Universidad de las Ciencias Informáticas

Facultad 8

Análisis, diseño e implementación del Portal WAP del SIIPOL Móvil

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

Autores

Maikel Bradshaw Venzant

Jorge Luis Rodríguez Martini

Tutor

Ing. Adonys Alea Boffill

Ciudad de La Habana, junio 2010

"Año 52 de la Revolución"

Declaración de autoría

Declaramos que somos los únicos autores del trabajo Análisis, diseño e implementación del Portal WAP del SIIPOL Móvil y autorizamos a la Facultad 8 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Autores:

_____

Maikel Bradshaw Venzant

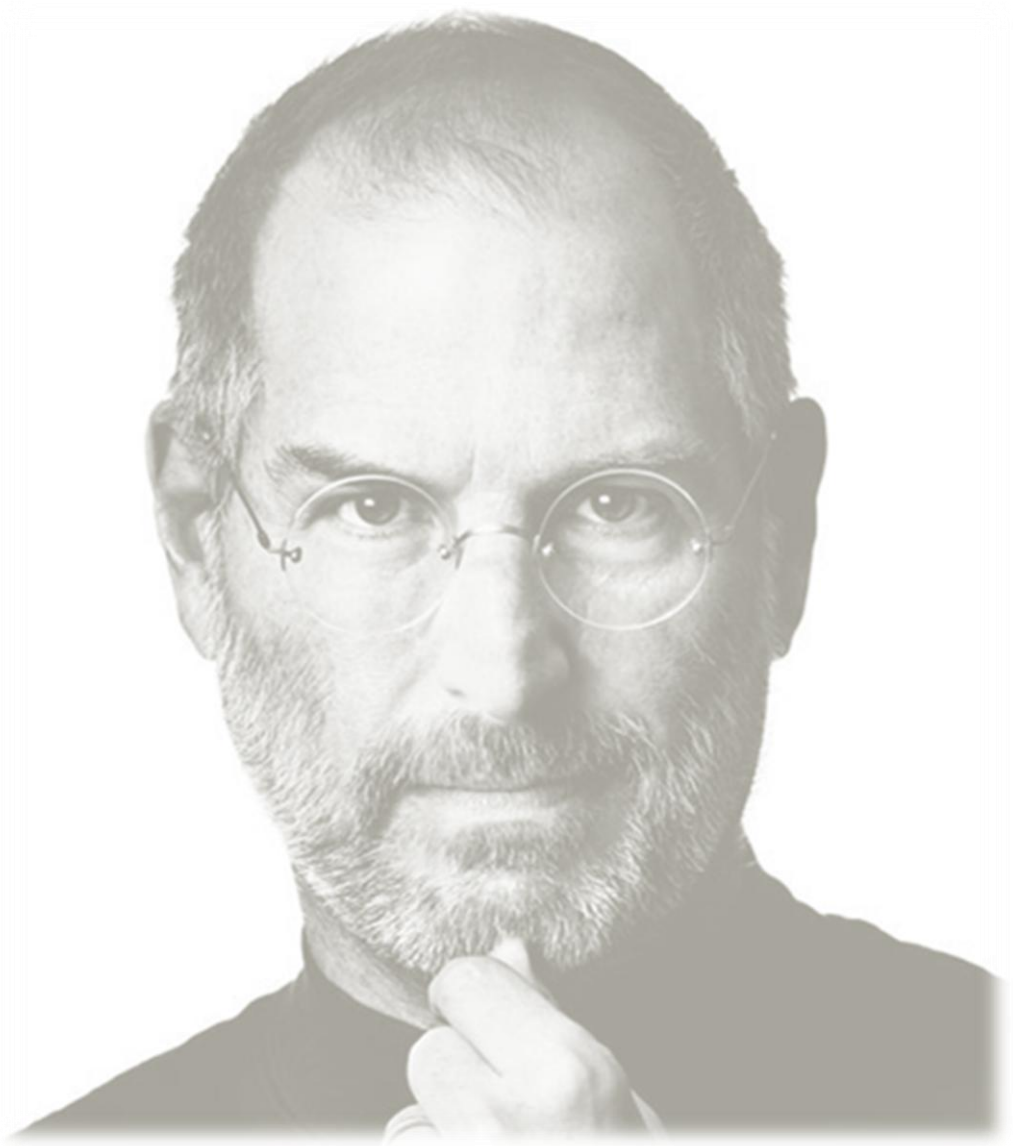
_____

Jorge Luis Rodríguez Martini

Tutor:

_____

Ing. Adonys Alea Boffill



“Tu tiempo es limitado, de modo que no lo malgastes viviendo la vida de alguien distinto. No quedes atrapado en el dogma, que es vivir como otros piensan que deberías vivir. No dejes que los ruidos de las opiniones de los demás acallen tu propia voz interior. Y lo que es más importante, ten el coraje para hacer lo que te dicen tu corazón y tu intuición. Ellos ya saben de algún modo en qué quieres convertirte realmente. Todo lo demás es secundario.”

Steve Jobs

Dedicatoria

Maikel:

A mis padres que me lo han dado todo, hoy les entrego el fruto de lo que crearon.

A mi champolito Jorge Luis, que ha servido de inspiración para que hoy me realice como ingeniero.

Jorge Luis:

A las personas más especiales que he tenido en toda mi vida, mis padres Mayra y Jorge. Ellos me han enseñado mucho de lo que sé, siempre han sabido guiarme y han confiado en mí.

A Yudeisy, el amor de mi vida, que siempre ha estado ahí para mí.

A ustedes va dedicado este trabajo que además de ser mi sueño, sé que es el de ustedes también.

Agradecimientos

Maikel:

A mis padres, por ser ambos un ejemplo a seguir y ser el motor impulsor que permitió que alcanzara hoy uno de mis sueños.

A mi abuela Fina, por consentirme siempre y darme todo lo que estuvo a su alcance.

A mis hermanos Jorge Luis y Elizabeth por formar parte de mi vida y brindarme siempre su apoyo.

A mi novia Idenia por entenderme y apoyarme siempre a pesar de la distancia que nos ha separado por tanto tiempo.

A nuestro tutor Adonys por haber contribuido a la realización de nosotros como ingenieros.

A nuestra oponente Dayana por soportarnos todo este tiempo.

A mis amigos del grupo 8207, que a pesar de la separación, siempre nos mantuvimos unidos.

A mis compañeros de aula por haber pasado 3 maravillosos cursos junto a ustedes.

A mis grandísimos amigos del Pikete, al Blue Team y a nuestras chicas por habernos enseñado muchas de las cosas que hoy sabemos.

Y en especial, agradecer a alguien que ha sabido ganarse un espacio inmenso en mi corazón, a esa auténtica amiga que eres tú **Olivia**, gracias por ser un apoyo incondicional todo este tiempo que hemos pasado juntos, jamás podré olvidarme de ti.

Jorge Luis:

A mi abuela Fela que siempre la tengo presente en todo lo que hago y la quiero con el alma. Eres lo más grande que tengo.

A mis padres por poder contar con ellos siempre y estar presentes para todo lo que me hiciera falta, por darme su apoyo incondicional y por convertirme en el hombre que soy hoy.

