Universidad de las Ciencias Informáticas

Facultad 8

Análisis, Diseño e Implementación del Módulo de Investigaciones Internas del SIIPOL.

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

Autores:

Adonys Alea Boffill

Yander Caceres Crúz

Tutora:

Ing. Diana García Vicente

- 2009 –
"Año del 50 Aniversario del Triunfo de la Revolución"

Todo es hermoso y constante,
todo es música y razón,
y todo como el diamante,
antes que luz, es carbón.

José Martí.

Declaración de Autoría.

Declaramos que somos los autores de este trabajo y autorizamos a la Facultad 8 de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con el mismo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año 2009.

Adonys Alea Boffill

_____
Firma del Autor

Yander Caceres Crúz

_____
Firma del Autor

Ing. Diana García Vicente

_____
Firma de la Tutora

Agradecimientos.

A nuestro Comandante en Jefe Fidel por brindarnos la oportunidad de estudiar es esta grandiosa Universidad que ha sido nuestra segunda casa durante estos cinco años y a la Revolución Cubana por darnos el derecho a todos de tener una educación gratuita y de alta calidad.

A nuestros padres por marcar el sendero y a nuestros hermanos por confiar siempre en nosotros… nada sería posible sin ustedes.

A nuestra amiga y tutora Diana y a Susel, por todos los consejos y la ayuda que nos han brindado, por su paciencia, por escucharnos y por darnos su apoyo en todo momento… muchas gracias a las dos por estar siempre presentes.

Al Peludo, Lorencito, el Hormigón y el Chucho… por todo y porque sí.

A todos nuestros compañeros de aula y proyecto, que estuvieron tanto tiempo a nuestro lado, que tanta ayuda nos brindaron y que junto a nosotros se esforzaron por lograr esta tarea. Shael, Arian, Eduardo, Jorgito, Adriel, Humberto, Geldri, Arlan y Amed… gracias por poner su granito de arena.

A todos aquellos que en algún momento nos preguntaron: ¿Cómo va la tesis?

A todos nuestros amigos…

Dedicatoria.

A mis padres, principalmente, pues es la satisfacción de uno de sus mayores deseos.
A mi hermano, que viene siguiendo mis pasos: espero que le sirva de ejemplo todo el esfuerzo realizado por mí.
A mis tías Ana y Milagros, que esperan mucho de mí: ellas son unos de los motivos de inspiración para mi desarrollo profesional.
Por último y no menos importante, a una semillita que viene creciendo desde hace casi 3 años: mi hijo, que ha tenido que crecer la mayor parte del tiempo lejos de mí, y también a su madre Leidiglay, que en todo este tiempo mágicamente ha sabido tanto darle su cariño, como hacerle sentir el mío.
Y en general, a todas las personas que me quieren…
Adonys.

A mis padres y a mi hermanita.
Yander.

Resumen.

El Cuerpo de Investigaciones Científicas, Penales y Criminalísticas de Venezuela tiene como objetivo fundamental detectar, procesar y esclarecer hechos delictivos ocurridos en ese país. Uno de los principales problemas que atentan contra su buen desempeño es el alto nivel de corrupción existente entre sus funcionarios. Precisamente, uno de los procesos que se realizan en este cuerpo policial, es el proceso de la investigación interna, o sea, el control de la disciplina de sus funcionarios. Este proceso es dirigido por una Dependencia denominada Inspectoría General, que regula y controla el trabajo del resto de las dependencias estadales que realizan esta función.

A raíz de la creación del ALBA, se desarrolla el nuevo Sistema de Investigación e Información Policial (SIIPOL), ahora con nuevas tecnologías, prestaciones y un incremento considerable en el campo de acción de su antecesor, enfocándose principalmente en dar seguimiento a los procesos investigativos, entre ellos el disciplinario, abarcando más áreas y acelerando el flujo de información entre ellas. Para desarrollar esta solución informática se ha modelado un sistema dividido en módulos, entre ellos el de Investigación Interna, que centraliza toda la información relacionada con los procesos disciplinarios.

La presente investigación se centra en analizar, diseñar e implementar el Módulo de Investigación Interna del SIIPOL, de manera que dé cumplimiento a los requisitos funcionales y no funcionales del mismo y las especificaciones de integración con el resto del SIIPOL, haciendo énfasis en la aplicación de adecuados mecanismos de diseño sobre la base de la arquitectura establecida para el sistema.

Abstract.

The CICPC of Venezuela's main purpose is to detect, prosecute and resolve crime in that country. One of the main problems that affect their good performance is the high level of corruption among its officials. Indeed, one of the processes on the police force is the process of disciplinary investigation, control of the discipline of their officers. This process is managed by a unit called *Inspectoría General*, which regulates and controls the work of all the state agencies that perform this function along the country.

Post ALBA creation, a new SIIPOL (*Sistema de Investigación e Información Policial*) is developed, now with new technologies, benefits and an increase in the scope of its predecessor, focusing primarily on supporting different investigation processes, including disciplinary investigation, covering more areas and accelerating the flow of information between them. To develop this software solution, a system divided into modules has been modeled, *Investigación Interna* among them, which centralizes all information related to disciplinary proceedings.

This research focuses on analyzing, designing and implementing the module of *Investigación Interna*, so that it covers functional and non-functional requirement specifications, and integrates with the rest of the system, making emphasis on the application of appropriate design mechanisms based on the defined architecture.

Índice de Contenidos.

Resumen. v
Abstract. vi
Introducción. 1
Capítulo 1: Fundamentación Teórica. 4
1.1. Introducción. 5
1.2. Software de Gestión Policial. 5
1.3. El Cuerpo de Investigaciones Científicas, Penales y Criminalísticas. 6
1.3.1. Procesos de Investigación Interna. 7
1.4. El Sistema de Investigación e Información Policial. 8
1.4.1. Módulo de Investigaciones Internas. 8
1.5. Metodología, Lenguajes y Herramientas de Desarrollo. 8
1.5.1. Proceso de Desarrollo de Software. 9
1.5.2. Lenguaje de Modelado. 11
1.5.3. Lenguaje de Programación. 12
1.5.4. Plataforma de Desarrollo. 14
1.5.5. Herramientas de Desarrollo. 15
1.5.6. Frameworks utilizados en la Solución. 17
1.6. Arquitectura Técnica. 20
1.7. Conclusiones. 22
Capítulo 2: Análisis, Diseño e Implementación de la Propuesta de Solución. 23
2.1. Introducción. 24
2.2. Modelo de Análisis. 25
2.3. Modelo de Diseño. 25
2.3.1. Diagrama de paquetes. 25
2.3.2. Diagrama de clases del diseño. 26
2.3.3. Realizaciones de Casos de Uso. 36
2.4. Modelo de Datos. 37
2.5. Modelo de Implementación. 37
2.5.1. Estándar de codificación. 38

2.6. Conclusiones. 40

Capítulo 3: Validación de la Solución Propuesta. 41

3.1. Introducción. 42
3.2. Tipos y Niveles de Pruebas. 42
3.3. Resultados obtenidos. 44
3.4. Conclusiones. 45

Conclusiones. 46
Recomendaciones. 47
Bibliografía 48
Glosario de Términos. 50
Anexos. 52

Anexo 1: Estructura organizacional del CICPC. 52
Anexo 2: Procesos de Investigación Interna. 53
Anexo 3: Diagrama de despliegue. 54
Anexo 4: Diagrama de Clases de Análisis 55
Anexo 5: Diagrama de Paquetes. 63
Anexo 6: Diagramas de Clases del Diseño. 64
Anexo 7: Realizaciones de Casos de Uso. 74
Anexo 8: Modelo de Datos. 77
Anexo 9: Diagramas de Implementación. 79
Anexo 10: Relación de Pruebas y No Conformidades. 82

Índice de Figuras.

Figura 1: Proceso de Desarrollo de Software 9
Figura 2: Proceso Unificado de Desarrollo de Software..... 10
Figura 3: Aplicación Cliente-Servidor..... 21
Figura 4: Arquitectura en tres capas..... 22
Figura 5: Comportamiento de las Pruebas al Módulo de Investigaciones Internas y SIIPOL. ... 44
Figura 6: Razón entre No Conformidades y Casos de Uso probados..... 45

Introducción.

Entre las cinco ciudades con mayor índice delictivo del orbe se encuentra Caracas, Venezuela. Declarado en varias ocasiones como el país más violento del mundo, cuenta con altos niveles de inseguridad ciudadana. La delincuencia se ha ido incrementando en gran medida debido a la pérdida de valores de la población, como consecuencia de la falta de oportunidades de trabajo y respeto a sus derechos como seres humanos. En los últimos años, las cifras de crímenes y homicidios han aumentado sustancialmente; fuentes de prensa revelan que en el pasado año ocurrieron alrededor de 14600 asesinatos con un bajo porcentaje de solución (menos del 10%); lo cual, sumado a la tensa situación en las calles y la corrupción del aparato policial, ha llegado al punto de generar cierto estado de paranoia en la sociedad.

En el advenimiento de la magistratura de Hugo Rafael Chávez Frías, se inicia el proceso constitutivo para la elaboración de la nueva constitución, donde se hace un fuerte hincapié en el delicado tema de la seguridad ciudadana y el considerable aumento de la actividad delictiva en la República Bolivariana de Venezuela. Se promulga el Decreto con fuerza de Ley de los “Órganos de Investigaciones Científicas, Penales y Criminalísticas”, en el cual se establece la creación del Cuerpo de Investigaciones Científicas, Penales y Criminalísticas, más conocido por las siglas CICPC. [5]

El Cuerpo de Investigaciones Científicas, Penales y Criminalísticas de la República Bolivariana de Venezuela, a través de su determinación científica, tiene como objetivos fundamentales detectar, procesar y esclarecer hechos delictivos tales como: asesinatos, robos, estafas y crímenes en general que puedan tener lugar en el país. Uno de los principales problemas que atenta contra el buen desempeño, es el alto nivel de corrupción que existe entre los funcionarios de la institución. [1]

Los procesos que se llevan a cabo en este cuerpo policial de investigaciones, va un poco más allá de las investigaciones penales y criminalísticas, ya que no menos importante se hace llevar a cabo el proceso de la investigación interna, o sea el control de la disciplina dentro del mismo cuerpo policial, porque la cruda realidad que vive el pueblo bolivariano sólo se puede enfrentar con disciplina y responsabilidad. Este proceso es centralizado y dirigido en una Dependencia denominada Inspectoría General, que rige, controla y revisa el trabajo del resto de las dependencias estadales que realizan la misma función en su región.

El CICPC cuenta en la actualidad con un Sistema Integrado de Información Policial, que almacena y centraliza toda la información de la nación relacionada con el proceso policial que se desarrolla para controlar y aminorar la ola delictiva y la inseguridad social. En dicho sistema está incluida la información de los procesos de la investigación interna de todas las dependencias adscritas al sistema policial del CICPC, procesos muy importantes para lo que es el control de la disciplina

interna de los funcionarios del CICPC. Éste es un sistema construido con tecnología obsoleta, difícil de manejar y de mantener, además de que hay muchos aspectos del proceso que obvia o que no soporta en su totalidad.

A raíz de la creación del ALBA, y la proliferación de las relaciones Cuba-Venezuela, se lleva a cabo la implementación del Sistema de Investigación e Información Policial (SIIPOL), ahora con nuevas tecnologías y prestaciones. Este nuevo sistema incrementa considerablemente el campo de acción y las prestaciones del antiguo, enfocándose principalmente en dar seguimiento a los procesos investigativos, entre ellos el disciplinario (o interno), abarcando más áreas y conectando las delegaciones estadales entre sí y con Inspectoría General, acelerando el flujo de información entre ellas. Esto propiciará un crecimiento en la rapidez de la investigación de las faltas disciplinarias y un aumento en el control de la disciplina interna en el cuerpo policial; para así hacer más eficiente el trabajo en el esclarecimiento de los hechos delictivos y en la solución de los casos investigativos penales.

A partir de un análisis exhaustivo del funcionamiento de la institución, así como sus procesos de negocio, y con la aplicación de la Ingeniería de Requerimientos, se generó un conjunto de requisitos funcionales y no funcionales, que resumen las capacidades que debe cumplir el nuevo sistema y que dan al traste con el antiguo Sistema Integrado de Información Policial. Para llevar a cabo la implementación de esta solución informática se ha modelado un sistema dividido en módulos, uno de los cuales es el de Investigaciones Internas; que controle, administre y centralice toda la información relacionada con los procesos disciplinarios.

Dada esta *situación,* el *problema científico* de la presente investigación está dado a partir de la siguiente cuestión: ¿Cómo garantizar el cumplimiento de los requisitos funcionales y no funcionales asociados al Módulo Investigación Interna del Sistema de Investigación e Información Policial?

El *objeto de estudio* establecido para el presente trabajo de diploma se basa en el proceso de desarrollo de software en Sistemas de Gestión de Información Policial. Y el *campo de acción* regido por la idea anterior se enmarca en el análisis, diseño e implementación del Módulo de Investigaciones Internas, perteneciente al sistema de gestión de información SIIPOL.

La *idea a defender* se presenta de la siguiente forma: “La metodología de desarrollo seleccionada para el análisis, diseño e implementación del Módulo Investigaciones Internas, garantizará el cumplimiento de los requisitos funcionales y no funcionales obtenidos en la Ingeniería de Requerimientos”.

El *objetivo general* es analizar, diseñar e implementar el Módulo de Investigaciones Internas del SIIPOL; de manera que dé cumplimiento a los requisitos funcionales y no funcionales, obtenidos como resultado de aplicar la Ingeniería de Requerimientos a los procesos de la Investigación Interna en el CICPC.

Los *objetivos específicos* son:

- Evaluar los resultados derivados de la aplicación de la Ingeniería de Requerimientos.
- Analizar, definir, estructurar y detallar mecanismos de diseño que garanticen el cumplimiento de las necesidades del Módulo de Investigaciones Internas.
- Implementar el Módulo diseñado respetando la arquitectura definida para el SIIPOL.
- Lograr la integración satisfactoria al SIIPOL.

Para dar cumplimiento a estos objetivos, se han definido una serie de *tareas investigativas*:

1. Investigar sobre otros sistemas de gestión de información.
2. Investigar sobre sistemas de gestión de información policial.
3. Estudiar metodología, herramientas y lenguaje para la solución del problema.
4. Realizar el diseño teórico de la investigación.
5. Analizar requisitos funcionales y no funcionales obtenidos de la Ingeniería de Requerimientos.
6. Estudiar el modelo de los Casos de Uso.
7. Modelar el análisis de los Casos de Uso.
8. Realizar el diseño de clases derivado del modelo de análisis.
9. Refinar el diseño utilizando patrones.
10. Implementar la propuesta de solución.
11. Probar los componentes desarrollados.
12. Integrar la solución al Sistema de Investigación e Información Policial a medida que se obtienen resultados satisfactorios.
13. Validar la solución desarrollada.

Con el desarrollo exitoso de las tareas investigativas anteriormente expuestas se espera lograr un Módulo funcional, que cumpla con los estándares de calidad requeridos y satisfaga a totalidad el conjunto de requerimientos funcionales y no funcionales, arrojados a partir de la aplicación de la Ingeniería de Requerimientos a los procesos de la Inspectoría General y las Delegaciones Estadales.

Capítulo 1: Fundamentación Teórica.

1.1. Introducción.

Se establecerán en este capítulo una serie de conceptos y definiciones que darán fundamento a la utilización de algunas de las tendencias actuales, dentro de una inmensa gama de buenas y nuevas normas que se explotan en el mundo del desarrollo de software, las cuales le dan solución a problemáticas y necesidades análogas a las que se muestran en este trabajo.

1.2. Software de Gestión Policial.

Sistemas de Gestión de Información.

Los sistemas de gestión de información constituyen actualmente una alternativa de imprescindible presencia en cualquier organización o empresa. Hoy en día en cada una de estas instituciones se maneja un gran flujo de información, tanto física como digital, haciéndose necesaria la gestión, el control y una forma rápida, segura y concisa de llegar a la información. Un sistema de gestión de la información es un sistema integrado, que posee la capacidad de gestionar los procesos del flujo de la información generados tanto dentro como fuera de la organización o empresa, mejorando así de forma gradual su posición con respecto a las demás instituciones de la misma rama.

En este nuevo mundo global y competitivo se debe tener en cuenta que las instituciones se enfrentan a un reto diario, donde el conocimiento, el descubrimiento, y la constante revolución de las técnicas y tecnologías es vital para lograr una ventaja competitiva.

Sistemas de Gestión de Información Policial.

Según lo expuesto anteriormente un sistema de gestión de información policial comparte el mismo concepto de un sistema de gestión de información, cuya característica primordial es la confidencialidad de los datos e información que maneja el sistema.

La utilidad de este tipo de sistema tiene una gran repercusión en la sociedad, pues los índices de delincuencia e indisciplina social en el mundo están muy altos. Este uso implica almacenamiento y control de la información recolectada por años, dando la posibilidad de rápidas consultas y avanzadas estadísticas que influyen positivamente en el desarrollo posterior de nuevas investigaciones, logrando agilizar las posibles respuestas a estas problemáticas. Como antes se mencionaba los sistemas de gestión de información policial proveen de gran confidencialidad de la información, pues permite el acceso a ésta sólo a las personas o entidades que tengan los suficientes permisos para realizar alguna acción deseada.

Sistema Integrado de Información Policial.

El Sistema Integrado de Información Policial es el sistema de gestión de información policial que posee actualmente el Cuerpo de Investigaciones Científicas Penales y Criminalísticas (CICPC) de la República Bolivariana de Venezuela. Este sistema cuenta con una serie de limitaciones tanto en el campo de acción como en las tecnologías usadas para el desarrollo del mismo, lo que dificulta el manejo y la rapidez del flujo de la información, además de que no se abarcan todos los procesos que se llevan a cabo en el cuerpo policial. Por otra parte las tecnologías usadas para la confección del mismo son obsoletas, se utilizó como lenguaje de programación Adabas-Natural y como gestor de base de datos SUN 6500, además de que está implementado de forma inflexible, lo que provoca limitaciones a la hora de poder realizar algún cambio o poder dar soporte al software que actualmente se explota.

Debido a la proliferación de las relaciones entre Cuba y Venezuela en el área de la informática, se negoció la propuesta de un nuevo sistema que englobe todos los procesos del CICPC y que permita un mejor manejo de la información, ahora con tecnologías más avanzadas que abran el camino de la reusabilidad y modificación del sistema en cualquier momento. Entre las nuevas perspectivas que tiene este nuevo sistema está la de centrarse más en el campo de los procesos de investigación del CICPC.

1.3. El Cuerpo de Investigaciones Científicas, Penales y Criminalísticas.

El CICPC es una institución que garantiza la eficiencia en la investigación del delito, mediante su determinación científica, asegurando el ejercicio de la acción penal que conduzca a una sana administración de justicia. Su misión se concentra en ser la institución indispensable, por su reconocida capacidad científica y máxima excelencia de sus recursos, con la finalidad de alcanzar el más alto nivel de credibilidad nacional e internacional en la investigación del fenómeno delictivo organizado y criminalidad violenta.

El CICPC ostenta entre sus objetivos fundamentales el de optimizar las acciones de investigación criminal con el fin de lograr el esclarecimiento de los hechos delictivos; capacitar el personal integrado en la institución, garantizando de esta forma la eficacia y la eficiencia; elevar el sentido de pertenencia a partir de la práctica de valores indispensables para el CICPC; garantizar las acciones y medios tendentes a mejorar la calidad de vida de sus miembros, en el aspecto educativo, cultural, deportivo, social y económico; consolidar la imagen de profesionalismo de la institución ante la

comunidad en general, fundamentada en una gerencia de alta capacidad de respuesta; lograr insertarse en la comunidad internacional como organismo de investigación penal de vanguardia; dotar al capital humano del CICPC de herramientas, mecanismos logísticos y de infraestructura que garanticen el óptimo desempeño de sus funciones; apoyar las políticas de Estado a través de estrategias dirigidas a la reducción de los delitos en todas sus modalidades; fortalecer organizacionalmente la institución y su sinergia con otros organismos de la Administración Pública Nacional y con instituciones privadas. (Ver Anexo 1)

El CICPC cuenta con el Sistema Integrado de Información Policial como aplicación Informática. Esta, en la actualidad se considera ineficiente ya que no cubre todos los procesos que se realizan a diario en la institución, muchos de ellos no se encuentran automatizados. La tecnología obsoleta que presenta, el alto costo en tiempo de respuesta a las peticiones, así como su interfaz poco amigable para los usuarios que interactúan con ella, dificultan más que agilizar las diversas operaciones que se efectúan dentro del Cuerpo de Investigación e Información Policial. A partir de la información antes mencionada se patentiza la necesidad de contar con sistema con tecnología de punta y que brinde solución instantánea a los diferentes problemas por los que atraviesa actualmente la institución.

1.3.1. Procesos de Investigación Interna.

Las dependencias subordinadas a Inspectoría General, se encargan junto a ésta de llevar a cabo los procesos de Investigación Interna del CICPC. En orden jerárquico se pueden nombrar la Dirección de Investigaciones Internas y las Delegaciones Estadales, teniendo estas últimas un marco de trabajo a nivel estatal.

Inicio de la Averiguación Preliminar: Las Averiguaciones Preliminares son el principio de muchas investigaciones disciplinarias, pueden iniciarse por una denuncia (verbal o escrita) u oficio (documento que da la orden de inicio de la investigación). Estas constituyen una investigación previa a la acusación del funcionario comprometido y centran su atención en demostrar que éste debe ser investigado. Este proceso tiene un plazo máximo de duración de 30 días.

Inicio de la Investigación Disciplinaria: Resultante de este proceso se obtiene un Expediente Disciplinario, puede ser iniciado por denuncia, oficio o por la conclusión de una Averiguación Preliminar. Esta investigación puede concluir con un proceso disciplinario y una sanción. Este proceso tiene un plazo máximo de duración de 90 días.

Sustanciación de Expedientes: Es el proceso que más trabajo y tiempo requiere pues constituye la investigación en sí. En este se realizan las diligencias (solicitudes de experticias,

experticias, entrevistas, inspecciones al sitio del suceso, etc.) necesarias para probar la veracidad de la acusación. La duración de este proceso depende y es siempre menor que el proceso al que se aplique, puede ser cualquiera de los dos anteriores.

Revisión y Aprobación: Es aplicable también a cualquiera de los dos primeros procesos descritos y por ende su tiempo es dependiente de ellos. Este proceso mantiene la integridad y correcta organización de los expedientes antes de ser remitidos a Inspectoría General.

Control de Investigación: Se realiza por los funcionarios adscritos a Inspectoría General y mantiene un control constante sobre los cambios que se hacen a los expedientes por parte de los investigadores. Al final de este proceso se decide si los funcionarios implicados deben ser investigados o procesados por el Consejo Disciplinario del CICPC.

De modo general, estos procesos pueden ser representados gráficamente para lograr una mejor comprensión de los mismos. (Ver Anexo 2)

1.4. El Sistema de Investigación e Información Policial.

Este nuevo sistema de gestión de información policial es la solución a las limitaciones antes mencionadas del sistema vigente en el CICPC. Éste pretende cubrir todos los procesos, que no estaban incluidos en la primera solución, incorporando al sistema muchas delegaciones cuyas características no le permitían la inclusión de las mismas en el sistema actualmente explotado. Además de esto, maximizar el flujo de la información entre estas delegaciones, y pretendiendo disminuir el tiempo de respuesta en el procesamiento de la información, las respuestas a consultas y la creación de estadísticas.

1.4.1. Módulo de Investigaciones Internas.

Investigaciones Internas constituye un módulo del SIIPOL. Este brinda funcionalidades que permiten iniciar, controlar y sustanciar los distintos tipos de investigaciones que se manejan como parte de los procesos de las dependencias de Inspectoría General, la Dirección de Investigaciones Internas y las Delegaciones Estadales. El módulo cuenta con 47 Casos de Uso y representa aproximadamente un 20% del total del sistema.

1.5. Metodología, Lenguajes y Herramientas de Desarrollo.

Se hace importante hacer una buena selección de las metodologías, lenguajes y herramientas de desarrollo a utilizar en la elaboración del sistema SIIPOL. En este segmento se hace referencia a

las principales características, funcionalidades y facilidades que brinda cada uno de los elementos seleccionados.

1.5.1.Proceso de Desarrollo de Software.

Un proceso de desarrollo de software tiene como propósito la producción eficaz y eficiente de un producto software, que reúna los requisitos del cliente. Dicho proceso, en términos globales se muestra en la Figura 1. El Proceso de Desarrollo del Software identifica cuál es el problema, analiza qué debe hacerse, diseña como podemos solucionar dicha problemática, hace la transferencia del diseño a código fuente, realiza pruebas, lo implanta en un entorno productivo y garantiza el soporte a posibles mejoras o cambios.

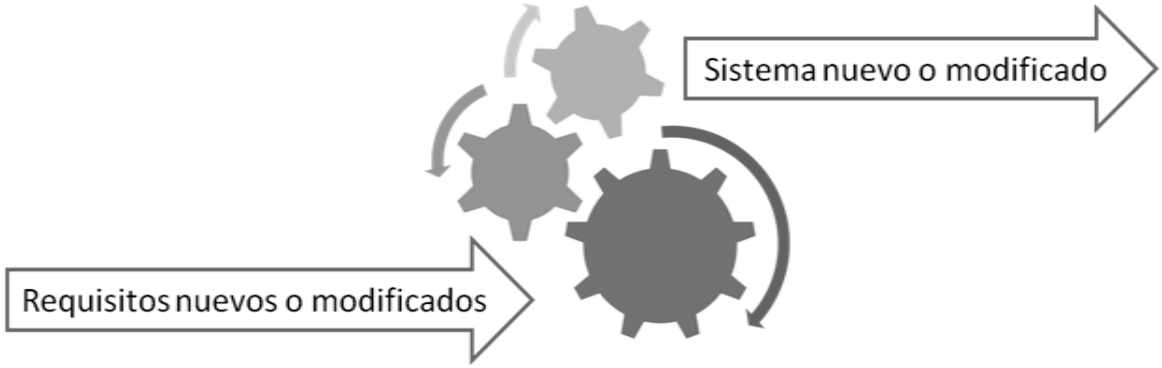


Figura 1: Proceso de Desarrollo de Software

Para cumplir estos propósitos con la calidad y eficiencia requerida, y con el fin de poder lograr un producto final que cumpla con las expectativas del cliente, se ha desarrollado para la industria del software cierta variedad de metodologías, que llevan a cabo de una manera u otra el cumplimiento de las fases del proceso de desarrollo del software. En lo adelante se describe la metodología utilizada para resolver la problemática que se expone en este trabajo.

Proceso Unificado de Desarrollo (RUP).

El Proceso Unificado de Desarrollo del Software o simplemente Proceso Unificado es un marco de desarrollo de software, que se caracteriza por estar dirigido por casos de uso, centrado en la arquitectura y por ser iterativo e incremental. El refinamiento más conocido y documentado del Proceso Unificado es el Proceso Unificado de Rational o simplemente RUP.

El Proceso Unificado de Rational no es simplemente un proceso, sino un marco de trabajo extensible que puede ser adaptado a organizaciones o proyectos específicos. De la misma forma, el

Proceso Unificado de Rational, cumple con las características antes mencionadas, por lo que muchas veces resulta imposible decir si un refinamiento particular del proceso ha sido derivado del Proceso Unificado o del RUP. Por dicho motivo, los dos nombres suelen utilizarse para referirse a un mismo concepto.

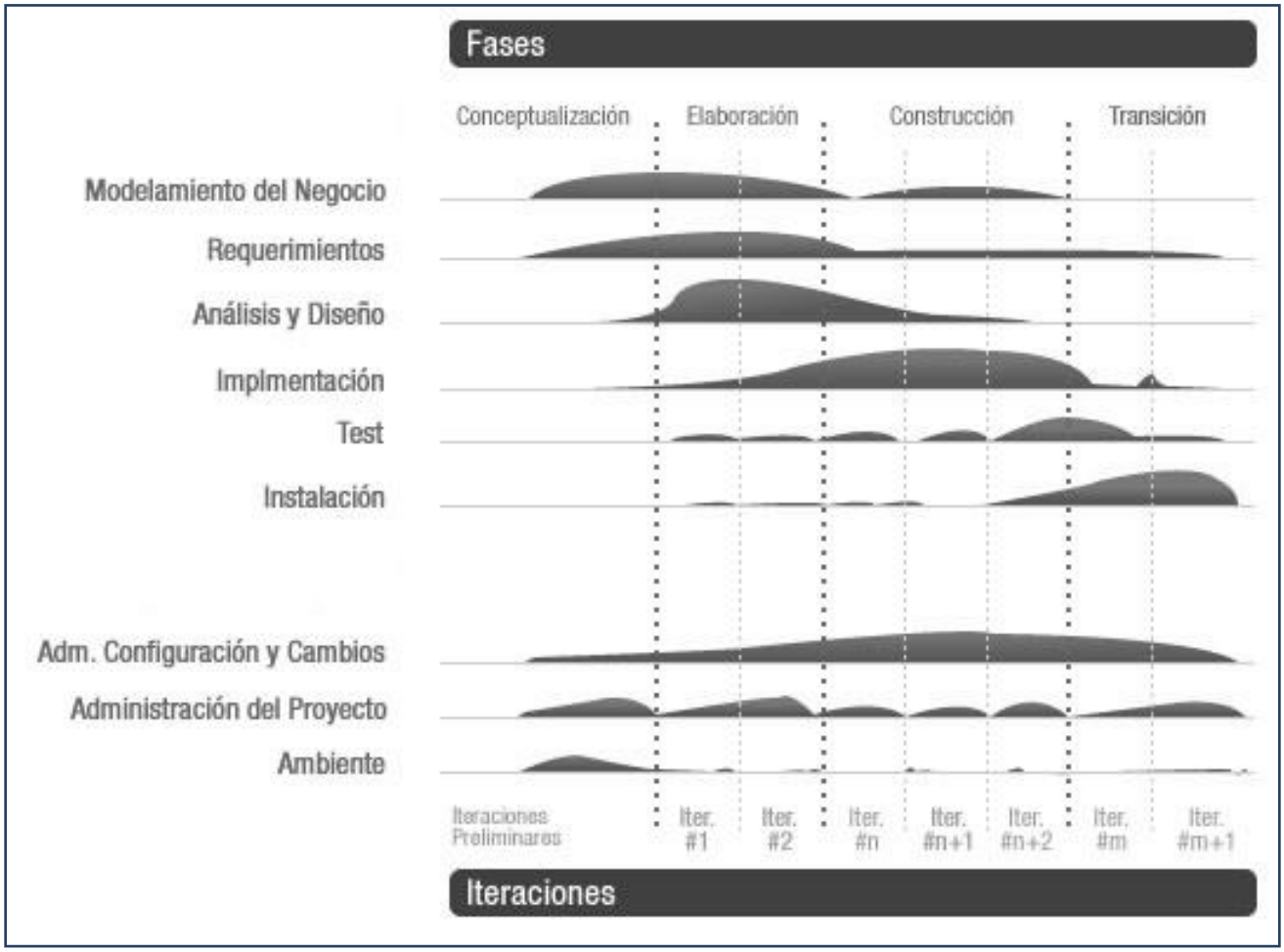


Figura 2: Proceso Unificado de Desarrollo de Software.

El Proceso Unificado tiene dos dimensiones:

1. Un eje horizontal que representa el tiempo y muestra los aspectos del ciclo de vida del proceso a lo largo de su desenvolvimiento
2. Un eje vertical que representa las disciplinas, las cuales agrupan actividades de una manera lógica de acuerdo a su naturaleza.

La primera dimensión representa el aspecto dinámico del proceso conforme se va desarrollando, se expresa en términos de fases, iteraciones e hitos.

La segunda dimensión constituye el aspecto estático del proceso: cómo es descrito en términos de componentes del proceso, disciplinas, actividades, flujos de trabajo, artefactos y roles.

Como se mencionaba anteriormente el ciclo de vida de RUP se caracteriza por ser:

Dirigido por casos de uso: Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso (cómo se llevan a cabo).

Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, pues describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. RUP se desarrolla mediante iteraciones, comenzando por los Casos de Uso (CU) relevantes desde el punto de vista de la arquitectura.

Iterativo e Incremental: Aunque la Figura 2 puede sugerir que los flujos de trabajo se desarrollan en cascada, la “lectura” de este gráfico tiene que ser vertical y horizontal. RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Por ejemplo, una iteración de elaboración centra su atención en el análisis y diseño, aunque refina los requerimientos y obtiene un producto con un determinado nivel, pero que irá creciendo incrementalmente en cada iteración.

Es práctico dividir el trabajo en partes más pequeñas o miniproyectos. Cada miniproyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto. Cada iteración se realiza de forma planificada es por eso que se dice que son miniproyectos.

Ya que RUP disminuye riesgos, se comporta de forma iterativa, utiliza componentes, aminora el tiempo de su realización y realiza pruebas constantemente, podemos asegurar la calidad del software. Es por estas razones que se tomó RUP como la metodología de desarrollo que reúne las condiciones pertinentes para ser parte de la solución del problema en cuestión en este trabajo.

1.5.2. Lenguaje de Modelado.

Un lenguaje de modelado es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar un diseño de software mayormente orientado a objetos. Algunas

organizaciones los usan extensivamente en combinación con una metodología de desarrollo de software, para avanzar de una especificación inicial a un plan de implementación y para comunicar dicho plan a todo un equipo de desarrolladores.

Lenguaje Unificado de Modelado (UML).

UML, conocido por sus siglas en inglés, Unified Modeling Language, en español: Lenguaje de Modelación Unificada, es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

Es importante resaltar que UML es un lenguaje para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Rational o RUP), pero no especifica en sí mismo qué metodología o proceso usar. Teniendo en cuenta la estrecha relación que existe entre RUP y UML, además de todas las características aquí referidas, no quedó duda a la hora de escoger como lenguaje de modelado a UML.

1.5.3.Lenguaje de Programación.

Un lenguaje de programación está constituido por un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones; que se pone a disposición del programador para que éste pueda comunicarse con los dispositivos hardware y software existentes, por lo que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Los lenguajes de programación pueden ser clasificados atendiendo a varios criterios, los cuales pueden ser: su nivel de abstracción o paradigma.

Lenguaje Java.

Java es un lenguaje de programación desarrollado por Sun Microsystems Inc. a principios de los años 90. Este lenguaje toma mucha de su sintaxis de C y C++, aunque tiene un modelo de objetos mucho más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación de punteros o memoria. Aunque a diferencia de éste, que combina la sintaxis para

programación genérica, estructurada y orientada a objetos, Java fue construido desde el principio para ser completamente orientado a objetos.

Además de ser un lenguaje orientado a objetos, otra cualidad muy importante es la independencia de la plataforma, esto significa que programas escritos en el lenguaje Java pueden ser ejecutados de igual forma en cualquier tipo de hardware. Es lo que posibilita ser capaz de escribir un programa una vez y que pueda ser ejecutado en dispositivos como computadoras, teléfonos móviles, tarjetas inteligentes, sintonizadores, impresoras, cámaras web, juegos o sistemas de navegación para automóviles; tal como afirma el axioma de Java, *"write once, run everywhere"*. El concepto de independencia de la plataforma de Java cuenta con un gran éxito en las aplicaciones en el entorno del servidor, como los Servicios Web, los Servlets y los Java Beans.

Otra de las ventajas que incluye Java es la incorporación del Recolector de Basura, éste es un mecanismo implícito de gestión de memoria, el cual permite que se eviten errores y fugas de memoria muy comunes en otros lenguajes de programación. La recolección de basura de Java es un proceso prácticamente invisible al desarrollador, es decir, el programador no tiene conciencia de cuándo la recolección de basura tendrá lugar, ya que ésta no tiene necesariamente que guardar relación con las acciones que realiza el código fuente. Debe tenerse en cuenta que la memoria es uno de los principales recursos que deben ser gestionados eficientemente.

En la parte del servidor, Java es más popular, principalmente desde la aparición de la especificación de Servlets y JSP (Java Server Pages). Hasta entonces, las aplicaciones web dinámicas de servidor que existían se basaban fundamentalmente en componentes CGI y lenguajes interpretados. Ambos tenían diversos inconvenientes entre los que se destaca fundamentalmente la lentitud, elevada carga computacional o de memoria y propensión a errores por su interpretación dinámica.

