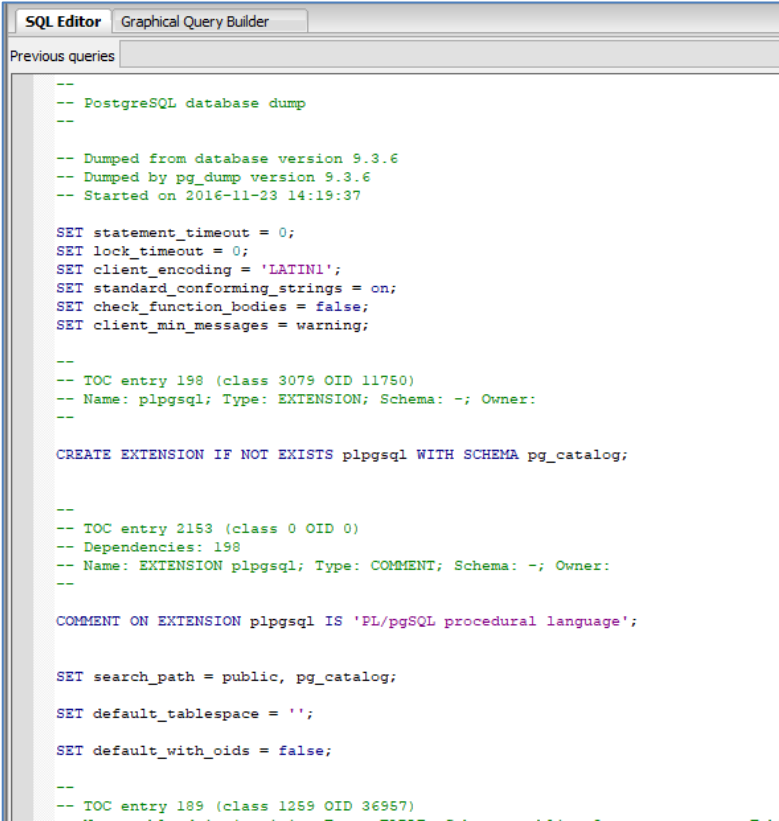
Figura n° 4.50 Diagrama Lógico de la Base de Datos.

4.1.8 IMPLEMENTACIÓN

Para la implementación de los casos de uso se utilizó el lenguaje PHP con el IDE de desarrollo NetBeans.

El Framework que se utilizó es Codeigniter, para la implementación de la base de datos se utilizó el sistema modelado de datos pgModeler y por el gestor de base de datos Postgresql, el modelado de software fue desarrollado con el software de modelador de sistemas Enterprise Architect v7.3 utilizando como herramienta de modelado del método Iconix, para las pruebas unitarias se utilizó PHPUnit”, tomadas en cuenta por su versatilidad en las aplicaciones web.

4.1.8.1 CREACIÓN DE LA BASE DE DATOS



```
SQL Editor Graphical Query Builder
Previous queries

--
-- PostgreSQL database dump
--

-- Dumped from database version 9.3.6
-- Dumped by pg_dump version 9.3.6
-- Started on 2016-11-23 14:19:37

SET statement_timeout = 0;
SET lock_timeout = 0;
SET client_encoding = 'LATIN1';
SET standard_conforming_strings = on;
SET check_function_bodies = false;
SET client_min_messages = warning;

--
-- TOC entry 198 (class 3079 OID 11750)
-- Name: plpgsql; Type: EXTENSION; Schema: -; Owner:
--

CREATE EXTENSION IF NOT EXISTS plpgsql WITH SCHEMA pg_catalog;

--
-- TOC entry 2153 (class 0 OID 0)
-- Dependencies: 198
-- Name: EXTENSION plpgsql; Type: COMMENT; Schema: -; Owner:
--

COMMENT ON EXTENSION plpgsql IS 'PL/pgSQL procedural language';

SET search_path = public, pg_catalog;

SET default_tablespace = '';

SET default_with_oids = false;

--
-- TOC entry 189 (class 1259 OID 36957)
```

Figura n° 4.51 Creación de Base de Datos (parte 1).


```

SQL Editor Graphical Query Builder
Previous queries

CREATE TABLE tbl_codigos (
  i_codigo character varying(6) NOT NULL,
  i_tabla character(3) NOT NULL,
  x_descripcion character varying(100) NOT NULL
);

ALTER TABLE public.tbl_codigos OWNER TO postgres;

--
-- TOC entry 187 (class 1259 OID 36947)
-- Name: tbl_cuantia_monetaria; Type: TABLE; Schema: public; Owner: postgres; Tablespace:
--

CREATE TABLE tbl_cuantia_monetaria (
  i_cuantia character(14) NOT NULL,
  m_pedido numeric(8,2),
  m_apagar numeric(8,2),
  f_registro date,
  f_actualizado date,
  m_acumulado numeric(8,2),
  i_tramite character varying(35) NOT NULL
);

ALTER TABLE public.tbl_cuantia_monetaria OWNER TO postgres;

--
-- TOC entry 185 (class 1259 OID 36941)
-- Name: tbl_deman_juridica; Type: TABLE; Schema: public; Owner: postgres; Tablespace:
--

CREATE TABLE tbl_deman_juridica (
  i_persona character varying(21) NOT NULL,
  i_demandado character(14) NOT NULL,
  i_entidad character(10) NOT NULL,
  flg_estado_rep character(1) NOT NULL
);

```

Figura nº 4.52 Creación de Base de Datos (parte 2).

```

SQL Editor Graphical Query Builder
Previous queries

-- TOC entry 189 (class 1259 OID 36957)
-- Name: tbl_admin_justicia; Type: TABLE; Schema: public; Owner: postgres; Tablespace:
--

CREATE TABLE tbl_admin_justicia (
  i_tramite character varying(35) NOT NULL,
  i_funcionario character(14) NOT NULL,
  flg_estado character(1)
);

ALTER TABLE public.tbl_admin_justicia OWNER TO postgres;

--
-- TOC entry 195 (class 1259 OID 36993)
-- Name: tbl_archivo; Type: TABLE; Schema: public; Owner: postgres; Tablespace:
--

CREATE TABLE tbl_archivo (
  i_archivo character(14) NOT NULL,
  x_descripcion character varying(500),
  f_registro date NOT NULL,
  x_nombre character varying(150) NOT NULL,
  x_url character varying(150) NOT NULL,
  x_icono character varying(10),
  i_expediente character(37),
  i_tramite character varying(35) NOT NULL
);

ALTER TABLE public.tbl_archivo OWNER TO postgres;

--
-- TOC entry 172 (class 1259 OID 36882)
-- Name: tbl_codigos; Type: TABLE; Schema: public; Owner: postgres; Tablespace:
--

```

Figura nº 4.53 Creación de Base de Datos (parte 3).

4.1.8.2 ESTRUCTURA DEL PROYECTO

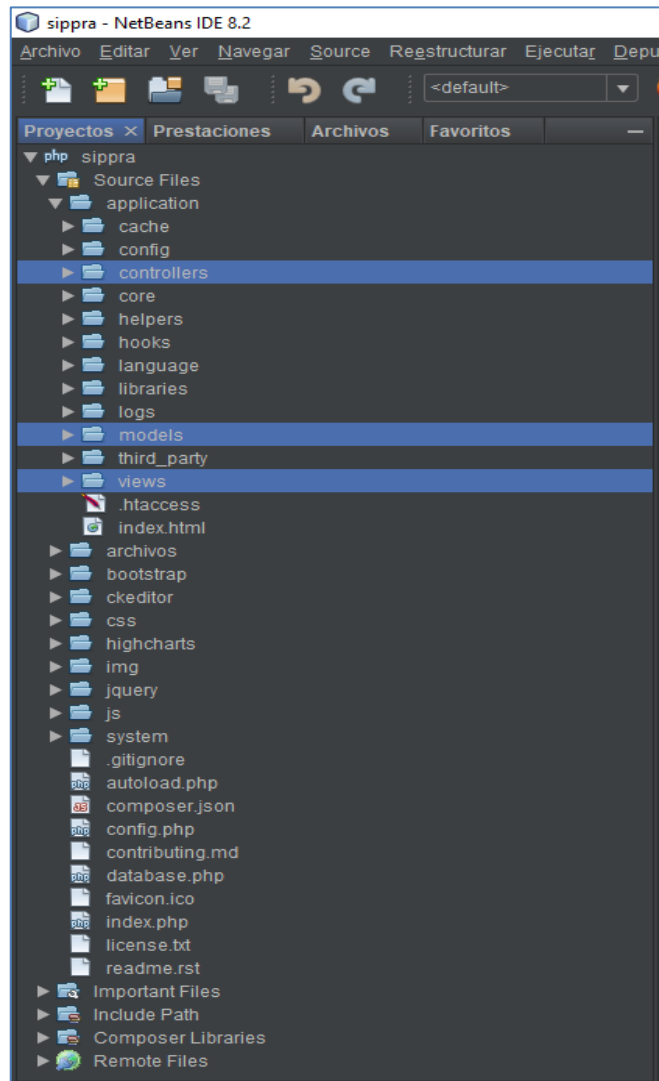


Figura n° 4.54 Estructura del proyecto.