A mi novia Yudeisy, que me ha brindado su amor incondicional y que ha estado a mi lado en todo momento.

A Adonys, porque además de ser tutor, es un gran amigo y siempre pude contar con él. Mucho de lo que sé te lo debo a ti.

A Damián por toda la ayuda que me brindó. Sin su ayuda no se hubiesen logrados los resultados obtenidos.

A mis tíos y tías que han sabido aconsejarme y me han dado ánimos para salir adelante.

A toda mi familia, por ser tan unida y por querernos tanto, porque en cualquier momento se puede contar con ustedes para cualquier cosa.

A todos mis compañeros de grupo (tanto del grupo anterior, como el de ahora), ustedes han sido mi familia en estos cinco años.

Al grupo de Yuneikys y sus Kini Kini (Yune, Ivonne y Alexey) por brindarme su amistad y por permancer siempre unidos durante estos cinco años.

Al equipo de Penales: Shael, Arian, Yander, Adriel y Eduardo, aprendí mucho trabajando con ustedes.

Y en especial a **Yadira**, tú has sido un gran apoyo para mí en todos estos años, eres una persona muy especial, siempre estarás presente en mí.

Agradecimientos comunes

A Fidel y a la Revolución por darnos la oportunidad de estudiar en esta universidad y de crecernos como hombres libres y de ciencia.

A la UCI por formarnos como profesionales y haber permitido nuestra superación durante estos años.

A todos aquellos que de una forma u otra, han contribuido a nuestra formación tanto profesional como personal.

Resumen

En Venezuela no existe ningún Portal WAP utilizado con fines policiales, en los cuales la seguridad de los datos que se manejan es un aspecto importante a tener en cuenta, por lo que el uso de esta tecnología al servicio de los órganos policiales se pondrá en práctica por primera vez en este país.

Según un estudio de la situación actual del CICPC, hay un conjunto de procesos que se deben informatizar para lograr una mayor eficiencia en las respuestas operativas, lo cual dio surgimiento al desarrollo del Portal WAP, el cual permitirá a los funcionarios conectarse mediante dispositivos móviles a la base de datos del SIIPOL de dicha institución, con el objetivo de realizar consultas sobre entidades operativas (armas, vehículos, objetos y personas).

En el presente trabajo se describen las principales características de las herramientas utilizadas para llevar a cabo la construcción del Portal. De igual manera se fundamenta la selección de la metodología RUP, la cual sirvió de guía al proceso de desarrollo.

Con la realización exitosa del proyecto, se incrementan notablemente las áreas desde donde un funcionario puede acceder a información de interés policial, especialmente hasta lugares donde se dificulta el acceso a las redes de computadoras. Lo anterior:

- ✓ Mejora la calidad de las decisiones operativas, basadas en una mayor disponibilidad de la información policial.
- ✓ Se incrementa la capacidad operativa en el trabajo preventivo de los funcionarios.
- ✓ Se presta un servicio más profesional a la población.

Palabras clave: WAP, móvil, portal.

Abstract

In Venezuela there is no WAP application used for law enforcement, in which the security of data handled is an important aspect to consider, so the use of this technology for law enforcement agencies will be implemented for the first time in this country.

According to a study of the current situation of the CICPC, there is a set of processes to be computerized for greater efficiency in the operational responses, which gave birth to the development of WAP Portal, which will allow staff to connect via mobile devices to the SIIPOL database of that institution, in order to perform consultations about operational entities (weapons, vehicles, objects and people).

In this work are described the main features of the tools used to carry out the construction of the Portal. Similarly, the selection is based on the RUP methodology, which guided the development process.

With the successful realization of this project, significantly increasing the areas from which an official can access information from police interest, especially to places where it is difficult to access computer networks. Above:

- ✓ Improves the quality of operational decisions based on greater availability of police information.
- ✓ Increased operational capacity in preventive work of the staff.
- ✓ It provides a more professional service to the public.

Keywords: WAP, mobile, portal.

Índice

Introducción

La seguridad ciudadana se ha convertido en un parámetro ineludible en la medición de la calidad de vida de los habitantes, constituyendo una preocupación constante para los organismos internacionales, los gobiernos y los mandatarios locales, que ven como los ciudadanos exigen y necesitan vivir con tranquilidad, para no ser víctimas de la delincuencia.

Venezuela es considerado uno de los países más violentos de América, dando muestra de ello, el aumento considerable de las cifras de crímenes y homicidios en los últimos años.

Actualmente, Venezuela cuenta con varios cuerpos de investigación contra el crimen y la delincuencia, entre los que se encuentra el Cuerpo de Investigaciones Científicas Penales y Criminalísticas (CICPC), de ahí la necesidad de dicha institución de dar respuesta rápida y confiable a todo hecho cometido fuera de la ley, de manera que su compleja organización interna no dificulte el tratamiento de la información y el accionar de la prestigiosa entidad.

El CICPC tiene como misión, garantizar la eficiencia en la investigación del delito mediante su determinación científica, asegurando que el ejercicio de la acción penal conduzca a una sana administración de justicia.

Esta institución cuenta con un Sistema de Investigación e Información Policial (SIIPOL), el cual surge como producto de la modernización realizada al Sistema Integrado de Información Policial. Este nuevo sistema incrementa considerablemente el campo de acción y las prestaciones del antiguo.

Según un estudio de la situación general que tiene actualmente la institución, hay un conjunto de procesos que se deben informatizar para una mayor eficiencia en las respuestas operativas, dando lugar al surgimiento del proyecto de desarrollo SIIPOL Móvil, el cual proveerá al CICPC de un Portal WAP y de una aplicación web. El Portal WAP permitirá a los funcionarios conectarse a la base de datos del SIIPOL mediante dispositivos móviles, con el objetivo de realizar consultas sobre entidades operativas (armas, vehículos, objetos y personas), fundamentalmente cuando estos se encuentren fuera de la institución. Por su parte la aplicación web contará con una plataforma de mensajería para darles indicaciones y proveer de información necesaria a los funcionarios del CICPC.

A raíz de la problemática antes expuesta se plantea como **Problema a Resolver**: ¿Cómo garantizar que los funcionarios del CICPC realicen consultas sobre entidades operativas en lugares donde no tengan acceso al SIIPOL?

El **Objeto de Estudio** de la presente investigación está dirigido al proceso de desarrollo de software, orientado a los sistemas de gestión de información, enfocado al **Campo de Acción** del desarrollo del Portal WAP del SIIPOL Móvil del CICPC.

Por lo antes expuesto se plantea como **Idea a Defender**:

Si se realiza el análisis, diseño e implementación de un sistema que permita realizar consultas sobre entidades operativas desde dispositivos móviles, se garantizará el acceso a información desde lugares donde no se pueda acceder al SIIPOL, logrando una mayor eficiencia en las respuestas operativas del CICPC.

Se define como **Objetivo General**: desarrollar el Portal WAP del SIIPOL Móvil.

Como **Objetivos Específicos**:

1. Elaborar el diseño teórico de la investigación.
2. Realizar el análisis y diseño del Portal WAP del SIIPOL Móvil.
3. Implementar una aplicación que de respuesta a las funcionalidades del Portal WAP del SIIPOL Móvil.
4. Validar la propuesta de solución.