La versatilidad y eficiencia de la tecnología Java, la portabilidad de su plataforma y la seguridad que aporta, la han convertido en la tecnología ideal para todo tipo de aplicaciones. Java ha sido probado, mejorado, ampliado y probado por una comunidad especializada de más de 6,5 millones de desarrolladores, la mayor y más activa del mundo. [4]

Por características como robustez, seguridad, independencia de la arquitectura, portabilidad y sencillez, además de distinguirse como lenguaje de programación apropiado para aplicaciones de gran envergadura, es escogido para el desarrollo del SIIPOL, con el objetivo de poder garantizar un sistema seguro y con altos niveles de eficiencia.

1.5.4. Plataforma de Desarrollo.

En un proyecto de software, para seleccionar la plataforma en la que se desarrollará el mismo, se debe tomar en cuenta el entorno de ejecución, el rendimiento, la escalabilidad, la portabilidad y la seguridad.

Plataforma Java.

La Plataforma Java es el nombre de un entorno o plataforma de computación originaria de Sun Microsystems Inc., capaz de ejecutar aplicaciones desarrolladas usando el Lenguaje de programación Java u otros lenguajes que compilen a bytecode y un conjunto de herramientas de desarrollo. En este caso, la plataforma no es un hardware específico o un sistema operativo, sino más bien una máquina virtual encargada de la ejecución de aplicaciones, y un conjunto de librerías estándar que ofrecen funcionalidad común. La Plataforma Java se compone de un amplio abanico de tecnologías, cada una de las cuales ofrece una parte del complejo de desarrollo o del entorno de ejecución en tiempo real. Además, las aplicaciones Java pueden usarse de forma variada, como por ejemplo ser incrustadas en una página Web.

El corazón de esta plataforma es el concepto común de un procesador virtual que ejecuta programas escritos en el lenguaje de programación Java. Este procesador es la máquina virtual de Java o JVM (Java Virtual Machine), que se encarga de traducir el bytecode en instrucciones nativas de la plataforma destino. Esto permite que una misma aplicación Java pueda ser ejecutada en una gran variedad de sistemas con arquitecturas distintas, siempre que se cuente con una implementación adecuada de la JVM.

Cuando Sun Microsystems Inc. decidió lanzar su nuevo estándar Java, llamado Java2, creó tres diferentes entornos para desarrollo y ejecución de aplicaciones. Éstos fueron J2SE, J2EE y J2ME.

J2SE (Java 2 Standard Edition) es la base de la tecnología Java. Permite el desarrollo de applets (aplicaciones que se ejecutan en un navegador web) y aplicaciones independientes (standalone). J2SE es el heredero directo del Java inicial. J2EE (Java 2 Enterprise Edition) está basado en J2SE, pero añade una serie de características necesarias en entornos empresariales, relativos a redes, acceso a datos y entrada/salida que requieren mayor capacidad de proceso, almacenamiento y memoria. La decisión de separarlos es debido a que no todas estas características son necesarias para el desarrollo de aplicaciones estándar.

Al igual que J2EE cubre unas necesidades más amplias que J2SE, se hace patente la necesidad de un subconjunto de J2SE para entornos más limitados. La respuesta de la empresa es J2ME (Java 2 Micro Edition), esta se basa en los conceptos de configuración y perfil. Una

configuración describe las características mínimas en cuanto a la configuración hardware y software. La configuración que usa J2ME es la CLDC (Connected Limited Device Configuration). Concretamente CLDC define cuáles son las características del lenguaje Java incluidas, qué funcionalidad será incluida en la máquina virtual de Java, las APIs necesarias para el desarrollo de aplicaciones en dispositivos móviles y los requerimientos Hardware de estos dispositivos.

Uno de los mayores beneficios de Java EE como plataforma es que puede comenzarse con poco o ningún coste. La implementación Java EE de Sun Microsystems Inc. puede ser descargada gratuitamente, y hay muchas herramientas de código abierto disponible para extender la plataforma o para simplificar el desarrollo.

Después de conocer todo esto, se puede concluir que la decisión más provechosa es sin lugar a dudas, la plataforma J2EE; por esta razón, se toma como plataforma de desarrollo para el SIIPOL.

1.5.5.Herramientas de Desarrollo.

En todo proceso de desarrollo de software se hace necesario el uso de herramientas. Estas facilitan y aumentan cuantiosamente la velocidad de la producción.

Herramienta Case Visual Paradigm for UML.

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadoras) son diversas aplicaciones informáticas concebidas para aumentar el rendimiento y la productividad en el desarrollo de software, reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software, en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores, entre otras.

Visual Paradigm for UML es una herramienta CASE profesional que utiliza UML 2.1 como lenguaje de modelado. Soporta el ciclo de vida completo del desarrollo de software: levantamiento de negocio, requisitos, análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite elaborar todos los diagramas y artefactos del Proceso Unificado, además de generar código fuente desde ellos u obtenerlos mediante ingeniería inversa, y generar la documentación del proyecto automáticamente en varios formatos como Web o .PDF. Esta herramienta CASE también brinda abundantes tutoriales de UML, demostraciones interactivas y proyectos de ejemplo; además de proporcionar una fuerte y fácil integración con otras herramientas de desarrollo

como Eclipse, ofrece aseguramiento para el control de versiones, logrando así acelerar al máximo la unión de las contribuciones individuales de cada uno de los integrantes del equipo de desarrollo. [19]

Por sus múltiples y relevantes prestaciones fue seleccionado Visual Paradigm for UML como herramienta CASE para el desarrollo del SIIPOL.

Entorno de Desarrollo Integrado Eclipse.

Un Entorno de Desarrollo Integrado, o en inglés Integrated Development Environment (IDE), es un programa compuesto por un conjunto de herramientas para un programador. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación; es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de otras aplicaciones existentes. Es posible que un mismo IDE pueda funcionar con varios lenguajes de programación, éste es el caso de Eclipse, al cual se le puede añadir soporte a lenguajes adicionales mediante plug-ins.

Eclipse fue desarrollado originalmente por IBM y en la actualidad por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios. Eclipse es principalmente una plataforma de programación, usada para crear entornos integrados de desarrollo (IDEs), aunque la definición que da el proyecto Eclipse acerca de su software es: "una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular". [20]

El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código.

El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la plataforma de cliente rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente le permite a Eclipse extenderse usando otros lenguajes de programación y diversas funcionalidades.

Por lo antes dicho es que se hace posible encontrar otros entornos basados en Eclipse para el desarrollo de aplicaciones J2EE, en los cuales se agrupan plug-ins afines a algún tipo específico de arquitectura de software, brindando así grandes prestaciones en cuanto a la rapidez en el proceso de desarrollo.

Conociendo todos estos argumentos, resulta fácil seleccionar Eclipse para el desarrollo del SIIPOL, con el objetivo de brindar un ambiente de desarrollo cómodo y fácilmente extensible.

1.5.6. Frameworks utilizados en la Solución.

Framework es un concepto sumamente genérico, se refiere a “ambiente de trabajo, y ejecución”. En general los framework son soluciones completas que contemplan herramientas de apoyo a la construcción (ambiente de trabajo o desarrollo) y motores de ejecución (ambiente de ejecución).

Framework de Presentación.

Es *Java Server Faces (JSF*) un framework de componentes (UI) del lado del servidor para aplicaciones Web basadas en la tecnología Java, es un framework para construir interfaces de usuario en aplicaciones web. JSF está compuesto por: Componentes UI definidos por etiquetas y XML, Eventos, Validadores y posee relación con datos de la lógica. JSF es diseñado para ser integrado con herramientas de desarrollo como: Java Sun Studio Creator, JDeveloper, Netbeans y Eclipse. JSF nos ofrece un marco de trabajo que facilita el desarrollo de aplicaciones, separando las diferentes capas de una arquitectura: presentación, reglas y entidades de negocio.

JSF fue creado dentro del Java Community Process de SUN, en el que han participado líderes de la industria como Oracle, BEA, IBM, etc. Es el framework oficial de SUN para el desarrollo de aplicaciones.

Se decide utilizar este framework porque cumple con el patrón arquitectónico Modelo Vista Controlador para aplicaciones web, posee una clara separación de roles, una gama de componentes extensibles a diferencia de JSP/Servlets, ofrece un mejor comportamiento y presentación que JSP, provee varias librerías de etiquetas para acceder y manipular los componentes. Además almacena automáticamente la información de los formularios actualizando el mismo en el momento de ser mostrado al lado del cliente y también encapsula la lógica de manipulación de los eventos y la forma en que se muestran los componentes de los desarrolladores, los cuales pueden limitarse a usar los componentes ya definidos.

Framework para lógica de negocio.

La primera versión de *Spring* fue escrita por Rod Johnson, que lo liberó cuando publicó su libro “Expert One-on-One J2EE Design and Development (Wrox Press, Octubre 2002)”. La versión 1.2.6 de *Spring* ganó un premio “Jolt de productividad” en el 2006. El framework comenzó su vida comercial en el 2002/2003 bajo la dirección de Rod Johnson y Juergen Holler, pero la primera versión completamente estable fue liberada en Marzo 2004. En este mismo año Spring comienza a considerarse como uno de los líderes en el campo de los frameworks full-stack en la plataforma Java/J2EE.

Spring provee soluciones a muchos problemas a los que se enfrentan los desarrolladores de Java, que quieren crear aplicaciones basadas en esta plataforma. El framework no se limita a la plataforma Java (existe una versión llamada Spring.Net para la plataforma de Microsoft), pero su mayor popularidad si se debe a su gran influencia en el desarrollo actual de Java.

Spring es probablemente más conocido por ofrecer funcionalidades requeridas para crear aplicaciones empresariales complejas, de forma efectiva por fuera de los modelos históricamente dominantes en la industria sobre la plataforma Java. Además de ello, se hizo conocido por introducir técnicas y funcionalidades anteriormente existentes sólo como posibilidad teórica o al menos no explotada dentro de las prácticas principales del desarrollo del software. Ésto resulta en un framework que ofrece un modelo consistente y hace que sea aplicable a la mayor parte de las aplicaciones que se pueden crear en la plataforma Java.

Framework de Acceso a Datos.

Manejar los datos de persistencia es la clave para las decisiones en un proyecto de software, dado que los mismos son un requerimiento usual de las aplicaciones. Toda solución tiene ventajas y desventajas, y no todas comparten el mismo ámbito ya que se ofrecen diferentes soluciones para un mismo problema, por lo que hay que valorar muy bien la solución de persistencia a usar en un proyecto de software.

Hay muchos debates en el mundo acerca de cómo persistir los objetos, y entre ellos se debate mucho el uso de las herramientas de persistencias como una buena solución para llevar a cabo la implementación del acceso a datos de un proyecto de software. Una de estas buenas soluciones es la utilizada para la ejecución de la capa de acceso a datos de la problemática en cuestión en dicho trabajo, que es el framework de Hibernate, el cual es una implementación ORM.

Cuando se hace referencia a la persistencia se habla de que es uno de los conceptos fundamentales en el desarrollo de aplicaciones. Si un sistema de información no preserva los datos entrados por los usuarios, cuando cierre el sistema operativo en el que se ejecuta la aplicación, se perderán todos los datos, y el sistema no se mantendrá igual cuando se vuelva a iniciar.

Hibernate es una herramienta que trata de dar una solución completa al problema del manejo de la persistencia de datos, mediando la interacción de la aplicación con las bases de datos relacionales mediante el mapeo objeto-relacional para la plataforma Java (y disponible también para .Net con el nombre de NHibernate), que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones.

Por estos motivos, y por muchas facilidades más es por lo que se decidió la utilización de esta herramienta para la solución, algunas otras buenas utilidades son como por ejemplo que: Hibernate posee un lenguaje propio llamado HQL para realizar consultas a la base de datos, lo permite que no haya que cambiar nada de código para cambiar de Sistema Gestor de Base de Datos, sólo cambiar un parámetro en el fichero de configuración de Hibernate, por otra parte pertenece a software libre y distribuido bajo los términos de la licencia GNU LGPL. Además, este framework implementa una especie de caché, guardando datos en su sesión; esto posibilita liberar la carga a la base de datos y permitir así multitud de consultas simultáneas sin colapsar la base de datos.

Framework para pruebas de Caja Blanca.

Es *JUnit* un conjunto de bibliotecas creadas por Erich Gamma y Kent Beck que son utilizadas en programación para hacer pruebas unitarias de aplicaciones Java. Es un framework que permite realizar la ejecución de clases Java de manera controlada, para poder evaluar el funcionamiento de cada uno de los métodos de la clase y comprobar el comportamiento esperado. Es decir, en función de algún conjunto de datos de entrada se evalúa el valor de retorno esperado; si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente. Es también un medio de controlar las pruebas de regresión, necesarias cuando una parte del código ha sido modificado y se desea ver que el nuevo código cumple con los requerimientos anteriores y que no se ha alterado su funcionalidad después de la nueva modificación.

En la actualidad las herramientas de desarrollo como NetBeans y Eclipse cuentan con módulos, los cuales permiten que la generación de las plantillas necesarias para la creación de las pruebas de una clase Java se realice de manera automática, facilitando al programador enfocarse en la prueba y el resultado esperado, y dejando a la herramienta la creación de las clases necesarias para ello.

El objetivo de las pruebas unitarias es aislar cada parte del programa y mostrar que las partes individuales son correctas. Proporcionan un contrato escrito que el trozo de código debe satisfacer. Estas pruebas aisladas proporcionan cinco ventajas básicas:

1. Fomentan el cambio: Las pruebas unitarias facilitan que el programador cambie el código para mejorar su estructura (lo que se ha dado en llamar refactorización), puesto que permiten hacer pruebas sobre los cambios y así asegurarse de que estos no han introducido errores.

2. Simplifica la integración: Puesto que permiten llegar a la fase de integración con un grado alto de seguridad de que el código está funcionando correctamente. De esta manera se facilitan las pruebas de integración.
3. Documenta el código: Las propias pruebas son documentación del código puesto que ahí se puede ver cómo utilizarlo.
4. Separación de la interfaz y la implementación: Dado que la única interacción entre los casos de prueba y las unidades bajo prueba son las interfaces de estas últimas, se puede cambiar cualquiera de los dos sin afectar al otro, a veces usando objetos mock (implementación falsa que simula el comportamiento de un objeto verdadero) para simular el comportamiento de objetos complejos.
5. Los errores: Están más acotados y son más fáciles de localizar, dado que se tienen pruebas unitarias que pueden desenmascararlos.

Son estas ventajas y todas las razones anteriores las que conllevan a la selección de esta herramienta para mantener el control y la calidad del software que se desarrollará.

Framework para la generación de Reportes.

Jasper Report es la librería para la generación de informes que ha sido utilizada en la solución de la problemática descrita, ya que es la librería más popular de código libre para la plataforma Java en el mundo. Se puede incluir fácilmente en cualquier aplicación de Java para la creación de sofisticados reportes. Se puede utilizar además para crear archivos de salida que se manejarán en el futuro, como hojas de cálculo.

Brinda las funcionalidades necesarias para dar cumplimiento a los requisitos impuestos. Provee una abundante fuente de datos, dando incluso la posibilidad de utilizar varios tipos de esta en un mismo reporte. El funcionamiento consiste en escribir un XML donde se recogen las particularidades del informe. Este XML lo tratan las clases del framework para obtener una salida. Esta salida puede ser un PDF, XML, HTML, CSV, XLS, RTF o TXT.

1.6. Arquitectura Técnica.

El desarrollo de este tipo de productos ha evolucionado a lo largo del último cuarto del siglo pasado, y principios del presente. La marcada tendencia al desarrollo de aplicaciones web e Internet ha llevado a la construcción de sistemas más grandes y complejos donde se requiere de mayor eficiencia, facilidad de mantenimiento y uso, interoperabilidad, seguridad y mejor tiempo de respuesta.

Una aplicación web es un sistema informático que los usuarios utilizan accediendo a un servidor a través de Internet o de una intranet. Las aplicaciones web son muy populares debido a la practicidad del navegador web como cliente ligero y la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software.

Con esta idea y con el fin de dar cumplimiento a los requerimientos funcionales y no funcionales del SIIPOL, se tiene como propuesta de solución para el sistema una aplicación web con la estructura que se muestra en el Diagrama de Despliegue (ver Anexo 3). Éste es un tipo de diagrama de UML que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes de hardware y software. En el mismo se muestra la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software.

En la figura 3 se da una representación informal de las relaciones anteriormente descritas.

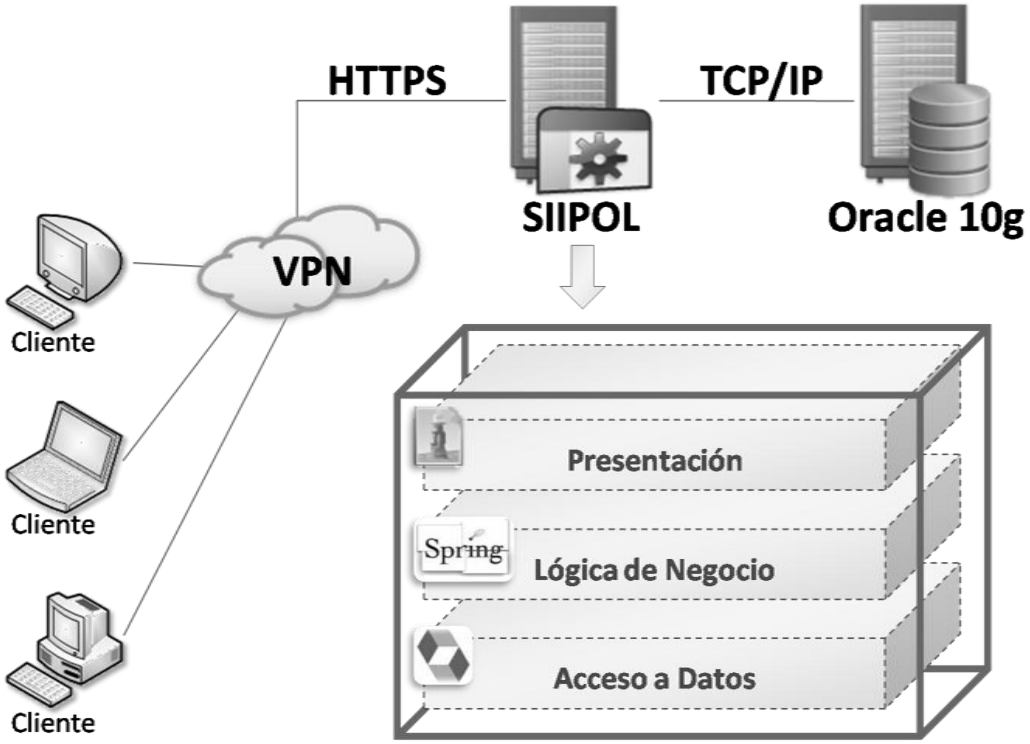


Figura 3: Aplicación Cliente-Servidor.

Se propone una Aplicación Web con arquitectura en tres capas como se muestra en la figura 4 y basada en tecnología Java, a la que se conectarán los clientes vía HTTPS.

El estilo arquitectónico en capas da ventajas sustanciales como lo son: la centralización de los aspectos de seguridad, transaccionalidad y no replicación de lógica de negocio en los clientes. Esto posibilita, que las modificaciones y mejoras sean automáticamente aprovechadas por el conjunto de

los usuarios reduciendo los costes de mantenimiento y garantizando a su vez una mayor sencillez de los clientes.

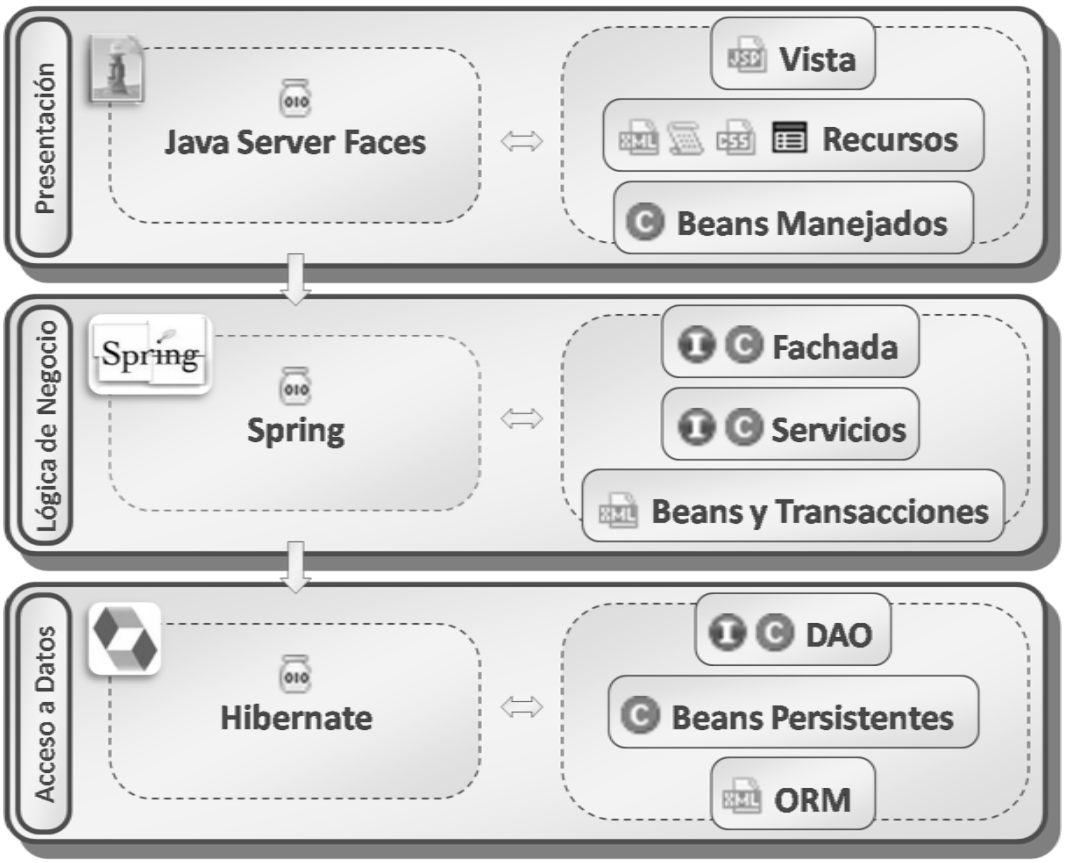


Figura 4: Arquitectura en tres capas.

1.7. Conclusiones.

En este capítulo se enmarcó el trabajo en su entorno de desarrollo. Se brindaron una serie de conceptos necesarios para la comprensión de la solución informática que da origen a esta investigación, como el de sistema de gestión de información; datos referentes a éstos a nivel mundial como fuente de estudio y consulta por parte de los autores; así como la explicación a modo de resumen del SIIPOL.

Capítulo 2: Análisis, Diseño e Implementación de la Propuesta de Solución.

2.1. Introducción.

Los flujos de Análisis y Diseño e Implementación son un paso fundamental entre los requisitos obtenidos y el resultado esperado. En la fase inicial, el análisis y el diseño se centran en establecer si el sistema que se ha concebido es factible, y en evaluar las tecnologías potenciales para la solución. La fase de elaboración se centra en la creación de una arquitectura inicial para el sistema, o sea la creación de una Arquitectura Candidata para proporcionar un punto de inicio para el trabajo de análisis principal.

Esta actividad tiene como finalidad transformar los requisitos en un diseño del sistema en creación, así como evolucionar una arquitectura sólida para el sistema y adaptar el diseño para que se ajuste al entorno de implementación, con un diseño pensado para el rendimiento.

Teniendo en cuenta la actividad de Implementación se puede decir que las actividades y productos de trabajo se organizan en un patrón de posibilidad para la disciplina. Además de que cada actividad representa un objetivo de alto nivel que necesita alcanzarse para completar la implementación con efectividad.

Un buen análisis y diseño de la aplicación proporcionan rapidez y precisión a la hora de la implementación.

Se tomaron una serie de Casos de Uso que cuentan con el criterio de que son significativos por la funcionalidad que describen dentro de los procesos de la Investigación Interna, dichos Casos de Uso son: Iniciar Investigación Disciplinaria y Gestionar Decisión del Consejo del submódulo de Investigación Disciplinaria, Iniciar Averiguación Preliminar del submódulo de Investigación Preliminar. Otros como Aprobar Diligencias Internas y Asignar Reasignar Investigaciones Preliminares\Disciplinarias, en este último caso se unen dos Casos de Uso, porque la funcionalidad se analiza, se diseña y se implementa de forma genérica para los dos tipos de expedientes, estos últimos son del submódulo Control de la Investigación. En el submódulo de Diligencias Internas se tomó como ejemplo Gestionar Acta Disciplinaria. Por último se seleccionaron una serie de Casos de Uso de forma común, porque como se mencionó anteriormente, el diseño, el análisis y la implementación se realizó de forma genérica, los cuales son Consultar Investigaciones Preliminares/Disciplinarias Asignadas, Ver Expediente Preliminar/Disciplinario, Remitir Expediente Preliminar/Disciplinario a Inspectoría y Gestionar Relación Objeto-Caso de asuntos internos.

2.2. Modelo de Análisis.

Como resultado del flujo de trabajo de requisitos se obtiene una vista externa del sistema, que en el lenguaje del cliente, describe lo que se espera de él a través del Diagrama de casos de uso. A partir del modelo de análisis es que se debe profundizar en los casos de usos detallándolos de manera que permitan reflejar una vista interna del sistema descrita con el lenguaje de los desarrolladores. En esta vista se especifican mejor los casos de uso y se determinan las clases necesarias para llevar a cabo las funcionalidades en ellos contenidos.

Este proceso se desarrolla fundamentalmente dentro de la fase de elaboración y se corresponde principalmente con el flujo de trabajo de análisis y diseño. El flujo de trabajo de análisis y diseño tiene un papel protagónico en la fase de elaboración.

En la propuesta de solución se realizó por cada caso de uso un diagrama de análisis unido con la realización de cada uno de ellos, que no es más que un diagrama para ver como colaboran cada uno de los elementos del diseño y de esa manera mostrar cómo funciona cada caso de uso desde una mejor perspectiva. (Ver Anexo 4)

2.3. Modelo de Diseño.

El Modelo de Diseño es la continuación del Modelo de Análisis, hasta obtener los objetos que interactúan para cumplir con los requisitos funcionales y no funcionales obtenidos. El diseño tiene el propósito de formular los modelos que se centran en los requisitos no funcionales y en el dominio de la solución y que prepara para la implementación y prueba del sistema. Pretende crear un plano del modelo de implementación, por lo que el grueso del esfuerzo está en las últimas iteraciones de elaboración y las primeras de construcción, lo que contribuye a una arquitectura estable y sólida. El modelo de diseño está muy cercano al de implementación, por lo que debemos guardarlo y mantenerlo a través del ciclo de vida completo del software.

2.3.1. Diagrama de paquetes.

El diagrama de paquetes muestra cómo un sistema está dividido en agrupaciones lógicas y las dependencias entre esas agrupaciones. Dado que normalmente un paquete está pensado como un directorio, los diagramas de paquetes suministran una descomposición de la jerarquía lógica de un sistema. Los Paquetes están normalmente organizados para maximizar la coherencia interna dentro de cada uno de ellos y minimizar el acoplamiento externo entre los mismos. Estos diagramas se usan para reflejar la organización de paquetes y sus elementos, además de que son muy utilizados en la

organización de diagramas de casos de uso y diagramas de clase, a pesar de que el uso de los diagramas de paquete no es limitado a estos elementos UML. (Ver Anexo 5)

En este diagrama de paquetes del Módulo de Investigaciones Internas, se representa solo una parte de las relaciones que existen entre los paquetes de cada submódulo, pues cada uno tiene la misma estructura interna. Ha sido representado, como se puede ver, las relaciones que existen con el submódulo *común*, cuya estructura representada en él como los demás se relaciona de la misma forma tanto interna como externamente.

2.3.2. Diagrama de clases del diseño.

El diagrama de clases de diseño representa la vista estática del sistema, o lo que es lo mismo, representa los elementos estructurales y sus relaciones, independientemente de su función.

La respuesta a un buen diseño está en la utilización de las relaciones entre los elementos de la llamada vista estática del sistema, para el desempeño de esta buena práctica se han utilizado en la solución del problema en cuestión patrones de diseño, los cuales han tomado un gran auge a partir del desarrollo del modelo orientado a objetos. Una arquitectura orientada a objetos bien estructurada está llena de patrones. Unas de las formas de medir la calidad de un sistema orientado a objetos está dada por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles.

En una aplicación del alcance que se requiere para cubrir los requisitos del módulo de Investigación Interna del SIIPOL, que cuenta numerosas clases, se ha de contar con el empleo de patrones para garantizar un diseño claro y lo más simple posible. En lo adelante se resume algunos de los patrones de diseño más significativos utilizados en construcción de la solución.

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos (Modelo/Vista/Controlador). El patrón MVC se ve frecuentemente en aplicaciones web y fue diseñado para reducir el esfuerzo de programación necesario en la implementación de sistemas múltiples.

El *Modelo* es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el Modelo y sus Vistas, y notificar a las Vistas cuando cambia el Modelo.

La *Vista* es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del Modelo y muestra los datos al usuario. Interactúa con el Modelo a través de una referencia al propio Modelo.

El *Controlador* es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo.

Al incorporar el modelo de arquitectura MVC a un diseño, las piezas de un programa se pueden construir por separado y luego unirlas en tiempo de ejecución. Si uno de los componentes posteriormente se observa que funciona mal, puede reemplazarse sin que las otras piezas se vean afectadas.

El patrón ***Alta Cohesión***, más que un diseño directamente implementable en código, se trata de un principio director que nos guiará en el diseño, es un objetivo subyacente a tener en cuenta continuamente. Es un principio evaluativo que aplica un diseñador mientras evalúa todas las decisiones de diseño. Se puede medir el nivel de cohesión en una clase cuanto más enfocado sea su comportamiento.

Cada elemento de nuestro diseño debe realizar una labor única dentro del sistema, que no se desempeñe en ninguno de los de los demás elementos del sistema. Por otra parte pensar en interfaces nos fuerza a que nuestros sistemas sigan los principios de alta cohesión.

Una diseño cohesionado facilita el cambio (objetivo principal de todos los patrones de diseño). Al realizar un cambio en una clase muy cohesionada, todos los métodos que pueden verse afectados, toda la información que necesitamos controlar, estará a la vista, en el mismo fichero.

El patrón ***Bajo Acoplamiento*** le da respuesta a la problemática de soportar bajas dependencias, bajo impacto del cambio e incremento de la reutilización. El acoplamiento es una medida de la fuerza con que un elemento está conectado a, tiene conocimiento de, confía en, otros elementos. Un elemento con bajo (o débil) acoplamiento no depende de demasiados otros elementos. Estos elementos pueden ser clases, subsistemas, sistemas entre otros, de ahí la importancia que se lleve a cabo el desempeño de este patrón, para de esta manera obtener una aplicación los más flexible posible.

El patrón ***Controlador*** aumenta el potencial de reutilización, y asegura que la lógica de la aplicación no se maneja en la capa de interfaz. Un Controlador es un objeto que no pertenece a la interfaz de usuario, responsable de recibir o manejar un evento del sistema y define el método para la operación del sistema.

El desempeño de este patrón no es más que asignar la responsabilidad de controlar el flujo de eventos del sistema a clases específicas. Esto facilita la centralización de actividades (validaciones, seguridad, etc.). El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión.

El patrón ***Experto*** no es más que asignar una responsabilidad al experto en información, la clase que tiene la información necesaria para realizar la responsabilidad. Un Modelo de Diseño podría definir cientos o miles de clases, y una aplicación podría requerir que se realicen cientos o miles de responsabilidades. Durante el diseño de objetos, cuando se definen las interacciones entre los objetos, tomamos decisiones sobre la asignación de responsabilidades a las clases. Si se realizan de la forma correcta, los sistemas tienden a ser más fáciles de entender, mantener y ampliar, y existen más oportunidades para reutilizar componentes en futuras aplicaciones.

Con el uso de este patrón se mantiene el encapsulamiento de la información, puesto que los objetos utilizan su propia información para llevar a cabo las tareas. Normalmente, esto conlleva un bajo acoplamiento, que como habíamos mencionado anteriormente da lugar a sistemas más robustos y más fáciles de mantener. Se distribuye el comportamiento entre las clases que contienen la información requerida, por tanto, se estimulan las definiciones de clases más cohesivas y ligeras.

Se utilizó el patrón ***Fachada*** primero que todo para proporcionar interfaces simples para subsistemas complejos, ya que cuando existen muchas dependencias entre clientes y clases que implementan una abstracción, este patrón proporciona independencia y portabilidad. Por otra parte teniendo en cuenta que el diseño está separado por capas, cada capa posee su propia fachada o interfaz unificada de alto nivel que facilite su uso.

Esto trae como consecuencia que al separar al cliente de los componentes del subsistema, se reduce el número de objetos con los que el cliente trata, facilitando así el uso del subsistema. También se promueve un débil acoplamiento entre el subsistema y sus clientes, eliminándose o reduciéndose las dependencias y no existen obstáculos para que las aplicaciones usen las clases del subsistema que necesiten.

El patrón ***polimorfismo*** es un principio fundamental para diseñar cómo se organiza el sistema para gestionar variaciones similares. Según el polimorfismo, un diseño basado en la asignación de responsabilidades puede extenderse fácilmente para manejar nuevas variaciones.

Este patrón trae como beneficio añadir fácilmente las extensiones necesarias para nuevas variaciones y las nuevas implementaciones se pueden introducir sin afectar a los clientes.

Luego de resumir algunos de los patrones más importantes para la realización del diseño se muestra a continuación los diagramas de clases significativas para cada uno de los casos de uso seleccionados.

Primero que todo hacer referencia a algunas de las clases más relevantes para la solución que van a ser utilizadas en los diagramas de diseño que se muestran en el Anexo 6.

Expediente (Bean Persistente)	
Atributo	**Tipo**
noExpediente	String
fecha	Date
funcionario	Funcionario
dependencia	DependenciaInterna

ActaInvestigativa (Bean Persistente)	
Atributo	**Tipo**
fechaEnCurso	Date
fechaConclusion	Date
fechaRemision	Date
funcionariosAsignados	Set<FuncionariosAsignados>
diligencias	Set<Diligencia>
asignada	Boolean

ActaDisciplinaria (Bean Persistente)	
Atributo	**Tipo**
modusOperandi	String
fechaVencimiento	Date
fechaOcurrencia	Date
fechaRemision	Date
funcionarioControlador	Funcionario
denunciante	Persona
sitioSuceso	SitioSuceso
entidadDenunciante	EnteExterno
relacionDisciplinariaPersonas	Set<AveriguacionPreliminarPersona>
relacionDisciplinariaElementos	Set<AveriguacionPreliminarElemento>
funcionariosInvestigados	Set<FuncionarioAbogado>
baseLegal	Set<Articulo>
tipoAveriguacion	NTipoAveriguacionPreliminar
nivelImportancia	NivelImportancia
horaAproximada	HoraAproximada

AveriguacionPreliminar (Bean Persistente)	
Atributo	**Tipo**

estadoAveriguacionPreliminar	NEstadoAveriguacionPreliminar

ExpedienteDisciplinario (Bean Persistente)	
Atributo	**Tipo**
estadoAveriguacionPreliminar	NEstadoAveriguacionPreliminar
estadoExpedienteDisciplinario	NEstadoExpedienteDisciplinario
tipoFalta	NTipoFalta
naturalezaFalta	NNaturalezaFalta

InvestigacionInternaDaoImpl (Dao)	
Atributo	**Tipo**

Responsabilidades:	
Nombre:	ejecutarProcedimiento
Descripción:	Realiza una serie de funcionalidades que se repite en cualquier lugar donde se vaya a realizar un procedimiento.
Nombre:	obtener
Descripción:	Obtiene cualquier tipo de elemento con solo saber la clase a que se hace referencia y uno de sus atributos.
Nombre:	obtenerTodos
Descripción:	Obtiene todos los elementos de una clase con solo saber la clase a que se hace referencia.
Nombre:	inicializarAtributos
Descripción:	Inicializa de una clase señalada los atributos que de ella fueron seleccionados.

CU Iniciar Investigación Disciplinaria.

IniciarInvestigacionInterna (Bean de respaldo)	
Atributo	**Tipo**
actaDisciplinaria	ActaDisciplinaria
fechaCreacion	Date
funcionarioActuante	Funcionario
delegada	Boolean
relacionarPersonaDisciplinariaManejado	RelacionarPersonaDisciplinariaManejado
relacionarObjetoDisciplinarioManejado	RelacionarObjetoDisciplinarioManejado
relacionarArmaDisciplinarioManejado	RelacionarArmaDisciplinarioManejado
relacionarVehiculoDisciplinarioManejado	RelacionarVehiculoDisciplinarioManejado
relacionarFuncionarioDisciplinarioManejado	RelacionarFuncionarioDisciplinarioManejado
gestionarBaseLegalSubviewManejado	GestionarBaseLegalSubviewManejado
Responsabilidades:	
Nombre:	agregarPregunta
Descripción:	Agrega preguntas a la reseña del hecho.