4.1.8.3 IMPLEMENTACIÓN DE LAS VISTAS

C_sistema
<pre> <?php class C_sistema extends CI_Controller { public function __construct() { parent::__construct(); \$this->load->model('auxiliar/M_sistema', 'sistema'); \$this->load->model('M_entidad', 'entidad'); \$this->load->model('M_tramite', 'tramite'); } public function ingresoAlSistema() { \$data['contenido'] = 'auxiliar/sistema/v_ingresoAlSistema'; \$data['titulo'] = 'Acceso al Sistema SIPRA'; \$this->load->view('frontend', \$data); } </pre>

Cuadro n° 4.28 Controlador C_sistema.

Frontend
<pre> <?php defined('BASEPATH') OR exit('No direct script access allowed'); \$this->load->view('head'); \$this->load->view('header'); \$this->load->view('contenido'); \$this->load->view('footer'); </pre>

Cuadro n° 4.29 Vista frontend.

head
<pre> <?php defined('BASEPATH') OR exit('No direct script access allowed'); ?> <!DOCTYPE html> <html lang="en"> <head> <meta name="viewport" content="width=device-width, initial-scale=1"> <meta charset="utf-8" name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no"> <link href="<?= base_url('bootstrap/css/bootstrap.min.css') ?>" rel="stylesheet" media="screen"> <link href="<?= base_url('jquery/jquery-ui.min.css') ?>" rel="stylesheet" media="screen"> <link href="<?= base_url('jquery/jquery-ui.timepicker.css') ?>" rel="stylesheet" media="screen"> <link href="<?= base_url('css/css_portal.css') ?>" rel="stylesheet" media="screen"> <link href="<?= base_url('css/css_menu.css') ?>" rel="stylesheet" media="screen"> <link href="<?= base_url('css/estilos.css') ?>" rel="stylesheet" media="screen"> <link href="<?= base_url('css/calendario.css') ?>" rel="stylesheet" media="screen"> <link href="<?= base_url('css/menu_vertical.css') ?>" rel="stylesheet" media="screen"> <link href="<?= base_url('css/css_persona.css') ?>" rel="stylesheet" media="screen"> <script src="<?= base_url('ckeditor/ckeditor.js') ?>" type="text/javascript"></script> <link rel="shortcut icon" href="<?= base_url('favicon.ico') ?>"> <link href='http://fonts.googleapis.com/css?family=Lobster' rel='stylesheet' type='text/css'> <link href='http://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'> <link href='http://fonts.googleapis.com/css?family=Dancing+Script' rel='stylesheet' type='text/css'> <link href='http://fonts.googleapis.com/css?family=Monoton' rel='stylesheet' type='text/css'> <link href='http://fonts.googleapis.com/css?family=Just+Another+Hand' rel='stylesheet' type='text/css'> <link href='http://fonts.googleapis.com/css?family=Tangerine:400,700' rel='stylesheet' type='text/css'> <link href='http://fonts.googleapis.com/css?family=Rye' rel='stylesheet' type='text/css'> <title>SIPRA SISTEMA INFORMATICO DE LA PROCURADURIA REGIONAL DE AYACUCHO</title> </head> <body> </pre>

Cuadro n° 4.30 Vista Head.

header
<pre> <?php defined('BASEPATH') OR exit('No direct script access allowed'); ?> <nav class="navbar navbar-default" style="padding-bottom: 1px; margin-bottom: 1px;"> </pre>

```

<div class="container-fluid">

  <div class="row color-header">
    <div class="col-md-1">
      <div>
        
      </div>
    </div>

    <div class="col-md-10">
      <?php if (isset($_SESSION['i_usuario'])) { ?>
        <div class="row">
          <div class="col-md-9">
            <div class="color-procuraduria-superior" style="text-align: center;">
              <h4 class="titulo_pagina" style="color: #FFFFFF;margin: 4px;">CONSEJO DE DEFENSA JURÍDICA DEL
ESTADO</h4>
            </div>
            <div>
              <h4 class="titulo_pagina" style="text-align: center; margin: 5px;"><?php if (isset($_SESSION['observacion']))
echo $_SESSION['observacion']; ?></h4>
              <h5 class="sub_titulo_pagina" style="text-align: center; margin: 5px;"><?php if
(isset($_SESSION['denominacion'])) echo $_SESSION['denominacion']; ?></h5>
            </div>
          </div>
          <div class="col-md-3">
            <div class="table-responsive" style="padding: 25px 0 0 0px;">
              <table width="100%">
                <tr>
                  <td colspan="1" class="js_dato" style="font-weight:bold;font-size: 11px;text-align: right; color: #3c3c3c;">
                    <span style="color: #902b2b;" class="glyphicon glyphicon-user"></span><?php if
(isset($_SESSION['i_usuario'])) echo ' ' . $_SESSION['usuario']; ?>
                  </td>
                </tr>
                <tr>
                  <td class="js_dato" style="font-size: 11px;text-align: right;color: #3c3c3c;">
                    <?php if (isset($_SESSION['descripcion_perfil'])) echo 'Perfil : ' . $_SESSION['descripcion_perfil']; ?>
                  </td>
                </tr>
                <tr>
                  <td style="text-align: right;"><hr class="color-linea-delgada"></td>
                </tr>
                <tr>
                  <td class="js_dato" style="text-align: right;">
                    <span style="color: #902b2b;" class="glyphicon glyphicon-book">
                    </span><? anchor('c_sistema/mostrarManualDeUsuario/', 'Manual de Usuario de SIPRA'); ?>
                  </td>
                </tr>
              </table>
            </div>
          </div>
        </div>
        <div class="col-md-offset-0 col-md-12">
          <nav class="navbar navbar-inverse menu_horizontal" role="navigation">
            <? menu_header(); ?>
          </nav>
        </div>
      </div>
      <?php } ?>
    </div>

    <div class="col-md-1">
      <div>
        
      </div>
    </div>
  </div>
</nav>

```

Cuadro nº 4.31 Vista Header.