Para dar cumplimiento a estos objetivos, se han definido las siguientes **Tareas Investigativas**:

1. Análisis de aplicaciones o soluciones similares.
2. Estudio de las herramientas y metodologías de desarrollo de software seleccionadas.
3. Realización del Modelo de Análisis de los casos de uso.

4. Realización del diseño de clases derivado del Modelo de Análisis.
5. Refinamiento del diseño empleando patrones.
6. Implementación de la propuesta de solución.
7. Validación de la propuesta de solución.
8. Análisis de los resultados de las pruebas de calidad realizadas a la aplicación.

Los **Métodos Científicos** utilizados son:

Teórico:

- ✓ Analítico Sintético: está dado por el análisis de los documentos generados por los analistas para el SIIPOL Móvil: especificación de casos de usos, glosario de términos, diagramas de entidades, entre otros, extrayendo y analizando los principales elementos relacionados con el objeto de estudio.

Empírico:

- ✓ Entrevista: entrevista con el analista del SIIPOL Móvil con el objetivo de obtener a través de una conversación planificada, información cualitativa de los eventos y fenómenos que puedan ocurrir en el sistema.

Estructura Capitular:

Capítulo 1. Fundamentación Teórica.

Se estudian los elementos teóricos de la investigación y las herramientas seleccionadas para llevar a cabo el desarrollo de la investigación, concluyendo con una propuesta de solución que cumpla con los requisitos funcionales y no funcionales establecidos para el Portal WAP.

Capítulo 2. Análisis, diseño e implementación de la propuesta de solución.

Se describen los principales artefactos generados durante el proceso de análisis, diseño e implementación del Portal WAP del SIIPOL Móvil, regido por la metodología seleccionada en el diseño teórico.

Capítulo 3. Validación de la propuesta de solución.

Resume el conjunto de pruebas realizadas al Portal WAP, mostrando las relaciones de las mismas y los casos de uso a través de No Conformidades en varias iteraciones, con el objetivo de asegurar la calidad y eficiencia del sistema.

Capítulo 1

Fundamentación teórica

1.1 Introducción

En este capítulo se describen los principales aspectos y conceptos de relevancia que han sido objeto de análisis a lo largo de la investigación, las tecnologías y metodología a utilizar, así como el lenguaje de programación elegido. Se expone además el estado del arte referente a las tendencias actuales de los portales WAP.

1.2 Fundamentación del tema

1.2.1 Telefonía móvil

Las tecnologías inalámbricas han tenido un desarrollo considerable en los últimos años. La telefonía celular da muestra de ello, desde sus inicios a finales de los años 70, ha revolucionado las actividades que se realizan a diario. Los teléfonos celulares se han convertido en una herramienta primordial para las personas, las hace sentir más seguras y más productivas.

A pesar de que la telefonía celular fue concebida para la voz únicamente, hoy en día es capaz de brindar servicios de datos, audio y video. Martin Cooper fue quien primero incursionó en esta tecnología, a él se le considera como el padre de la telefonía celular al introducir el primer radioteléfono en 1973 en los Estados Unidos.

El teléfono móvil es un dispositivo inalámbrico que permite tener acceso a la red de telefonía celular. Como característica más relevante se resalta su portabilidad, facilitando la comunicación desde casi cualquier lugar. Aunque su principal función es la comunicación por voz, su rápido desarrollo ha incorporado otras funciones tales como: acceso a Internet, envío de mensajes de texto, agenda, reproducción de vídeo y música, GPS (Sistema de Posicionamiento Global), entre otras (1).

Las primeras conexiones se efectuaban mediante una llamada telefónica a un número del operador, a través del cual se transmitían los datos de manera similar a como lo haría un módem en una computadora. Posteriormente nació GPRS (Servicio General de Paquetes vía Radio) para la transmisión de datos por paquetes, que permitió acceder a Internet a través del protocolo TCP/IP (Protocolo de Control de Transmisión / Protocolo de Internet). Se puede utilizar para servicios tales como FTP (Protocolo de Transferencia de Archivos), Telnet, servicio de mensajería, Internet y correo electrónico, utilizando los mismos protocolos que un ordenador convencional. La velocidad del GPRS es de 54 a 144 Mbps y se tarifa en función de la cantidad de información transmitida y recibida (2).

Con la aparición de la telefonía móvil digital, fue posible acceder a páginas de Internet especialmente diseñadas para móviles, lo cual dio lugar a la tecnología WAP.

Cada año que pasa es mayor el número de personas en el mundo que usan teléfonos móviles. Como se puede apreciar en la Figura 1, el número de usuarios de teléfonos móviles en todo el mundo ascendió a cerca de 1500 millones a finales del pasado año.

Figura 1. Usuarios de teléfonos móviles

1.2.2 WAP

El Protocolo de Aplicaciones Inalámbricas (WAP) ha sido un factor clave para la evolución de la telefonía móvil, permitiendo la implementación de un gran número de servicios, ya que WAP es un estándar bastante seguro, que permite que los usuarios accedan a la información a través de dispositivos inalámbricos como PDAs[1] (Asistente Personal Digital), teléfonos celulares y los denominados teléfonos inteligentes (3). Está soportado por la mayoría de las redes inalámbricas actuales, siendo un factor clave en el desarrollo de sistemas que brindan servicio a los usuarios que hacen uso de dispositivos con estas características.

En la siguiente figura se pueden observar los diferentes componentes en la arquitectura del protocolo WAP (4):

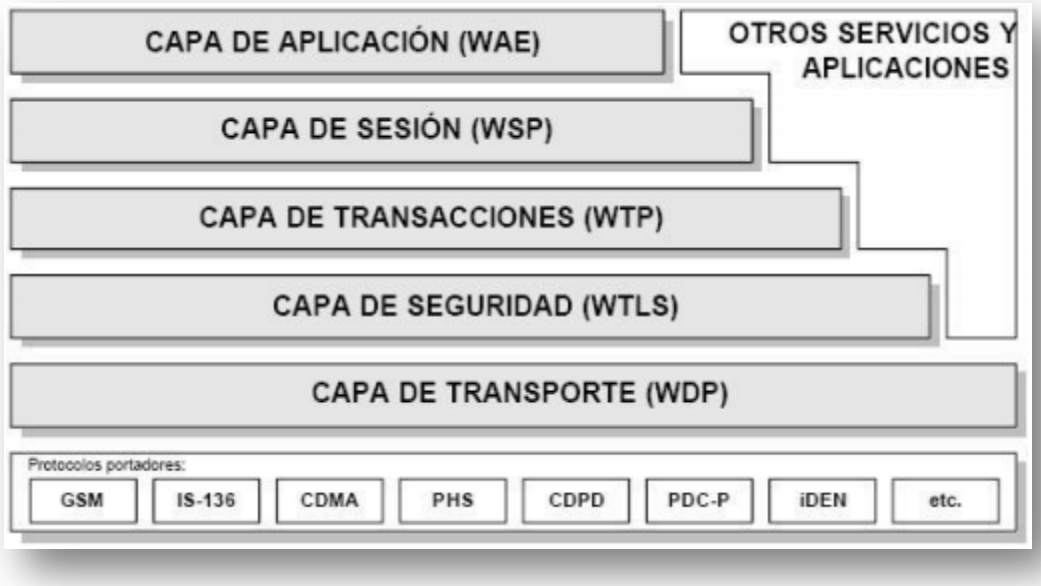


Figura 2. Arquitectura del protocolo WAP

[1] PDAs: es un computador de mano originalmente diseñado como agenda electrónica. Un PDA puede funcionar como teléfono móvil, fax, explorador de Internet, organizador personal, GPS, etc.