IniciarInvestigacionDisciplinariaManejado (Bean de respaldo)	
Atributo	**Tipo**
nomencladorInvestigacionPenalFacade	NomencladorInvestigacionPenalFacade
nomencladorFacade	NomencladorFacade
administracionFacade	AdministracionFacade
analisisInformacionFacade	AnalisisInformacionFacade
listaNaturalezaFalta	List<SelectItem>
listaTipoFalta	List<SelectItem>
listaNivelImportancia	List<SelectItem>
listaDependenciasInternas	List<SelectItem>
naturalezaFalta	NNaturalezaFalta
tipoFalta	NTipoFalta
nivelImportancia	NivelImportancia
dependenciaInterna	DependenciaInterna
idHoraSeleccionada	String
funcionariosContraventores	List<FuncionarioAbogado>
personasRelacionadas	Set<AveriguacionPreliminarPersona>
elementosRelacionados	Set<AveriguacionPreliminarElemento>
Responsabilidades:	
Nombre:	incluir
Descripción:	Organiza toda la información y crea el Expediente Disciplinario para hacerlo persistente.
Nombre:	validarId
Descripción:	Valida que el funcionario actuante no se repita entre los funcionarios mencionados.
Nombre:	construirEntrevista
Descripción:	Construye la entrevista asociada al expediente en cuestión.

CU Gestionar Decisión del Consejo Disciplinario.

GestionarDecisionConsejoDisciplinarioManejado (Bean de respaldo)	
Atributo	**Tipo**
expedienteDisciplinario	ExpedienteDisciplinario
tiposSanciones	List<SelectItem>
sancionSeleccionada	NTSancion
Observaciones	String
fechaDesicion	Date
listaFuncionarios	List<SelectItem>
funcionarioSeleccionado	FuncionarioAbogado
listaDecisiones	List<FuncionarioAbogado>
Responsabilidades:	
Nombre:	adicionar
Descripción:	Adiciona una decisión al funcionario contraventor seleccionado.
Nombre:	deshacer
Descripción:	Elimina la decisión incluida al funcionario contraventor.
Nombre:	incluirDecision
Descripción:	Incluye todas las decisiones tomadas a los funcionarios contraventores del Expediente Disciplinario.

CU Iniciar Averiguación Preliminar.

IniciarInvestigacionInterna (Bean de respaldo)	
Atributo	**Tipo**
actaDisciplinaria	ActaDisciplinaria
fechaCreacion	Date
funcionarioActuante	Funcionario
delegada	Boolean
relacionarPersonaDisciplinariaManejado	RelacionarPersonaDisciplinariaManejado
relacionarObjetoDisciplinarioManejado	RelacionarObjetoDisciplinarioManejado
relacionarArmaDisciplinarioManejado	RelacionarArmaDisciplinarioManejado
relacionarVehiculoDisciplinarioManejado	RelacionarVehiculoDisciplinarioManejado
relacionarFuncionarioDisciplinarioManejado	RelacionarFuncionarioDisciplinarioManejado
gestionarBaseLegalSubviewManejado	GestionarBaseLegalSubviewManejado
Responsabilidades:	
Nombre:	agregarPregunta
Descripción:	Agrega preguntas a la reseña del hecho.

IniciarAveriguacionPreliminarManejado (Bean de respaldo)	
Atributo	**Tipo**
nomencladorInvestigacionPenalFacade	NomencladorInvestigacionPenalFacade
nomencladorFacade	NomencladorFacade
administracionFacade	AdministracionFacade
analisisInformacionFacade	AnalisisInformacionFacade
horaFalta	Date
credencialDenunciante	String
tipoAveriguacion	String
tipoDenunciante	String
noRif	String
denunciante Agraviado	Boolean
modificarDenunciante	Boolean
identificador	Integer
funcionarioDenunciante	Funcionario
denunciante	Persona
estadosVenezolanos	List<EstadoVenezolano>
listaHoraAproximada	List<SelectItem>
listaNivelImportancia	List<SelectItem>
listaTipoAveriguacion	List<SelectItem>
listaFuncionarios	List<SelectItem>
funcionariosMencionados	List<FuncionarioAbogado>
personasRelacionadas	Set<AveriguacionPreliminarPersona>
elementosRelacionados	Set<AveriguacionPreliminarElemento>
Responsabilidades:	
Nombre:	incluir
Descripción:	Organiza toda la información y crea la Averiguación Preliminar para hacerla persistente.

Nombre:	validarId
Descripción:	Valida que el funcionario actuante no se repita entre los funcionarios mencionados.
Nombre:	construirEntrevista
Descripción:	Construye la entrevista asociada al expediente en cuestión.

CU Aprobar Diligencia Interna.

RevisarAprobarDiligenciasInternasManejado (Bean de respaldo)	
Atributo	**Tipo**
diligenciaInternaDTO	List<DiligenciaInternaDTO>
funcionario	Funcionario
comentario	String
actaDisciplinaria	ActaDisciplinaria
table	UIData
Para cada responsabilidad:	
Nombre:	acetarDiligencias
Descripción:	Acepta la diligencia y le cambia el estado para Aprobado.
Nombre:	aceptarTodas
Descripción:	Acepta todas las diligencias y les cambia el estado para Aprobado.
Nombre:	rechazarDiligencia
Descripción:	Anula la diligencia y le cambia el estado para Anulado.
Nombre:	verDiligencia
Descripción:	Permite ver la diligencia seleccionada.

CU Asignar Reasignar Investigaciones Preliminares/Disciplinarias.

AsignarReasignarExpedienteInternoManejado (Bean de respaldo)	
Atributo	**Tipo**
noExpediente	String
credencial	String
fechaApertura	Date
estado	String
nivelImportancia	String
listaEstados	List<SelectItem>
listaEstadosControlados	List<SelectItem>
nivelesImportancia	List<SelectItem>
Responsabilidades:	
Nombre:	verExpediente
Descripción:	Permite ver el Expediente seleccionado.
Nombre:	ejecutarAccion
Descripción:	Realizar la acción de asignar o resignar.

CU Gestionar Acta Disciplinaria.

GestionarActaDisciplinariaManejado (Bean de respaldo)	
Atributo	**Tipo**
fechaRedaccion	Date

horaRedaccion	Date
descripción	String
relacionarFuncionarioDisciplinarioManejado	RelacionarFuncionarioDisciplinarioManejado
funcionariosMencionados	List<FuncionarioAbogado>
funcionarios	HashMap<Integer, Integer>
Responsabilidades:	
Nombre:	incluir
Descripción:	Organiza la información y crea el Acta Disciplinaria Interna para hacerla persistente.

VerActaDisciplinariaManejado (Bean de respaldo)	
Atributo	**Tipo**
gestionarBaseLegalSubviewManejado	GestionarBaseLegalSubviewManejado
Responsabilidades:	
Nombre:	modificar
Descripción:	Permite ir a modificar la diligencia en caso de que cumpla con las precondiciones.

ModificarActaDisciplinariaManejado (Bean de respaldo)	
Atributo	**Tipo**
nomencladorInvestigacionPenalFacade	NomencladorInvestigacionPenalFacade
gestionarBaseLegalSubviewManejado	GestionarBaseLegalSubviewManejado
disciplinarioManejado	RelacionarFuncionarioDisciplinarioManejado
actaDisciplinariaInterna	ActaDisciplinariaInterna
fechaRedaccion	Date
horaRedaccion	Date
funcionariosMencionados	List<FuncionarioAbogado>
func	HashMap<Integer, Integer>
Responsabilidades:	
Nombre:	actualizar
Descripción:	Organiza la información y modifica el Acta Disciplinaria Interna para hacerla persistente los cambios.

Investigaciones Preliminares/Disciplinarias Asignadas.

ConsultarExpedientesInternosAsignadosManejado (Bean de respaldo)	
Atributo	**Tipo**
desdeFecha	Date
hastaFecha	Date
numeroExp	String
estadoExp	String
Importancia	NivelImportancia
credencial	String
funcionario	Funcionario
listImportancia	List<SelectItem>
dataTable	UIData

esJefe	Boolean
listaTipoFalta	List<SelectItem>
tipoFalta	NTipoFalta
Responsabilidades:	
Nombre:	consultar
Descripción:	Realiza la consulta con los parámetros seleccionados.

Ver Expediente Preliminar/Disciplinario.

VerExpedienteInternoManejado (Bean de respaldo)	
Atributo	**Tipo**
actaDisciplinaria	ActaDisciplinaria
funcionario	Funcionario
cantDiligencias	int
solicitudesNoRespondidas	int
cantComunicaciones	int
listDTO	List<SelectItem>
diligencia	DTOIncluirDiligencia
tipoActaDisciplinaria	Boolean
noExpedientePreliminar	String
listObjetosAveriguacion	List<AveriguacionPreliminarElemento>
listPersonasAveriguacion	List<AveriguacionPreliminarPersona>
listVehiculosAveriguacion	List<AveriguacionPreliminarElemento>
listArmasAveriguacion	List<AveriguacionPreliminarElemento>
analisisInformacionFacade	AnalisisInformacionFacade
administracionFacade	AdministracionFacade
tablaPersonas	UIData
tablaArmas	UIData
tablaVehiculos	UIData
tablaObjetos	UIData
funcionariosMencionados	List<FuncionarioAbogado>
Responsabilidades:	
Nombre:	verDecision
Descripción:	Permite ver la decisión de un expediente ya haya sido decidido.
Nombre:	consultarComunicaciones
Descripción:	Permite ir a consultar las comunicaciones asociadas al expediente.
Nombre:	consultarDiligencias
Descripción:	Permite ir a consultar las diligencias asociadas al expediente.
Nombre:	remitirExpediente
Descripción:	Permite ir a remitir un expediente cuando tenga todas las diligencias aprobadas.
Nombre:	incluirDiligencia
Descripción:	Da la posibilidad de poder realizar una diligencia seleccionada.
Nombre:	setObject
Descripción:	Permite cargar todos los elementos necesarios para ver el expediente.

Remitir Expediente Preliminar/Disciplinario a Inspectoría.

RemitirActasDisciplinariasManejado (Bean de respaldo)	
Atributo	**Tipo**
averiguacionPreliminar	AveriguacionPreliminar
expedienteDisciplinario	ExpedienteDisciplinario
diligenciasAsociadas	List<DiligenciaInternaDTO>
observaciones	String
tablaDiligencias	UIData
gestionAdministrativaFacade	GestionAdministrativaFacade
Responsabilidades:	
Nombre:	remitirExpediente
Descripción:	Permite remitir el expediente a Inspectoría General.

Gestionar Relación Objeto Caso de Asuntos Internos.

RelacionarElementoDisciplinarioManejado (Bean de respaldo)	
Atributo	**Tipo**
analisisInformacionFacade	AnalisisInformacionFacade
comentario	String
elementoIncluir	Elemento
elementoSeleccionado	Elemento
averiguacionElementoSeleccionada	AveriguacionPreliminarElemento
razonPorDefecto	NTRelAveriguacionPreliminarElemento
razonSelecionada	NTRelAveriguacionPreliminarElemento
razones	List<SelectItem>
elementos	Map<Integer, AveriguacionPreliminarElemento>
tabla	HtmlDataTable
Responsabilidades:	
Nombre:	verDatosElementos
Descripción:	Permite ver datos de cualquier tipo de elemento.
Nombre:	asociarRelacion
Descripción:	Permite asociar una nueva relación con el expediente.
Nombre:	modificarRelacion
Descripción:	Permite modificar una relación seleccionada en el expediente.
Nombre:	disociarRelacion
Descripción:	Permite disociar una relación seleccionada en el expediente.

2.3.3.Realizaciones de Casos de Uso.

Para representar gráficamente cómo las clases y subsistemas (elementos estructurales) interactúan para llevar a cabo las funcionalidades descritas en un caso de uso, se confeccionan diagramas de interacción, específicamente diagramas de secuencia.

Debido a que la arquitectura establecida para el proyecto pauta una gran cantidad de aspectos de funcionamiento de cada una de las capas arquitectónicas del sistema, la confección de diagramas de secuencia detallados que incluyan todas las clases y su interacción no aporta mucha información.

Por tal motivo, se decidió confeccionar diagramas de secuencia con un nivel de abstracción mucho mayor, en el que los elementos estructurales que interactúen sean los subsistemas que componen el sistema. A este tipo de diagramas se le denomina Diagrama de Contrato entre Paquetes, precisamente porque definen las responsabilidades que asume un paquete en relación con los otros. (Ver Anexo 7)

2.4. Modelo de Datos.

Los modelos de datos aportan la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos, así como la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de sistemas de información.

Un modelo de datos es por tanto una colección de conceptos bien definidos matemáticamente que ayudan a expresar las propiedades estáticas y dinámicas de una aplicación con un uso de datos intensivo, entre las que se cuentan los sistemas de gestión de información.

Diagrama de clases persistentes.

Las clases persistentes son aquellas clases que tienen un mayor tiempo de vida, más allá del tiempo de ejecución de la aplicación. Por lo general las clases persistentes coinciden con los conceptos de información que se usan en el sistema. También se les denomina en ocasiones: clases de dominio.

Diagrama de tablas del modelo relacional.

En el diagrama de tablas del modelo relacional se representan las tablas de la base de datos relacional y las relaciones entre ellas. Es, por tanto, la vista física de almacenamiento de la información del sistema.

(Ver Anexo 8)

2.5. Modelo de Implementación.

El Modelo de Implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes podemos encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos.

Diagramas de subsistemas de implementación.

Este artefacto describe cómo se implementan los componentes, congregándolos en subsistemas organizados en capas y jerarquías, y señala las dependencias entre éstos.

Diagrama de componentes.

Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, librerías compartidas, módulos, ejecutables, o paquetes.

En lo adelante se representa la dependencia de los componentes en dos diagramas diferentes, uno para representar los archivos java, interfaces, XML de configuración y propiedades que son utilizados para la programación del lado del servidor, y el otro diagrama representa la dependencia entre los archivos o páginas JSP.

(Ver Anexo 9)

2.5.1. Estándar de codificación.

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un proyecto de software, es necesario establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Cuando el proyecto de software incorpore código fuente previo, o bien cuando realice el mantenimiento de un sistema de software creado anteriormente, el estándar de codificación debería establecer cómo operar con la base de código existente.

La legibilidad del código fuente repercute directamente en lo bien que un programador comprende un sistema de software. La mantenibilidad del código es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento. Aunque la legibilidad y la mantenibilidad son el resultado de muchos factores, una faceta del desarrollo de software en la que todos los programadores influyen especialmente es en la técnica de codificación. El mejor método para asegurarse de que un equipo de programadores mantenga un código de calidad es establecer un estándar de codificación sobre el que se efectuarán luego revisiones del código de rutinas.

Teniendo en cuenta el lenguaje de programación y los frameworks utilizados a continuación se hará referencia a algunas pautas significativas que se tienen en cuanta en la solución de la

problemática, el resto de los patrones a seguir estarán incluidos en la referencia al documento de Guía de estilo de código para el proyecto CICPC.

Convenciones de nombres

Todo lo referido a la convención de nombres se regirá en última instancia por la “Convención de código de Java”, lo que quiere decir que lo que no esté especificado se hará tal como venga indicado en dicha convención.

Como principio todos los nombres serán en español exceptuando aquellos que correspondan al nombre de un patrón conocido, ejemplo: ServiceLocator, Facade, Builder, ect.

Clases

Los nombres de las clases deben ser sustantivos, cuando son compuestos tendrán la primera letra de cada palabra que lo forma en mayúsculas. Los nombres de las clases deben ser simples y descriptivos. Además se debe usar palabras completas, evitar acrónimos y abreviaturas (a no ser que la abreviatura sea mucho más conocida que el nombre completo, como URL o HTML).

Métodos

Los métodos deben ser verbos, cuando son compuestos tendrán la primera letra en minúscula, y la primera letra de las siguientes palabras que lo forma en mayúscula.

Ejemplo: enviarSolicitud().

Variables

Excepto las constantes, todas las instancias y variables de clase o método empezarán con minúscula. Las palabras internas que lo forman (si son compuestas) empiezan con su primera letra en mayúsculas. Los nombres de variables no deben empezar con los caracteres subguión "_" o signo del dólar "$", aunque ambos están permitidos por el lenguaje.

Los nombres de las variables deben ser cortos pero con significado, estos nombres de ser de un solo caracter estarán incorrectos, excepto para variables índices temporales, como son las de ciclos, para hacer intercambio de valores, etc. Nombres comunes para variables temporales son i, j, k, m, y n para enteros; c, d, y e para caracteres.

Constantes

Los nombres de las variables declaradas como constantes deben ir totalmente en mayúsculas separando las palabras con un subguión ("_").

Ejemplo: ANCHURA_MAXIMA = 999;

2.6. Conclusiones.

En este capítulo se pudo llevar a cabo el cumplimiento de los requisitos funcionales y no funcionales en las fases de elaboración y construcción del ciclo de vida de RUP. Después de hacerse realizado un minucioso trabajo en el análisis, diseño e implementación del modulo de Investigación Interna del SIIPOL, se pudo comprobar que luego de un buen análisis de los casos de uso y de las clases que satisfacen a estos, se da pie a la realización de un diseño, que ayudado con los patrones utilizados para lograr una mayor simpleza y reusabilidad, propicia el desarrollo de una cómoda implementación de el problema en cuestión.

Capítulo 3: Validación de la Solución Propuesta.

3.1. Introducción.

Las Pruebas de software son los procesos que permiten verificar y revelar la calidad de un sistema determinado. Estas se integran dentro de las diferentes fases del ciclo de desarrollo de software. Así se ejecuta un programa y mediante técnicas experimentales se trata de descubrir qué errores tiene. Para determinar el nivel de calidad se deben efectuar unas medidas o pruebas que permitan comprobar el grado de cumplimiento respecto de las especificaciones iniciales del sistema.

Según Edsger Wybe Dijkstra "El *testing* puede probar la presencia de errores pero no la ausencia de ellos"; estos errores pueden ser clasificados como errores de programación (*bugs*) y defectos de forma. En un defecto de forma, el programa no realiza lo que el usuario espera. Por el contrario, un error de programación puede describirse como un fallo en la semántica de un programa de ordenador. Éste podría presentarse, o no, como un defecto de forma si se llegan a dar ciertas condiciones de cálculo.

El proceso de prueba es clave a la hora de detectar errores o fallas. Conceptos como estabilidad, escalabilidad, eficiencia y seguridad se relacionan a la calidad de un producto bien desarrollado. Las aplicaciones de software han crecido en complejidad y tamaño, y por consiguiente también en costos. Hoy en día es crucial verificar y evaluar la calidad de lo construido para minimizar el costo de su reparación y la aceptación por parte del cliente final.

3.2. Tipos y Niveles de Pruebas.

Para lograr encontrar los diferentes tipos de errores anteriormente descritos existen dos tipos de pruebas:

Pruebas de caja negra: Son pruebas que se llevan a cabo sobre la interfaz del software. El objetivo es demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y se produce un resultado correcto, y que la integridad de la información externa se mantiene (no se ve el código). Estas pruebas permiten detectar los errores de forma.

También conocidas como Pruebas de Comportamiento, estas pruebas se basan en la especificación del programa o componente a ser probado para elaborar los casos de prueba. El componente se ve como una “Caja Negra” cuyo comportamiento sólo puede ser determinado estudiando sus entradas y las salidas obtenidas a partir de ellas. Para realizarlas se selecciona un conjunto de posibles entradas y salidas sobre las que se realizaran las pruebas. Hay que tener en cuenta que en todo programa existe un conjunto de entradas que causan un comportamiento erróneo en nuestro sistema, y como consecuencia producen una serie de salidas que revelan la presencia de defectos.

La confección de los casos de prueba aplicados a la solución de software se basa en el método empleado para el diseño de casos de prueba del SIIPOL, el cual, por las características y complejidad del sistema, es un híbrido obtenido a partir de los métodos de Particiones de Equivalencia y Análisis de Valores Limites, el cual se basa principalmente en la especificación de casos de uso. El mismo está detallado en el documento Estrategias de Pruebas.

Pruebas de caja blanca: Se comprueban los caminos lógicos del software Se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el esperado (sobre el código). Estas detectan los errores de programación.

Las pruebas de caja blanca se llevan a cabo en primer lugar, sobre un módulo concreto, para luego realizar las de caja negra sobre varios subsistemas.

Es importante tener en cuenta que estas pruebas no descubren todos los errores del código. Por definición, sólo prueban unidades de forma independiente. Por lo tanto, no descubrirán errores de integración, problemas de rendimiento y otros problemas que afectan a todo el sistema en su conjunto. Además, puede no ser trivial anticipar todos los casos especiales de entradas que puede recibir en realidad la unidad de programa bajo estudio. Estas pruebas sólo son realmente efectivas si se usan en conjunto con otras pruebas de software.

Para llevar a cabo este tipo de pruebas se utilizó el *framework* JUnit, con éste se desarrollan componentes de software que se encargan de ejecutar y probar partes del sistema.

A la hora de evaluar dinámicamente un sistema de software, la estrategia seleccionada debe permitir comenzar por los componentes más simples y pequeños e ir avanzando progresivamente hasta probar todo el sistema en su conjunto. Más concretamente, los pasos a seguir son:

Pruebas Unitarias: Comienzan con la prueba de cada módulo. Es una forma de probar el correcto funcionamiento de las partes separadas del sistema. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. Luego, con las Pruebas de Integración, se podrá asegurar el correcto funcionamiento del sistema o subsistema en cuestión. La idea es escribir casos de prueba para cada función o método en el módulo de forma que cada caso sea independiente del resto.

Pruebas de Integración: Son aquellas que se realizan en el ámbito del desarrollo de software una vez que se han aprobado las pruebas unitarias. Únicamente se refieren a la prueba o pruebas de todos los elementos unitarios que componen un proceso, hecha en conjunto, de una sola vez. Consiste en realizar pruebas para verificar que un gran conjunto de partes de software funcionan juntos.

Pruebas de Sistema: El software ensamblado totalmente con cualquier componente hardware que requiere se prueba para comprobar que se cumplen los requisitos funcionales.

Pruebas de Aceptación: El cliente comprueba que el software funciona según sus expectativas.

El sistema SIIPOL y el Módulo de Investigaciones Internas, como parte del mismo, fue sometido a diversas pruebas correspondientes a los niveles descritos. Dada su importancia, los resultados más relevantes fueron obtenidos durante las pruebas de liberación, aceptación y piloto.

3.3. Resultados obtenidos.

El sistema fue sometido a varias iteraciones de prueba por parte de los equipos de Calidad Interna y Desarrollo donde se detectaron una gran cantidad de No Conformidades (ver Anexo 4). Estas iteraciones de prueba aseguraron que los errores del sistema disminuyeran considerablemente antes de enfrentarse a las Pruebas de Liberación a desarrollarse por la Dirección de Calidad UCI.

En la figura 5 se muestra la cantidad de No Conformidades que se detectaron en cada una de las iteraciones de prueba que se efectuaron al sistema y otras que fueron descubiertas fuera de las iteraciones de prueba. En la misma se observa una disminución sustancial de los errores encontrados, llegando a cero en las Pruebas Piloto (en el caso del Módulo de Investigaciones Internas). Estos datos aseguraron la aceptación del sistema por parte del cliente final. Durante estas últimas pruebas, se obtuvieron Peticiones de Cambio por parte del cliente, las cuales traen como consecuencia la inclusión de nuevas funcionalidades al sistema.

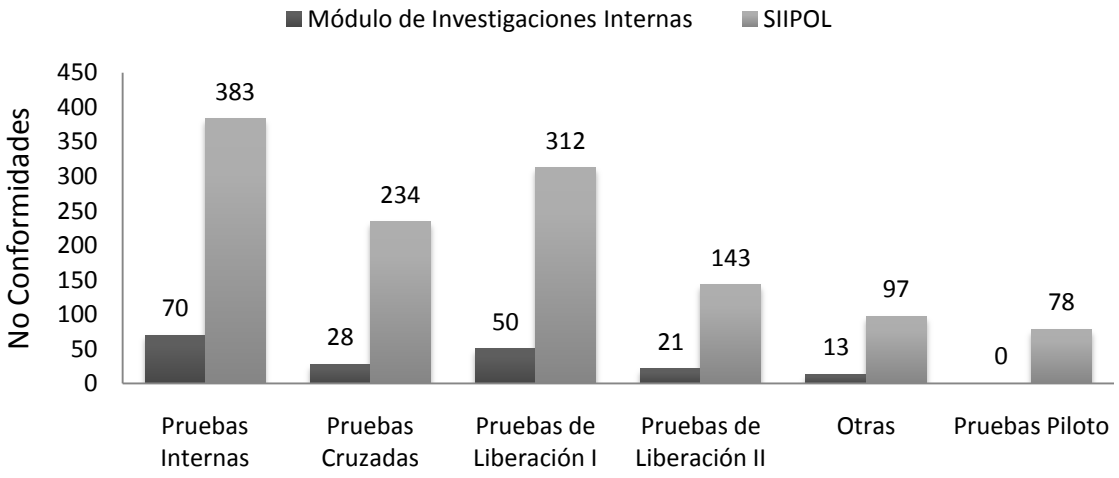


Figura 5: Comportamiento de las Pruebas al Módulo de Investigaciones Internas y SIIPOL.

De igual manera es importante destacar la baja razón de problemas en relación con la cantidad de Casos de Uso que conforman el Módulo de Investigaciones Internas, tal y como se muestra en la figura 6. De manera general, el módulo obtuvo muy buena aceptación por parte del cliente, todas las no conformidades fueron resueltas así como la solución de la mayoría las Peticiones de Cambio, presentando así un sistema más sólido y funcional.

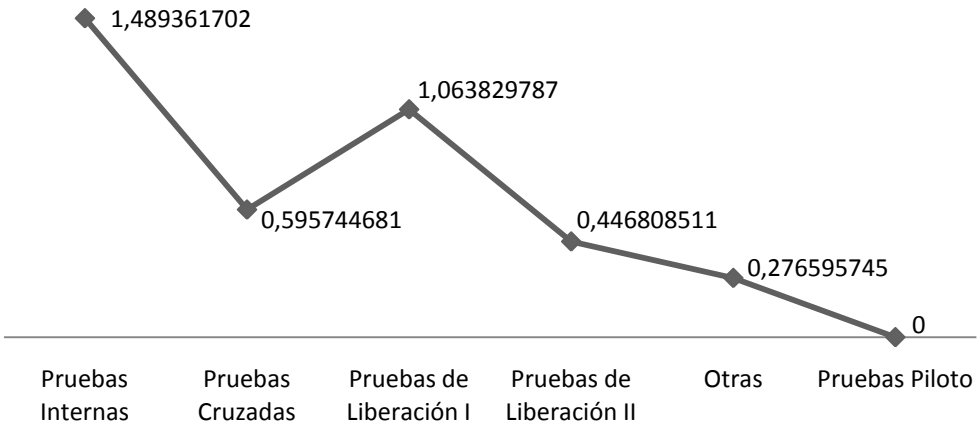


Figura 6: Razón entre No Conformidades y Casos de Uso probados.

Como parte importante de los resultados obtenidos, se destaca el gran aporte social del Módulo de Investigaciones Internas, el cual servirá de apoyo a los funcionarios encargados de las investigaciones internas del CICPC. Todo esto traerá consigo un mejor funcionamiento del órgano policial.

3.4. Conclusiones.

Finalizada la etapa de pruebas se obtiene un módulo que cumple con los requisitos funcionales y no funcionales obtenidos de la Ingeniería de Requerimientos, se integra completamente al sistema y es aceptado por el cliente final de la aplicación. Estos resultados fueron garantizados gracias a la aplicación de las diferentes pruebas descritas en el capítulo.

El Módulo de Investigaciones Internas obtuvo buenos resultados en las pruebas aplicadas, encontrándose una cantidad de No Conformidades inferior a la media del sistema.

Conclusiones.

Con la culminación de este trabajo de diploma se evaluaron los resultados derivados de la aplicación de la Ingeniería de Requerimientos a los procesos de la Investigación Interna en el CICPC. Con la realización los flujos de Análisis y Diseño se garantizaron la flexibilidad y simpleza del módulo a través de la aplicación de patrones de diseño. Posteriormente teniendo en cuenta el buen desempeño de los flujos antes mencionados, se llevó a cabo la implementación del módulo respetando la arquitectura definida para el SIIPOL. Logrando de esta manera una integración satisfactoria del módulo Investigación Interna al Sistema de Investigación e Información Policial, tal así que se obtiene por parte del cliente la aceptación durante las pruebas de Aceptación y Piloto.

Se destaca el aporte tanto a la sociedad como al cuerpo policial del CICPC, del resultado de la implementación del módulo, pues se le brinda un mayor número de procesos automatizados referentes a la investigación interna. Se proporciona de esta manera mayor facilidad de control sobre las indisciplina dentro del mismo cuerpo policial, lo que trae consigo eficiencia y compromiso a la hora de resolver los innumerables delitos que ocurren en el hermano país Venezolano anualmente. Dando de esta manera un aporte considerable a la tranquilidad y bienestar social a una nación que se encuentra catalogadas como una de las de mayor índice de delincuencia en el mundo.

Recomendaciones.

- Refactorizar el módulo de Investigaciones Internas.
- Implementar los nuevos Casos de Uso resultantes de las Peticiones de Cambio de las pruebas.
- Poner este trabajo a disposición de la comunidad universitaria como referencia para proyectos similares, teniendo en cuenta las reglas de confidencialidad establecidas.
- Estudiar la posibilidad de realizar un sistema similar para los órganos del MININT de Cuba, específicamente para el departamento de Control Interno.

Bibliografía

1. **División de Desarrollo y Fortalecimiento Institucional.** Cuerpo de Investigaciones Científicas, Penales y Criminalísticas. [En línea] [Citado el: 1 de febrero de 2009.] http://www.cicpc.gov.ve/templates/misi%C3%B3n1.html.

2. **Pereira Ojeda, Maikel y Gago Martinez, Yudanis.** *Análisis, Diseño e Implementación del módulo Experticias Criminalísticas del Sistema de Investigación e Información Policial.* Ciudad de la Habana : s.n., 2007.

3. **Rivero Guevara, Humberto.** *Análisis, diseño e implementación del módulo Aprehensión del SIIPOL.* Cuidad de la Habana : s.n., 2007.

4. **Sun Microsystems, Inc.** Conozca más sobre la tecnología Java. [En línea] [Citado el: 17 de Febrero de 2009.] http://www.java.com/es/about/.

5. *Decreto con Fuerza de Ley de los Órganos de Investigaciones Científicas, Penales y Criminalísticas.* **Gaceta Oficial de la República Bolivariana de Venezuela.** 5551, 2001.

6. **Pressman, Roger.** *Ingeniería de software. Un enfoque práctico.* s.l. : McGraw.Hill/Interamericana de España, 2002.

7. **Larman, Craig.** *UML y Patrones.* s.l. : Prentice Hall., 2000.

8. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* 2000.

9. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El lenguaje unificado de modelado.* s.l. : Addison Wesley., 2000.

10. **Zukowski, John.** *Java 2 J2SE 1.4.* s.l. : Anaya Multimedia, S.A., 2003.

11. **Becerril, Francisco.** *Java a su alcance.* s.l. : McGRAW-HILL, 1998.

12. **Englander, Robert.** *Developing Java Beans.* s.l. : O'Reilly.

13. **Flanagan, David.** *Java en pocas palabras.* s.l. : McGraw-Hill, 1999. 970-10-2070-7.

14. **Akif, Mohammad, y otros.** *Java y XML.* s.l. : Anaya Multimedia, 2001. 84-415-1364-5.

15. **Departamento de Ingeniería de Software, UCI.** *Conferencias de ISW.* Cuidad de La Habana : s.n., 2008-2009.

16. **Kruchten, Philippe.** *Rational Unified Process, An Introduction, Third Edition.* 2003.

17. **Mann, Kito.** *JavaServer Faces in Action.* s.l. : Manning, 2005.

18. **Scrum Alliance, Inc.** ScrumAlliance. [En línea] [Citado el: 21 de marzo de 2009.] http://www.scrumalliance.org/.

19. **Visual-Paradigm.** Sitio Web oficial Visual-Paradigm. [En línea] [Citado el: 25 de febrero de 2009.] http://www.visual-paradigm.com/product/vpuml/.

20. **Eclipse Fundation.** Eclipse. [En línea] [Citado el: 21 de febrero de 2009.] http://www.eclipse.org.

21. **Geary, David y Horstmann, Cay.** *Core JavaServer™ Faces, Second Edition.* s.l. : Prentice Hall, 2007.

22. **Walls, Craig y Breidenbach, Ryan.** *Spring in Action, Second Edition.* s.l. : Manning, 2008. 1-933988-13-4.

23. **Suárez, Héctor.** *Manual Hibernate.* s.l. : Java Hispano, 2003.

24. **Bauer, Christian y King, Gavin.** *Hibernate in Action.* s.l. : Manning, 2005.

25. **Bauer, Christian y King, Gavin.** *Java Persistence with Hibernate.* s.l. : Manning., 2007.

26. **Peak, Patrick y Heudecker, Nick.** *Hibernate Quickly.* s.l. : Manning., 2006.

27. **Red Hat.** *RichFaces Developer Guide.* s.l. : Red Hat., 2007.

28. **Forman, Ira y Forman, Nate.** *Java Reflection in Action.* s.l. : Manning, 2005. 1-932394-18-4.

29. **Kayal, Dhrubojyoti.** *Pro Java™ EE Spring Patterns.* s.l. : Apress, 2008. 978-1-4302-1009-2.

30. **Frómeta, Maykell.** *Guía de estilo de código para el proyecto CICPC.* Cuidad de La Habana : s.n., 2007.

Glosario de Términos.

Adabas-Natural: Lenguaje de Programación en que está implementado el sistema anterior.

API: es el conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Applet: es un componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador web. El applet debe ejecutarse en un contenedor, que lo proporciona un programa anfitrión, mediante un plugin, o en aplicaciones como teléfonos móviles que soportan el modelo de programación por applets.

Arquitectura Candidata: Es el proceso donde se estudia y se propone la Arquitectura de un Software en términos de sus componentes fundamentales, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución; que concluye con las pruebas a la arquitectura y posterior aprobación de la misma como Línea base para el resto de las fases de desarrollo de RUP.

Bytecode: Es un código intermedio más abstracto que el código máquina. Habitualmente es tratado como un fichero binario que contiene un programa ejecutable.

Bean: En el lenguaje Java es un componente software reutilizable que evita programar los distintos componentes uno a uno. Existen con la finalidad de ahorrar tiempo al programar.

Código Fuente: Es un conjunto de líneas de texto que son las instrucciones que debe seguir el ordenador para ejecutar dicho programa.

Datos de persistencia: Son los datos que poseen un mayor tiempo de vida, pues quedan persistidos aunque el programa se deje de ejecutar.

Diagrama de Paquetes: Muestra como un sistema está dividido en agrupaciones lógicas mostrando las dependencias entre esas agrupaciones.

Diseño de Clases: Un diseño de clases representa la vista estática del sistema, o lo que es lo mismo, representa los elementos estructurales y sus relaciones, independientemente de su función.

Estándar de Codificación: Son pautas a seguir para desarrollar un software en equipo, para lograr la uniformidad del código y reusabilidad del mismo.

Frameworks full-stack: Son frameworks que brindan servicios en todas las capas lógicas de una aplicación.

IDE: Entorno de desarrollo integrado (programa compuesto por un conjunto de herramientas para un desarrollador).

Ingeniería de Requerimientos: Comprende todas las tareas relacionadas con la determinación de las necesidades o de las condiciones a satisfacer para un software nuevo o modificado.

Java Bean: Es una de las API que forman parte del estándar de construcción de aplicaciones empresariales J2EE.

Módulo funcional: Es una parte del sistema que este operativo y pueda comenzar a usarse por el cliente.

plug-ins: Son programas que expanden las características de programas principales como el browser y le agregan capacidades multimedia. Se 'conectan' a una aplicación y corren como parte de esa aplicación.

Pruebas Unitarias: Pruebas que se centran en la verificación de los más pequeños elementos comprobables del software. Normalmente se aplican a todos los componentes para verificar que funcionen como se espera y se aplican paralelamente al propio desarrollo.

Requisitos Funcionales y no Funcionales: Condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente.