Contenido = auxiliar/sistema/v_ingresoAlSistema

```

<? form_open('c_sistema/logueo', array('class' => 'form-horizontal', 'role' => 'form')); ?>
<div class="row ">
  <div class="col-md-offset-4 col-md-3 ">
    <div class="panel panel-primary">
      <div class="panel-heading"><?php echo $titulo ?></div>
      <div class="panel-body">
        <? validation_erroses(validation_errors()); ?>
        <? informacion_error($this->session->flashdata("error")); ?>
        <div class="form-group">
          <label for="login" class="col-sm-3 control-label label_izquierdo">Usuario</label>

```

```

<div class="col-sm-7">
  <?=> form_input(array('type' => 'text', 'name' => 'login', 'id' => 'login', 'class' => 'form-control input-sm dato_caja',
'placeholder' => 'Usuario ...', 'value' => set_value('login'), 'onkeyup' => "javascript:this.value=this.value.toUpperCase()); ?>
</div>
</div>

<div class="form-group">
  <label for="password" class="col-sm-3 control-label label_izquierdo">Password</label>
  <div class="col-sm-7">
    <?=> form_input(array('type' => 'password', 'name' => 'password', 'id' => 'password', 'class' => 'form-control input-sm
dato_caja', 'value' => set_value('password'), 'onkeyup' => "javascript:this.value = this.value.toUpperCase()); ?>
  </div>
</div>
<hr>
<div class="form-group " >
  <div class="col-sm-12" align="center">
    <?=> form_button(array('type' => 'submit', 'content' => 'Iniciar Sesión', 'class' => 'btn btn-primary')); ?>
    <?=> anchor('http://www.google.com.pe/', 'Cancelar', array('class' => 'btn btn-default')); ?>
  </div>
</div>
</div>
</div>
</div>
</div>
<?=> form_close(); ?>

```

Cuadro n° 4.32 Vista v_ingresoAlSistema.

```

footer
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
?>
<footer class="color-footer">
  <hr class="color-procuraduria-inferior">
  <p style="margin: 0; padding-top: 1px; color: #000000;text-align: center;font-family: cursive; font-size: 11px;">
    <?php echo 'Dirección: ';
    if (isset($_SESSION['via']) && strlen($_SESSION['via'])) echo $_SESSION['via'];
    if (isset($_SESSION['manzana']) && strlen($_SESSION['manzana'])) echo ' / Manzana: ' . $_SESSION['manzana'];
    if (isset($_SESSION['lote']) && strlen($_SESSION['lote'])) echo ' Lote: ' . $_SESSION['manzana'];
    if (isset($_SESSION['hu']) && strlen($_SESSION['hu'])) echo ' - ' . $_SESSION['hu'];
    if (isset($_SESSION['telefono']) && strlen($_SESSION['telefono'])) echo ' / Teléfono: ' . $_SESSION['telefono'];

  ?>
</p>
<p style="margin: 0; padding-bottom: 15px; color: #000000;text-align: center;font-family: cursive; font-size: 14px;">E-
mail: procuraduria_publica@yahoo.es</p>
</footer>

<script src="<?=> base_url('jquery/jquery-1.12.3.min.js') ?>"></script>
<script src="<?=> base_url('jquery/jquery-ui.min.js') ?>"></script>
<script src="<?=> base_url('jquery/jquery-ui.timepicker.js') ?>"></script>
<script src="<?=> base_url('bootstrap/js/bootstrap.js') ?>"></script>
<script src="<?=> base_url('js/js-ubigeo.js') ?>"></script>
<script src="<?=> base_url('js/js-calendario.js') ?>"></script>
<script src="<?=> base_url('js/js-persona.js') ?>"></script>
<script src="<?=> base_url('js/js-direccion.js') ?>"></script>
<script src="<?=> base_url('js/js-funciones.js') ?>"></script>

<script src="<?=> base_url('js/js-entidad.js') ?>"></script>
<script src="<?=> base_url('js/js-funcionario.js') ?>"></script>

<script src="<?=> base_url('highcharts/highcharts.js') ?>"></script>
<script src="<?=> base_url('highcharts/highcharts-3d.js') ?>"></script>
<script src="<?=> base_url('highcharts/modules/exporting.js') ?>"></script>

<script type="text/javascript">
  var path = '<?php echo base_url() ?>index.php';
</script>
</body>
</html>

```

Cuadro n° 4.33 Vista Footer.

```

Vista Registrar Usuario
<div class="row">
  <div class="col-sm-offset-1 col-sm-10">
    <div class="panel panel-primary">
      <div class="panel-heading"><?php echo $titulo ?></div>
      <div class="panel-body">
        <?=> validacion_errores(validation_errors()); ?>
      </div>
    </div>
  </div>
</div>

```

```

<div class="table-responsive">
  <?form_open('c_usuario/listarUsuarios', array('class' => 'form-horizantal', 'role' => 'form'))?>
  <table>
    <tr>
      <td class="label_izquierdo">Usuario</td>
      <td>
        <input type="text" class="dato_caja" size="35" name="datousuario" id="datousuario" value="">
      </td>
      <td>
        <button type="submit" class="btn btn-info btn-xs" onclick="return validarBusqueda();">
          <span class="glyphicon glyphicon-search"></span>
        </button>
      </td>
    </tr>
  </table>
  <?form_close()?>
</div>
<hr>
<?anchor('c_usuario/registrarUsuario', 'Agregar', array('class' => 'btn btn-success'))?>
<?anchor('c_sistema/inicio', 'Cancelar', array('class' => 'btn btn-default'))?>
<div style="max-height: 400px;overflow-y: auto;overflow-x: hidden;">
  <div class="table-responsive">
    <table class="table table-condensed table-bordered table-striped">
      <thead>
        <tr>
          <th class="encabezadotabla">Perfil</th>
          <th class="encabezadotabla">Nº Documento</th>
          <th class="encabezadotabla">Ap. Paterno</th>
          <th class="encabezadotabla">Ap. Materno</th>
          <th class="encabezadotabla">Nombres</th>
          <th class="encabezadotabla">Logín</th>
          <th class="encabezadotabla">Estado</th>
          <th class="encabezadotabla">Domicilio</th>
          <th class="encabezadotabla" width="7%">Acción </th>
        </tr>
      </thead>
      <tbody>
        <?php foreach ($usuarios as $u) { ?>
          <tr>
            <td class="contenidotabla" style="vertical-align: middle;"><?=$u->tipofuncion?></td>
            <td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?=$u->n_documento?>
            <td class="contenidotabla" style="vertical-align: middle;"><?=$u->x_apaterno?></td>
            <td class="contenidotabla" style="vertical-align: middle;"><?=$u->x_apmaterno?></td>
            <td class="contenidotabla" style="vertical-align: middle;"><?=$u->x_nombres?></td>
            <td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?=$u->x_login?>
            <td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?php
              if ($u->flag_estado)
                echo 'ACTIVO';
              else
                echo 'PASIVO';
            ?></td>
            <td class="contenidotabla" style="vertical-align: middle;"><?=$u->domicilio?></td>
            <td style="vertical-align: middle;text-align: center;">
              <?anchor('c_usuario/editarUsuario' . $u->i_usuario, 'Actualizar Usuario')?>
              <a href="base_url('index.php/c_usuario/recetearClaveDeUsuario/' . $u->i_usuario)"?>
                class="btn btn-info btn-xs" onClick="return confirm('¿Está Seguro de Recetear clave de Usuario?')"><span class="glyphicon
                glyphicon-refresh" title="Recetear Clave"></span></a>
            </td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>
</div>
</div>
</div>
</div>

```

Cuadro n° 4.34 Vista Registrar Usuario.

```

Listar Usuarios Del Sistema
<div class="row">
  <div class="col-sm-offset-1 col-sm-10">
    <div class="panel panel-primary">
      <div class="panel-heading"><?php echo $titulo ?></div>
      <div class="panel-body">
        <?validacion_errores(validation_errors())?>
        <div class="table-responsive">