A continuación se explican cada una de las capas que componen la arquitectura del protocolo WAP.

Capa de Aplicación

El Entorno Inalámbrico de Aplicación (WAE) es un entorno de propósito general, basado en la combinación del World Wide Web (W3C)[2] y tecnologías de comunicaciones móviles.

Este entorno incluye un micronavegador, el cual posee básicamente las siguientes funcionalidades:

- ✓ Un lenguaje denominado WML (Lenguaje de Marcas Inalámbrico).
- ✓ Un lenguaje denominado WMLS, que es un lenguaje de script para dotar de cierto dinamismo a sus documentos.

Además de un conjunto de formatos de contenido bien definidos entre los que se encuentran imágenes, entradas en la agenda de teléfonos e información de calendario, denominados también como Servicios Telefónicos de Valor Añadido (TeleVAS).

Capa de Sesión

El Protocolo Inalámbrico de Sesión (WSP) proporciona a la Capa de Aplicación (WAP) una interfaz con dos servicios de sesión, un servicio orientado a conexión, el cual funciona por encima de la Capa de Transacciones y otro no orientado a conexión que funciona por encima de la Capa de Transporte y proporciona además servicios de datagramas seguros y no seguros.

Actualmente, esta capa consiste en servicios adaptados a aplicaciones basadas en la navegación web, proporcionando las siguientes funcionalidades:

- ✓ Semántica y funcionalidades del HTTP 1.1 (Protocolo de Transferencia de Hipertexto) en una codificación compacta.
- ✓ Negociación de las características del protocolo.

[2] El W3C es un consorcio internacional que tiene como objetivo desarrollar protocolos comunes y promover la evolución de la web.

- ✓ Suspensión, reanudación y cambio de sesión.

Capa de Transacciones

El Protocolo Inalámbrico de Transacción (WTP) funciona por encima de un servicio de datagramas, tanto seguros como no seguros, proporcionado las siguientes funcionalidades:

Tres clases de servicios de transacciones:

- ✓ Peticiones inseguras de un solo camino.
- ✓ Peticiones seguras de un solo camino.
- ✓ Transacciones seguras de dos caminos (petición - respuesta).

Capa de Seguridad

La Capa Inalámbrica de Seguridad de Transporte (WTLS) es un protocolo basado en el estándar SSL (Capa de Conexión Segura), utilizado en el entorno web para la proporción de seguridad en la realización de transferencia de datos. Este protocolo ha sido especialmente diseñado para el Protocolo de Transporte de WAP y optimizado para ser utilizado en canales de comunicación de banda estrecha. Para este protocolo se han definido las siguientes características:

- ✓ Integridad de los datos: asegura que los datos intercambiados entre el terminal y un servidor de aplicaciones no han sido modificados y que la información no es corrupta.
- ✓ Privacidad de los datos: asegura que la información intercambiada entre el terminal y un servidor de aplicaciones no pueda ser entendida por terceras partes que puedan interceptar el flujo de datos.
- ✓ Autentificación: contiene servicios para establecer la autenticidad del terminal y del servidor de aplicaciones.

Capa de Transporte

En esta capa se encuentra el Protocolo Inalámbrico de Datagramas (WDP) el cual proporciona un servicio fiable a los protocolos de las capas superiores de WAP y permite la comunicación de forma transparente sobre los protocolos portadores válidos.

Debido a que este protocolo proporciona una interfaz común a los protocolos de las capas superiores, las capas de Seguridad, Sesión y Aplicación pueden trabajar independientemente de la red inalámbrica que da soporte al sistema.

La arquitectura WAP consta de varias partes, en la Figura 3, se muestra su funcionamiento cuando se ejecutan peticiones realizadas por usuarios de terminales móviles.

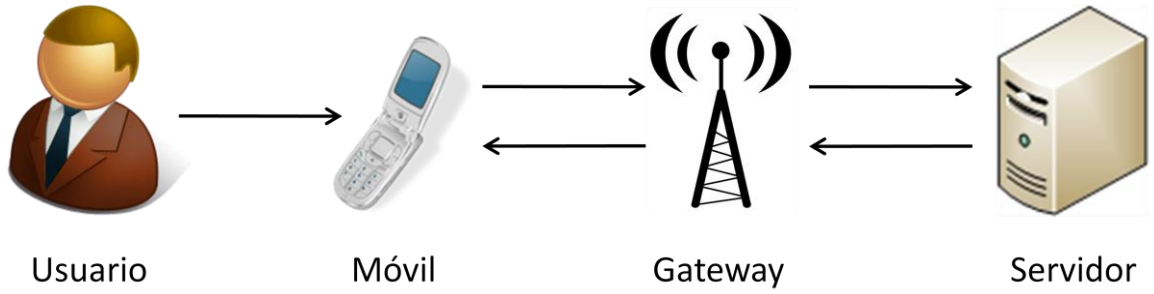


Figura 3. Funcionamiento de WAP

1. El usuario solicita la página WAP que quiere visualizar.
2. El navegador del móvil envía la petición con la dirección de la página solicitada y la información sobre el abonado al WAP Gateway (software capaz de conectarse a la red de telefonía móvil y a Internet).
3. El Gateway examina la petición y la envía al servidor donde se encuentra la información solicitada.
4. El servidor añade la información HTTP (Protocolo de Transferencia de Hipertexto) o HTTPS (Protocolo de Transferencia de Hipertexto Seguro) pertinente y envía la información de vuelta al Gateway.

5. En el Gateway se examina la respuesta del servidor, se valida el código en busca de errores y se genera la respuesta que se envía al móvil.

6. El micro navegador reconoce la información recibida y si el código es correcto lo muestra en pantalla.

El Gateway es la parte más importante del esquema anterior. Su función es convertir los protocolos de comunicación WAP (ejemplo: WSP, WTP, WTLS) a protocolos HTTP, TCP/IP y viceversa.

El auge en el desarrollo de los portales WAP se debe en gran medida a que los sitios web no fueron construidos teniendo en cuenta las limitaciones y características que poseen los dispositivos móviles para la visualización del contenido.

La mayoría de estos dispositivos no soportan scripts en su navegador, ni algunas características concernientes al lenguaje utilizado en la construcción de las páginas web, lo cual ocasiona que no se puedan visualizar las páginas y en caso contrario no lo hacen de la forma esperada.

El problema que se presenta al tener pantallas muy pequeñas es que se necesita de una mayor interacción por parte de los usuarios, los cuales tienen que recorrer hacia todas direcciones la pantalla con el objetivo de visualizar el recurso web de forma completa.