Servlet: Es un objeto que se ejecuta en un servidor o contenedor JEE, fue especialmente diseñado para ofrecer contenido dinámico desde un servidor web.

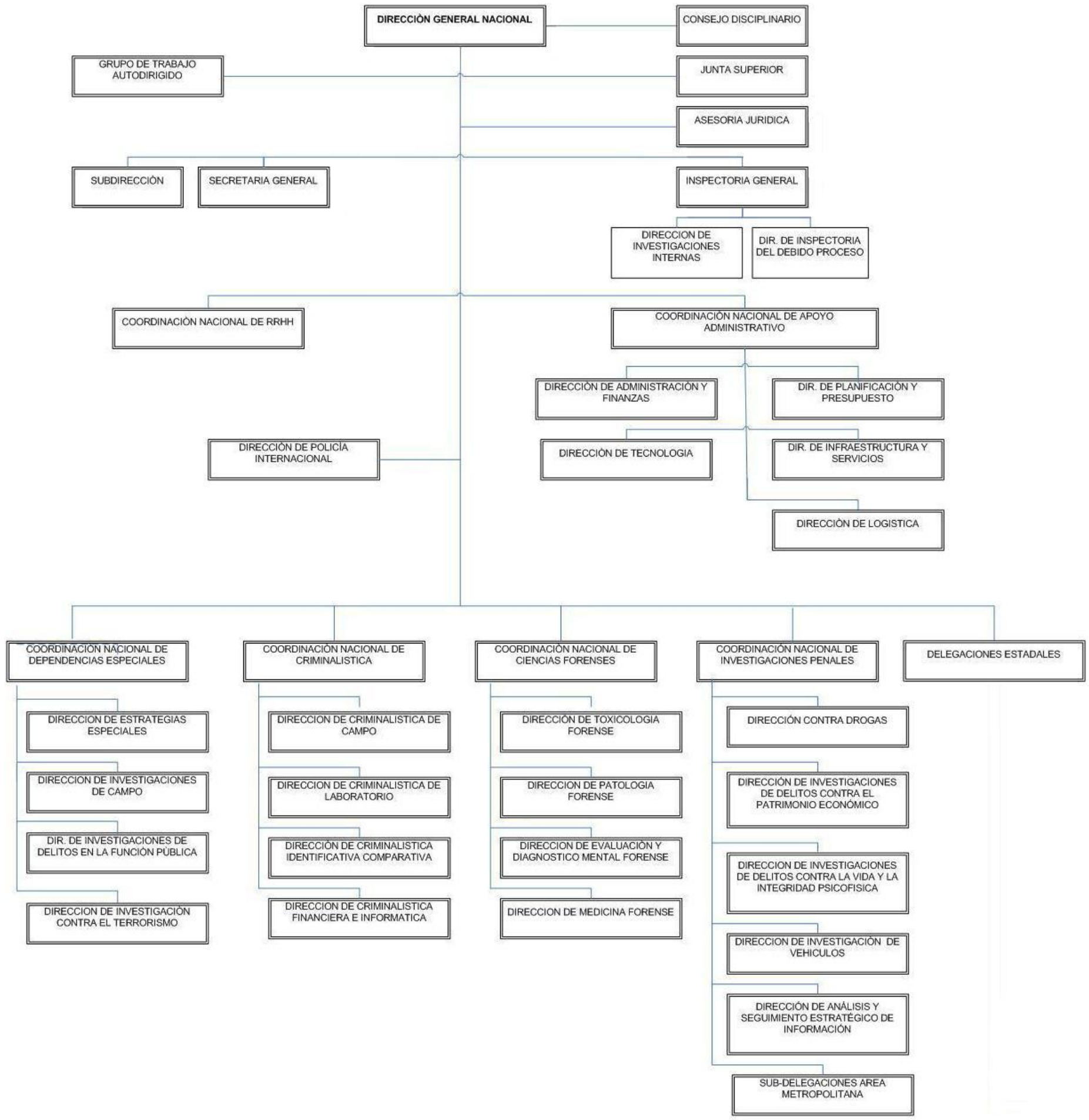
Sistema Integrado de Información Policial: Sistema de gestión de información policial que actalmente es usado por el CICPC.

standalone: independiente

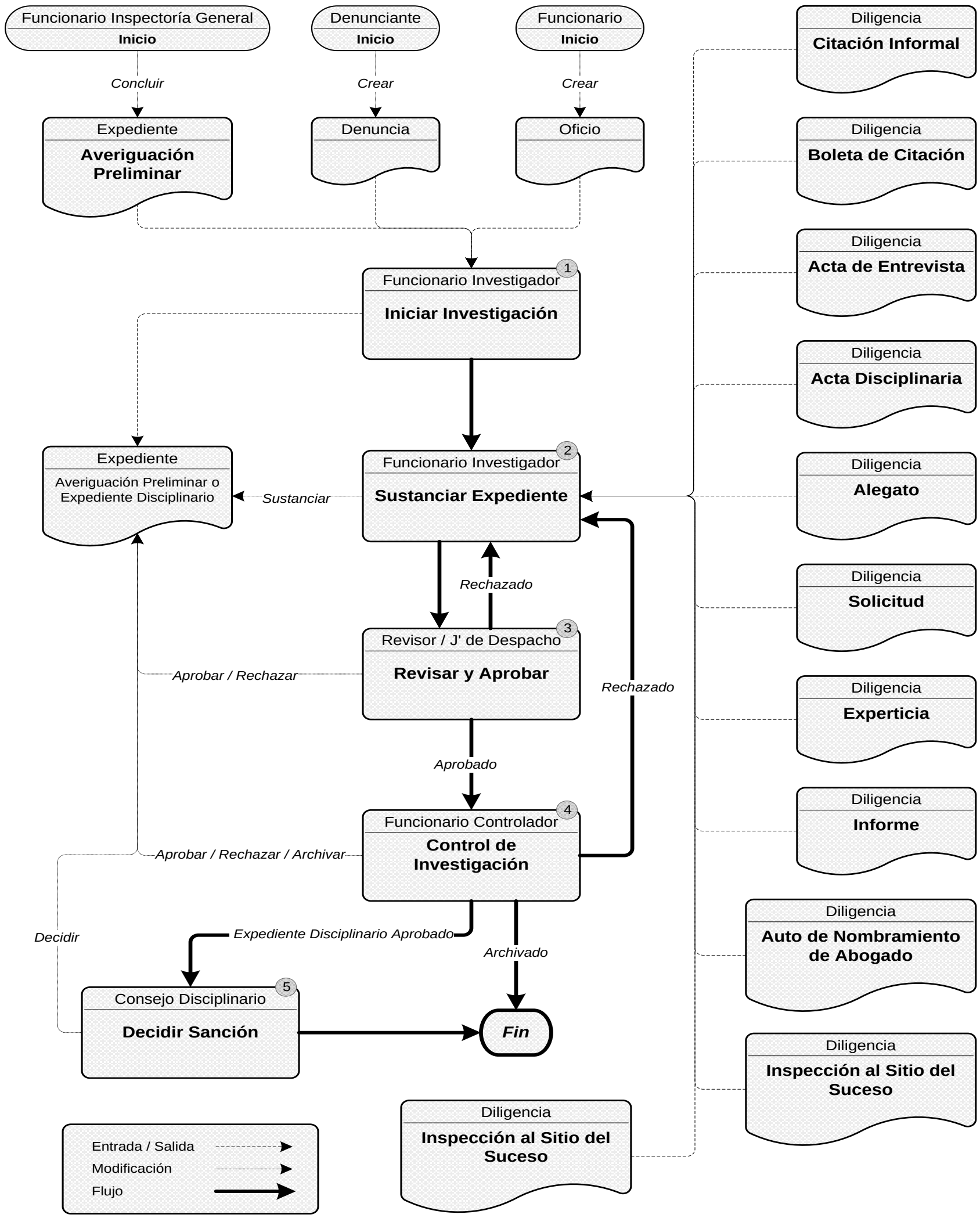
"write once, run everywhere": Es la característica más sobresaliente de java frente a otros lenguajes de programación, por su operación independiente de la plataforma, con lo cual una aplicación se compila una sola vez, y el código de bytes de java resultante (archivos .class) puede ser interpretado en cualquier sistema operativo, que tenga una máquina virtual instalada.

Anexos.

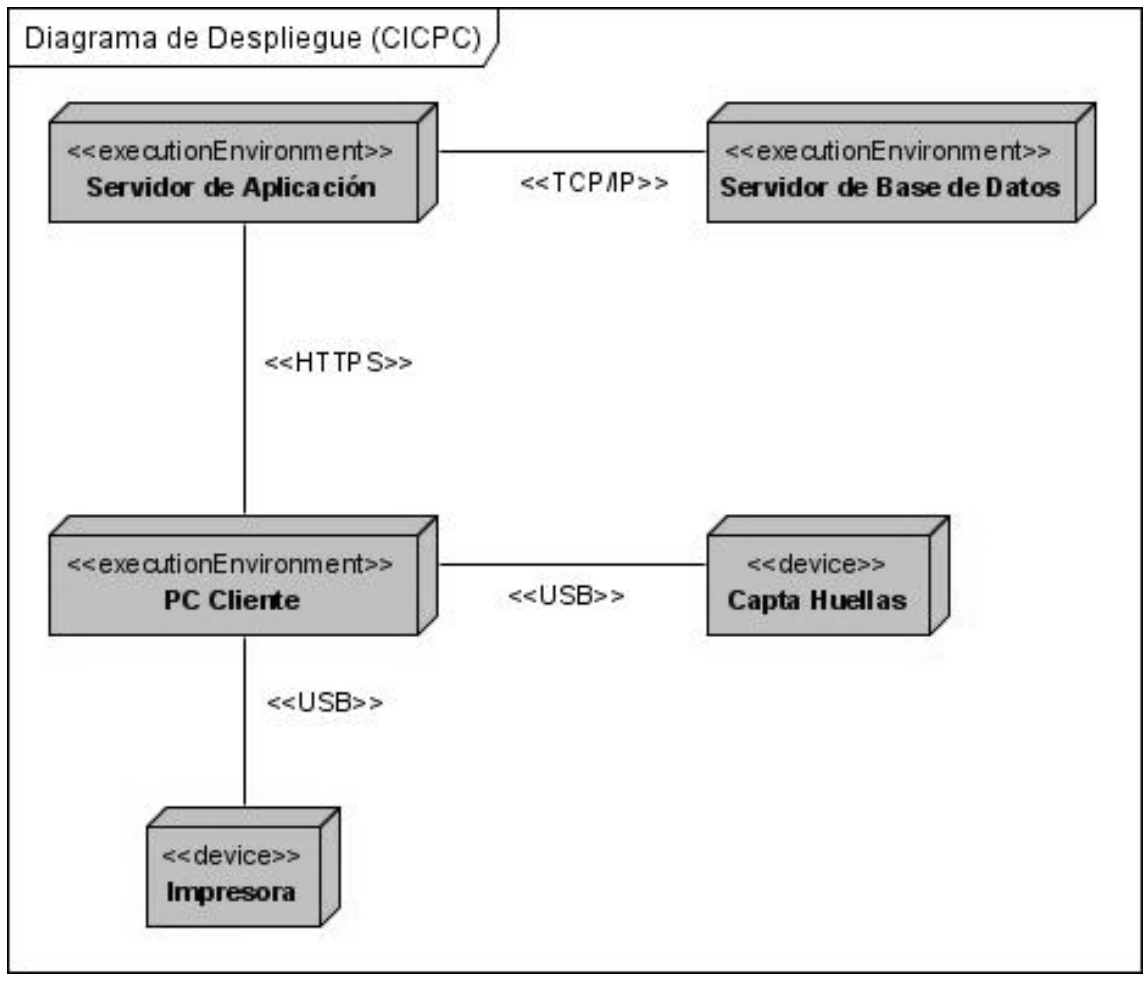
Anexo 1: Estructura organizacional del CICPC.



Anexo 2: Procesos de Investigación Interna.



Anexo 3: Diagrama de despliegue.



Anexo 4: Diagrama de Clases de Análisis

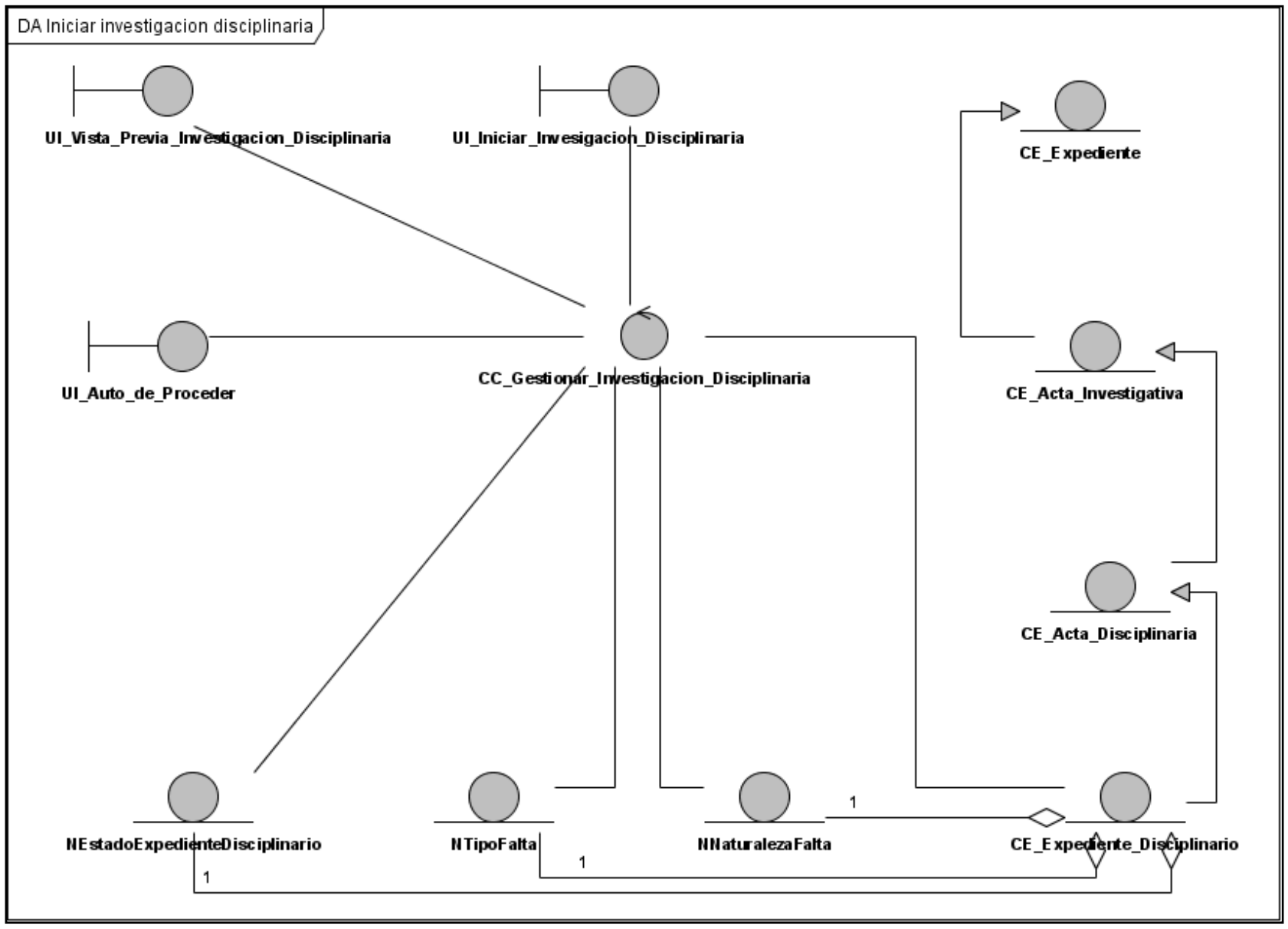


Diagrama de Análisis 1: Iniciar Investigación Disciplinaria.

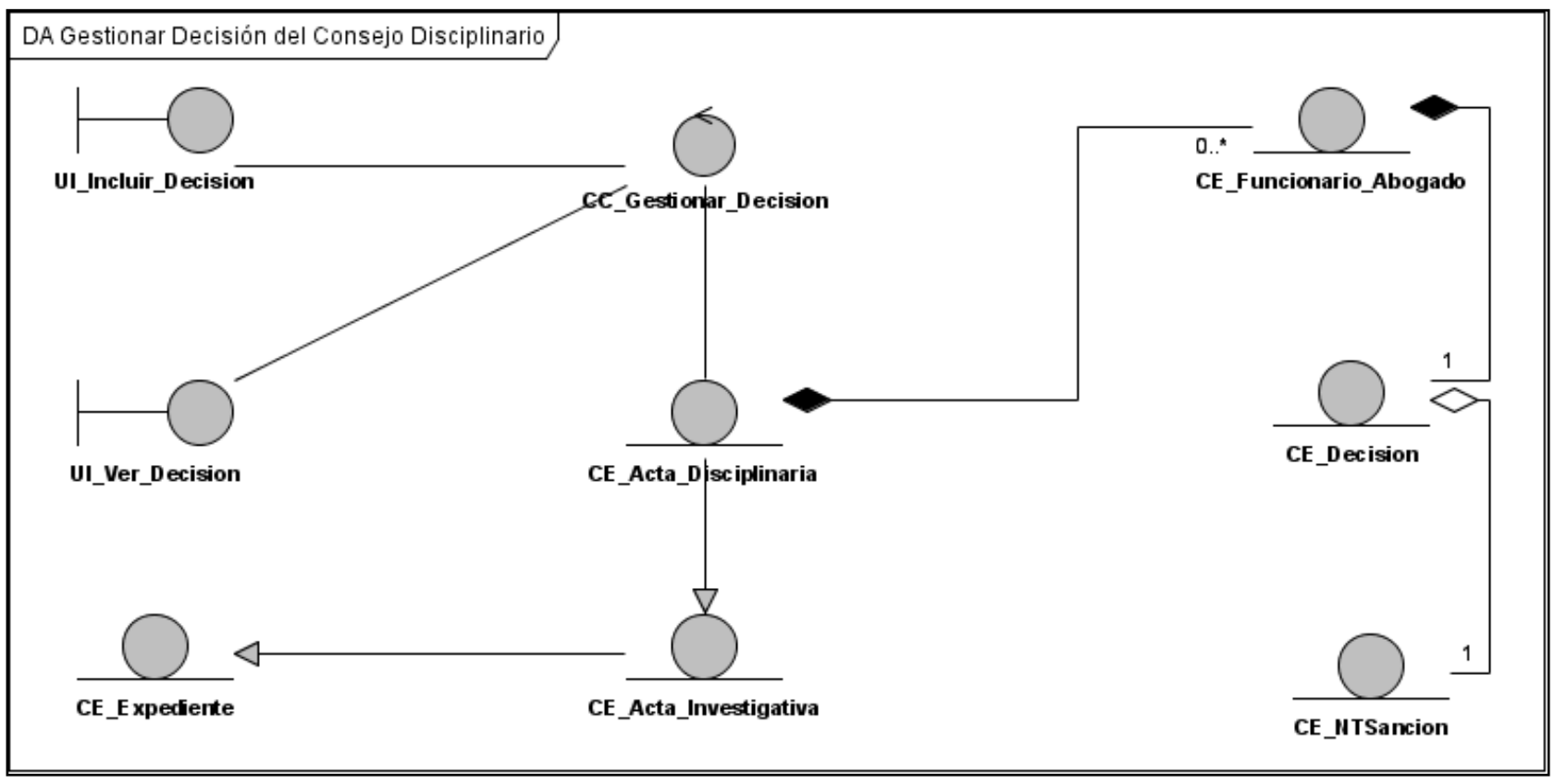


Diagrama de Análisis 2: Gestionar Decisión del Consejo Disciplinario.

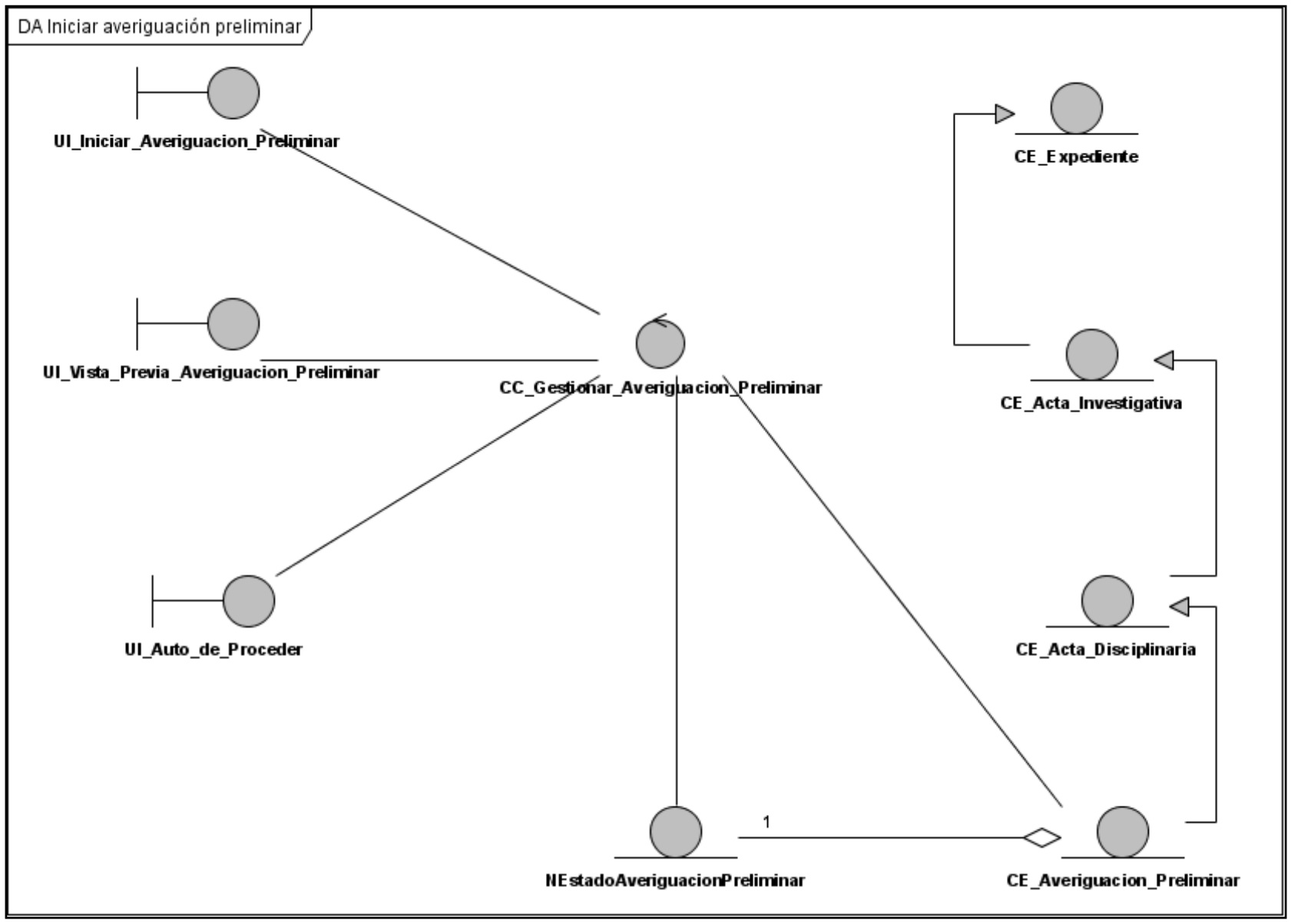


Diagrama de Análisis 3: Iniciar Averiguación Preliminar.

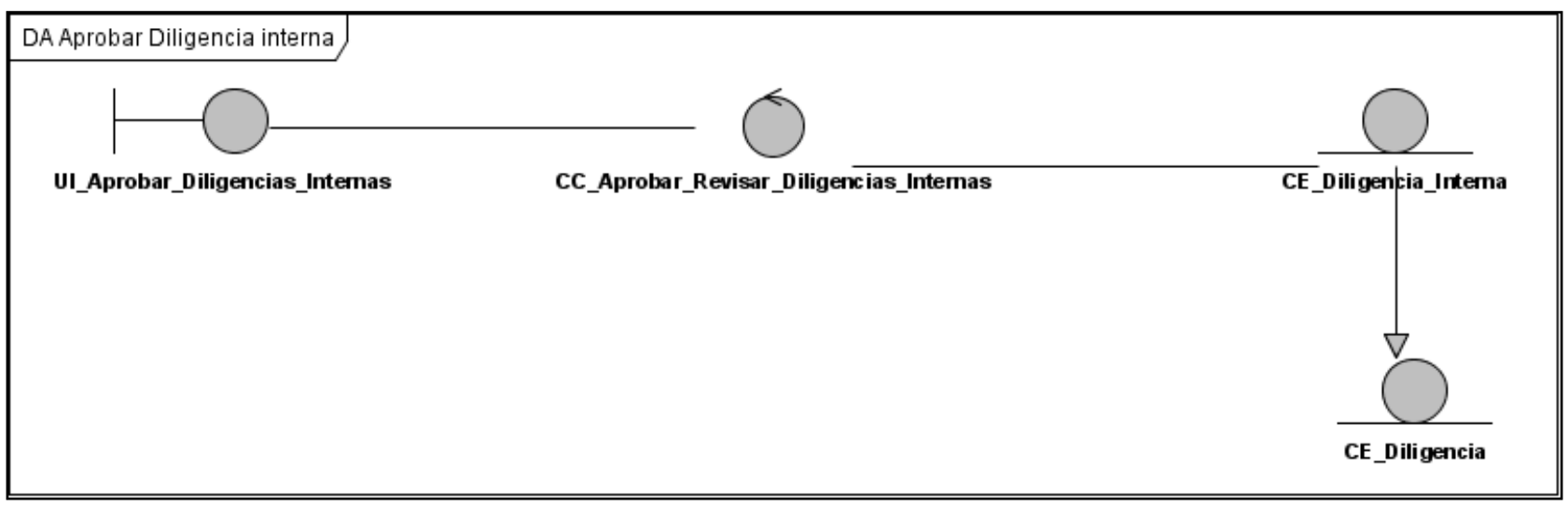


Diagrama de Análisis 4: Aprobar Diligencia Interna.

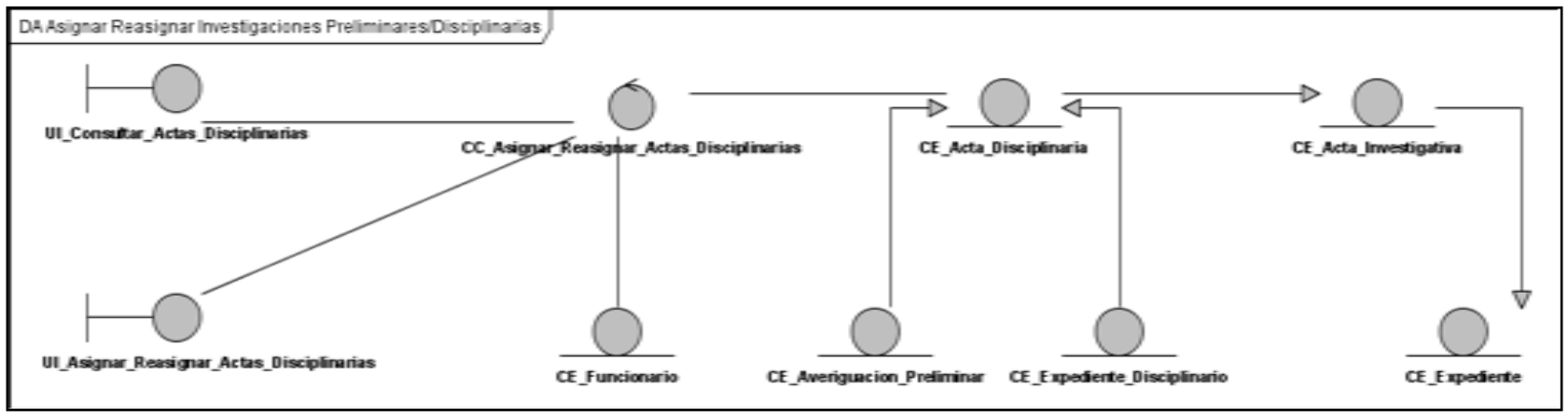


Diagrama de Análisis 5: Asignar Reasignar Investigaciones Preliminares/Disciplinarias.

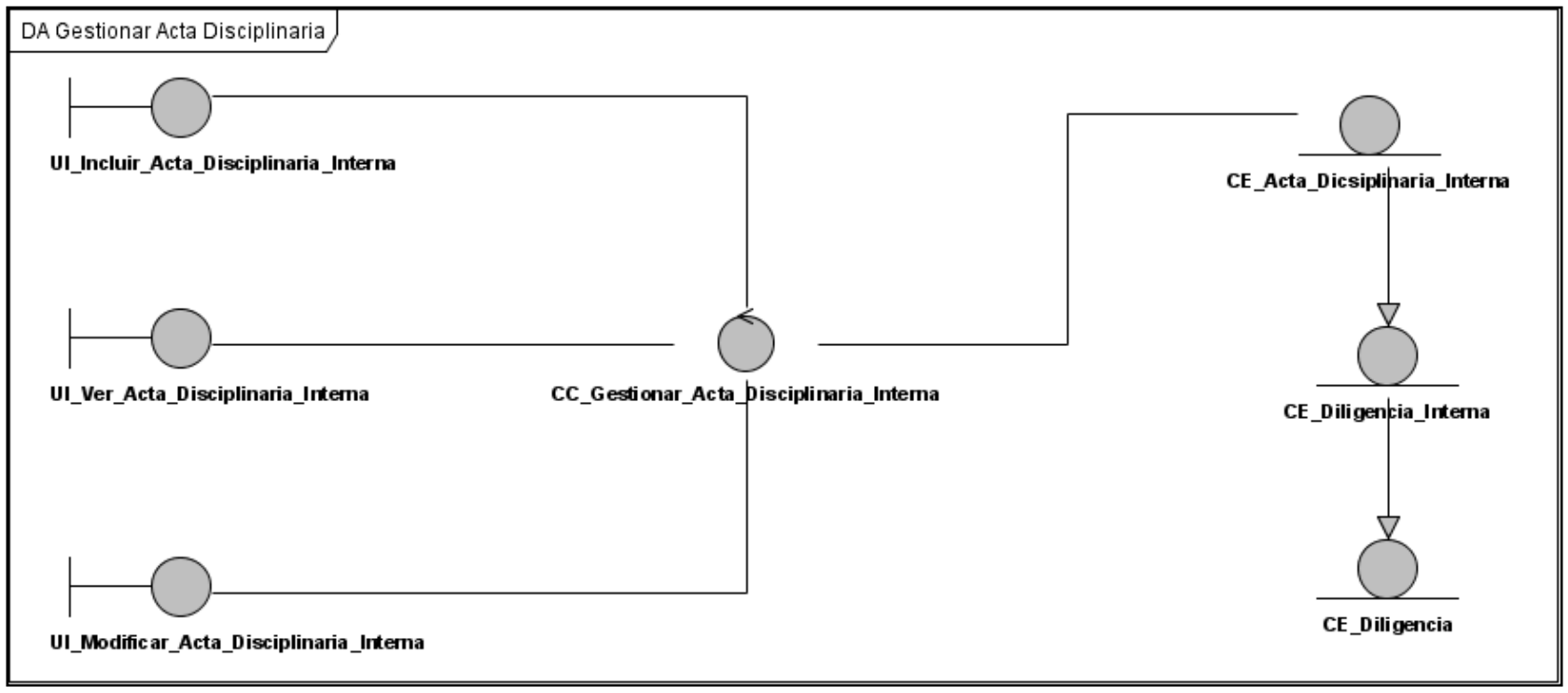


Diagrama de Análisis 6: Gestionar Acta Disciplinaria.

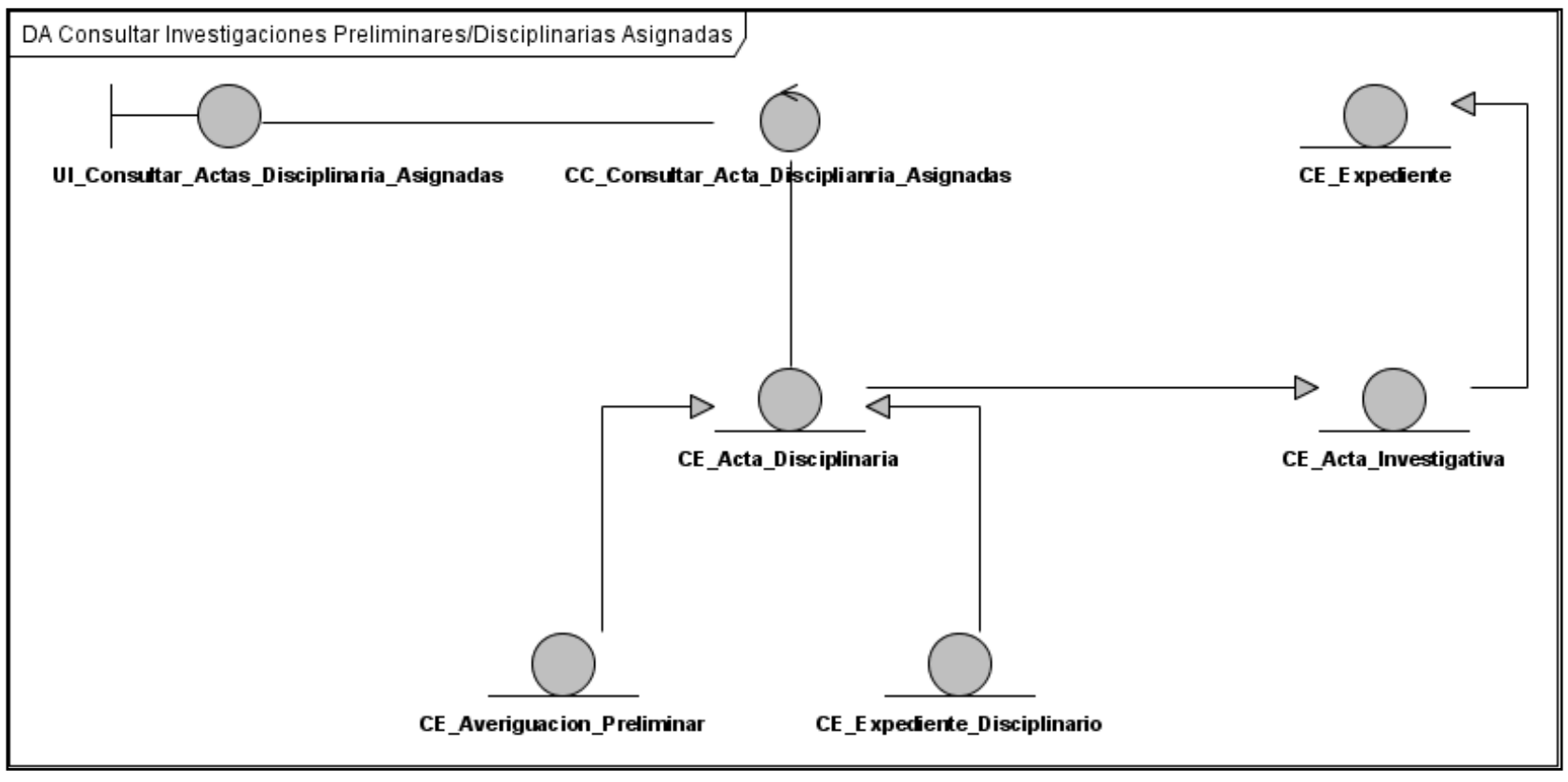


Diagrama de Análisis 7: Consultar Investigaciones Preliminares/Disciplinarias Asignadas.