```

```

<?= form_open('c_usuario/listarUsuarios', array('class' => 'form-horizontal', 'role' => 'form')); ?>
<table>
<tr>
<td class="label_izquierdo">Usuario</td>
<td>
<input type="text" class="dato_caja" size="35" name="datousuario" id="datousuario" value="">
</td>
<td>
<button type="submit" class="btn btn-info btn-xs" onclick="return validarBusqueda();">
<span class="glyphicon glyphicon-search"></span>
</button>
</td>
</tr>
</table>
<?= form_close(); ?>
</div>

<hr>
<?= anchor('c_usuario/registrarUsuario', 'Agregar', array('class' => 'btn btn-success')); ?>
<?= anchor('c_sistema/inicio', 'Cancelar', array('class' => 'btn btn-default')); ?>
<div style="max-height: 400px;overflow-y: auto;overflow-x: hidden;">
<div class="table-responsive">
<table class="table table-condensed table-bordered table-striped">
<thead>
<tr>
<th class="encabezadotabla">Perfil</th>
<th class="encabezadotabla">N° Documento</th>
<th class="encabezadotabla">Ap. Paterno</th>
<th class="encabezadotabla">Ap. Materno</th>
<th class="encabezadotabla">Nombres</th>
<th class="encabezadotabla">Login</th>
<th class="encabezadotabla">Estado</th>
<th class="encabezadotabla">Domicilio</th>
<th class="encabezadotabla" width="7%">Acción </th>
</tr>
</thead>
<tbody>
<?php foreach ($usuarios as $u) { ?>
<tr>
<td class="contenidotabla" style="vertical-align: middle;"><?= $u->tipofuncion; ?></td>
<td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?= $u->n_documento; ?></td>
<td class="contenidotabla" style="vertical-align: middle;"><?= $u->x_apaterno; ?></td>
<td class="contenidotabla" style="vertical-align: middle;"><?= $u->x_apmaterno; ?></td>
<td class="contenidotabla" style="vertical-align: middle;"><?= $u->x_nombres; ?></td>
<td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?= $u->x_login; ?></td>
<td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?php
if ($u->flg_estado)
echo 'ACTIVO';
else
echo 'PASIVO';
?></td>
<td class="contenidotabla" style="vertical-align: middle;"><?= $u->domicilio; ?></td>
<td style="vertical-align: middle;text-align: center;">
<?= anchor('c_usuario/editarUsuario/' . $u->i_usuario, '<span class="glyphicon glyphicon-pencil"
title="Actualizar Usuario"></span>', 'class="btn btn-success btn-xs"') ?>
<a href="<?= base_url('index.php/c_usuario/recetearClaveDeUsuario/' . $u->i_usuario); ?>"
class="btn btn-info btn-xs" onClick="return confirm('¿Está Seguro de Recetear clave de Usuario?')">
<span class="glyphicon glyphicon-refresh" title="Recetear Clave"></span></a>
</td>
</tr>
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>

```

Cuadro n° 4.35 Vista Listar Usuarios Del Sistema.

Agregar Expediente /Caso

```

<?= form_open('c_expediente/guardarExpediente', array('class' => 'form-horizontal', 'role' => 'form')); ?>
<input type="hidden" name="idtramite" id="idtramite" value="<?= $tramite->i_tramite; ?>" />
<div class="row">
<div class="col-md-offset-3 col-md-5">
<div class="panel panel-primary">

```

```

<div class="panel-heading"><?php echo $titulo ?></div>
<div class="panel-body">
  <div class="borde-contenido">

    <? validation_erros(validation_erros()); ?>
    <?= informacion_error($this->session->flashdata("error")); ?>
    <div class="form-group">
      <? form_label('Tipo Trámite', 'tipotramite', array('class' => 'col-sm-3 control-label label_izquierdo')); ?>
      <div class="col-sm-5">
        <? form_dropdown('tipotramite', $tipotramite, 0, 'class="combobox" onchange="jsEstablecerTipoExpediente()'
id="tipotramite"); ?>
      </div>
    </div>

    <div class="form-group">
      <label for="numdoc" class="col-sm-3 control-label label_izquierdo">N° Expediente</label>
      <div class="col-sm-7">
        <? form_input(array('type' => 'text', 'name' => 'numdoc', 'id' => 'numdoc', 'class' => 'form-control input-sm
dato_caja', 'value' => set_value('numdoc'), 'onkeyup' => "javascript:this.value = this.value.toUpperCase()")); ?>
      </div>
    </div>

    <div class="form-group">
      <label for="campofecha" class="col-sm-3 control-label label_izquierdo">Fecha Registro</label>
      <div class="col-sm-3">
        <? form_input(array('type' => 'text', 'name' => 'campofecha', 'id' => 'campofecha', 'class' => 'form-control input-
sm dato_caja', 'value' => set_value(date("d/m/Y")))); ?>
      </div>
    </div>

    <div class="form-group">
      <label for="observacion" class="col-sm-3 control-label label_izquierdo">Observación</label>
      <div class="col-sm-7">
        <? form_textarea(array('rows' => '5', 'type' => 'text', 'name' => 'observacion', 'id' => 'observacion', 'class' =>
'form-control input-sm dato_caja', 'value' => set_value('observacion'), 'onkeyup' => "javascript:this.value =
this.value.toUpperCase()")); ?>
      </div>
    </div>
  </div>
  <hr>
  <div class="form-group">
    <div class="col-sm-12" align="center">
      <? form_button(array('type' => 'submit', 'content' => 'Guardar', 'class' => 'btn btn-primary')); ?>
      <? anchor('c_expediente/listarExpedientes/'. $tramite->i_tramite, 'Cancelar', array('class' => 'btn btn-default')); ?>
    </div>
  </div>
</div>
</div>
<? form_close(); ?>

```

Cuadro n° 4.36 Vista Agregar Expediente /Caso.

Listar Expediente/ Caso

```

<div class="row">
  <div class="col-md-offset-1 col-md-9">
    <div class="panel panel-primary">
      <div class="panel-heading"><?php echo $titulo ?></div>
      <div class="panel-body">
        <?php
        if (admin_tramite_permitido($tramite->i_tramite)) {
          echo anchor('c_expediente/registrarExpediente/'. $tramite->i_tramite, 'Agregar', array('class' => 'btn btn-info'));
        }
        <? anchor('c_tramite/adminTramite/'. $tramite->i_tramite, 'Salir', array('class' => 'btn btn-default')); ?>
        <div class="table-responsive">
          <table class="table table-condensed table-bordered table-striped">
            <thead>
              <tr>
                <th class="encabezadotabla" width="18%">Documento Matriz</th>
                <th class="encabezadotabla" width="18%">Expediente</th>
                <th class="encabezadotabla">Distrito Judicial/Fiscal</th>
                <th class="encabezadotabla">Registrado</th>
                <th class="encabezadotabla">Observación</th>
                <th class="encabezadotabla">Estado</th>
                <th class="encabezadotabla" width="10%">Acción </th>
              </tr>
            </thead>
            <tbody>
              <?php foreach ($expedientes as $e) { ?>
                <tr>

```



```

<td class="contenidotabla" style="vertical-align: middle;"><?= $e->n_documento; ?></td>
<td class="contenidotabla" style="vertical-align: middle;"><?= $e->n_expediente; ?></td>
<td class="contenidotabla" style="vertical-align: middle;"><?= $e->tiposexpediente; ?></td>
<td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?= $e->f_registro; ?></td>
<td class="contenidotabla" style="vertical-align: middle;"><?= $e->x_observacion; ?></td>
<td class="contenidotabla" style="vertical-align: middle;"><?= $e->estado; ?></td>
<td style="vertical-align: middle;text-align: center;">
<?php
    if (admin_tramite_permitido($tramite->i_tramite) && (int) $e->n_exp_raiz == 0) {
        ?>
        <a href="<?= base_url('index.php/c_expediente/editarExpediente/' . trim($e->i_tramite) . '/' . trim($e->i_expediente)); ?>" class="btn btn-primary btn-xs"><span class="glyphicon glyphicon-pencil" title="Actualizar Expediente"></span></a>
        <a href="<?= base_url('index.php/c_expediente/eliminarExpediente/' . trim($e->i_tramite) . '/' . trim($e->i_expediente)); ?>" class="btn btn-danger btn-xs" onClick="return confirm('¿Está Seguro de Quitar el Expediente?')"><span class="glyphicon glyphicon-remove" title="Quitar Expediente"></span></a>
        <?php } ?>
    </td>
</tr>
<?php } ?>
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>

```

Cuadro n° 4.37 Vista Listar Expediente/ Caso.