No existe una uniformidad en los formatos de contenidos digitales (imágenes, sonidos, documentos) soportados por los navegadores, ya que esa capacidad varía entre modelos de diferentes fabricantes, como también entre modelos de un mismo productor de dispositivos móviles. Esto trae aparejado que se haga muy difícil construir una aplicación donde cada contenido disponible para los usuarios pueda ser interpretado y visualizado de manera correcta.

La principal importancia de la tecnología WAP radica en que permite acceder a información e intercambiar mensajes en ambos sentidos. Esto es comparable al avance que representó la introducción de la telefonía móvil sobre la fija, pero en este caso es todavía más poderosa ya que también incluye datos.

Los principales beneficios de las aplicaciones WAP se resumen en que permiten:

- ✓ Consultar información específica desde cualquier lugar y en todo momento.

- ✓ Enviar y recibir información en todo el territorio de cobertura del operador.

Este protocolo no permite acceder libremente a Internet, ya que requiere que los sitios estén desarrollados teniendo en cuenta las especificaciones de los dispositivos, además de que no se disfrutan al máximo de las ventajas de multimedia (imágenes y sonidos) que habitualmente se utilizan en Internet.

Las páginas que muestran los dispositivos móviles, están escritas en un lenguaje llamado WML (Lenguaje de Marcado Inalámbrico). Se trata de una versión adaptada de HTML para la tecnología móvil, que tiene en cuenta las limitaciones de los móviles, tales como la cantidad de información que se puede visualizar, forma de navegación y entrada de datos (5).

1.2.3 Repositorio de Descripción de Dispositivos

El contenido web que se envía a dispositivos móviles ha sido adaptado para que tenga en cuenta una serie de factores tales como el tamaño de la pantalla y la compatibilidad con el lenguaje de etiquetado y el formato de imagen. Dicha información se almacena en Repositorios de Descripción de Dispositivos (DDR), los cuales son bases de datos de descripciones que los desarrolladores utilizan para adaptar la información a los diferentes dispositivos (6).

WURFL (Archivo Universal de Recursos Inalámbrico) es una base de datos en formato XML que contiene información sobre capacidades y características de muchos terminales móviles. Su principal objetivo es conocer las capacidades de los dispositivos móviles para poder adaptar el contenido que se entrega, de manera que los desarrolladores sean capaces de construir mejores aplicaciones y mejores servicios para los usuarios (7).

Una buena parte de los datos recogidos en WURFL provienen de una especificación devenida en estándar, creada por la Open Mobile Alliance (OMA[3]) denominada User Agent Profile (Perfil de Agente de Usuario).

Como el resto de las herramientas de su tipo, basa su funcionalidad en el parámetro llamado UserAgent (Agente de Usuario) de los dispositivos móviles, útil para poder determinar las características de estos. Tomando el UserAgent se puede consultar la base de datos de WURFL y conocer datos importantes del

[3] OMA: es una organización que desarrolla estándares abiertos para la industria de la telefonía móvil.

dispositivo móvil tales como: ancho y alto de la pantalla, el tipo de imágenes que puede desplegar, tipo de lenguaje estándar soportado para visualizar sitios web, si es que soporta la reproducción de archivos en formato de audio MP3, etc.

En esta base de datos se emplea un algoritmo de fall-back (recurrir a algo), consistente en que cada objeto mapeado dentro del fichero wurfl.xml posee un atributo que tiene como valor asociado el identificador del dispositivo del cual hereda. Esto se utiliza para recorrer el árbol de forma recursiva y así ir recuperando los valores de las propiedades de un equipo determinado a medida que se va descendiendo hasta la raíz, que es a su vez un objeto genérico con el que se asegura que ninguna propiedad se quede sin valor. Siguiendo este algoritmo, cuando un nuevo dispositivo se da de alta, solo se definen las características que le son diferentes con respecto a su predecesor.

1.2.4 Sistemas similares

Actualmente se utiliza cada vez con más frecuencia la tecnología WAP, la cual no sólo ofrece a los usuarios una variedad de información sobre temas específicos, sino que también les permiten el acceso a ofertas de servicios en forma de programas ejecutables.

Las aplicaciones concebidas originalmente como portales web son adecuadas también para el tratamiento y proceso de información, de hecho, los contenidos web ya existentes pueden ser modificados sin grandes esfuerzos de programación para su utilización WAP.

La investigación realizada arroja como conclusión, que la tecnología WAP tiene en la actualidad infinidad de posibilidades, para clientes tanto corporativos como individuales. Algunos ejemplos son:

- ✓ Agendas corporativas WAP.
- ✓ Gestión de pedidos.
- ✓ Servicios de localización.
- ✓ Servicios de mensajería.
- ✓ Tiendas virtuales.

- ✓ Comercio electrónico móvil.
- ✓ Servicios bancarios en línea.
- ✓ Venta y reserva de pasajes.
- ✓ Información del tiempo, tráfico, horarios, guía turística, entre otros.

Aunque la utilización de estos portales se ha popularizado gradualmente, su uso con fines policiales no ha sido extensivo. En Venezuela es primera vez que se utiliza la tecnología WAP para brindar servicios de este tipo, donde la seguridad de la información tiene un orden prioritario, debido a la sensibilidad y confidencialidad de los datos que se manejan.

Existe un sistema similar en Perú, utilizado por la policía para el registro y control de vehículos. Esta solución permite acceder vía WAP a un completo registro de autos robados para que la policía pueda chequear si el vehículo ha sido denunciado, ver su número de motor, chapa, entre otros datos.

1.2.5 Metodología de desarrollo de software

El concepto de metodología dentro de la ingeniería del software es uno de los más confusos. La constante innovación tecnológica hace que cada vez sea necesaria la aplicación de nuevas metodologías adaptadas a las necesidades actuales. No existe una metodología claramente superior a las demás, todas en esencia, han sido creadas con fines determinados.

Existe un grupo de metodologías comúnmente conocidas como tradicionales, propuestas casi todas con anterioridad a los años 90, las cuales pretendían ayudar a los profesionales indicando pautas para realizar y documentar cada una de las tareas del desarrollo del software. Sin embargo, la mayoría asume que un proyecto informático es casi una extensión de un proyecto burocrático tradicional, por tanto, los pasos que sugieren para llevar a cabo cada tarea, están cargados de reiteraciones y ambigüedades, no suelen tener en cuenta aspectos como la calidad, la satisfacción, la competitividad y los beneficios.

La mayor parte de las metodologías tradicionales ya no se ponen en práctica, están obsoletas desde casi todos los puntos de vista. Sólo algunas metodologías tradicionales han sido revisadas y adaptadas.

Por decisiones tecnológicas de la dirección del proyecto SIIPOL Móvil, están establecidas las herramientas, metodología y lenguaje de desarrollo a utilizar, por tal motivo no forma parte del presente trabajo investigar acerca de éstas, sino centrarse en el estudio de la metodología previamente seleccionada.