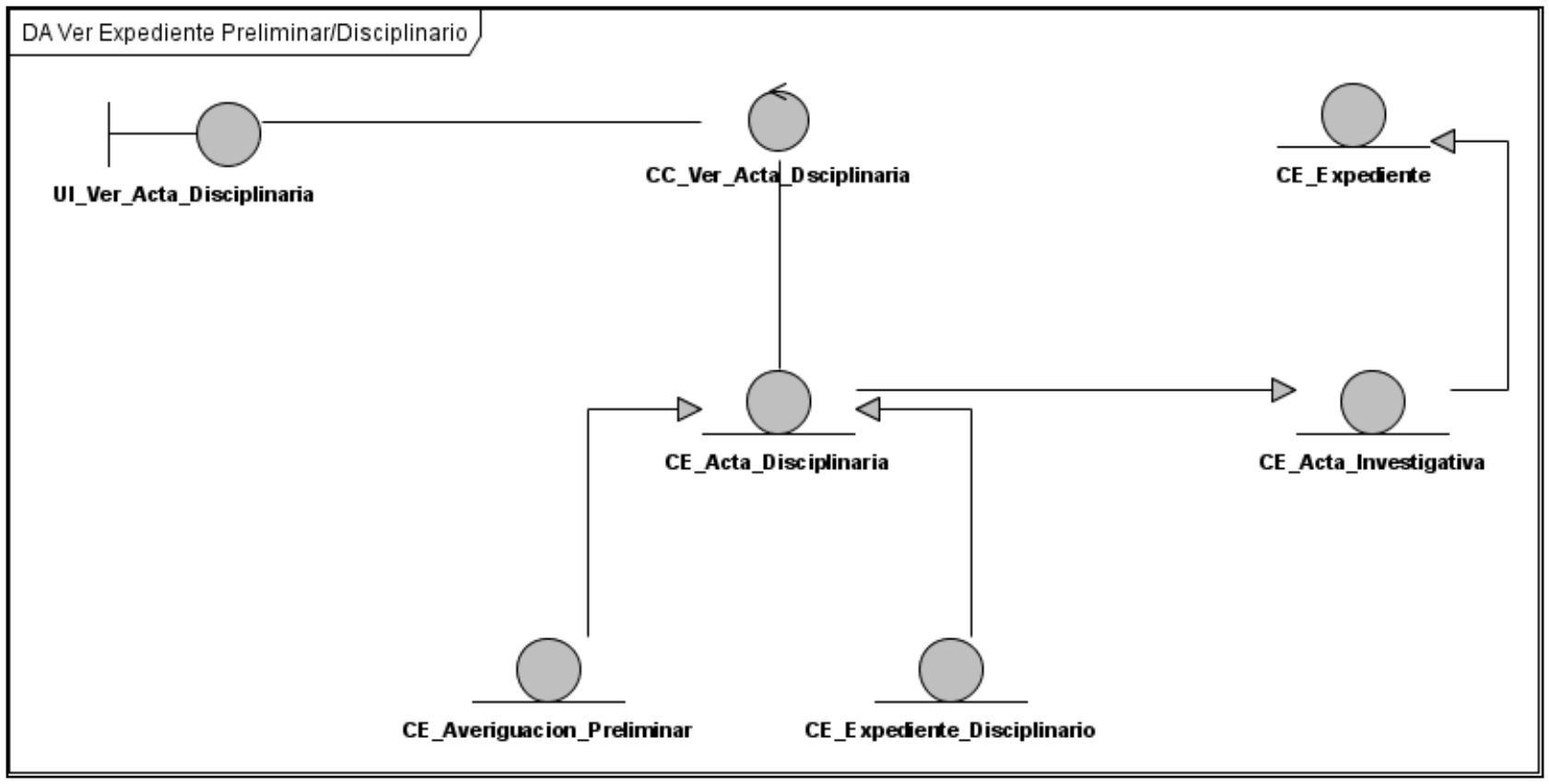


Diagrama de Análisis 8: Ver Expediente Preliminar/Disciplinario.

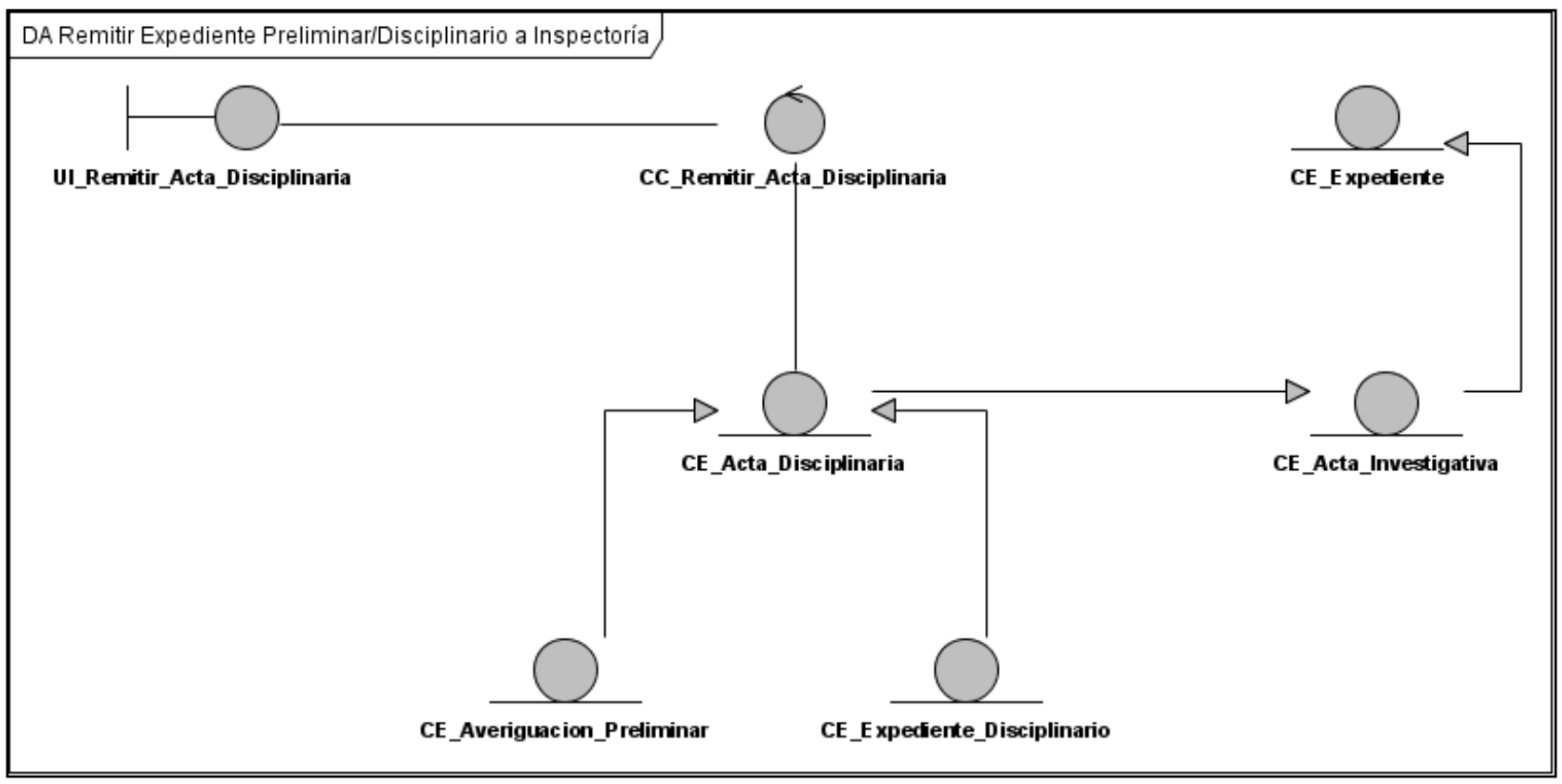


Diagrama de Análisis 9: Remitir Expediente Preliminar/Disciplinario a Inspectoría.

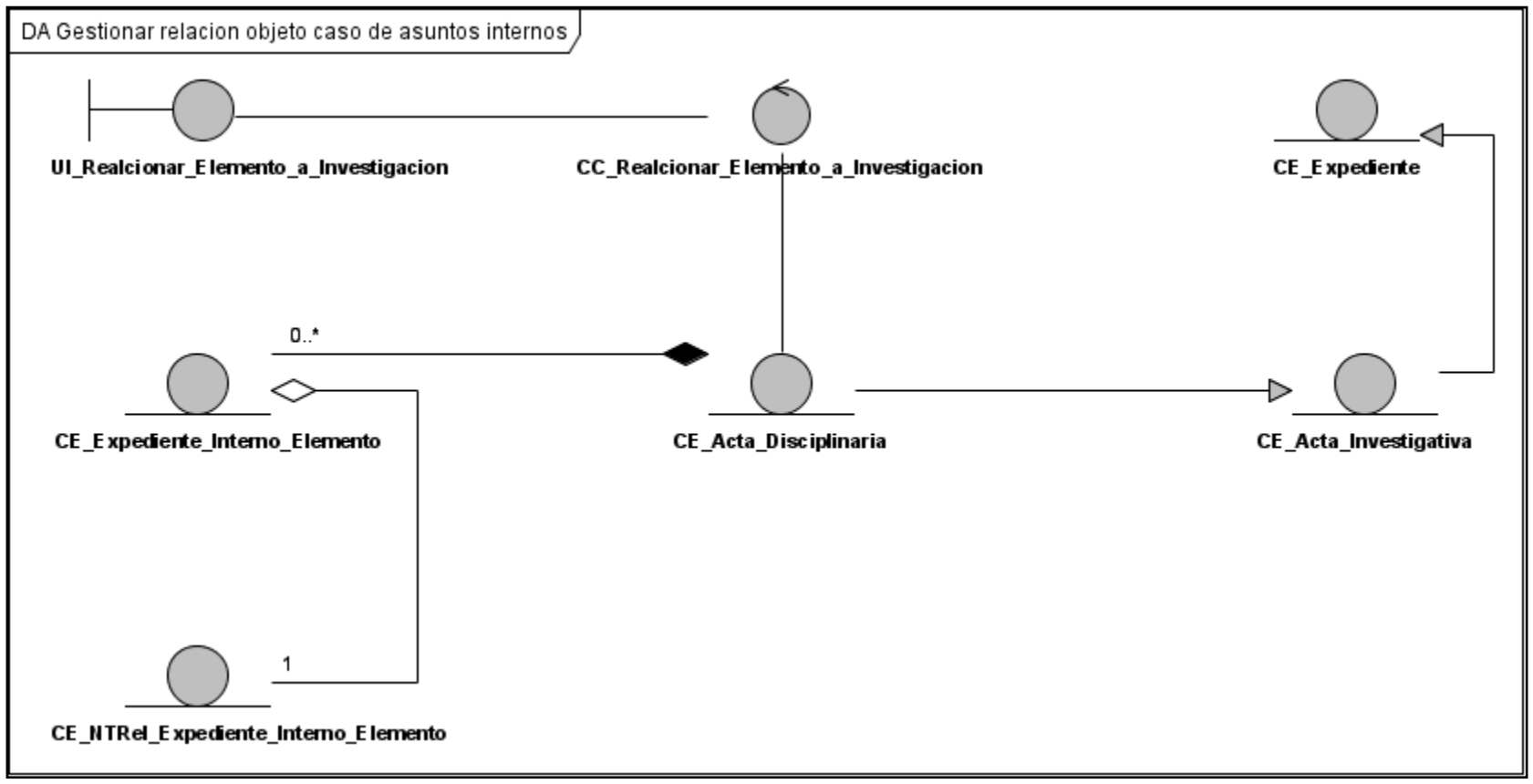
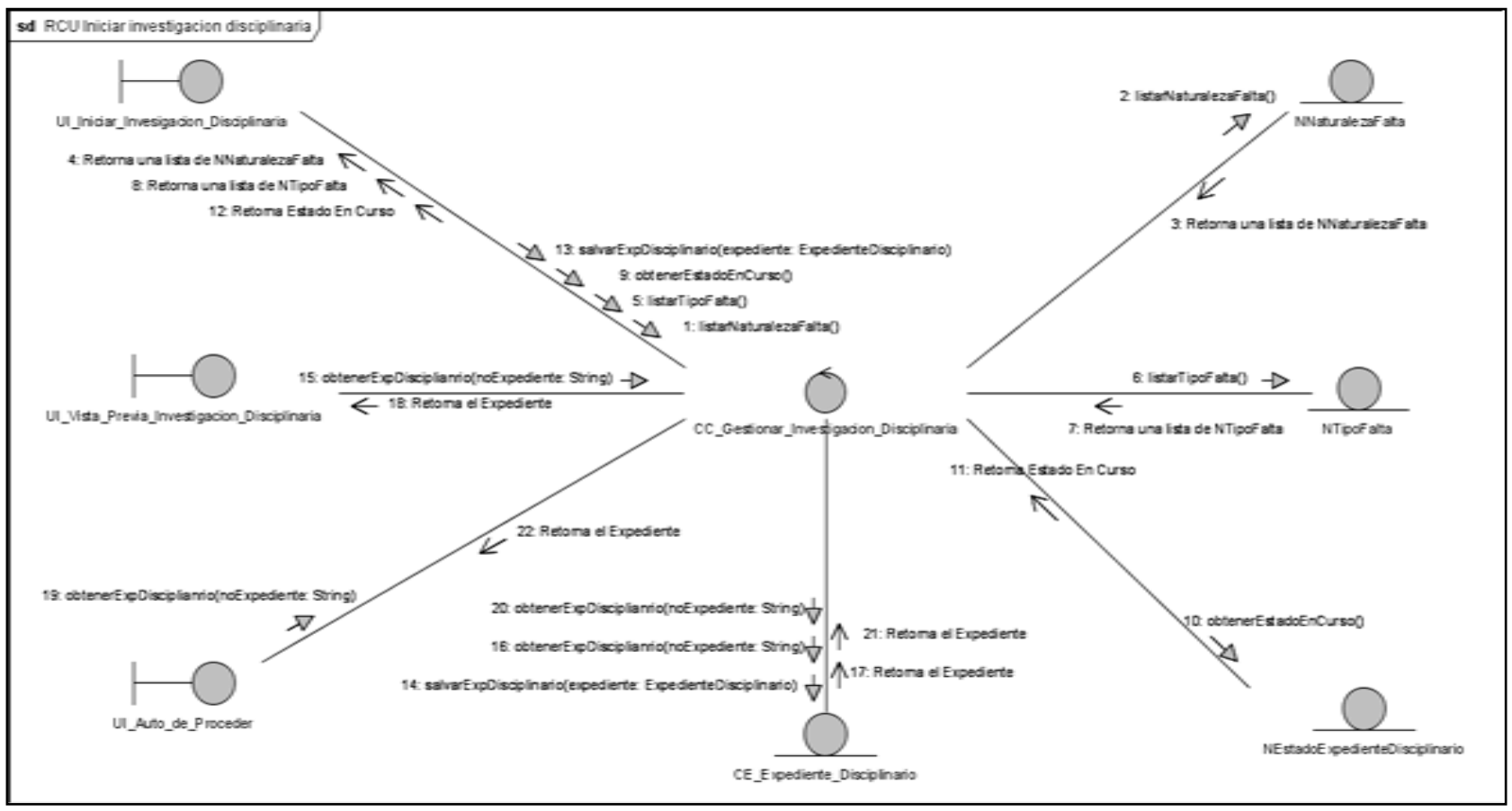
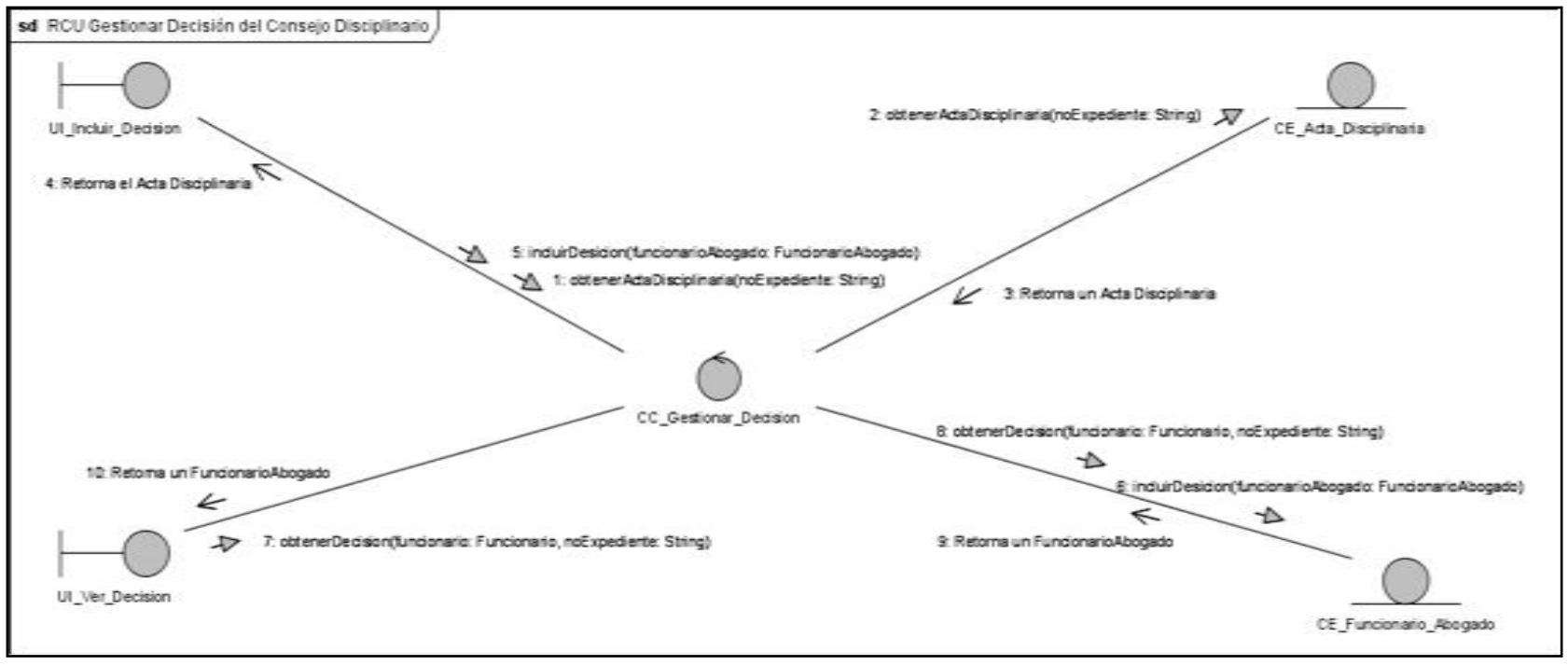


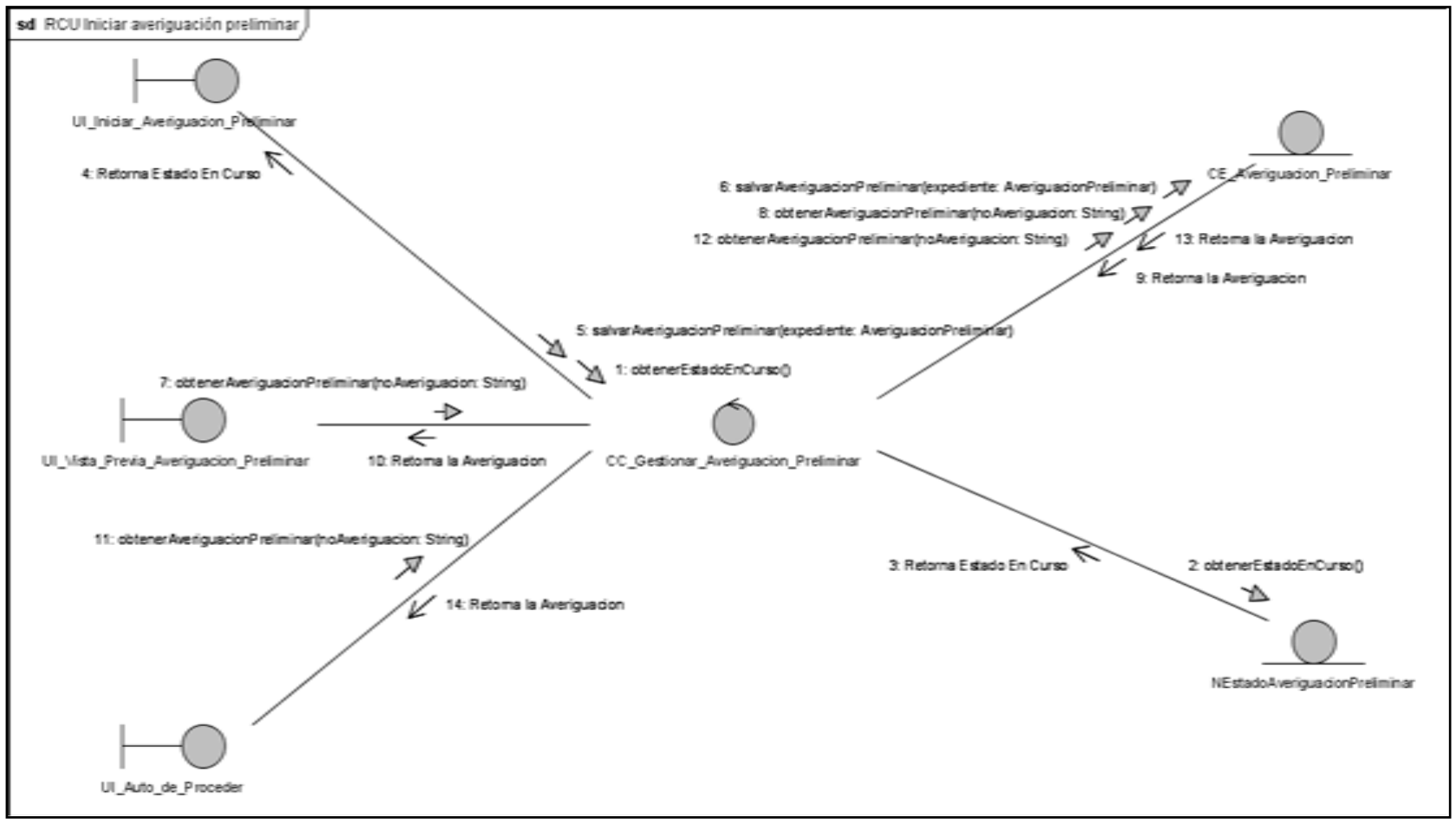
Diagrama de Análisis 10: Gestionar Relación Objeto Caso de Asuntos Internos.Realización de los Casos de Usos Significativos



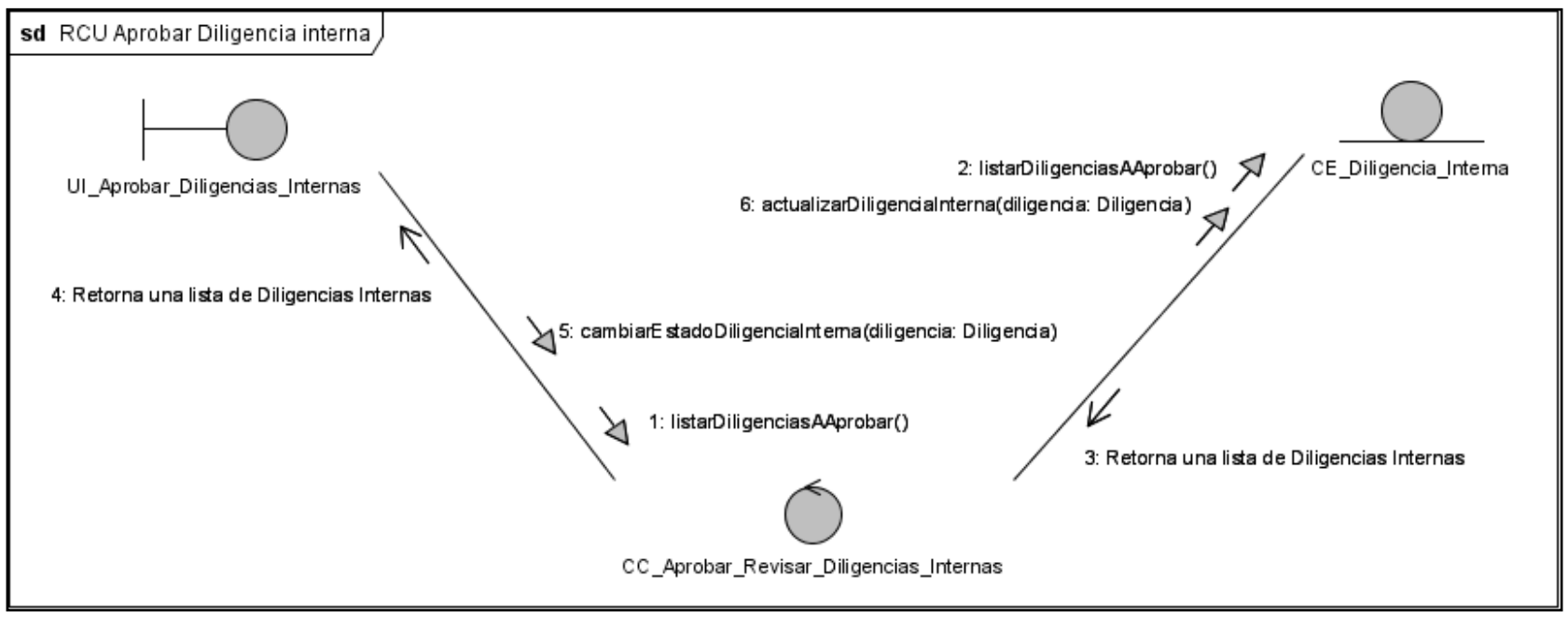
Realización de Caso de Uso 1: Iniciar Investigación Disciplinaria.



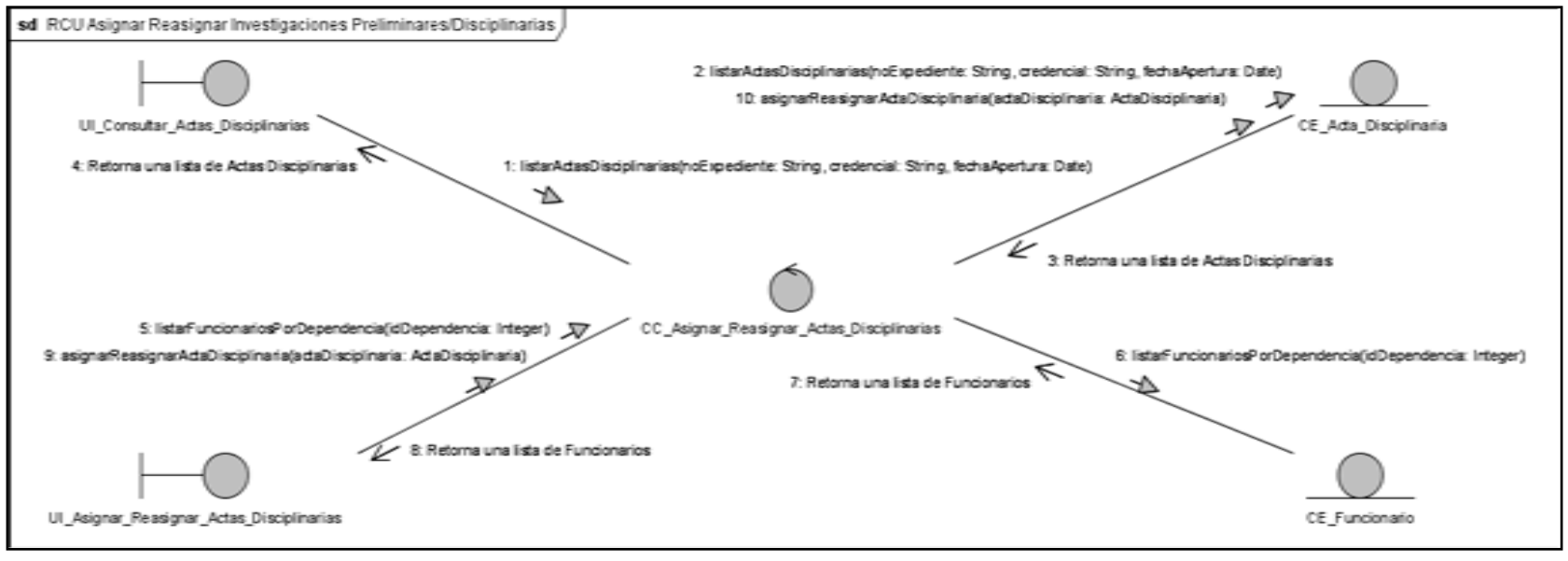
Realización de Caso de Uso 2: Gestionar Decisión del Consejo Disciplinario.



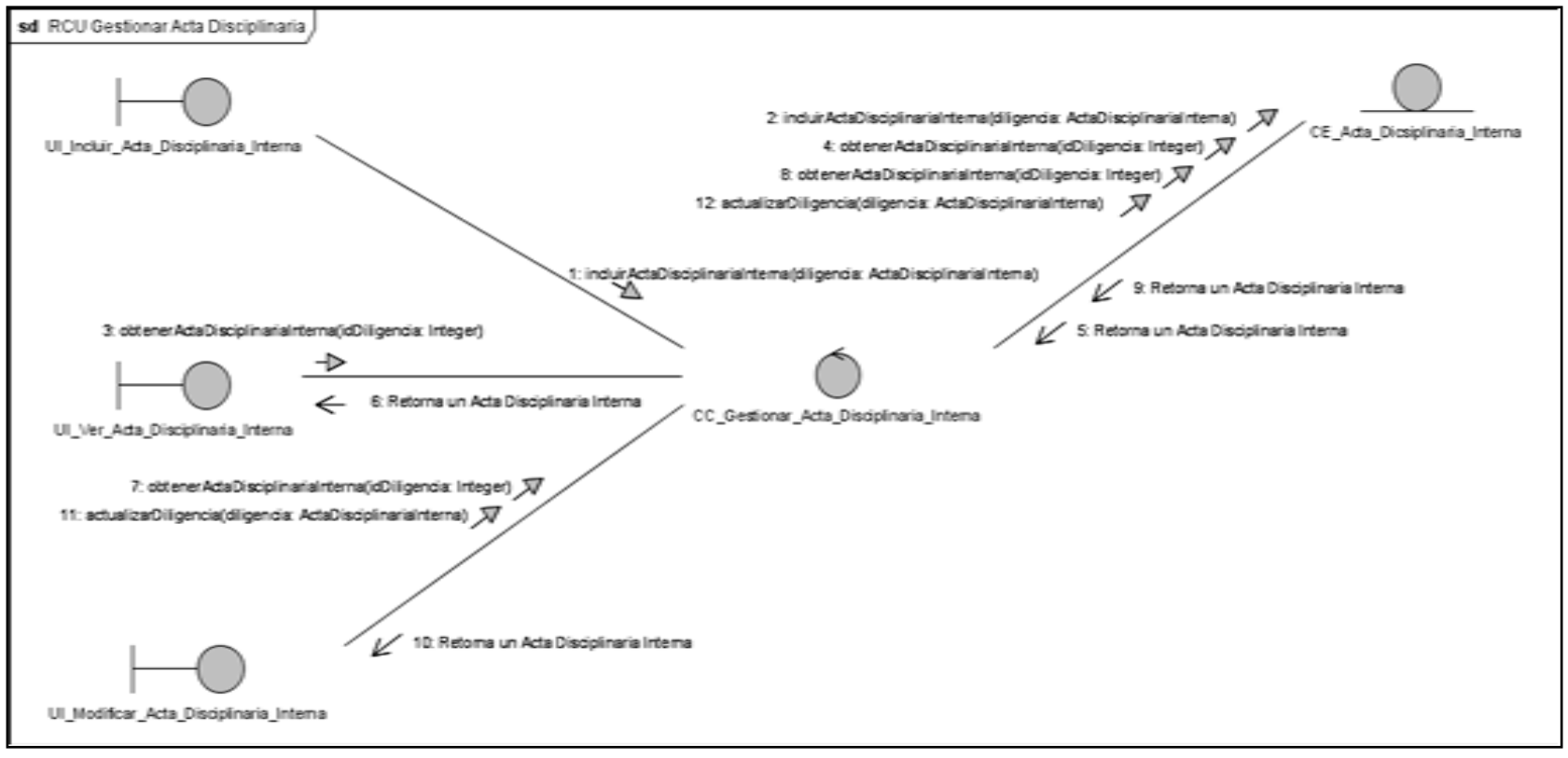
Realización de Caso de Uso 3: Iniciar Averiguación Preliminar.



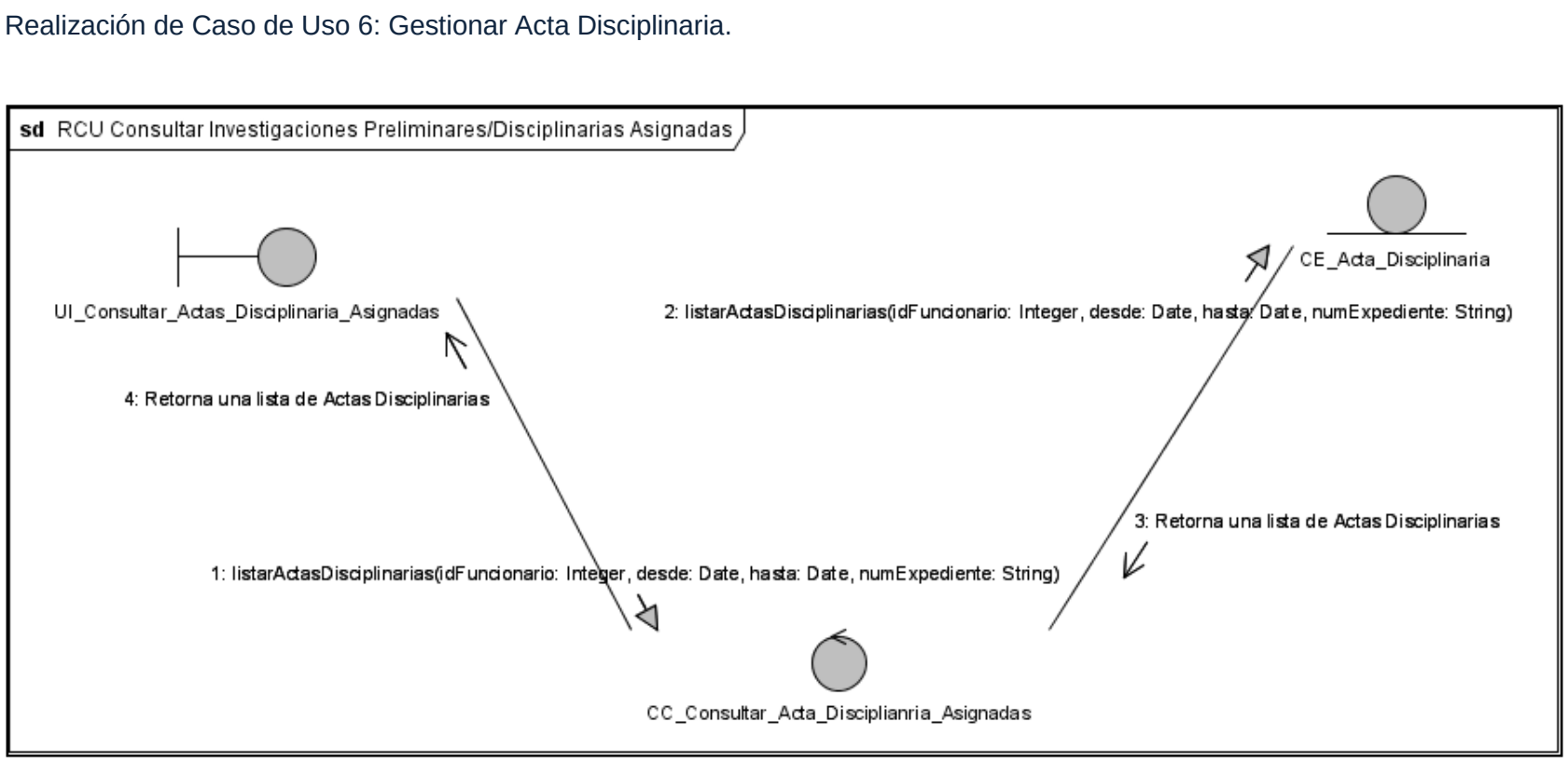
Realización de Caso de Uso 4: Aprobar Diligencia Interna.