Listar Procesos de un expediente

```

<?php
$this->load->view('proceso/vjs_verContenidoProceso');
?>

<input type="hidden" name="idtramite" id="idtramite" value="<?= $tramite->i_tramite; ?>" />
<div class="row">
    <div class="col-md-offset-1 col-md-9">
        <div class="panel panel-primary">
            <div class="panel-heading"><?php echo $titulo ?></div>
            <div class="panel-body">
                <?php
                    if (admin_tramite_permitido($tramite->i_tramite)) {
                        echo anchor('c_proceso/registrarProceso/' . $tramite->i_tramite, 'Agregar', array('class' => 'btn btn-info'));
                    }
                ?>
                <?= anchor('c_tramite/adminTramite/' . $tramite->i_tramite, 'Salir', array('class' => 'btn btn-default')); ?>
            <div style="max-height: 400px;overflow-y: auto;overflow-x: hidden;">
                <div class="table-responsive">
                    <table class="table table-condensed table-bordered table-striped">
                        <thead>
                            <tr>
                                <th class="encabezadotabla" width="3%">N°</th>
                                <th class="encabezadotabla" width="20%">Trámite</th>
                                <th class="encabezadotabla">Resumen</th>
                                <th class="encabezadotabla">Procesado</th>
                                <th class="encabezadotabla" width="22%">Usuario</th>
                                <th class="encabezadotabla" width="8%">Acción </th>
                            </tr>
                        </thead>
                        <tbody>
                            <?php
                                $contador = sizeof($procesos);
                                foreach ($procesos as $pr) {
                                    ?>
                                    <tr>
                                        <td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?= $contador; ?></td>
                                        <td class="contenidotabla" style="vertical-align: middle;"><?= $pr->i_tramite; ?></td>
                                        <td class="contenidotabla" style="vertical-align: middle;">
                                            <button style="background-color: #ffffff;border: none;text-align: left;" title="Ver Contenido"
                                            onclick="verContenidoProceso(<?= '&quot;' . $pr->i_proceso . '&quot;' ; ?>)"><?= $pr->x_resumen; ?></button>
                                        </td>
                                        <td class="contenidotabla" style="vertical-align: middle;text-align: center;"><?= $pr->f_proceso; ?></td>
                                        <td class="contenidotabla" style="vertical-align: middle;"><?= $pr->nomusuario; ?></td>
                                        <td style="vertical-align: middle;text-align: center;">
                                            <?php if (admin_tramite_permitido($tramite->i_tramite)) { ?>
                                                <a href="<?= base_url('index.php/c_proceso/editarProceso/' . trim($pr->i_tramite) . '/' . trim($pr->i_proceso)); ?>" class="btn btn-primary btn-xs"><span class="glyphicon glyphicon-pencil" title="Actualizar Proceso"></span></a>
                                                <a href="<?= base_url('index.php/c_proceso/eliminarProceso/' . trim($pr->i_tramite) . '/' . trim($pr->i_proceso)); ?>" class="btn btn-danger btn-xs" onClick="return confirm('¿Está Seguro de Eliminar el Proceso?')"><span class="glyphicon glyphicon-remove" title="Eliminar Proceso"></span></a>
                                            }
                                        </td>
                                    </tr>
                                }
                            </tbody>
                    </table>
                </div>
            </div>
        </div>
    </div>
</div>

```

```
<?php } ?>
</td>
</tr>
<?php
$contador--;
}
?>
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>
</div>
```

Cuadro n° 4.38 Vista Listar Procesos de un expediente.

4.1.8.4 IMPLEMENTACIÓN DE LOS CASOS DE USOS

Registrar Usuario
<pre> public function registrarUsuario() { \$data['titulo'] = "REGISTRAR NUEVO USUARIO DEL SISTEMA"; \$data['tiposusuario'] = \$this->sistema->cargarCodigosTabla('TFU'); \$data['contenido'] = 'usuario/v_registrarUsuario'; \$this->load->view('frontend', \$data); } public function guardarUsuario() { \$aux = \$this->input->post(); if (\$this->form_validation->run('guardar_usuario')) { \$registro = array(); \$registro['i_usuario'] = trim(\$aux['idper1']) . trim(\$aux['perfil']); \$registro['c_tipofuncion'] = trim(\$aux['perfil']); \$registro['x_login'] = trim(\$aux['login']); \$registro['c_password'] = md5(trim(\$aux['login'])); \$registro['f_registro'] = date("Y/m/d H:i"); \$registro['lg_estado'] = '1'; \$registro['i_persona'] = trim(\$aux['idper1']); if (sizeof(\$this->usuario->buscarUsuario(\$registro['i_usuario'])) == 0) { \$this->usuario->guardarUsuario(\$registro); redirect(base_url() . 'index.php/c_usuario/listarUsuarios', 301); } else { \$this->session->set_flashdata("error", "Usuario(a) que intentas ingresar ya existe en la Base de Datos"); redirect(base_url() . 'index.php/c_usuario/registrarUsuario', 301); } } else { \$this->registrarUsuario(); } } </pre>

Cuadro n° 4.39 Registrar Usuario.

Listar Usuarios Del Sistema
<pre> public function listarUsuarios() { \$dato = \$this->input->post('datousuario'); if (isset(\$dato)) { \$data['usuarios'] = \$this->usuario->listarUsuarios(\$dato, trim(\$_SESSION['i_usuario'])); foreach (\$data['usuarios'] as \$us) { \$us->tipofuncion = \$this->sistema->descripcionDeCodigo(\$us->c_tipofuncion, 'TFU')->x_descripcion; \$auxPer = \$this->direccion->buscarDomicilioDePersona(\$us->i_persona); \$us->domicilio = ""; if (sizeof(\$auxPer) > 0) { \$us->domicilio = \$auxPer[0]['x_hab_urbana'] . ' - ' . \$auxPer[0]['x_via'] . ' . ' . \$auxPer[0]['n_num_via'] . ' . ' . \$auxPer[0]['x_manzana'] . ' . ' . \$auxPer[0]['x_lote']; } } } else { \$data['usuarios'] = \$this->usuario->listarUsuarios("", trim(\$_SESSION['i_usuario'])); foreach (\$data['usuarios'] as \$us) { \$us->tipofuncion = \$this->sistema->descripcionDeCodigo(\$us->c_tipofuncion, 'TFU')->x_descripcion; \$auxPer = \$this->direccion->buscarDomicilioDePersona(\$us->i_persona); \$us->domicilio = ""; if (sizeof(\$auxPer) > 0) { \$us->domicilio = \$auxPer[0]['x_hab_urbana'] . ' - ' . \$auxPer[0]['x_via'] . ' . ' . \$auxPer[0]['n_num_via'] . ' . ' . \$auxPer[0]['x_manzana'] . ' . ' . \$auxPer[0]['x_lote']; } } } \$data['titulo'] = 'Mantenimiento de Usuarios'; \$data['contenido'] = 'usuario/v_listaUsuarios'; \$this->load->view('frontend', \$data); } </pre>

Cuadro n° 4.40 Listar Usuarios Del Sistema.

Agregar Expediente /Caso
<pre> public function registrarExpediente(\$idTramite) { \$data = detalle_tramite(\$idTramite); \$data['contenido'] = 'tramite/v_adminTramite'; \$data['tipotramite'] = \$this->sistema->cargarCodigosTabla('TDT'); unset(\$data['tipotramite'][3], \$data['tipotramite'][4]); \$data['titulo'] = 'Registrar Nuevo Trámite'; \$data['subcontenido'] = 'expediente/v_registrarExpediente'; \$this->load->view('frontend', \$data); } public function guardarExpediente() { </pre>

```

$aux = $this->input->post();
if ($this->form_validation->run('guardar_expediente')) {
    if (sizeof($this->expediente->buscarExpedientePorId($aux['idtramite'], trim($aux['tipotramite']) . trim($aux['numdoc'])) == 0)
    {
        $expediente = array();
        $expediente['i_expediente'] = trim($aux['tipotramite']) . trim($aux['numdoc']);
        $expediente['n_expediente'] = trim($aux['numdoc']);

        if (sizeof($this->expediente->buscarExpedientesPorTramiteJudicial($aux['idtramite'])) == 0) {
            $expediente['n_exp_raiz'] = '1';
        } else {
            $expediente['n_exp_raiz'] = '0';
        }
        $expediente['c_tipoexpediente'] = trim($aux['tipotramite']);
        $expediente['f_registro'] = $aux['campofecha'];
        $expediente['x_observacion'] = trim($aux['observacion']);
        $expediente['flg_estado'] = '1';
        $expediente['i_tramite'] = trim($aux['idtramite']);
        $this->expediente->guardarExpediente($expediente);
        redirect(base_url() . 'index.php/c_expediente/listarExpedientes/' . $aux['idtramite'], 301);
    } else {
        $this->session->set_flashdata("error", "El documento ya fue registrado");
        redirect(base_url() . 'index.php/c_expediente/registrarExpediente/' . $aux['idtramite'], 301);
    }
} else {
    $this->registrarExpediente($aux['idtramite']);
}
}