1.2.6.1 Proceso Unificado de Rational (RUP)

RUP constituye junto con el Lenguaje Unificado de Modelado (UML), la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos. RUP define claramente quién está haciendo qué, cuándo lo hace y cómo alcanzar un determinado objetivo. Divide el proceso de desarrollo en ciclos, teniendo un producto al final de cada uno, cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante:

- ✓ Inicio: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos.
- ✓ Elaboración: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos.
- ✓ Construcción: se enfoca en la elaboración de un producto completamente operativo y eficiente.
- ✓ Transición: se instala el producto y se capacita al usuario. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados.

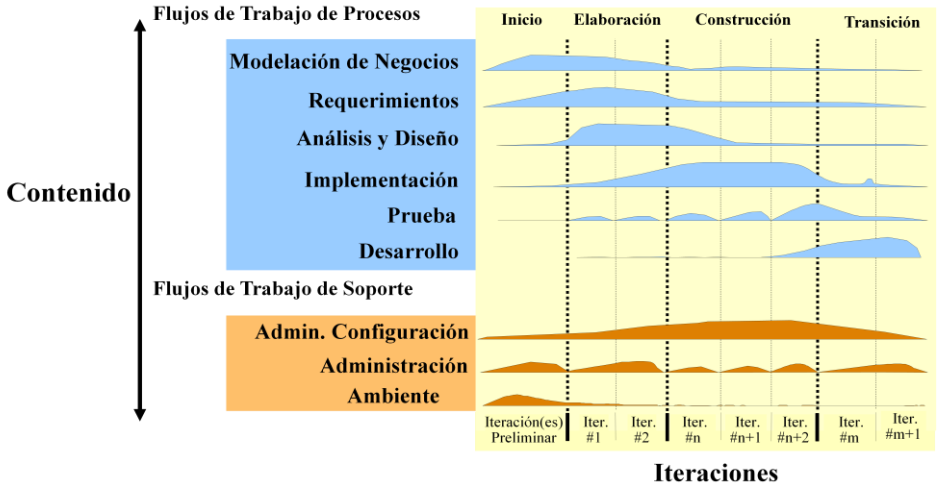


Figura 4. Estructura de RUP

El proceso de software propuesto por RUP tiene tres características esenciales (8):

- ✓ Guiado por casos de uso: un caso de uso es una descripción de una secuencia de interacciones de un sistema con una persona u otro sistema que usa alguno de sus servicios. Los casos de uso responden a un proceso o parte de un proceso dentro del negocio, guiando de esta manera el ciclo de desarrollo del software.

- ✓ Iterativo incremental: se dice que un proceso es iterativo incremental cuando en cada iteración se alcanza una mayor cercanía a los objetivos finales, añadiendo algo nuevo en cada una de ellas.

- ✓ Centrado en la arquitectura: la arquitectura es fundamentalmente una actividad centrada en el diseño, que muestra una visión común del sistema completo, por lo que repercute en todo el ciclo de vida de desarrollo del software. Las decisiones de la arquitectura se concentran en lo que es esencial en un sistema: la influencia de los requerimientos y las actividades importantes como el diseño colaborativo, el desarrollo e implementación del sistema, la planeación de su evolución y su adaptación.

El Proceso Unificado de Rational brinda una guía para encontrar, organizar, documentar y seguir los cambios de los requisitos funcionales y restricciones. Utiliza una notación de casos de uso y escenarios para representar los requisitos.

RUP utiliza como lenguaje de modelado UML, para visualizar, especificar, construir y documentar los artefactos de un software. El modelado visual ayuda a mantener la consistencia entre los artefactos del sistema: requisitos, diseños e implementaciones, por tanto, ayuda a mejorar la capacidad del equipo para gestionar la complejidad del software. UML utiliza diagramas y una semántica bien definida con el propósito de elaborar los artefactos de un sistema a través de las distintas etapas de su ciclo de vida, principalmente durante el análisis y el diseño del mismo. Ofrece nueve diagramas para modelar sistemas (9):

- ✓ Diagramas de Casos de Uso.
- ✓ Diagramas de Secuencia.
- ✓ Diagramas de Colaboración.
- ✓ Diagramas de Estado.
- ✓ Diagramas de Actividad.
- ✓ Diagramas de Clases.
- ✓ Diagramas de Objetos.
- ✓ Diagramas de Componentes.
- ✓ Diagramas de Implementación.

Se decide utilizar RUP como metodología para el desarrollo del Portal WAP, ya que reduce riesgos debido a que es un proceso iterativo, utiliza componentes, lo cual reduce el tiempo de realización del proyecto y lleva a cabo constantemente la ejecución de pruebas, con lo que se puede asegurar la calidad del producto final. Es adaptable, característica que permite ajustarla a las necesidades particulares del proyecto. Otra de las facilidades por la cual se escogió esta metodología fue debido a la distancia

geográfica que existe entre el cliente y el equipo de desarrollo, para lo cual RUP genera la documentación necesaria para validar los artefactos, además de poseer varios elementos de planificación que permiten llevar el control del desarrollo del proyecto.

1.2.6 Plataforma de desarrollo

J2EE (Java 2 Enterprise Edition) se ha convertido en una de las plataformas de programación más usada por los desarrolladores. Su principal ventaja consiste en que su entorno de desarrollo es independiente de la plataforma sobre la que se trabaje, o sea, sus aplicaciones funcionan sin importar cuál sea el sistema operativo sobre el que estén operando.

La plataforma J2EE proporciona un conjunto de herramientas que crean un escenario ideal para el desarrollo y despliegue de aplicaciones.

Como características más distintivas se destacan las siguientes:

- ✓ Portabilidad: los componentes pueden ser reubicados sin requerir cambios significativos.
- ✓ Diversidad de ambientes: puede ser desplegada en cualquier sistema sin que esto afecte su desempeño.
- ✓ Oportunidad en su aparición: los componentes que se desarrollan son implementados y publicados para su integración a las aplicaciones en el momento en que son requeridos.
- ✓ Reutilización: el diseño de componentes de software, con independencia de los servicios que provee, permite reforzar el concepto de reutilización, proporciona al integrador de aplicaciones la facilidad de generar soluciones completas, aprovechando los beneficios de los componentes disponibles.

Entre los principales beneficios que se obtienen con la aplicación de J2EE, se encuentran:

- ✓ Enriquecimiento y robustez de soluciones.
- ✓ Generación de componentes reutilizables dentro del ámbito de los distintos sistemas del proyecto.

- ✓ Detección de mecanismos de seguimiento integrales a los proyectos.
- ✓ Disminución de tiempos de investigación y de elaboración de prototipos.
- ✓ Reducción de tiempos en la replicación del conocimiento.