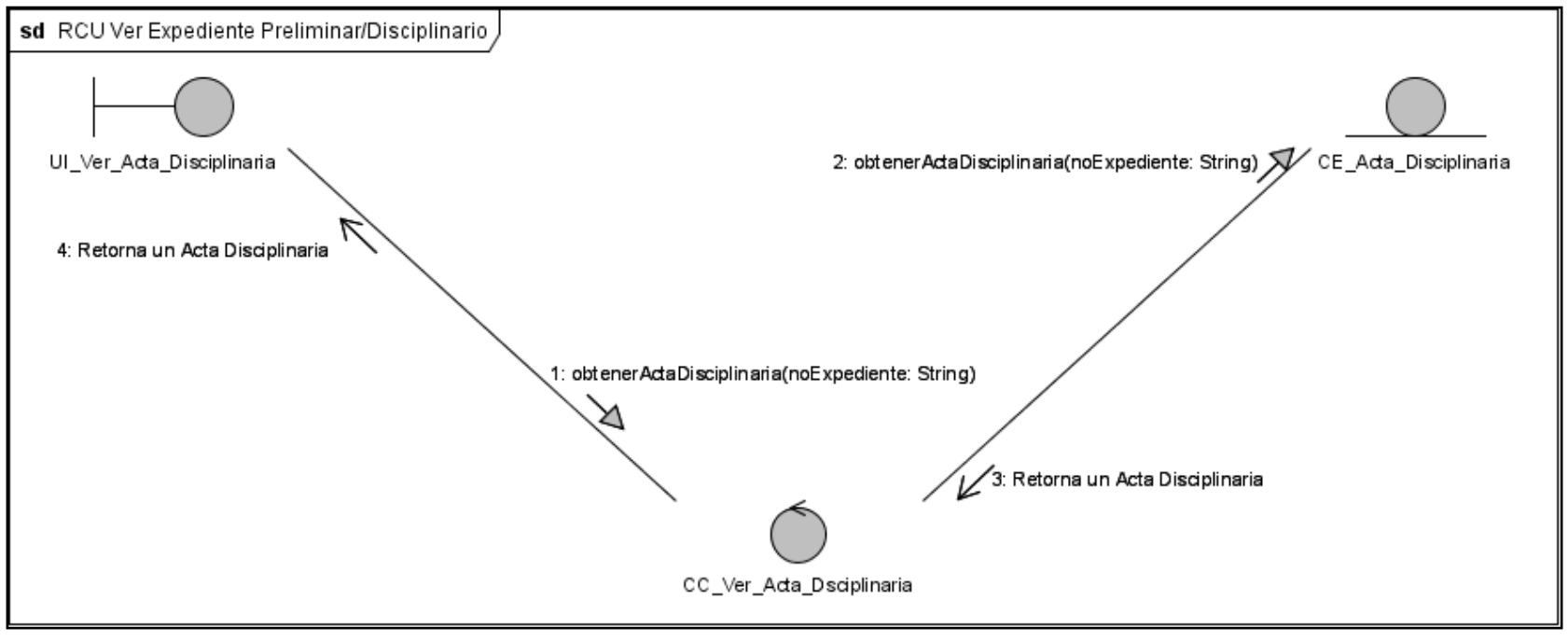
Realización de Caso de Uso 5: Asignar Reasignar Investigaciones Preliminares/Disciplinarias.



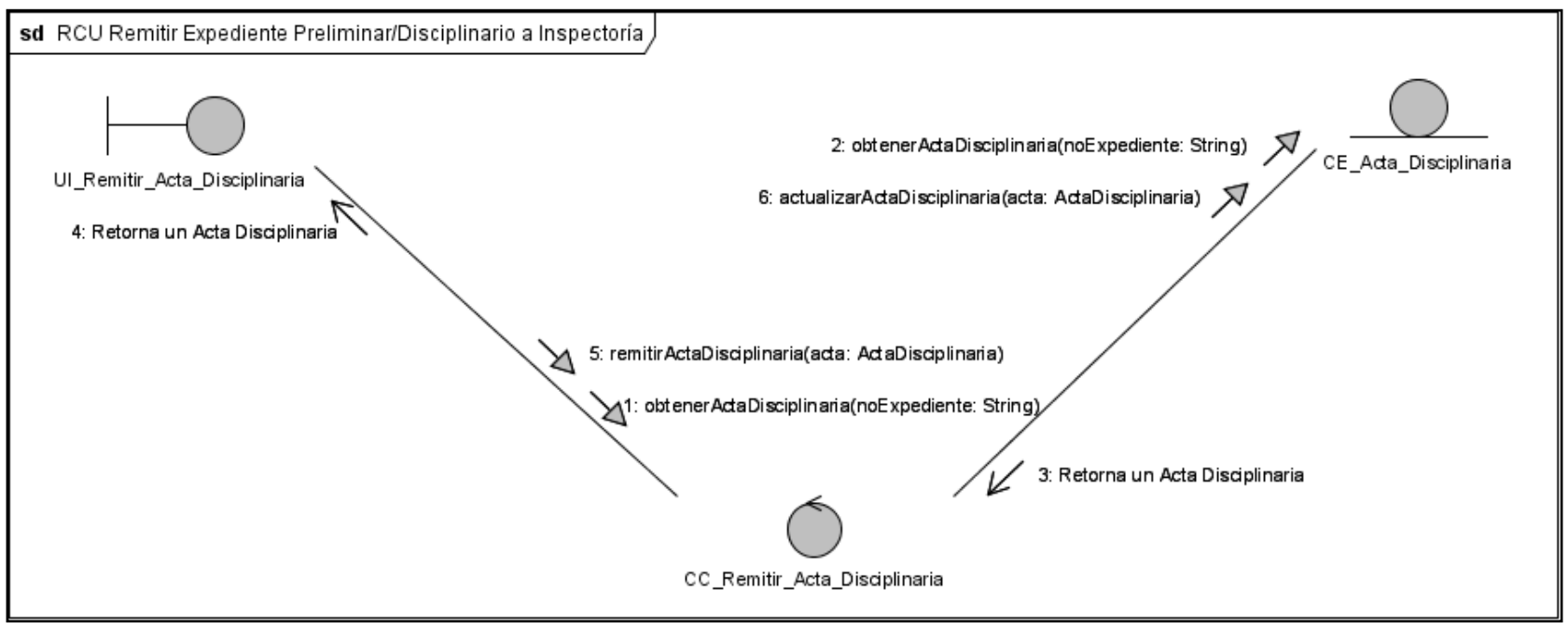
Realización de Caso de Uso 6: Gestionar Acta Disciplinaria.



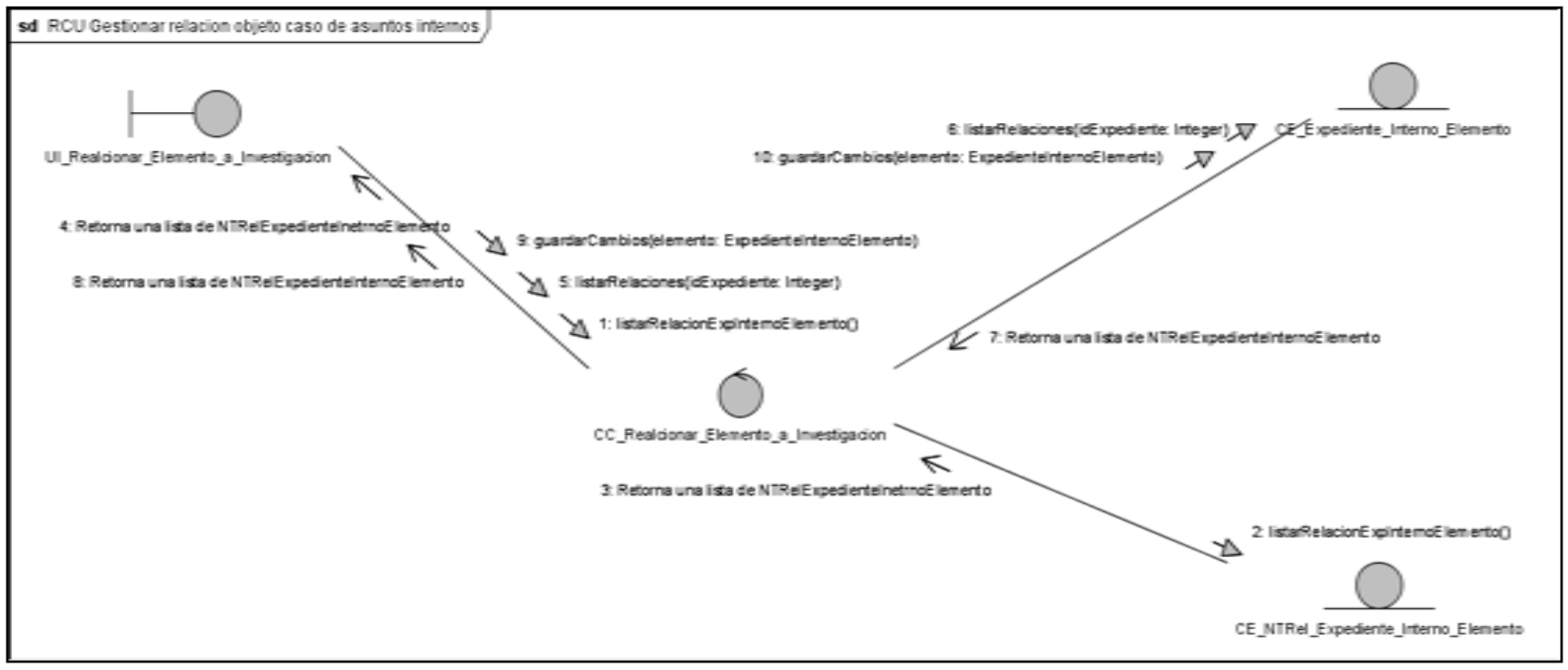
Realización de Caso de Uso 7: Consultar Investigaciones Preliminares/Disciplinarias Asignadas.



Realización de Caso de Uso 8: Ver Expediente Preliminar/Disciplinario.



Realización de Caso de Uso 9: Remitir Expediente Preliminar/Disciplinario a Inspectoría.



Realización de Caso de Uso 10: Gestionar Relación Objeto Caso de Asuntos Internos.

Anexo 5: Diagrama de Paquetes.

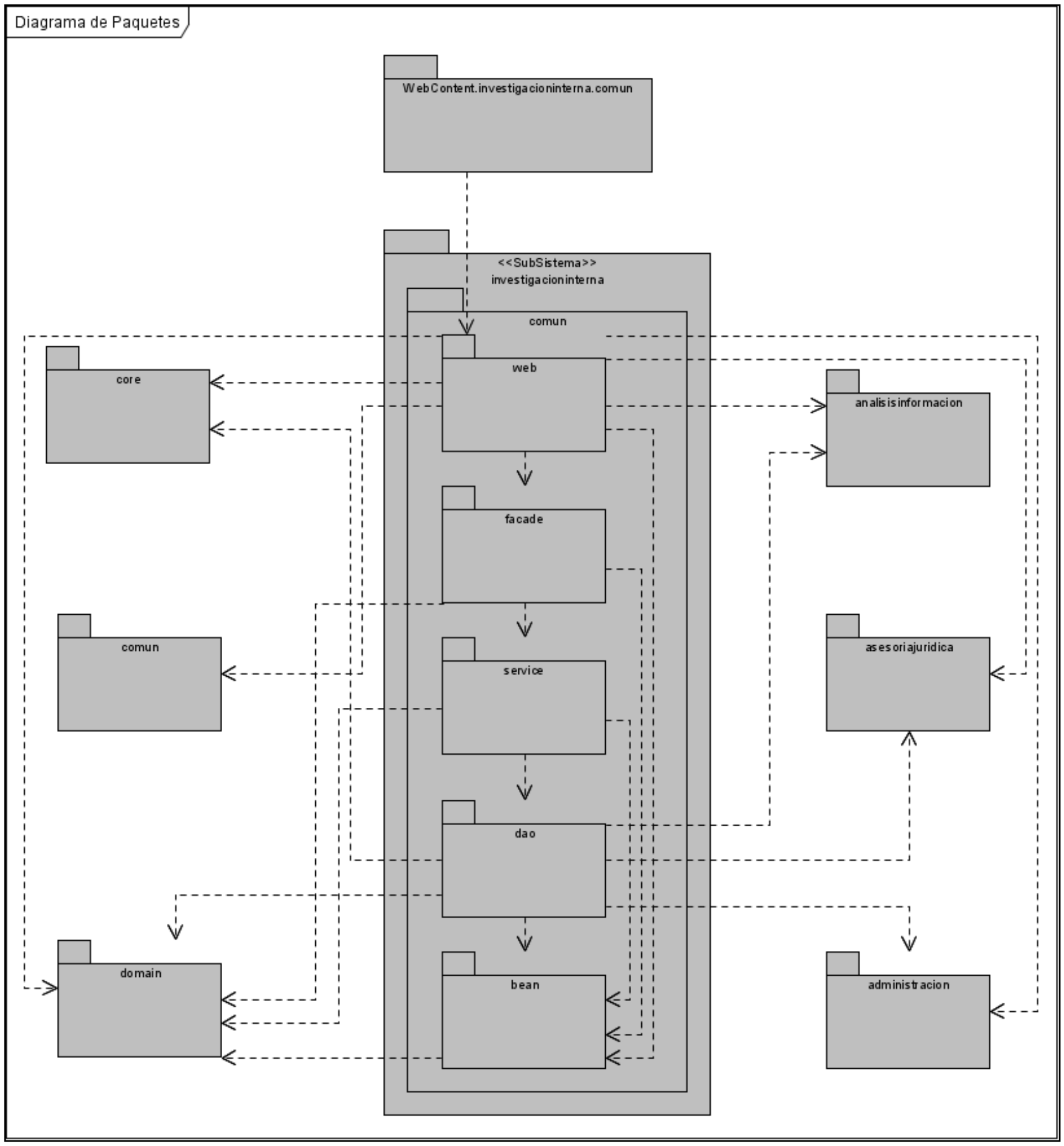


Diagrama de Paquetes 1: Diagrama de Paquetes del Módulo Investigación Interna.

Anexo 6: Diagramas de Clases del Diseño.

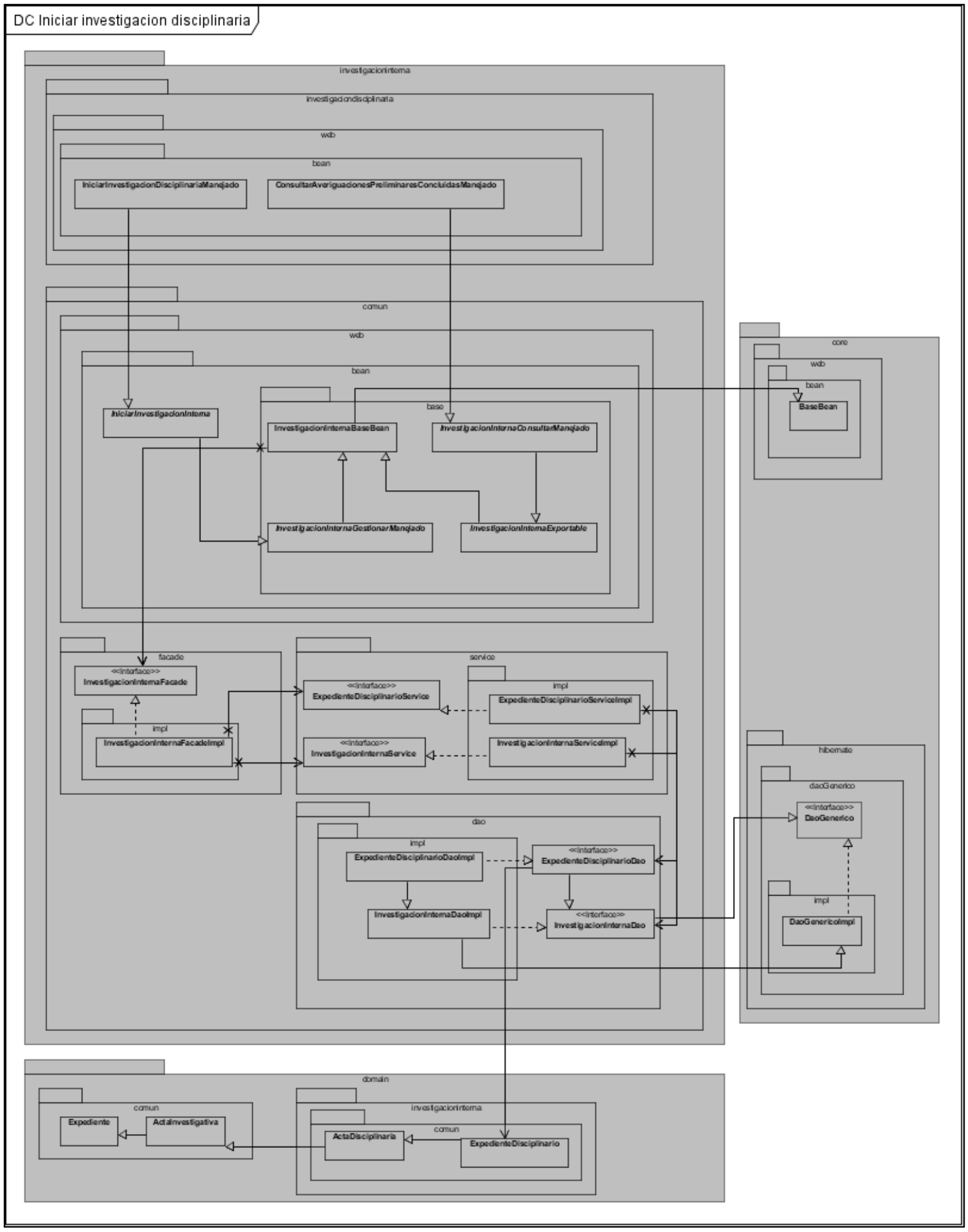


Diagrama de Clases del Diseño 1: Iniciar Investigación Disciplinaria.

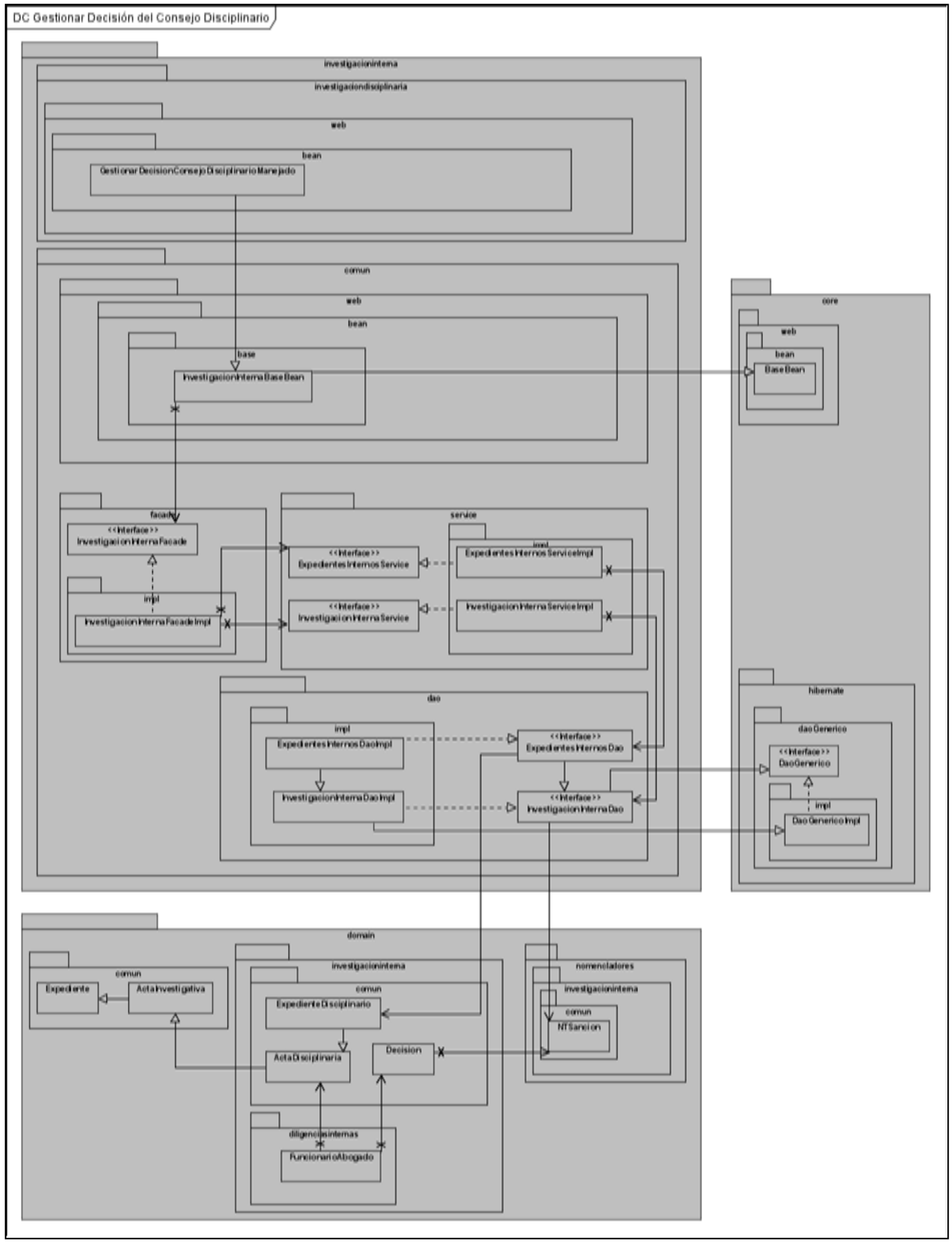


Diagrama de Clases del Diseño 2: Gestionar Decisión del Consejo Disciplinario.

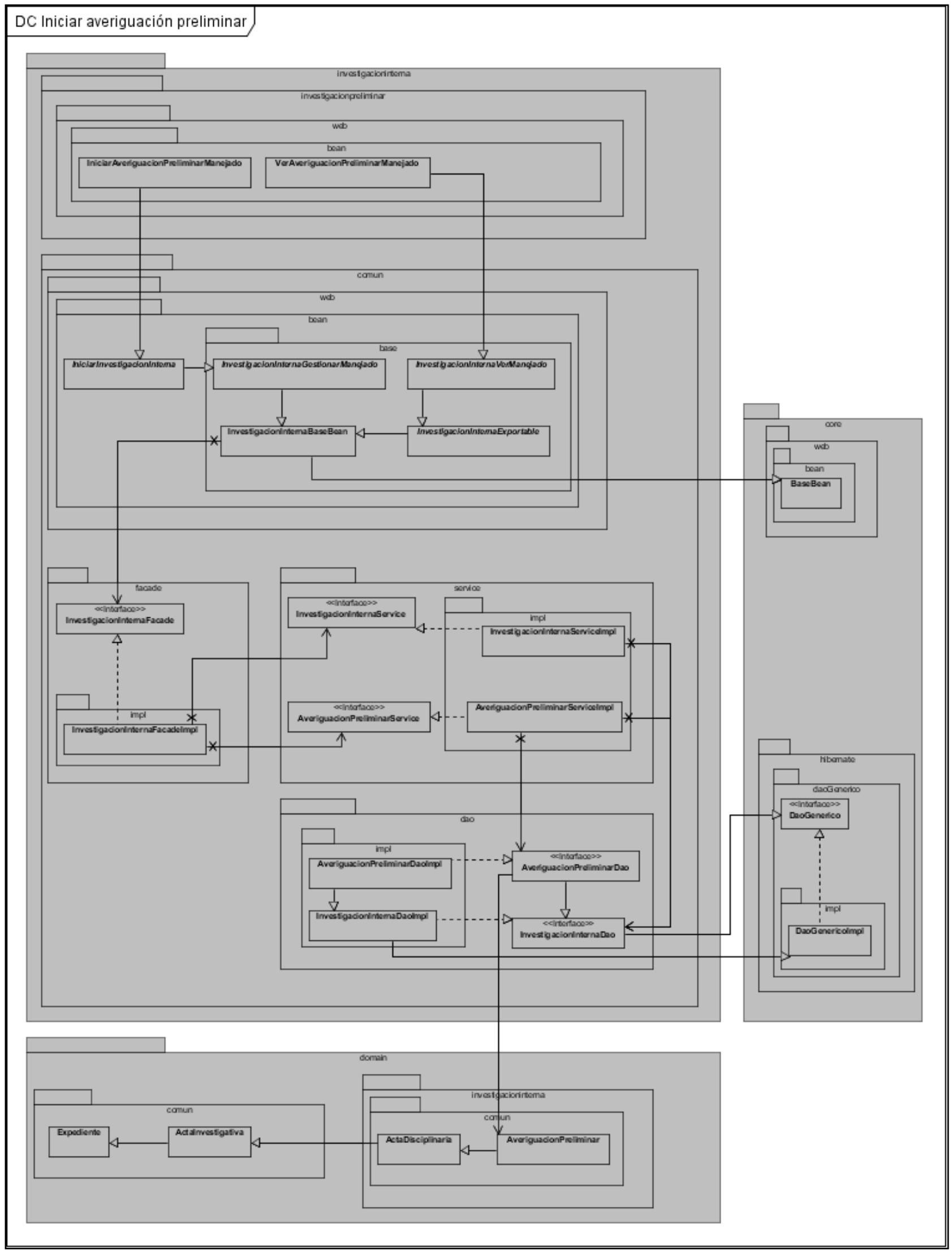


Diagrama de Clases del Diseño 3: Iniciar Averiguación Preliminar.

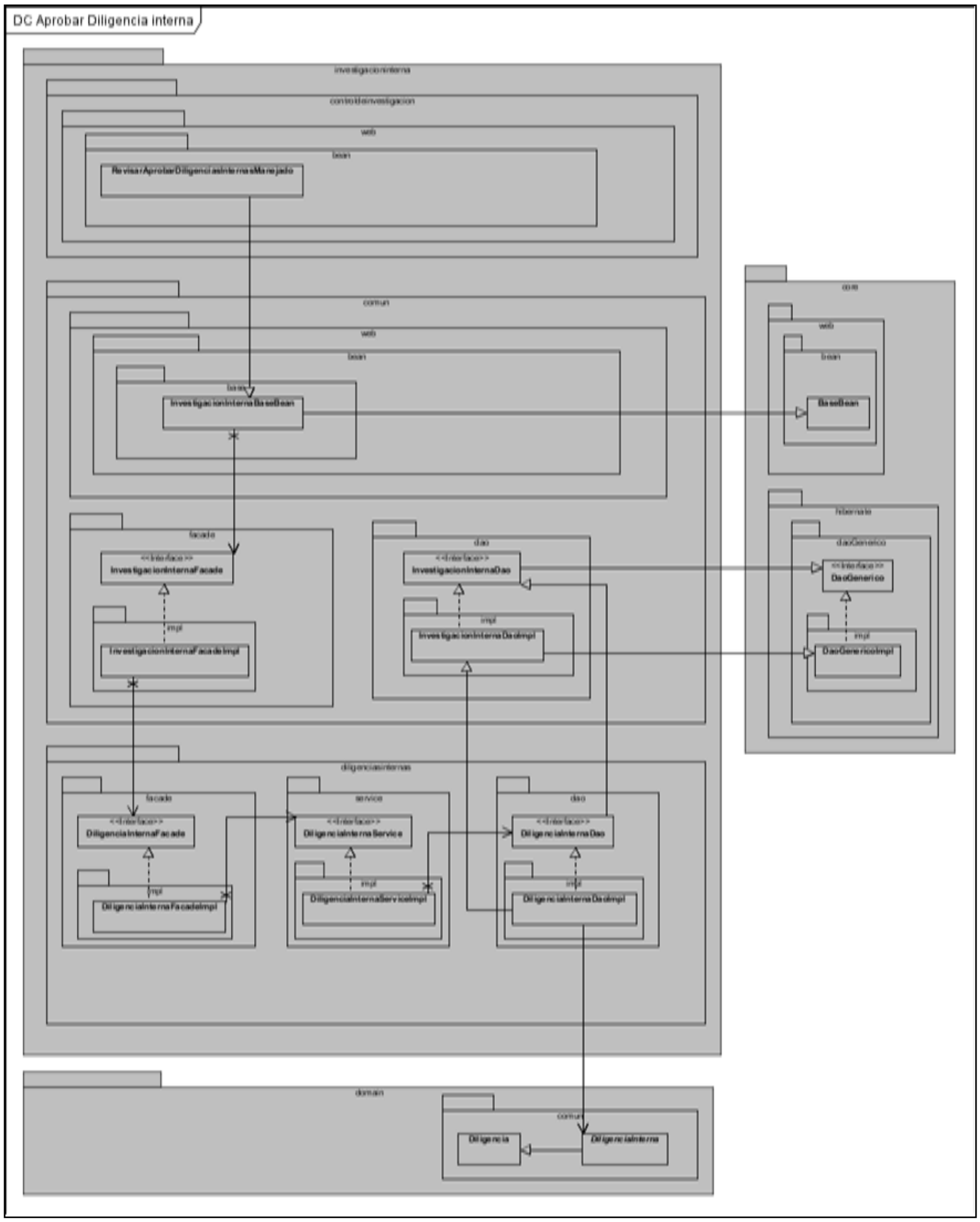


Diagrama de Clases del Diseño 4 Aprobar Diligencia Interna

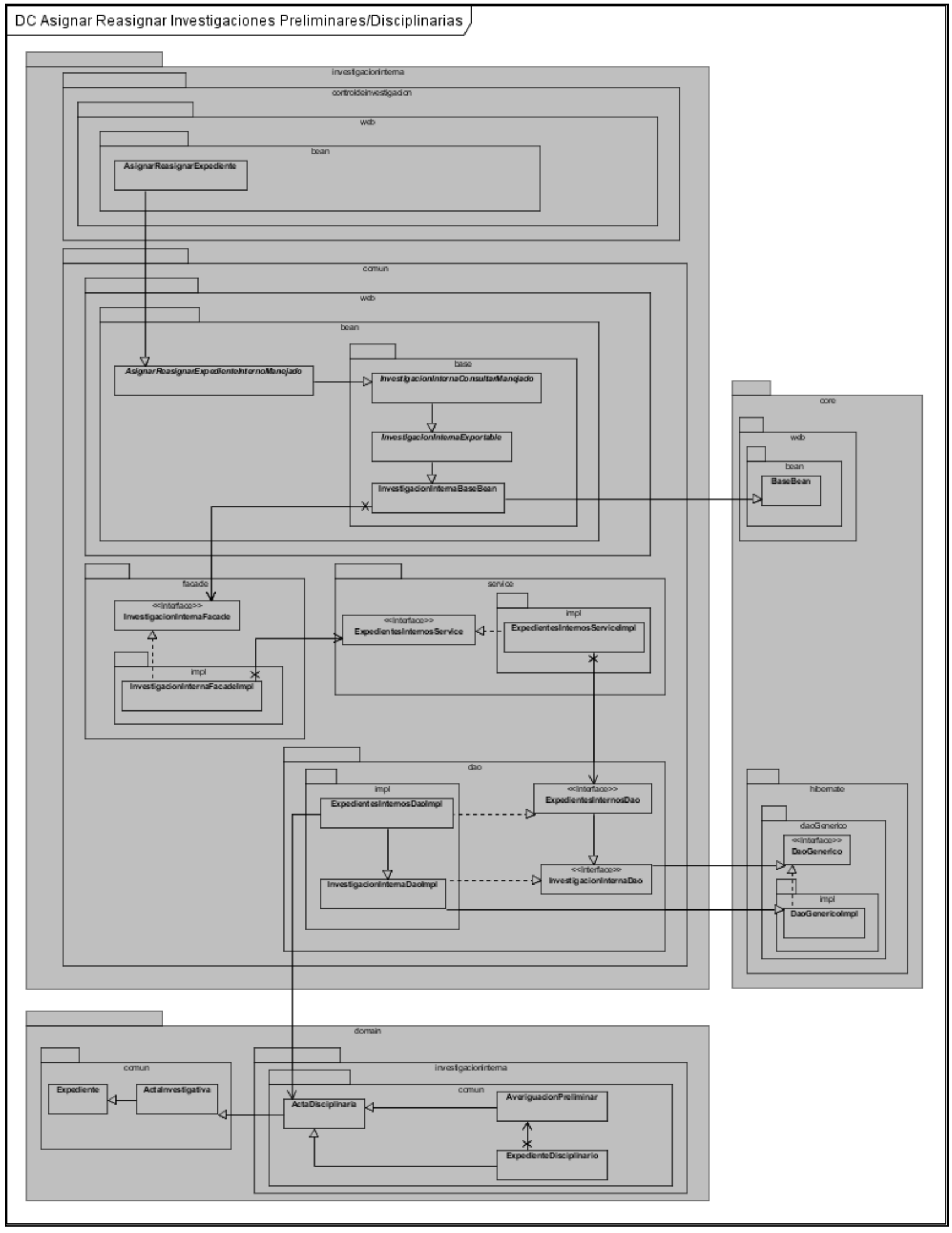


Diagrama de Clases del Diseño 5: Asignar Reasignar Investigaciones Preliminares/Disciplinarias.

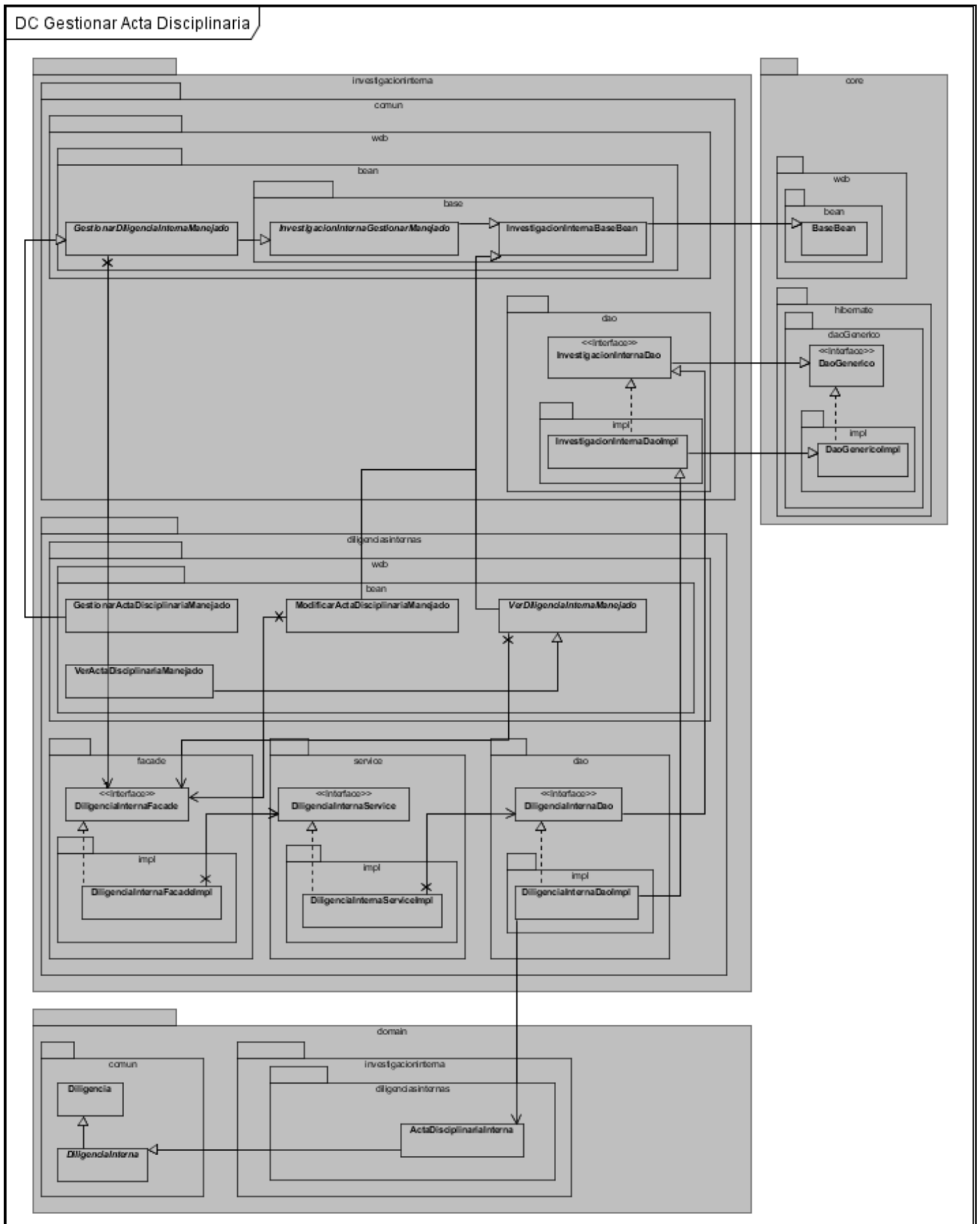


Diagrama de Clases del Diseño 6: Gestionar Acta Disciplinaria.