```

Cuadro n° 4.41 Agregar Expediente /Caso.

```

Listar Expediente/ Caso
public function listarExpedientes($idTramite) {
    $data = detalle_tramite($idTramite);
    $data['expedientes'] = $this->expediente->listarExpedientesPorTramite($idTramite);
    foreach ($data['expedientes'] as $e) {
        $e->tipoexpediente = $this->sistema->descripcionDeCodigo($e->c_tipoexpediente, 'TDT')->x_descripcion;
        $e->estado = 'PASIVO';
        if ((int) $e->flg_estado == 1) {
            $e->estado = 'ACTIVO';
        }
    }
    $data['contenido'] = 'tramite/v_adminTramite';
    $data['titulo'] = 'Expedientes del Trámite';
    $data['subcontenido'] = 'expediente/v_listarExpedientes';
    $this->load->view('frontend', $data);
}

```

Cuadro n° 4.42 Listar Expediente/ Caso.

```

Listar Procesos de un expediente
public function listarProcesos($idTramite) {
    $data = detalle_tramite($idTramite);
    $data['procesos'] = $this->proceso->listarProcesoPorTramite($idTramite);
    $data['contenido'] = 'tramite/v_adminTramite';
    $data['titulo'] = 'Actos Procesales';
    $data['subcontenido'] = 'proceso/v_listarProcesos';
    $this->load->view('frontend', $data);
}

```

Cuadro n° 4.43 Listar Procesos de un expediente.

4.2 REVISIÓN DE CÓDIGO

4.2.1 PRUEBAS UNITARIAS

CASO DE USO	CONTROLADOR	METODO	ESTADO
Registrar Usuario	Usuario	<ul style="list-style-type: none"> registrarUsuario() 	Exitoso
Listar	Usuario	<ul style="list-style-type: none"> listarUsuarios() 	Exitoso

Usuarios Del Sistema			
Agregar Expediente /Caso	Expediente	<ul style="list-style-type: none"> • registrarExpediente(\$idTramite) • guardarExpediente() 	Exitoso
Listar Expediente/ Caso	Expediente	<ul style="list-style-type: none"> • listarExpedientes() 	Exitoso
Listar Procesos de un expediente	Proceso	<ul style="list-style-type: none"> • listarProcesos() 	Exitoso

Cuadro n° 4.44 Resultado de pruebas unitarias.

4.2.2 PRUEBAS FUNCIONALES

N°	PROPÓSITO	INICIALIZACIÓN	DESCRIPCIÓN DE DATOS DE ENTRADA	RESULTADO ESPERADO	EVALUACIÓN DE PRUEBA
1	Registrar Usuario	El Administrador del sistema debe de haberse autenticado e iniciado sesión (Activa).	El Administrador hace clic en el menú (Administración -> Usuario del Sistema); luego el sistema muestra un formulario (mantenimiento de usuarios) y allí se observa el botón (Agregar), el sistema muestra un formulario y procedemos a ingresar los datos del usuario, el Administrador los registra haciendo un clic en el botón (Guardar).	El sistema muestra un mensaje de confirmación de registro indicando que los datos se guardaron correctamente	EXITOSA
2	Listar Usuarios Del Sistema	El Administrador del sistema debe de haberse autenticado e iniciado sesión (Activa).	El Administrador hace clic en el menú (Administración -> Usuario del Sistema); luego el sistema muestra un formulario (Mantenimiento de Usuarios) (No necesita ningún dato de entrada)	El sistema muestra una lista de todos los usuarios registrados.	EXITOSA
3	Agregar Expediente /Caso	El Usuario del sistema debe de haberse autenticado e iniciado sesión (Activa).	El Usuario hace clic en el menú (Trámite Documentario -> Administrar tramite) luego el sistema muestra un formulario (Administrar Tramite) donde se observa un botón icono (Abrir Tramite) al cual se le hace clic, luego el sistema nos muestra una pantalla con un menú lateral en donde el usuario hace clic al	El sistema muestra un mensaje de confirmación de registro indicando que los datos se guardaron correctamente	EXITOSA

			menú (Expedientes); el sistema muestra un formulario (Expedientes de Tramite) después el usuario hace clic al botón (Agregar) luego el sistema muestra un formulario (Registrar Nuevo Tramite) en donde el usuario llena los campos correctamente y luego procede a realizar clic al botón (Guardar).		
4	Listar Expediente/ Caso	El Usuario del sistema debe de haberse autenticado e iniciado sesión (Activa).	El Usuario hace clic en el menú (Trámite Documentario -> Administrar tramite) luego el sistema muestra un formulario (Administrar Tramite) donde se observa un botón icono (Abrir Tramite) al cual se le hace clic, luego el sistema nos muestra una pantalla con un menú lateral en donde el usuario hace clic al menú (Expedientes); el sistema muestra un formulario (Expedientes de Tramite (No necesita ningún dato de entrada)	El sistema muestra una lista de todos expedientes caso para dicho trámite previamente registrado.	EXITOSA
5	Listar Procesos de un expediente	El Usuario del sistema debe de haberse autenticado e iniciado sesión (Activa).	El Usuario hace clic en el menú (Trámite Documentario -> Administrar tramite) luego el sistema muestra un formulario (Administrar Tramite) donde se observa un botón icono (Abrir Tramite) al cual se le hace clic, luego el sistema nos muestra una pantalla con un menú lateral en donde el usuario hace clic al menú (Acto Procesal); (No necesita ningún dato de entrada)	El sistema muestra lista de los Procesos de un expediente previamente registrado.	EXITOSA

Cuadro n° 4.45 Pruebas funcionales.

4.3 RESULTADO DEL PROCESO DE DESARROLLO DE SISTEMA.

Se dio uso a la metodología Iconix para la construcción del “Sistema De Información Web Para Optimizar La Gestión De Archivos En La Procuraduría De Ayacucho”; siguiendo paso a paso las actividades de la metodología para la construcción del sistema.

IMPLEMENTACION DEL SISTEMA DE INFORMACIÓN WEB PARA OPTIMIZAR LA GESTIÓN DE ARCHIVOS EN LA PROCURADURÍA DE AYACUCHO			
N°	TAREA	ARTEFACTO	TIEMPO ESTIMADO (DIAS)

1	Identificar las necesidades y requisitos que debe cumplir el sistema.	Requisitos funcionales y no funcionales.	5
2	Identificar el mundo real y los objetos de dominio del negocio.	Modelo de dominio.	4
3	Definir los requisitos de comportamiento.	Casos de uso.	4
4	Realizar análisis de robustez para eliminar ambigüedad de los casos de uso y determinar los defectos del modelo de dominio.	Diagrama de robustez.	4
5	Asignar comportamiento a los objetos.	Diagrama de secuencia.	3
6	Finalizar el modelo estático.	Diagrama de clases.	4
7	Generar código.	Código fuente.	100
8	Realizar pruebas de aceptación.	Pruebas funcionales y unitarias.	5

Cuadro n° 4.46 Tiempo Estimado para la Implementación de Software.

La realización del sistema de información se logró gracias al empleo del lenguaje PHP y a la utilización del framework Codeigniter; también se implementó la base de datos en su totalidad; se realizó de forma metódica el análisis, diseño e implementación de todos los casos de uso planteados en el análisis de requisitos final, en este informe (tesis) se documenta solo cinco casos de uso los cuales son los más representativos.

4.4 DISCUSIÓN DE RESULTADOS

- a. Dentro de los resultados obtenidos, el sistema de información resultante tiene en gran medida en función a otros sistemas del mismo rubro, las siguientes características como usabilidad, funcionalidad y fiabilidad; lo que hace que este sistema cumpla con los estándares, para que los usuarios finales se familiaricen y adapten de forma rápida al uso de este.
- b. En los objetivos se hace énfasis de cómo debería funcionar el sistema frente al usuario, para optimizar la gestión documental de la Procuraduría; un medio para lograr esto es generar unos artefactos óptimos para lograr implementar el objetivo principal.
- c. Los resultados obtenidos se pueden integrar a las diversas instituciones de la región de Ayacucho, que requieran de un sistema para la automatización de documentos referentes a la procuraduría pública.
- d. Las limitaciones que se hallaron en esta investigación, fueron la falta de

documentación como manuales de usuario, manual de procedimientos y manual funciones, todo ello en concordancia a la gestión de los documentos de la Procuraduría de Ayacucho, lo que nos dificultó entender la lógica de negocio o del problema en cuestión a solucionarse planteada en esta tesis.