1.2.7.1 Lenguaje Java

Java es un lenguaje de desarrollo de propósito general, válido para realizar todo tipo de aplicaciones profesionales. Aunque los lenguajes de programación se encuentran fuera del ámbito de lo que es una plataforma, Java fue diseñado para usarse con la plataforma de igual nombre. Incluye una combinación de características que lo hacen único, siendo adoptado por multitud de fabricantes como herramienta básica para el desarrollo de aplicaciones. Entre ellas se destacan las siguientes:

- ✓ Lenguaje orientado a objetos: maneja conceptos tales como, encapsulación, herencia, polimorfismo, etc.
- ✓ Disponibilidad de un amplio conjunto de librerías: la programación de aplicaciones con Java se basa no solo en el empleo del juego de instrucciones que componen el lenguaje, sino también en la posibilidad de utilizar un amplio conjunto de clases.
- ✓ Lenguaje simple: Java posee una curva de aprendizaje muy rápida. Resulta más sencillo que otros lenguajes como C++, ya que se han eliminado características, tales como los punteros.
- ✓ Distribuido: proporciona una colección de clases para su uso en aplicaciones de red que permiten establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.
- ✓ Interpretado y compilado: Java es compilado en la medida en que su código fuente se transforma en una especie de código máquina. Por otra parte, es interpretado, ya que se puede ejecutar directamente sobre cualquier máquina que tenga el intérprete y el sistema de ejecución en tiempo real.
- ✓ Robusto: fue diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Sus características de memoria liberan

a los programadores de una familia entera de errores, ya que se ha prescindido por completo de los punteros y la recolección de basura elimina la necesidad de liberación explícita de memoria.

- ✓ Seguro: dada la naturaleza distribuida de Java, la seguridad se impuso como una necesidad de vital importancia, implementando barreras de seguridad en el lenguaje y en el sistema de ejecución en tiempo real.
- ✓ Indiferente a la arquitectura: Java está diseñado para soportar aplicaciones que se ejecuten en los más variados entornos de red.
- ✓ Portable: la indiferencia a la arquitectura representa sólo una parte de su portabilidad. Java especifica el tamaño de sus tipos de datos básicos y el comportamiento de sus operadores aritméticos, de manera que los programas son iguales en todas las plataformas.
- ✓ Dinámico: el lenguaje Java y su sistema de ejecución en tiempo real son dinámicos en la fase de enlazado. Las clases se enlazan a medida que son necesitadas.

1.2.7 Herramientas de desarrollo

Una herramienta de desarrollo es un programa especializado que da soporte a una tarea concreta dentro de las actividades de desarrollo de software. Han sido creadas para ayudar y asistir a los ingenieros en el desarrollo de sistemas informáticos (12).

Dentro de sus objetivos se encuentran: facilitar la creación de sistemas que cumplan con las necesidades y los requisitos de los usuarios, garantizando la calidad de los mismos, completar la creación de aplicaciones en un tiempo razonable, así como aumentar la productividad de los ingenieros.

Las herramientas de desarrollo aumentan y facilitan cuantiosamente la velocidad de la producción, por lo cual se hace indispensable su uso en todo proceso de desarrollo de software.

1.2.7.1 Visual Paradigm

Visual Paradigm es una herramienta CASE (Ingeniería de Software Asistida por Ordenador) que aumenta la productividad en el desarrollo de software, reduciendo el coste de las aplicaciones informáticas en términos de tiempo y dinero. Soporta el ciclo de vida completo del desarrollo de software: Análisis y

Diseño orientados a objetos, Construcción, Pruebas y Despliegue. Visual Paradigm ayuda a una rápida construcción de aplicaciones con calidad, permitiendo a partir de la documentación que de éste se genera, una mayor comprensión de los procesos que se llevan a cabo durante el desarrollo del software.

Características fundamentales:

- ✓ Fácil de usar, ya que sus componentes se encuentran relacionados, por lo que se hace sencillo la creación de cualquier tipo de diagrama.
- ✓ Soporta UML 2.1.
- ✓ Puede utilizar técnicas de ingeniería inversa para llevar de código fuente a diagramas de clases, manteniendo de esta manera la sincronización entre el modelo y el código.
- ✓ Permite la generación de código a partir de diagramas para plataformas como .Net, Java y PHP.
- ✓ Permite la exportación e importación de ficheros, brindando la posibilidad de intercambiar información con aplicaciones como Visio y Rational Rose.
- ✓ Brinda la posibilidad de generar informes.
- ✓ Incluye el modelado colaborativo con Subversion.

1.2.7.2 Entorno de Desarrollo Integrado

Un IDE (Entorno de Desarrollo Integrado) es un conjunto de herramientas utilizadas por los programadores, que incluye por lo general, un buen editor de código, administrador de proyectos y archivos, enlace a compiladores e integración con sistemas controladores de versiones o repositorios, además de brindar facilidades para la construcción de interfaces gráficas de usuario (11).

Las características principales de un Entorno de Desarrollo Integrado son las siguientes:

- ✓ Permite trabajar con varios proyectos a la vez, encontrar código duplicado y personalizar el entorno.
- ✓ Incorpora completamiento de código.

- ✓ Permite la integración con aplicaciones de control de versiones.
- ✓ Permite comparar archivos.
- ✓ Mantiene un historial local de los archivos.
- ✓ Incorpora funcionalidades a través de plugins[4].
- ✓ Permite el trabajo con múltiples lenguajes de programación.

Eclipse

Eclipse es un IDE que trabaja sobre un amplio rango de sistemas operativos, con una arquitectura abierta y basada en plugins, que facilita las tareas de edición, compilación y ejecución de programas durante su fase de desarrollo. Aunque soporta varios lenguajes de programación, es con el lenguaje Java con el que mejor se integra.

Para el caso de Java es necesario, disponer del Kit de Desarrollo de Software el cual incluye las herramientas de desarrollo necesarias, ofreciendo un IDE con un compilador interno y un modelo completo de los archivos fuente de Java, lo cual permite técnicas avanzadas de refactorización y análisis de código. Eclipse realiza las pruebas unitarias con JUnit, además, a través de plugins libremente disponibles es posible añadir control de versiones con Subversion e integración con Hibernate.

1.2.7.3 Sistema de Control de Versiones

Un sistema de control de versiones es un sistema de gestión de archivos y directorios, cuya principal característica es que mantiene el registro de los cambios y modificaciones que se han realizado sobre estos a lo largo del tiempo (12). De esta forma, el sistema es capaz de recordar las versiones antiguas de los datos, lo que permite examinar el historial de cambios o recuperar versiones anteriores de un fichero.

[4] Plugins: es una aplicación que se relaciona con otra para aportarle una función nueva.

TortoiseSVN

Es un cliente gratuito de código abierto para el sistema de control de versiones Subversion. Se encarga del manejo de ficheros y directorios a lo largo del tiempo, los cuales se almacenan en un repositorio central.

Entre las características de TortoiseSVN se destacan:

- ✓ Integración con el explorador de Windows.
- ✓ Íconos sobreimpresionados: el estado de cada carpeta y fichero versionado se indica por pequeños íconos. De esta forma, se puede ver fácilmente el estado en el que se encuentra la información.
- ✓ Fácil acceso a los comandos de Subversion.
- ✓ Proporciona una serie de herramientas internas, tales como:
  - TortoiseMerge: permite ver las diferencias entre ficheros de texto y fusionar los cambios.
  - TortoiseBlame: hace más fácil la lectura de los ficheros de autoría.
  - SubWCRev: es un programa de consola para Windows que puede utilizarse para leer el estado de una copia de trabajo local y opcionalmente realizar sustituciones de palabras claves en un fichero.