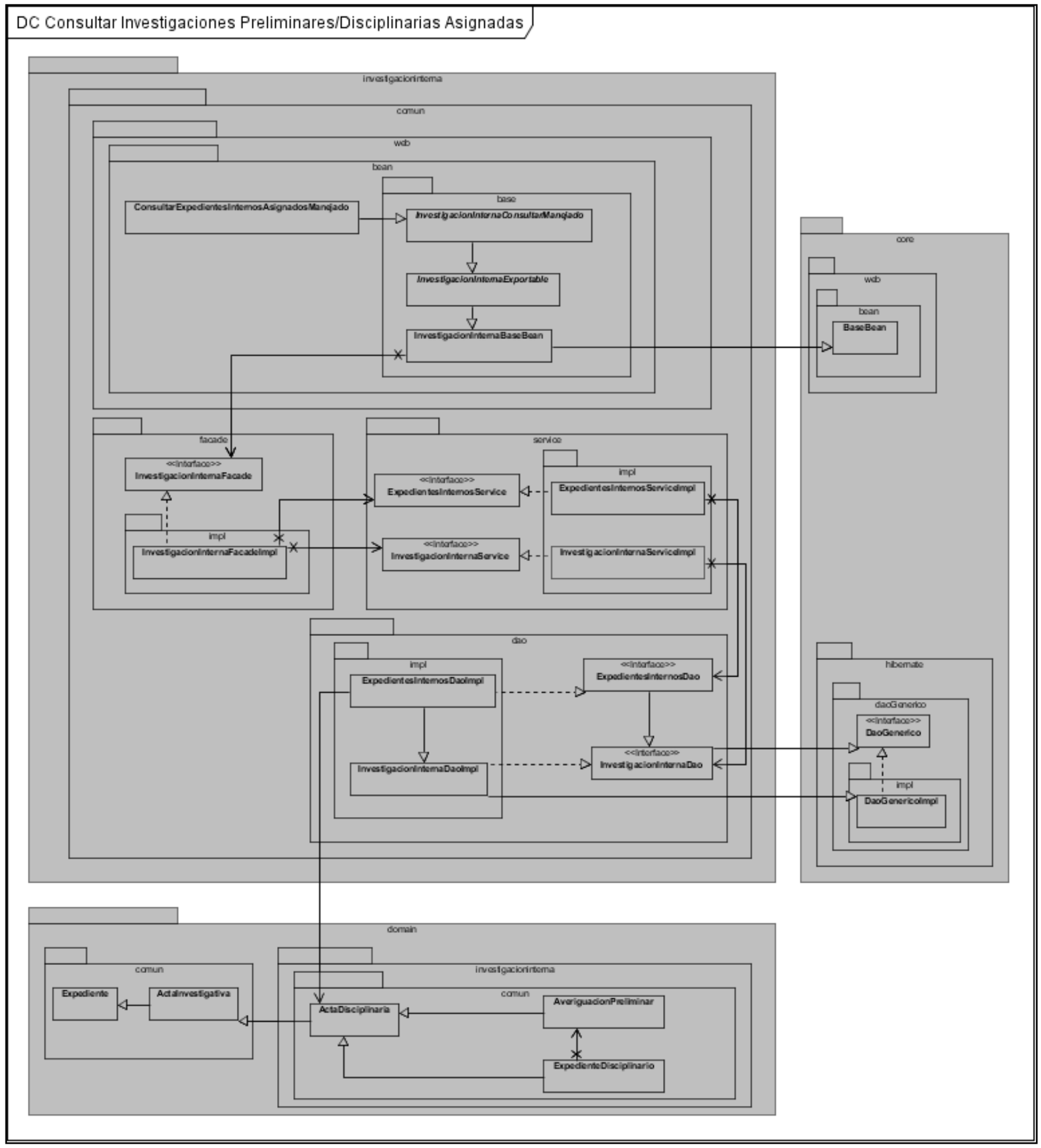


Diagrama de Clases del Diseño 7: Consultar Investigaciones Preliminares/Disciplinarias Asignadas.

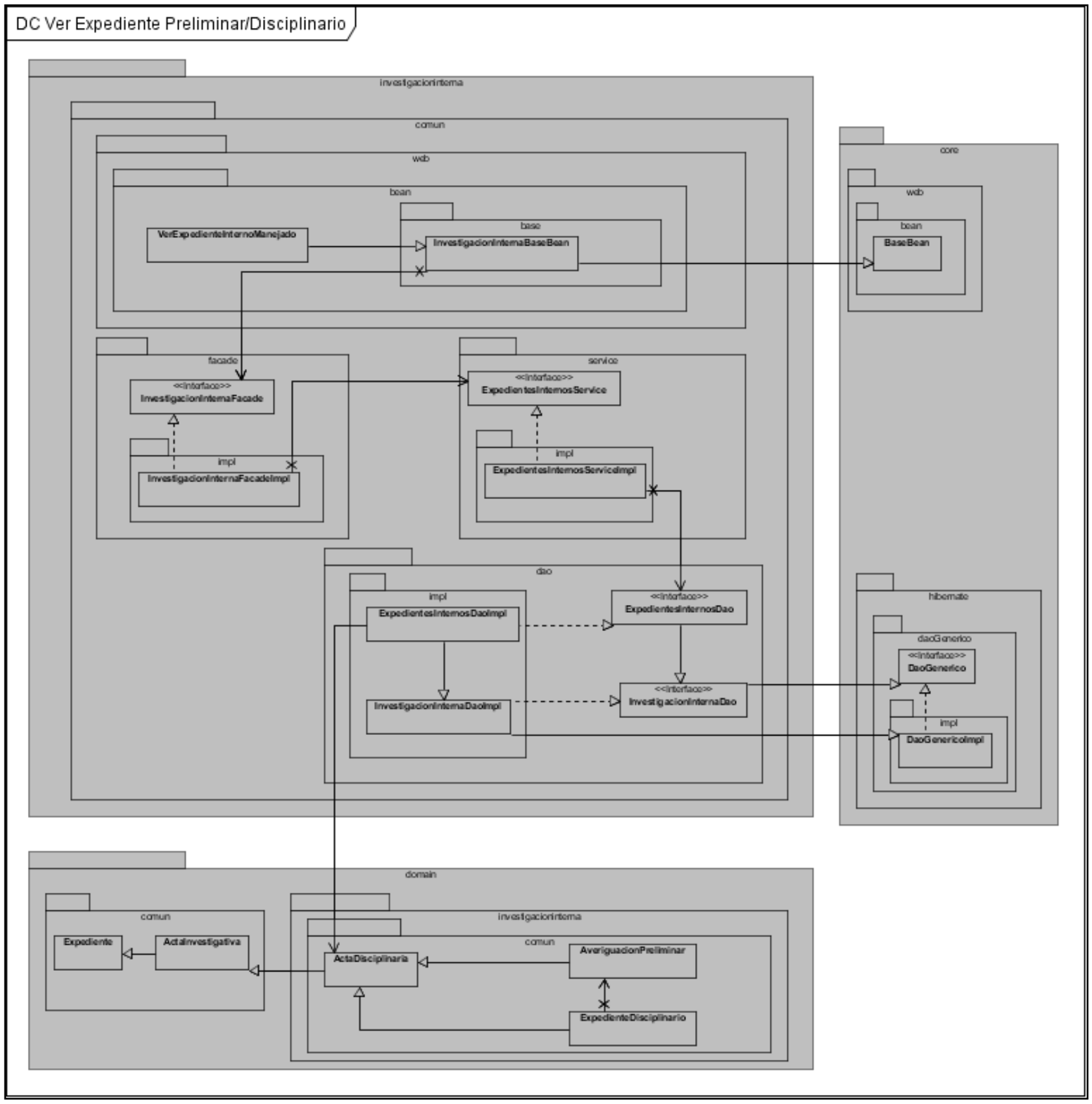


Diagrama de Clases del Diseño 8: Ver Expediente Preliminar/Disciplinario.

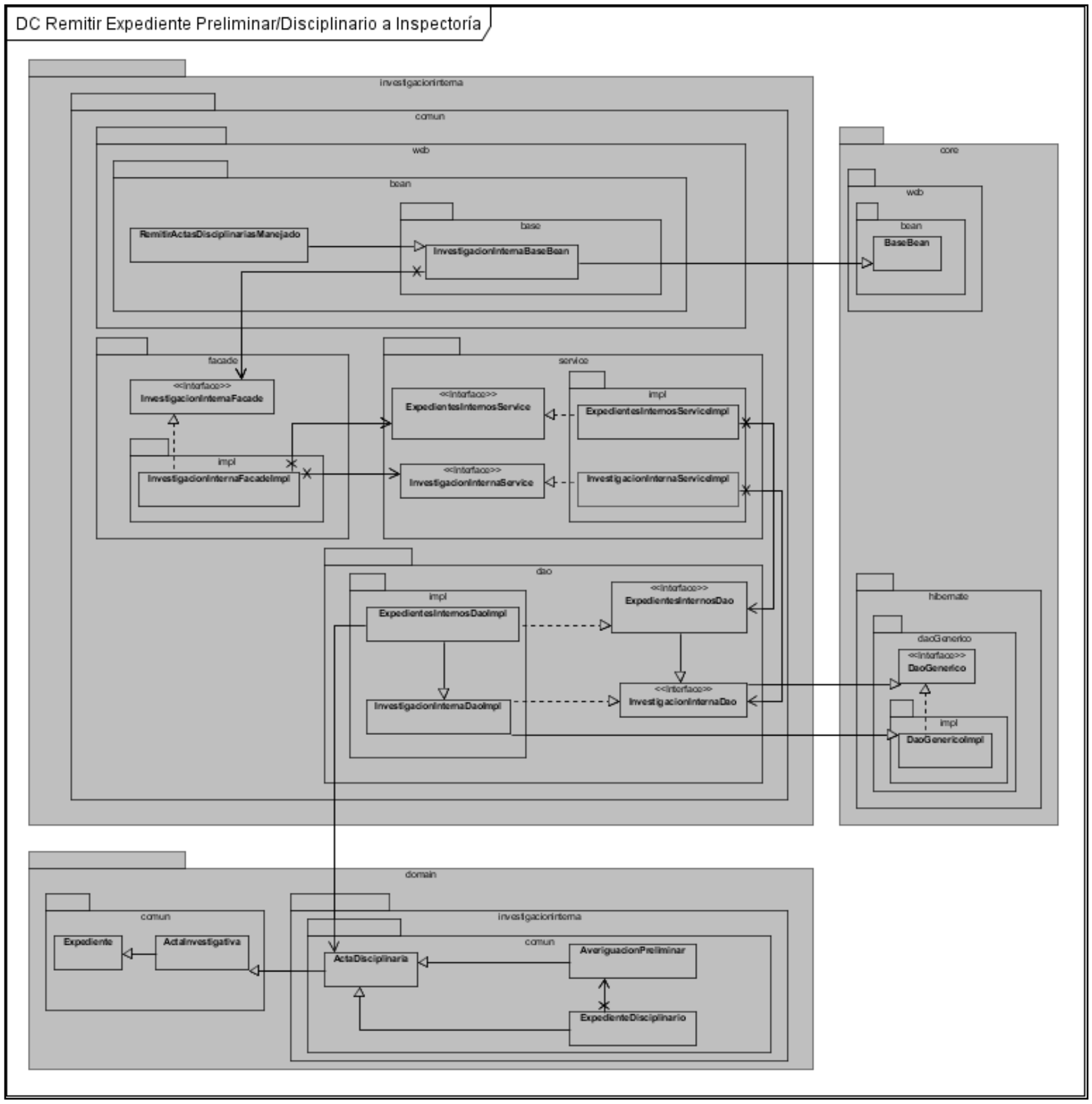


Diagrama de Clases del Diseño 9: Remitir Expediente Preliminar/Disciplinario a Inspectoría.

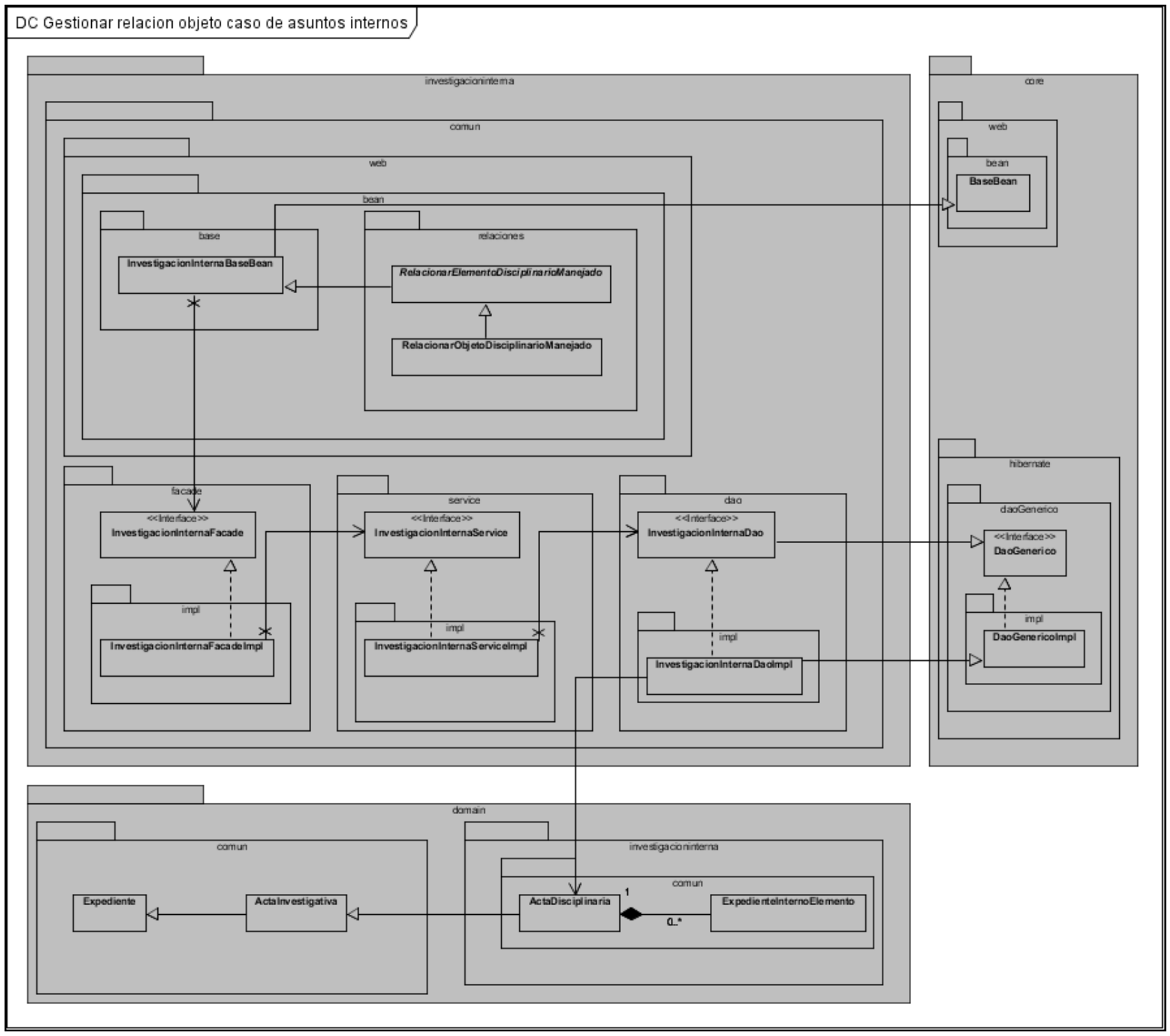


Diagrama de Clases del Diseño 10: Gestionar Relación Objeto Caso de Asuntos Internos.

Anexo 7: Realizaciones de Casos de Uso.

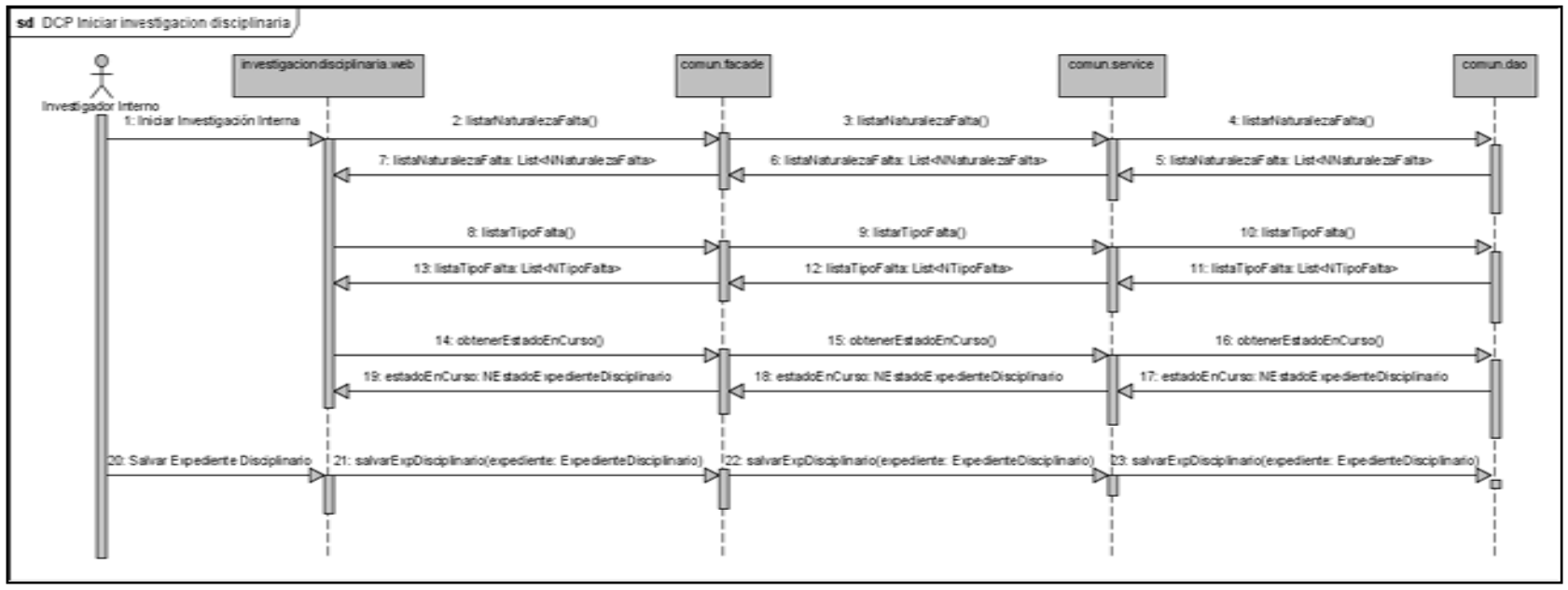


Diagrama de Contrato 1: Iniciar Investigación Disciplinaria.

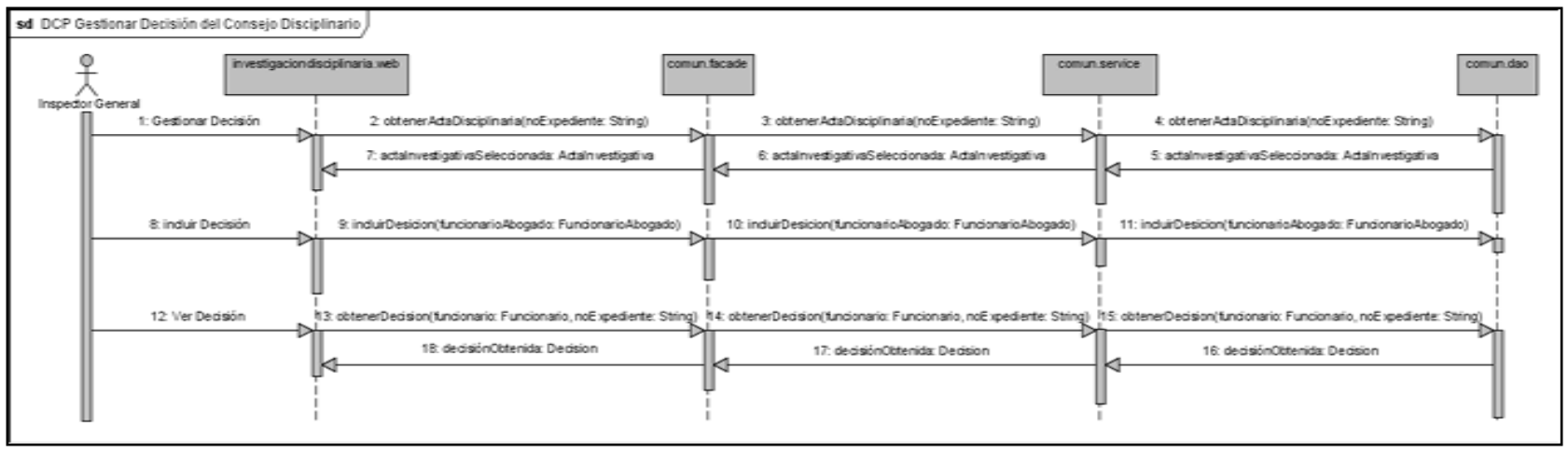


Diagrama de Contrato 2: Gestionar Decisión del Consejo Disciplinario.

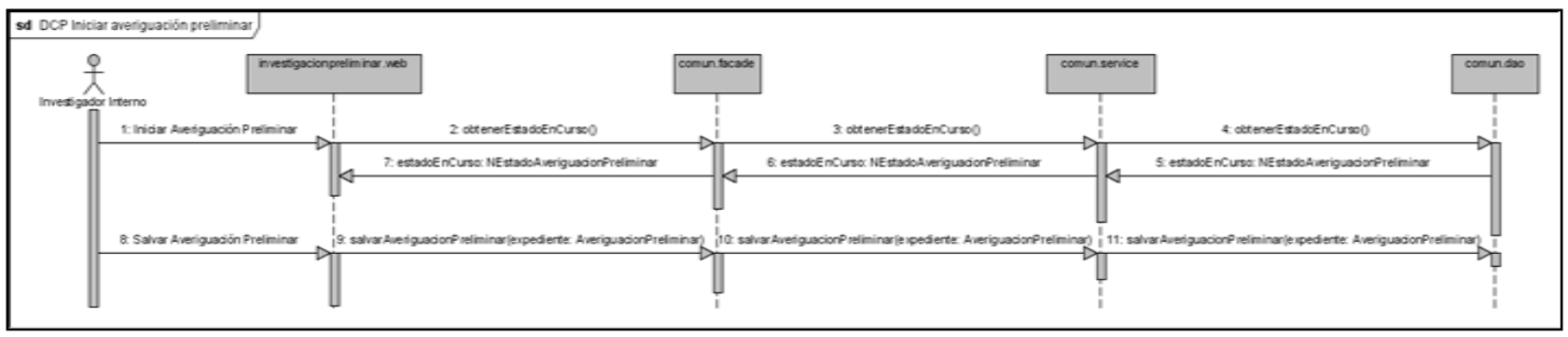


Diagrama de Contrato 3: Iniciar Averiguación Preliminar.

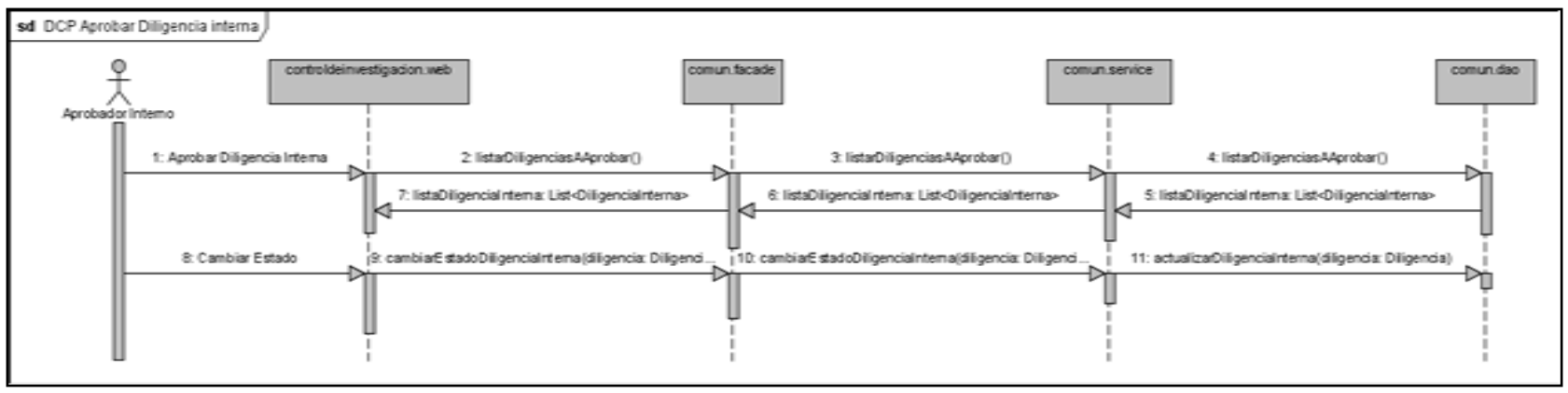


Diagrama de Contrato 4: Aprobar Diligencia Interna.

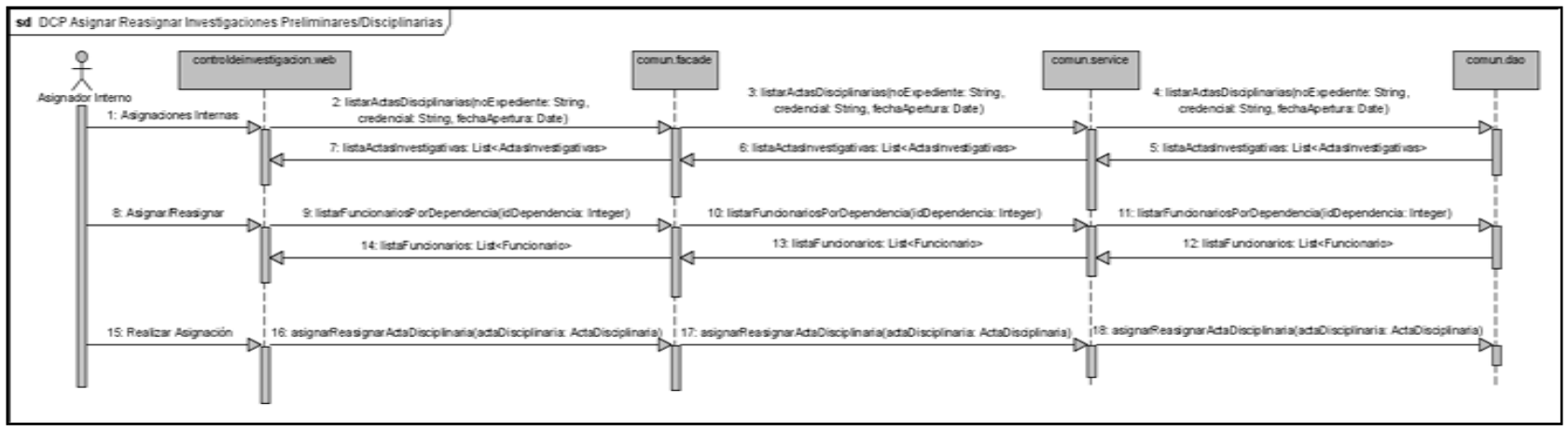


Diagrama de Contrato 5: Asignar Reasignar Investigaciones Preliminares/Disciplinarias.

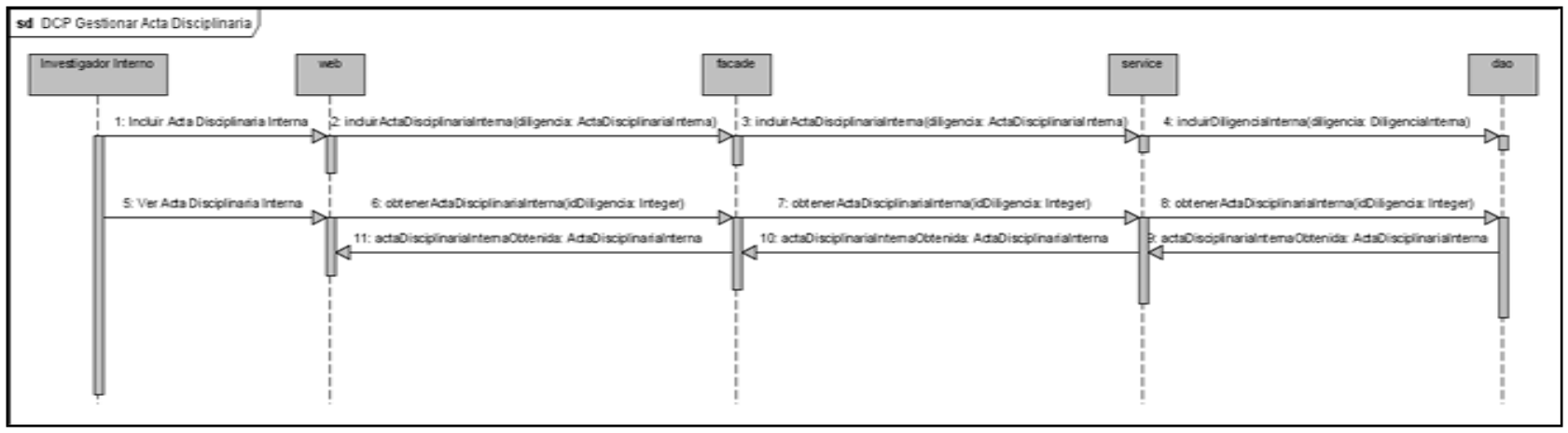


Diagrama de Contrato 6. Gestionar Acta Disciplinaria.

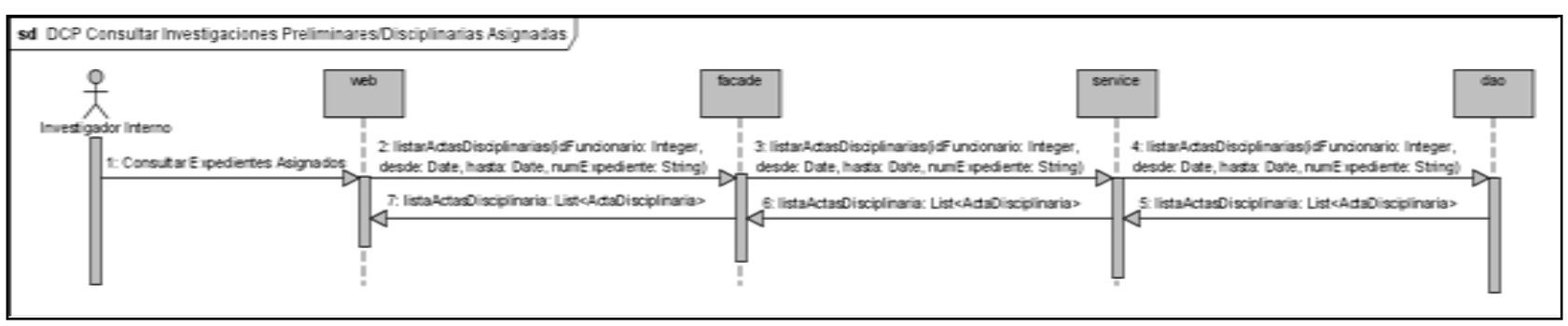


Diagrama de Contrato 7: Consultar Investigaciones Preliminares/Disciplinarias Asignadas.

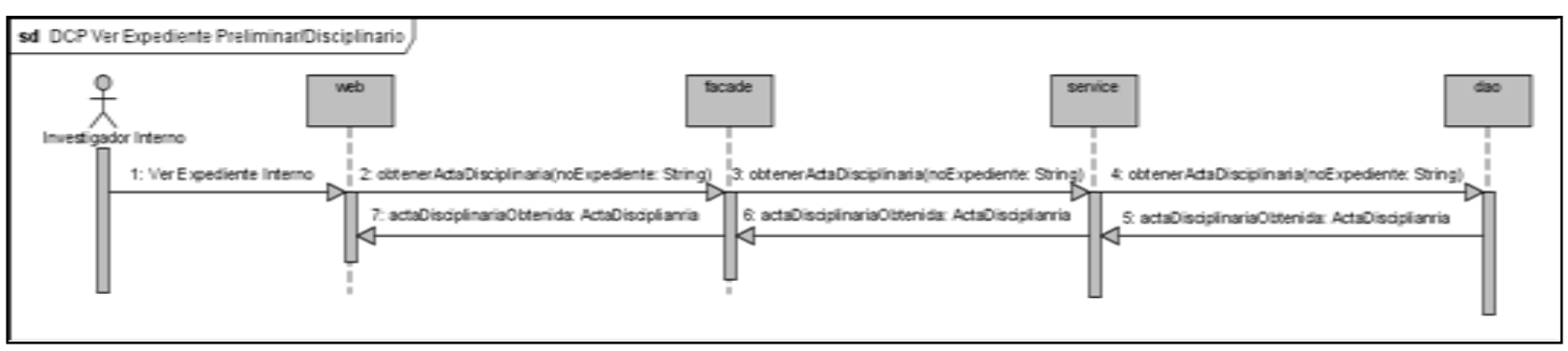


Diagrama de Contrato 8: Ver Expediente Preliminar/Disciplinario.

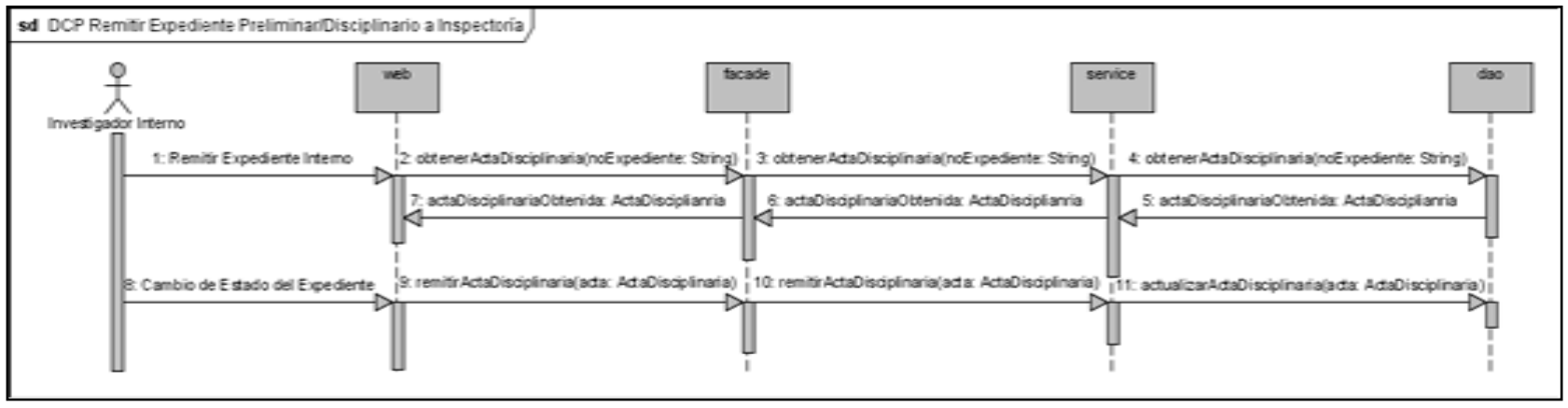


Diagrama de Contrato 9: Remitir Expediente Preliminar/Disciplinario a Inspectoría.

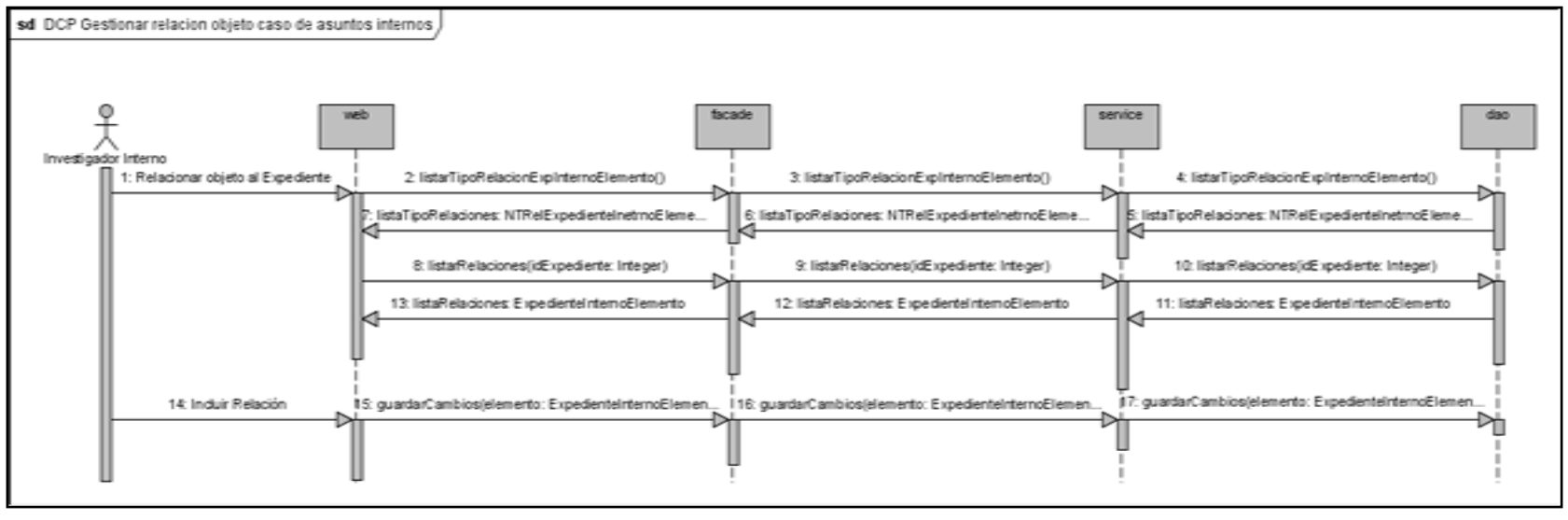


Diagrama de Contrato 10: Gestionar Relación Objeto Caso de Asuntos Internos.