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

- a. En relación al objetivo general de este trabajo de investigación, se puede concluir que, se logró automatizar en su totalidad el sistema de información de gestión documental para la Procuraduría de Ayacucho; el sistema se desarrolló bajo el patrón (MVC) con ayuda del framework CodeIgniter, lo cual hace que el sistema se escalable, portable y seguro.
- b. En relación al objetivo específico, “Aplicar la Usabilidad en el sistema de información web ayuda al registro de documentos”, se puede concluir que, se logró en su totalidad la capacidad de “*registro de documentos*” en el sistema web, que va de acuerdo a la lógica de negocio, el cual es objeto de estudio de esta tesis; todo ello se puede comprobar en las artefactos creados en la etapa de resultados de esta investigación y en la aplicación de software en sí la cual se adjunta como anexo en esta tesis.
- c. De acuerdo al objetivo específico, “Utilizar la Funcionalidad en el sistema de información web ayuda a la clasificación de documentos.”, se puede concluir que, se logró en su totalidad la característica de “*clasificación de documentos*” en el sistema web, que va de acuerdo a la lógica de negocio el cual es objeto de estudio de esta tesis; todo ello se puede comprobar en los artefactos creados en la etapa de resultados de la tesis y en la aplicación de software en sí, la cual se adjunta como anexo en esta tesis.
- d. Según el objetivo específico, “Aplicar la Fiabilidad en el sistema de información web mejora la descripción y recuperación de documentos”, se puede concluir que, se logró en su totalidad la capacidad de albergar la “*descripción y recuperación de documentos*” en el sistema web que va de acuerdo a la lógica de negocio el cual es objeto de estudio de esta tesis; todo ello se puede comprobar en los artefactos creados en la etapa de resultados de la investigación y en la aplicación de software en sí, la cual se adjunta como anexo en esta tesis.

5.2 RECOMENDACIONES

- a. Se recomienda adecuar el “Sistema de Información Web para optimizar la gestión de archivos en la Procuraduría de Ayacucho” subirlo a la nube, para que cualquier institución adjunta al Gobierno Regional de Ayacucho lo pueda usar de forma directa y evitar la instalación local cada vez que así se requiera.
- b. Se recomienda investigar e implementar un plan de gestión y respaldo de la base de datos del sistema de web creado, para todas las entidades adjuntas al Gobierno Regional de Ayacucho que adoptaron este sistema.
- c. La Subgerencia De Desarrollo Institucional E Informática del Gobierno Regional de Ayacucho debería capacitar al personal administrativo, para el manejo y administración del “Sistema de Información Web para optimizar la gestión de archivos en la Procuraduría de Ayacucho”.
- a. Gestionar la adquisición de servidores dedicados tanto para los sistemas como para la base de datos, con la finalidad de garantizar el normal desenvolvimiento del sistema, priorizando performance y calidad para su funcionamiento.

BIBLIOGRAFÍA

1. Amavizca Valdez, L., García Ruíz, A., Jiménez López, E., Duarte Guerrero, G., & Vázquez Brindis, J. (2014). Aplicación de la metodología semi-ágil ICONIX para el desarrollo de software: Recuperado de LACCEI: <http://www.laccei.org/LACCEI2014-Guayaquil/RefereedPapers/RP246.pdf>
2. Andreu, R., J. Ricart y J. Valor (1996). Estrategia y sistemas de información, McGraw-Hill, Madrid.
3. Antonia H. (1991). Archivística General Teoría y Práctica. Edita: Diputación provincial de Sevilla, España.
4. Avalos, H., Lucio, F., Quispe, P., Henry, G. (2014). Sistema de asignación y control de expedientes en el área de fiscalización de la ONP, alineado a DBR y BPM, utilizando BONITASOFT, Universidad San Martín de Porres, Lima – Perú.
5. Bernal, C. (2006). Metodología De La Investigación: Para Administración, Economía, Humanidades Y Ciencias Sociales. (2da. Ed.). México: Pearson Educación
6. CampusMvp (2019), Desarrollador web: Front-end, back-end y full stack. ¿Quién es quién?, Recuperado el 05 de julio, 2019, <https://www.campusmvp.es/recursos/post/Desarrollador-web-Front-end-back-end-y-full-stack-Quien-es-quien.aspx>
7. Carrasco, 2006. Metodología de Investigación Científica. Primera edición. Impreso en Perú.
8. Cervantes, J., Gómez, M., González P. y García A. (2016). Introducción a la Programación Orientada a Objetos. Libro de la Universidad Autónoma Metropolitana – Unidad Cuajimalpa. © 2016 Por esta edición, Mexico.
9. CodeIgniter (2019), Welcome to CodeIgniter. Recuperado el 05 de julio, 2019, https://www.codeigniter.com/user_guide/general/welcome.html
10. Correo. (2018), Procuraduría Regional con limitaciones para defender intereses del Estado. (2018). Recuperado el 03 de septiembre del 2018, <https://diariocorreo.pe/edicion/ayacucho/procuraduria-regional-con-limitaciones-para-defender-intereses-del-estado-710055/>.

11. Correo. (2018), Procuraduría tiene tres abogados para atender más de siete mil casos. (2018). Recuperado el 03 de septiembre del 2018, <https://diariocorreo.pe/edicion/ayacucho/procuraduria-tiene-tres-abogados-para-atender-mas-de-siete-mil-casos-679241/>.
12. Correo. (2018), Tres mil 500 casos y 90 laudos arbitrales lleva la Procuraduría Regional de Ayacucho. (2018). Recuperado el 03 de septiembre del 2018, <https://diariocorreo.pe/edicion/ayacucho/tres-mil-500-casos-y-90-laudos-arbitrales-lleva-la-procuraduria-regional-de-ayacucho-798994/>.
13. Correo. (2019), Procuraduría Regional con más de 24 mil casos en curso. (2019). Recuperado el 12 de marzo del 2019, <https://diariocorreo.pe/edicion/ayacucho/procuraduria-regional-con-mas-de-24-mil-casos-en-curso-866153/>
14. Digital Guide. (2017). CodeIgniter, el peso pluma de los frameworks PHP. Recuperado el 12 de agosto, 2019, de <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/codeigniter-framework-php-rapido-y-versatil/>
15. Fernández, V. (2006). Desarrollo de Sistemas de Información: una Metodología Basada en el Modelado, Univ. Politécnica de Catalunya, 2010. Editor Upc Edicions.
16. Galindo, J. y Camps, J. (2008). Diseño e implementación de un marco de trabajo (framework) de presentación para aplicaciones JEE. Una publicación de Universitat Oberta de Catalunya. Edita: S/E, España.
17. Gallego, A., (2003). Desarrollo Web con PHP y MySQL. Editorial Anaya Multimedia (Grupo Anaya) S. A., España.
18. García Bravo, D. (2000). Sistemas de información en la empresa. Conceptos y aplicaciones, Pirámide, Madrid.
19. Gauchat, J. (2003). El gran Libro de HTML5, CSS y JavaScript. Colombia: Marcombo.
20. Graw Mc, (1993). Sistemas informáticos. – Hardware, componentes físicos. – Software, componentes lógicos. Un enfoque práctico. 3ªEd. R. S. PRESSMAN.

21. Gutiérrez, J. (2008). Laboratorio de Redes Neuronales. Armenia, Colombia: Conceptos Gráficos.
22. Hernández, R. S., Collado, C. F., LucioBaptista, y de la Luz Casas Pérez Ma de la Luz Casas, M. (2014). *Metodología de la investigación*, volumen 1. Mcgraw-hill México.
23. Hernández, R.; Fernández, C. y P. Baptista (2010). *Metodología de la Investigación*. Quinta Edición. México. McGraw-Hill.
24. ICA (2019), Internaional Council on Archives.. Recuperado el 02 de julio de 2019, <https://www.ica.org/es>.
25. International Organization for Standardization. (2014). ISO/IEC 25000:2014 SQuaRE (System and Software Quality Requirements and Evaluation)
26. Joyanes, L. (2007). Algoritmos y Programación Orientada a Objetos. McGraw Hill, España.
27. Joyanes, L. (2008). Fundamentos de Programación – Algoritmos, estructuras de datos y objetos. Edición 4ta. Editorial McGraw Hill, España.
28. Larman C., (1999). UML y patrones. Introducción al Análisis y Diseño Orientado a Objetos, 1era edición, Editorial Prentice Hall, España.
29. Martin Gavilán, C (2009). Concepto y función de archivo - Clases de archivos - El Sistema Archivístico Español. Edita: S/E, España.
30. Mazón, B, Catuche, J, Chimarro, V y Rivas, W (2015). Fundamentos de Programación Orientada a Objetos en Java. Libro de la Universidad Técnica de Machala. Primera Edición 2015. ISBN: 978-9978-316-72-6, Ecuador.
31. MDN web docs. (2019), CSS, Recuperado el 07 de junio del 2019, <https://developer.mozilla.org/es/docs/Web/CSS>
32. MDN web docs. (2019), HTML, Recuperado el 07 de junio del 2019, <https://developer.mozilla.org/es/docs/Web/HTML>
33. Menguzzato, M. y J. J. Renau. (1991). La Dirección Estratégica de la empresa. Un enfoque innovador del Management, Ariel, Barcelona.
34. Moreno García, M. (2016). TEMA 1 Sistemas de información.
35. Moreno, F (2000). Introducción a la OOP. Edita: © Grupo EIDOS Versión

1.0.0, España.

36. Morrison, (2008). Javascript, Recuperado el 29 de Octubre del 2018 de, <http://support.oreilly.com/oreilly/topics/head-first-javascript-by-michael-morrison>.
37. Moscoso, J. (2018). Desarrollo de una aplicación web para la mejora de la gestión de trámite documentario en la Dirección Regional de Agricultura de Madre de Dios, Universidad inca Garcilaso de la Vega, Lima – Perú.
38. Pérez, A. (2007). Desarrollo De Herramientas Web De Gestión Docente. Tesis de Ingeniero Técnico de Telecomunicación, Especialidad Telemática. Escuela Técnica Superior De Ingeniería De Telecomunicación Universidad Politécnica De Cartagena. Cartagena – España.
39. Porras, E. (2010). Comparación de dos procesos de Desarrollo de Software Usando Los Métodos ICONIX Y XP, Caso: Comercialización de la Tara en la Región Ayacucho. Tesis de Maestría, Universidad Nacional de Ingeniería, Lima, Perú.
40. Porras, E. (2011). La Metodología Ágil y Formal ICONIX para el Desarrollo de Software: Teoría y Práctica. Ayacucho, Perú: Ami Ayacucho.
41. Rafael L., Carlos D., y Joaquín G. (2011). Introducción a la gestión de sistemas de información, Edita: Publicacions de la Universitat Jaume I. Servei de Comunicació i Publicacions, España.
42. Raul H (2002). Sistemas de información en la era digital. Edita: Programa Avanzado de perfeccionamiento en Management de la FUNDACION OSDE, con la supervisión académica y certificación de la Universidad Nacional de San Martín, Argentina.
43. Rosenberg, D. y K. Scott (1999) Use Case Driven Object Modeling with UML: A Practica/ Approach. Primera Edición. EUA. Addison-Wesley.
44. Rosenberg, D. y K. Scott (2001) Applying Use Case Driven Object Modeling with UML: Un Anotated e-Commerce Sample. Primera Edición. United States. Addison-Wesley.
45. Rosenberg, D., & Stephens, M. (2007). Use Case Driven Object Modeling

with UML Theory and Practice: Theory and Practice. New York Edita: Apress, EEUU.

46. Rosenberg, Doug, Stephens, Matt y Collins-Cope, Mark. (2005). Agile Development with ICONIX Process: People, Process, and Pragmatism. EE.UU Edita: Apress, 2005. ISBN 1-59059-464-9, EE.UU.
47. Saiz, F. J., Olmeda, R., Olmeda, C. (2013). Implantación de un sistema de gestión de Archivo para una fundación en medicina, Universidad Carlos III de Madrid, España.
48. Santana Bueno, M (2011). Curso de Internet a distancia para sacerdotes, religiosos y religiosas. Edita: Diócesis de Canarias - Islas Canarias, España.
49. Silberschatz, A., Korth, H. y Sudarshan, S. (2006). Fundamentos De Base De Datos. Edita: McGraw-Hill/Interamericana de España, S.A.U., España.
50. Moreno, Maria. (2019), TEMA 1 Sistemas de Información, Universidad de Salamanca, Departamento de Informática.
51. Tutoriales Programación Ya. (2019). ¿Qué es PHP? Recuperado de <http://www.tutorialesprogramacionya.com/phpya/temarios/descripcion.php?punto=1&cod=23&inicio=0>

ANEXO A

Título general: Sistema de Información Web para optimizar la gestión de archivos en la procuraduría de Ayacucho, 2018.

PROBLEMA	OBJETIVOS	HIPÓTESIS	VARIABLES	METODOLOGÍA	POBLACIÓN
<p>Problema general:</p> <p>¿Cómo un Sistema de información web optimiza la gestión de archivos en la Procuraduría de Ayacucho, 2018?</p> <p>Problemas específicos:</p> <p>¿De qué manera la usabilidad del sistema de información web ayuda al registro de documentos?</p> <p>¿De qué manera la funcionalidad del sistema de información web optimiza la clasificación de documentos?</p> <p>¿De qué manera la fiabilidad del sistema de información web mejora la descripción y recuperación de documentos?</p>	<p>Objetivo general:</p> <p>Desarrollar un Sistema de información web para optimizar la gestión de archivos en la Procuraduría de Ayacucho, 2018, con la finalidad de optimizar la gestión de archivos.</p> <p>Objetivos específicos:</p> <p>Aplicar la Usabilidad en el sistema de información web ayuda al registro de documentos.</p> <p>Utilizar la Funcionalidad en el sistema de información web ayuda a la clasificación de documentos.</p> <p>Aplicar la Fiabilidad en el sistema de información web mejora la descripción y recuperación de documentos</p>	<p>Hipótesis general:</p> <p>El sistema de información Web optimiza la gestión de archivos de la procuraduría de Ayacucho, 2018.</p> <p>específicos:</p> <p>La Usabilidad en el sistema de información web ayuda al registro de documentos.</p> <p>La Funcionalidad en el sistema de información web ayuda a la clasificación de documentos.</p> <p>la Fiabilidad en el sistema de información web mejora la descripción y recuperación de documentos</p>	<p>Variable Independiente:</p> <p>SISTEMA DE INFORMACIÓN WEB</p> <p>INDICADORES:</p> <ul style="list-style-type: none"> • <i>Usabilidad.</i> • <i>Funcionalidad.</i> • <i>Fiabilidad,</i> <p>Variable Dependiente:</p> <p>GESTIÓN DE ARCHIVOS</p> <p>INDICADORES:</p> <ul style="list-style-type: none"> • <i>Registro de documentos.</i> • <i>Clasificación de documentos.</i> • <i>Descripción y recuperación de documentos.</i> 	<p>Nivel de investigación</p> <p>Descriptivo</p> <p>Tipo de investigación.</p> <p>Aplicada</p> <p>Diseño de investigación</p> <p>No experimental correlacional.</p>	<p>La población</p> <p>La población está representada por los actores que hacen uso del sistema de información Web de gestión de archivo de la procuraduría de Ayacucho, 2018.</p> <p>La muestra.</p> <p>Se tomará una muestra de tipo aleatoria simple, al 95% de confianza y 5% de error de las los actores que hacen uso sistema de información Web de gestión de archivo de la procuraduría de Ayacucho, 2018.</p>

Cuadro n° 4.47 Matriz DE Consistencia.

ANEXO B

- ✓ Disco compacto conteniendo el sistema y la base datos