Anexo 8: Modelo de Datos.

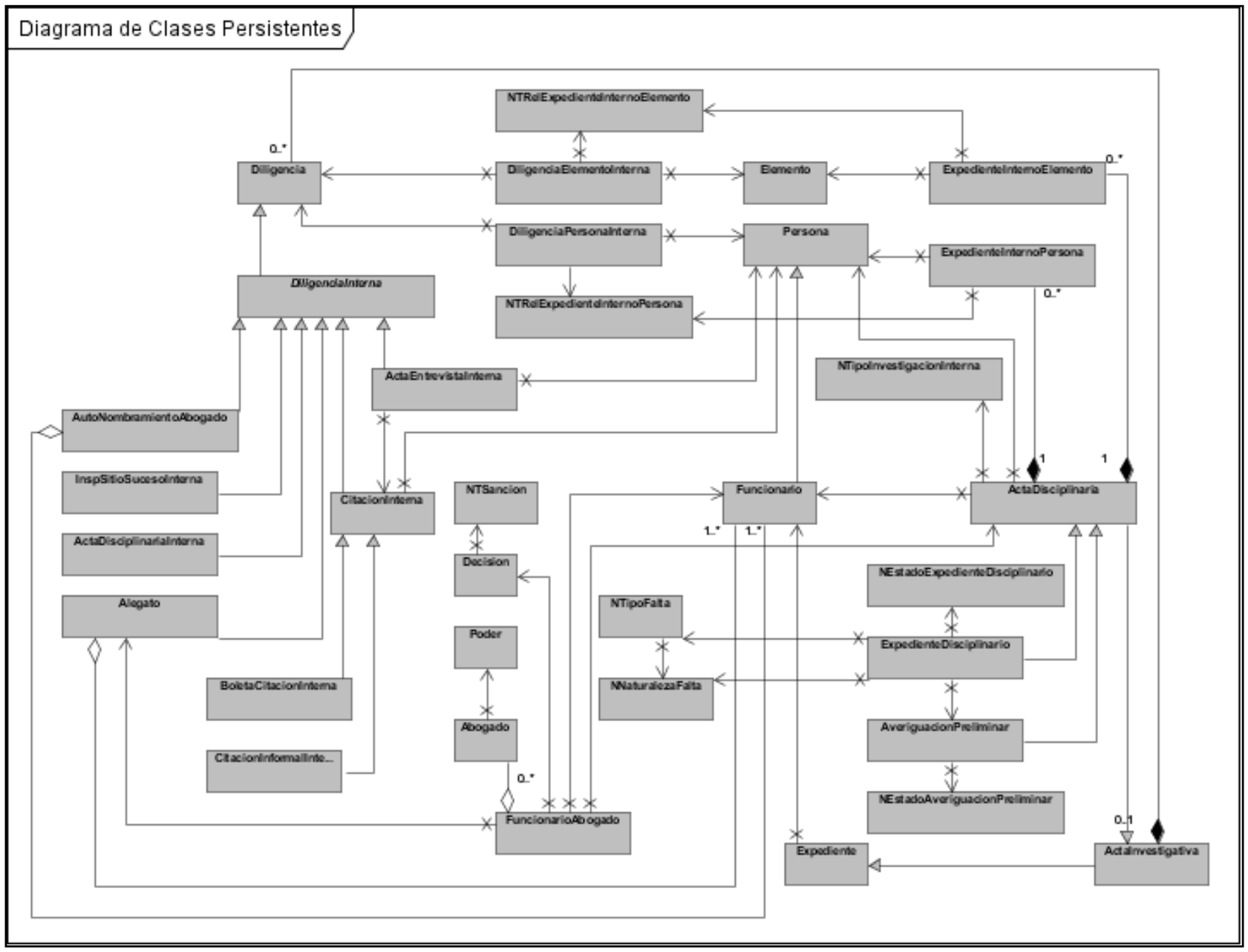


Diagrama de Clases Persistentes 1

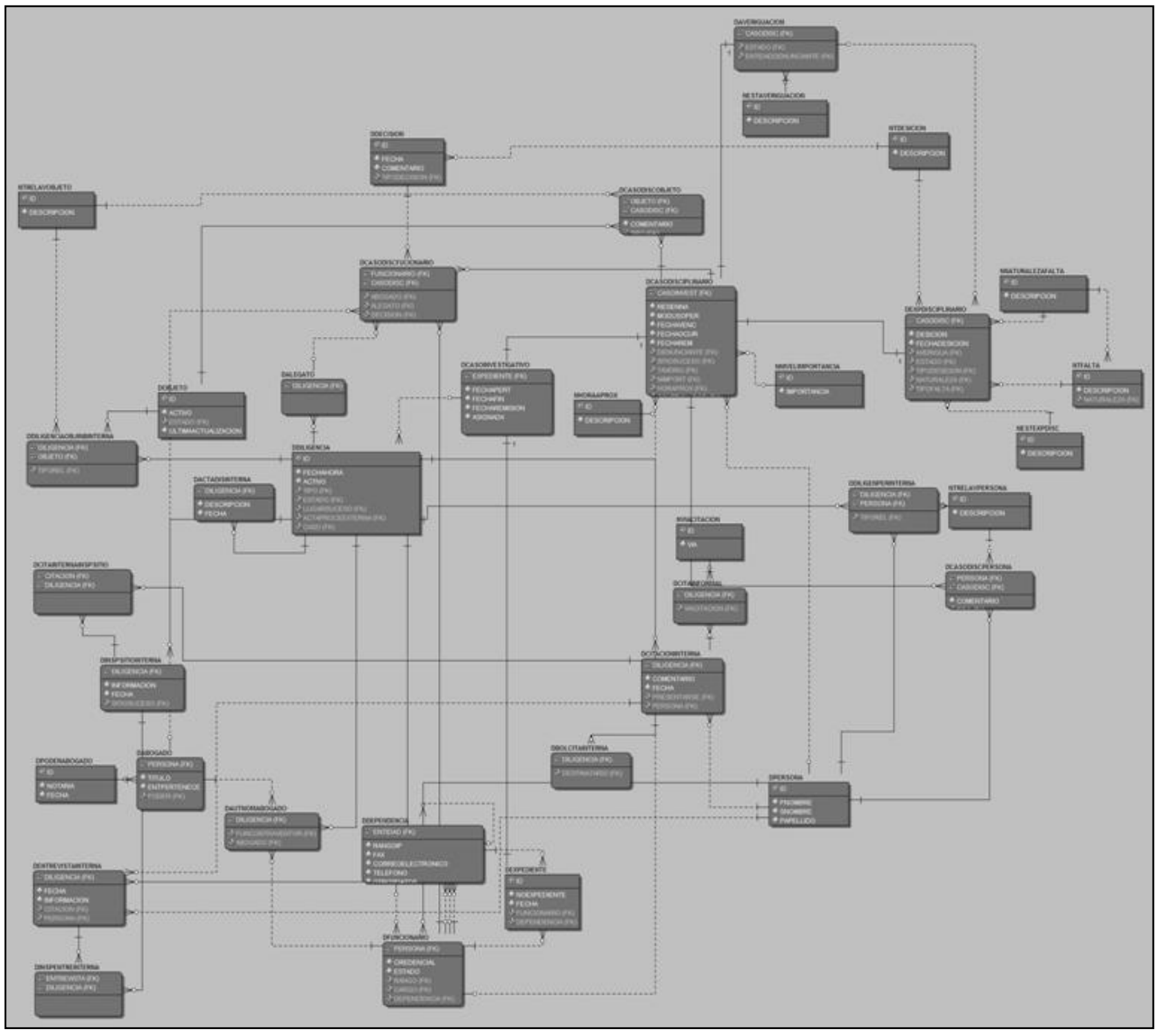


Diagrama Entidad Relación 1

Anexo 9: Diagramas de Implementación.

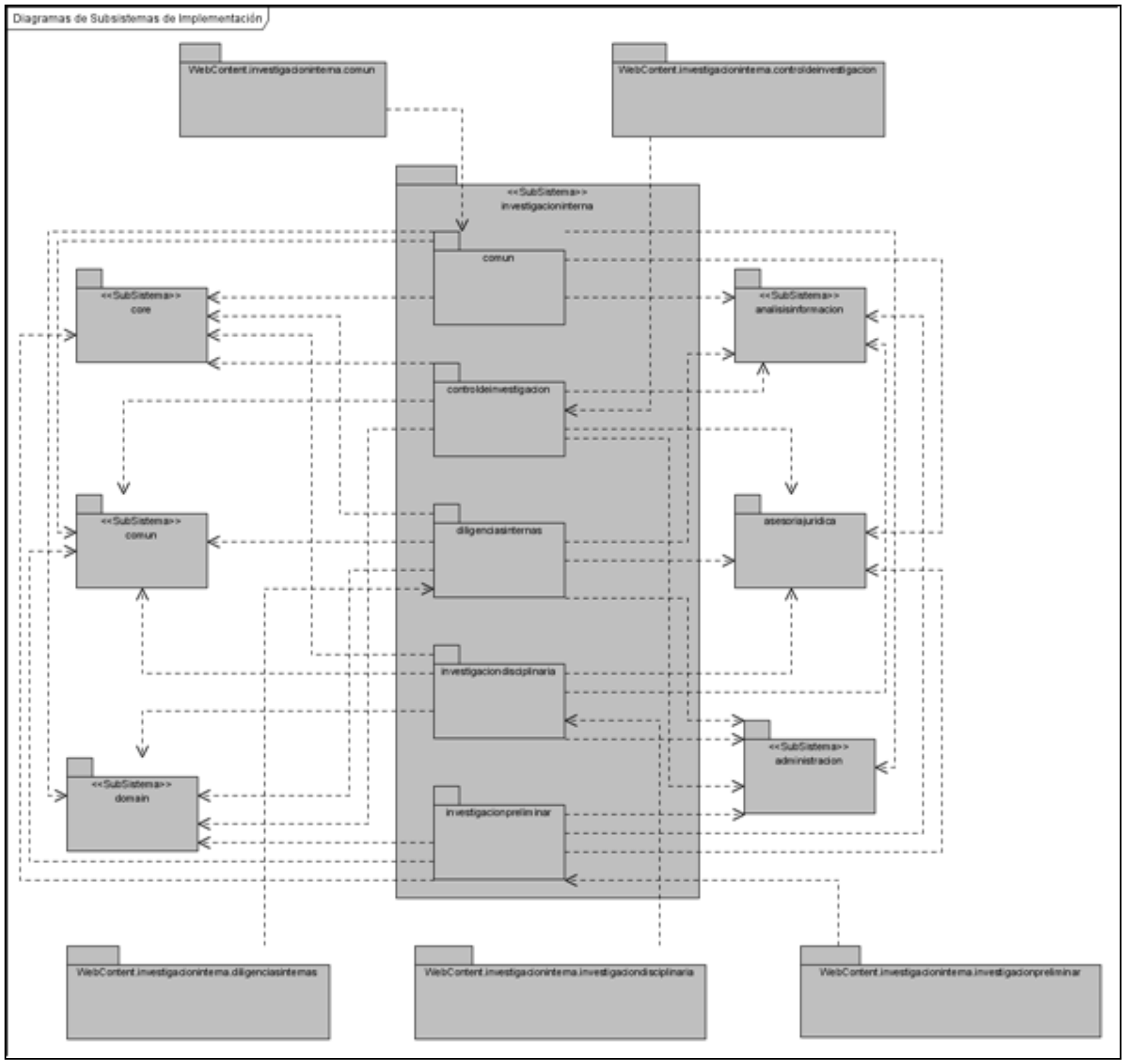


Diagrama de Subsistema de Implementación 1

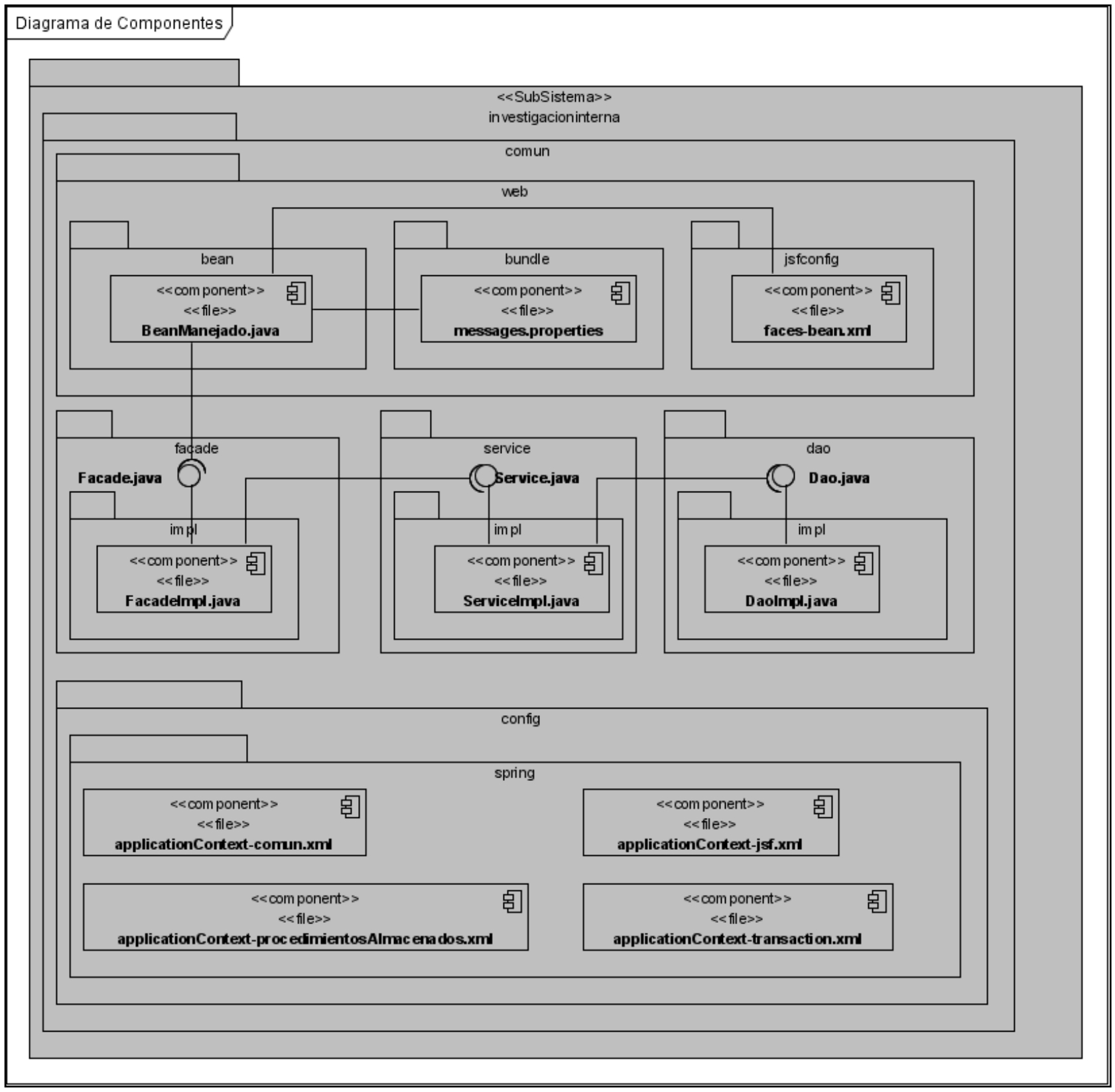


Diagrama de Componentes 1

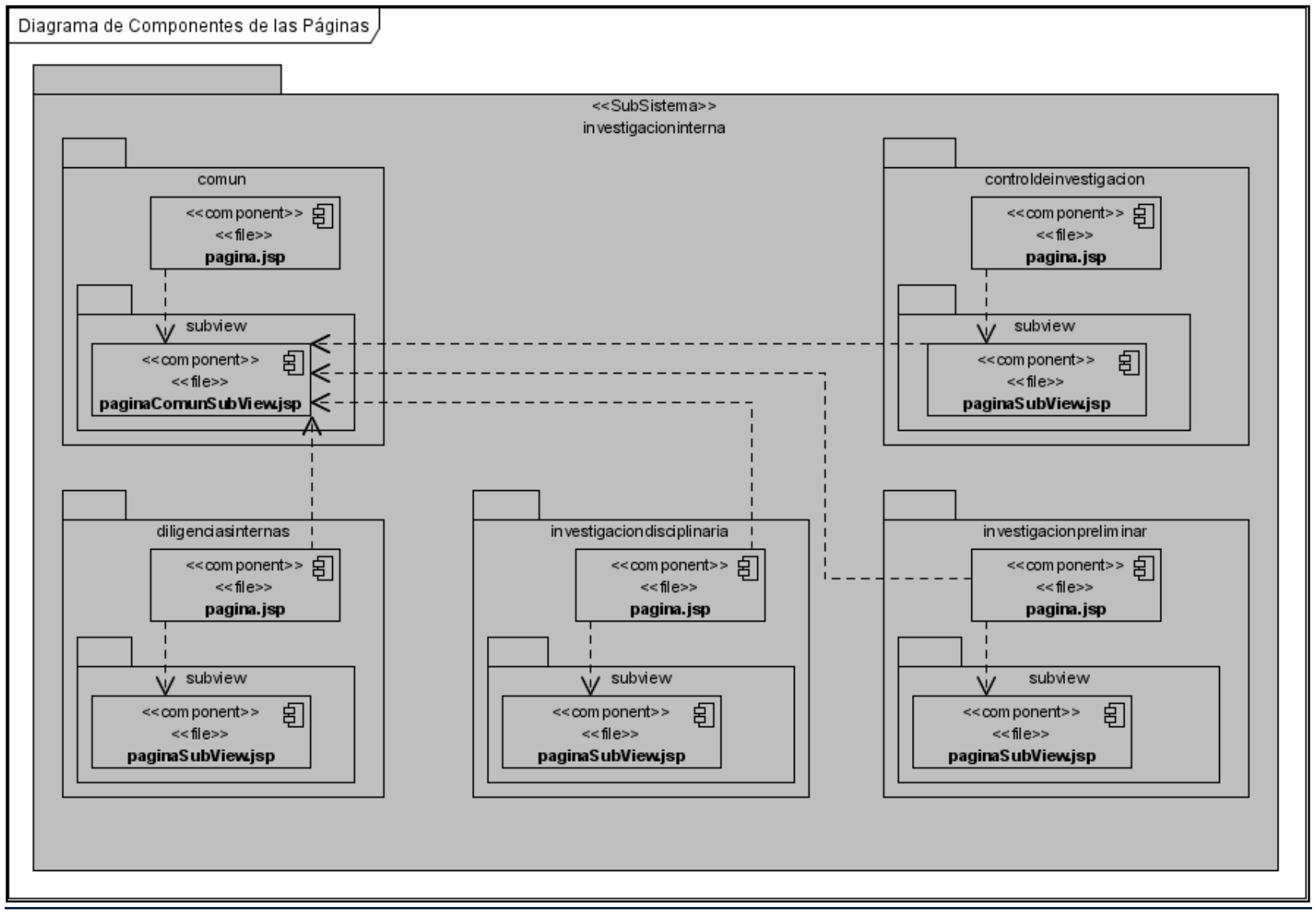


Diagrama de Componentes 2

Anexo 10: Relación de Pruebas y No Conformidades.

Prueba	Solución	Caso de Uso
Pruebas Internas	Corregido	CU Aprobar Diligencias Internas.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias Controladas.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias Controladas.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias Controladas.
Pruebas Internas	Corregido	CU Asignar/Reasignar Investigaciones Disciplinarias Controladas.
Pruebas Internas	Corregido	CU Consultar Diligencias Internas.
Pruebas Internas	Corregido	CU Consultar Diligencias Internas.
Pruebas Internas	Corregido	CU Consultar Expedientes Controlados Asignados.
Pruebas Internas	Corregido	CU Consultar Expedientes Controlados Asignados.
Pruebas Internas	Corregido	CU Consultar Fotos de Funcionarios.
Pruebas Internas	Corregido	CU Consultar Fotos de Funcionarios.
Pruebas Internas	Corregido	CU Consultar Fotos de Funcionarios.
Pruebas Internas	Corregido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Disciplinarias Controladas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas Internas	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas Internas	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas Internas	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas Internas	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas Internas	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas Internas	Corregido	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas Internas	Corregido	CU Gestionar Acta Disciplinaria.
Pruebas Internas	Corregido	CU Gestionar Acta Disciplinaria.
Pruebas Internas	Corregido	CU Gestionar Acta Disciplinaria.
Pruebas Internas	Corregido	CU Gestionar Acta Disciplinaria.
Pruebas Internas	Corregido	CU Gestionar Alegato.
Pruebas Internas	Corregido	CU Gestionar Alegato.
Pruebas Internas	Corregido	CU Gestionar Auto de Nombramiento de Abogado.
Pruebas Internas	Corregido	CU Gestionar Auto de Nombramiento de Abogado.
Pruebas Internas	Corregido	CU Gestionar Auto de Nombramiento de Abogado.
Pruebas Internas	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas Internas	Corregido	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas Internas	Corregido	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas Internas	Corregido	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas Internas	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas Internas	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas Internas	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas Internas	Corregido	CU Iniciar Investigación Disciplinaria.

Pruebas Internas	Corregido	CU Remitir Expediente Disciplinario a Inspectoría.
Pruebas Internas	Corregido	CU Remitir Expediente Disciplinario a Inspectoría.
Pruebas Internas	Corregido	CU Remitir Expediente Disciplinario a Inspectoría.
Pruebas Internas	Corregido	CU Remitir Expediente Disciplinario a Inspectoría.
Pruebas Internas	Corregido	CU Remitir Expediente Disciplinario a Inspectoría.
Pruebas Internas	Corregido	CU Remitir Expediente Disciplinario a Inspectoría.
Pruebas Internas	Corregido	CU Remitir Expediente Disciplinario a Inspectoría.
Pruebas Internas	Corregido	CU Revisar Diligencias Internas.
Pruebas Internas	Corregido	CU Ver Expediente Disciplinario.
Pruebas Internas	Corregido	CU Ver Expediente Disciplinario.
Pruebas Internas	Funciona correctamente	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Funciona correctamente	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas Internas	Funciona correctamente	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas Internas	Funciona correctamente	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas Internas	Funciona correctamente	CU Iniciar Investigación Disciplinaria.
Pruebas Internas	No será corregido	CU Iniciar Averiguación Preliminar.
Pruebas Internas	No será corregido	CU Iniciar Investigación Disciplinaria.
Pruebas Internas	Inválido	CU Aprobar Diligencias Internas.
Pruebas Internas	Inválido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas Internas	Inválido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Inválido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Internas	Inválido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Inválido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Internas	Inválido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas Internas	Inválido	CU Gestionar Acta de Entrevista Interna.
Pruebas Internas	Inválido	CU Gestionar Acta de Entrevista Interna.
Pruebas Internas	Inválido	CU Gestionar Acta de Entrevista Interna.
Pruebas Internas	Inválido	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas Internas	Inválido	CU Gestionar Acta Disciplinaria.
Pruebas Internas	Inválido	CU Gestionar Acta Disciplinaria.
Pruebas Internas	Inválido	CU Gestionar Acta Disciplinaria.
Pruebas Internas	Inválido	CU Gestionar Acta Disciplinaria.
Pruebas Internas	Inválido	CU Gestionar Auto de Nombramiento de Abogado.
Pruebas Internas	Inválido	CU Gestionar Auto de Nombramiento de Abogado.
Pruebas Internas	Inválido	CU Iniciar Averiguación Preliminar.
Pruebas Internas	Inválido	CU Revisar Diligencias Internas.
Pruebas Internas	Inválido	CU Revisar Diligencias Internas.
Pruebas Internas	Duplicado	CU Consultar Diligencias Internas.
Pruebas Internas	Duplicado	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas Internas	Duplicado	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas Internas	Duplicado	CU Iniciar Averiguación Preliminar.
Pruebas Internas	Duplicado	CU Iniciar Averiguación Preliminar.
Pruebas Cruzadas	Corregido	CU Asignar/Reasignar Investigaciones Preliminares.
Pruebas Cruzadas	Corregido	CU Consultar Expedientes Preliminares Controlados.
Pruebas Cruzadas	Corregido	CU Consultar Expedientes Preliminares Controlados.
Pruebas Cruzadas	Corregido	CU Consultar Fotos Funcionarios.
Pruebas Cruzadas	Corregido	CU Consultar Fotos Funcionarios.
Pruebas Cruzadas	Corregido	CU Consultar Fotos Funcionarios.
Pruebas Cruzadas	Corregido	CU Consultar Fotos Funcionarios.
Pruebas Cruzadas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Cruzadas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Cruzadas	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Cruzadas	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas Cruzadas	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas Cruzadas	Corregido	CU Gestionar Auto de Nombramiento de Abogado.
Pruebas Cruzadas	Corregido	CU Gestionar Auto de Nombramiento de Abogado.
Pruebas Cruzadas	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas Cruzadas	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas Cruzadas	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas Cruzadas	Corregido	CU Gestionar Decisión del Consejo Disciplinario.
Pruebas Cruzadas	Corregido	CU Gestionar Relación de Objeto al Caso de Asuntos Internos.
Pruebas Cruzadas	Corregido	CU Gestionar Relación de Objeto al Caso de Asuntos Internos.
Pruebas Cruzadas	Corregido	CU Gestionar Relación de Persona al Caso de Asuntos Internos.

Pruebas Cruzadas	Corregido	CU Gestionar Relación de Vehículo al Caso de Asuntos Internos.
Pruebas Cruzadas	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas Cruzadas	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas Cruzadas	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas Cruzadas	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas Cruzadas	Corregido	CU Remitir Expediente Disciplinario a Inspectoría.
Pruebas Cruzadas	Corregido	CU Ver Expediente Disciplinario.
Pruebas Cruzadas	Funciona correctamente	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas Cruzadas	Funciona correctamente	CU Gestionar Boleta de Citación Interna.
Pruebas Cruzadas	Funciona correctamente	CU Gestionar Relación de Objeto al Caso de Asuntos Internos.
Pruebas Cruzadas	Funciona correctamente	CU Iniciar Averiguación Preliminar.
Pruebas Cruzadas	Funciona correctamente	CU Ver Expediente Disciplinario.
Pruebas Cruzadas	No será corregido	CU Consultar Fotos Funcionarios.
Pruebas Cruzadas	Inválido	CU Consultar Expedientes Preliminares Controlados.
Pruebas Cruzadas	Inválido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Cruzadas	Inválido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas Cruzadas	Inválido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas Cruzadas	Inválido	CU Gestionar Relación de Persona al Caso de Asuntos Internos.
Pruebas Cruzadas	Inválido	CU Iniciar Averiguación Preliminar.
Pruebas Cruzadas	Inválido	CU Ver Expediente Disciplinario.
Pruebas Cruzadas	Duplicado	CU Gestionar Relación de Persona al Caso de Asuntos Internos.
Pruebas de Liberación I	Corregido	CU Consultar Comunicaciones Internas.
Pruebas de Liberación I	Corregido	CU Consultar Diligencias Internas.
Pruebas de Liberación I	Corregido	CU Consultar Diligencias Internas.
Pruebas de Liberación I	Corregido	CU Consultar Fotos de Funcionarios.
Pruebas de Liberación I	Corregido	CU Consultar Fotos de Funcionarios.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Disciplinarias Controladas.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Disciplinarias Controladas.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Disciplinarias Controladas.
Pruebas de Liberación I	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas de Liberación I	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas de Liberación I	Corregido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas de Liberación I	Corregido	CU Gestionar Acta Disciplinaria.
Pruebas de Liberación I	Corregido	CU Gestionar Acta Disciplinaria.
Pruebas de Liberación I	Corregido	CU Gestionar Alegato.
Pruebas de Liberación I	Corregido	CU Gestionar Alegato.
Pruebas de Liberación I	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Corregido	CU Gestionar Decisión del Consejo Disciplinario.
Pruebas de Liberación I	Corregido	CU Gestionar Decisión del Consejo Disciplinario.
Pruebas de Liberación I	Corregido	CU Gestionar Relación de Arma al Caso de Asuntos Internos.

Pruebas de Liberación I	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas de Liberación I	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas de Liberación I	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas de Liberación I	Corregido	CU Iniciar Averiguación Preliminar.
Pruebas de Liberación I	Corregido	CU Ver Expediente Disciplinario.
Pruebas de Liberación I	Corregido	CU Ver Expediente Disciplinario.
Pruebas de Liberación I	Corregido	CU Ver Expediente Disciplinario.
Pruebas de Liberación I	Corregido	Todos los CU Consultar.
Pruebas de Liberación I	Funciona correctamente	CU Gestionar Auto de Nombramiento de Abogado.
Pruebas de Liberación I	Funciona correctamente	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Funciona correctamente	CU Revisar Diligencias Internas.
Pruebas de Liberación I	Funciona correctamente	CU Ver Expediente Disciplinario.
Pruebas de Liberación I	No será corregido	CU Gestionar Acta Entrevista Interna.
Pruebas de Liberación I	Inválido	CU Consultar Expedientes Preliminares Controlados.
Pruebas de Liberación I	Inválido	CU Consultar Expedientes Preliminares Controlados.
Pruebas de Liberación I	Inválido	CU Consultar Fotos de Funcionarios.
Pruebas de Liberación I	Inválido	CU Consultar Información de Archivo Expedientes Internos.
Pruebas de Liberación I	Inválido	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas de Liberación I	Inválido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas de Liberación I	Inválido	CU Consultar Registro Histórico de Expedientes Internos.
Pruebas de Liberación I	Inválido	CU Gestionar Acta de Entrevista Interna.
Pruebas de Liberación I	Inválido	CU Gestionar Acta Disciplinaria.
Pruebas de Liberación I	Inválido	CU Gestionar Acta Disciplinaria.
Pruebas de Liberación I	Inválido	CU Gestionar Boleta de Citación Interna.
Pruebas de Liberación I	Inválido	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas de Liberación I	Inválido	CU Gestionar Relación de Objeto al Caso de Asuntos Internos.
Pruebas de Liberación I	Duplicado	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas de Liberación I	Duplicado	CU Consultar Investigaciones Disciplinarias Asignadas.
Pruebas de Liberación II	Corregido	CU Consultar Diligencia Interna.
Pruebas de Liberación II	Corregido	CU Consultar Fotos de Funcionarios.
Pruebas de Liberación II	Corregido	CU Consultar Información de Archivo Expedientes Internos.
Pruebas de Liberación II	Corregido	CU Consultar Investigaciones Preliminares Asignadas.
Pruebas de Liberación II	Corregido	CU Gestionar Acta de Entrevista Interna.
Pruebas de Liberación II	Corregido	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas de Liberación II	Corregido	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas de Liberación II	Corregido	CU Gestionar Acta Disciplinaria.
Pruebas de Liberación II	Corregido	CU Gestionar Acta Disciplinaria.
Pruebas de Liberación II	Corregido	CU Gestionar Participación en el Expediente Disciplinario.
Pruebas de Liberación II	Corregido	CU Gestionar Participación en el Expediente Disciplinario.
Pruebas de Liberación II	Corregido	CU Gestionar Participación en el Expediente Disciplinario.
Pruebas de Liberación II	Corregido	CU Gestionar Relación de Arma al Caso de Asuntos Internos.
Pruebas de Liberación II	Corregido	CU Gestionar Relación de Objeto al Caso de Asuntos Internos.
Pruebas de Liberación II	Corregido	CU Gestionar Relación de Persona al Caso de Asuntos Internos.
Pruebas de Liberación II	Corregido	CU Gestionar Relación de Persona al Caso de Asuntos Internos.
Pruebas de Liberación II	Corregido	CU Gestionar Relación de Vehículo al Caso de Asuntos Internos
Pruebas de Liberación II	Corregido	CU Gestionar Relación de Vehículo al Caso de Asuntos Internos
Pruebas de Liberación II	Corregido	CU Iniciar Investigación Disciplinaria.
Pruebas de Liberación II	Corregido	CU Iniciar Investigación Disciplinaria.
Pruebas de Liberación II	Corregido	CU Iniciar Investigación Disciplinaria.
Pruebas de Liberación II	Funciona correctamente	CU Consultar Antecedentes Disciplinarios de Funcionarios.
Pruebas de Liberación II	Funciona correctamente	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas de Liberación II	Funciona correctamente	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas de Liberación II	Funciona correctamente	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas de Liberación II	Funciona correctamente	CU Gestionar Acta de Inspección al Sitio del Suceso en Internas.
Pruebas de Liberación II	Inválido	CU Consultar Comunicaciones Internas.
Pruebas de Liberación II	Inválido	CU Gestionar Decisión del Consejo Disciplinario.
Pruebas de Liberación II	Inválido	CU Iniciar Averiguación Preliminar.
Otras	Corregido	CU Aprobar Diligencias Internas.
Otras	Corregido	CU Consultar Registro Histórico de Expedientes Internos.
Otras	Corregido	CU Gestionar Acta de Entrevista.
Otras	Corregido	CU Gestionar Relación de Objeto al Caso de Asuntos Internos.
Otras	Corregido	CU Gestionar Relación de Vehículo al Caso de Asuntos Internos.
Otras	Corregido	CU Gestionar Relación de Vehículo al Caso de Asuntos Internos.

Otras	Corregido	CU Iniciar Averiguación Preliminar.
Otras	Corregido	CU Iniciar Averiguación Preliminar.
Otras	Corregido	CU Iniciar Averiguación Preliminar.
Otras	Corregido	CU Iniciar Averiguación Preliminar.
Otras	Corregido	CU Revisar Diligencias Internas.
Otras	Corregido	CU Ver Expediente Disciplinario.
Otras	Corregido	CU Ver Expediente Disciplinario.
Otras	Inválido	CU Asignar/Reasignar Investigaciones Disciplinarias.
Otras	Inválido	CU Asignar/Reasignar Investigaciones Preliminares.
Otras	Inválido	CU Consultar Diligencias Internas.
Otras	Inválido	CU Consultar Fotos de Funcionarios.
Otras	Inválido	CU Consultar Investigaciones Disciplinarias Asignadas.
Otras	Inválido	CU Consultar Investigaciones Disciplinarias Controladas.
Otras	Inválido	CU Consultar Investigaciones Preliminares Asignadas.
Otras	Inválido	CU Consultar Investigaciones Preliminares Asignadas.
Otras	Inválido	CU Consultar Investigaciones Preliminares Asignadas.
Otras	Inválido	CU Gestionar Acta Disciplinaria.
Otras	Inválido	CU Gestionar Acta Disciplinaria.
Otras	Inválido	CU Gestionar Boleta de Citación Interna.
Otras	Inválido	CU Gestionar Notificación de Amenaza.
Otras	Inválido	CU Gestionar Notificación de Amenaza.
Otras	Inválido	CU Gestionar Notificación de Amenaza.
Otras	Inválido	CU Revisar Diligencias Internas.
Otras	Inválido	CU Ver Expediente Disciplinario.
Otras	Inválido	CU Ver Expediente Disciplinario.
Otras	Inválido	CU Ver Expediente Disciplinario.
Otras	Inválido	CU Ver Expediente Disciplinario.
Pruebas Piloto	Inválido	CU Consultar Comunicaciones Internas.
Pruebas Piloto	Inválido	CU Consultar Libro de Control de Investigaciones Internas.
Pruebas Piloto	Inválido	CU Gestionar Acta de Entrevista Interna.
Pruebas Piloto	Inválido	CU Gestionar Acta de Entrevista Interna.
Pruebas Piloto	Inválido	CU Gestionar Acta Disciplinaria.
Pruebas Piloto	Inválido	CU Iniciar Averiguación Preliminar.
Pruebas Piloto	Inválido	CU Iniciar Averiguación Preliminar.
Pruebas Piloto	Inválido	CU Iniciar Investigación Disciplinaria.
Pruebas Piloto	Inválido	CU Iniciar Investigación Disciplinaria.
Pruebas Piloto	Inválido	CU Remitir Expediente Disciplinario a Inspectoría.