1.2.7.4 Interfaz de Programación de Aplicaciones (API)

Una API representa una interfaz de comunicación entre componentes de software. Se trata del conjunto de llamadas a bibliotecas que ofrecen acceso a determinados servicios y representa un método para conseguir abstracción en la programación (13).

Alembik

Es un API de Java que provee servicios de transcodificación para diferentes archivos de multimedia (imagen, audio video, etc.). Alembik utiliza el fichero wurfl.xml para buscar las características de los dispositivos móviles.

Los parámetros de transcodificación dependen del tipo de dispositivo móvil. Pueden especificar: tamaño, dimensión y el formato de salida del archivo deseado.

1.2.7.5 Frameworks

Un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado para ayudar a desarrollar y unir los diferentes componentes de un proyecto (14). Son desarrollados para brindar a los programadores y diseñadores una mejor organización y estructura a sus proyectos. Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo tales como el uso de patrones.

Algunas ventajas del uso de los frameworks son:

- ✓ Modularidad y reducción de la complejidad: la aplicación está formada por subsistemas especializados.
- ✓ Fortaleza al cambio: los módulos pueden ser cambiados conservando la arquitectura global de la aplicación.
- ✓ Documentación: la documentación del framework promueve el uso correcto del mismo y disminuye el esfuerzo necesario para el mantenimiento.
- ✓ Estructura: el desarrollo basado en frameworks establece una estructura sobre la cual las aplicaciones pueden ser construidas, liberando al desarrollador de tomar el 100% de las decisiones de diseño.
- ✓ Distribución de funciones: permite paralelizar el trabajo ya que la solución puede desarrollarse como un conjunto de piezas independientes que encajarán en el framework usado.
- ✓ Eficiencia: el desarrollador puede concentrarse en los requerimientos funcionales de la aplicación.
- ✓ Aplicaciones ricas: posibilidad de dar más funcionalidad a los usuarios de la aplicación.

SpringMVC

Es utilizado para la capa de presentación. Es un framework para desarrollar aplicaciones Java, que implementa una arquitectura Modelo Vista Controlador (MVC). Dos de sus objetivos más importantes son: permitir que el desarrollo se concentre en la lógica del negocio y que se haga empleando buenos principios de diseño orientado a objetos.

Algunas de las ventajas de utilizar Spring MVC son:

- ✓ Ofrece una división limpia entre el modelo, la vista y el controlador.
- ✓ Es muy flexible, ya que implementa toda su estructura mediante interfaces. Todas las partes del framework son configurables a través de plugins.
- ✓ Provee interceptores como controladores que permiten factorizar el comportamiento común en el manejo de múltiples peticiones.
- ✓ No obliga a utilizar JSP (Páginas de Servidor de Java), ofrece mejor integración con otras tecnologías como Velocity o FreeMaker.
- ✓ Tiene una interfaz bien definida para la integración con la capa de negocio.

Spring

Para la lógica de negocio es utilizado Spring, un framework basado en la técnica de Inversión de Control (IoC) y una implementación de desarrollo según la Programación Orientada a Aspectos (AOP), dando soporte y simplificando la complejidad propia del software, promueve el bajo acoplamiento a partir de la inyección de dependencias entre los objetos (17).

Está formado por un conjunto de componentes reutilizables para realizar tareas comunes. Sus dos principales fortalezas son la utilización de patrones reutilizables y la integración con otras herramientas o incluso otros frameworks, con el fin de obtener los beneficios que el desarrollador desea de cada una de ellas.

Spring está diseñado con interfaces para que el desarrollador pueda utilizarlas, promoviendo de esta forma la reutilización de código. Otro enfoque bajo el cual fue construido y diseñado, es que una clase tenga una única función, con el propósito de que se enfoque en resolver su tarea de manera más centrada.

Entre sus características claves se encuentran:

- ✓ Implementa la filosofía POJO (Objeto Java), la cual revalora la simplicidad de las clases Java aportando manejo de transacciones de forma no intrusiva.
- ✓ Incorpora un módulo para la implementación de la seguridad, llamado Spring Security.
- ✓ Brinda soporte para el acceso y la publicación de servicios web.
- ✓ Provee un paquete de pruebas para componentes del framework integrado con JUnit.

Hibernate

La persistencia de datos de cualquier aplicación de software ha sido uno de los aspectos más importantes que influye directamente en el desempeño y tiempos de desarrollo de cualquier aplicación de software.

Una de las técnicas más utilizadas hoy en día es el mapeo de objetos a tablas de un modelo de bases de datos relacional. Las ventajas que ofrece esta estrategia han dado lugar al desarrollo de diversos frameworks de persistencia encargados de resolver problemas como el mapeo de relaciones entre objetos.

Hibernate es uno de estos frameworks, ampliamente utilizados en el desarrollo de aplicaciones. Es una herramienta para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional y el modelo de objetos de una aplicación mediante archivos XML (16). Permite además desarrollar un modelo de datos utilizando todo el poder de un lenguaje orientado a objetos, incluyendo asociación, herencia, polimorfismo, composición y el framework de colección de Java. Está diseñado para ser flexible en cuanto al esquema de tablas utilizado, para poder adaptarse a una base de datos ya existente. Hibernate simplifica el proceso de creación de aplicaciones y reduce drásticamente el tiempo de desarrollo, mantenimiento y resolución de errores.

Entre las principales características de Hibernate están las siguientes:

- ✓ Permite diseñar objetos persistentes que incluyen polimorfismo, relaciones, colecciones y un gran número de tipos de datos.
- ✓ Brinda soporte para el manejo de transacciones, concurrencia y caché.
- ✓ Ofrece soporte para un amplio conjunto de colecciones de Java.
- ✓ Incorpora un lenguaje para el manejo de consultas a la base de datos (HQL - Lenguaje de Consultas de Hibernate).
- ✓ Es no intrusivo.

Hay un conjunto de factores que hacen de Hibernate una potente herramienta, entre ellos:

- ✓ Productividad: usado conjuntamente con otras herramientas reduce significativamente el tiempo de desarrollo de aplicaciones.
- ✓ Mantenibilidad: al tener menos líneas de código hace que el sistema sea más comprensible, centrándose en la lógica de negocio.
- ✓ Rendimiento: la mayoría de las optimizaciones son más sencillas disponiendo de un ORM (Mapeo de Objeto Relacional), resultando más fáciles de optimizar las consultas.
- ✓ Flexibilidad: puede integrarse en una arquitectura J2EE y permite configurarlo con JDBC (Conectividad de Base de Datos de Java).

JUnit

Es un framework simple y de código abierto que permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de las clases se comportan como se espera. Es seleccionado para la realización de dos tipos de pruebas fundamentalmente:

- ✓ Pruebas unitarias: consisten en probar la correcta funcionalidad del módulo en cuestión como si actuara independiente del resto.
- ✓ Pruebas de integración: se prueba la correcta integración de cada módulo.

El framework provee al desarrollador de herramientas, clases y métodos que le facilitan la tarea de realizar pruebas en el sistema y así asegurar su consistencia y funcionalidad, determinando si el código cumple con las funcionalidades específicas para el que fue realizado, evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera y comprobar la correcta funcionalidad de las aplicaciones.

Es además un medio para controlar las pruebas de regresión, necesarias cuando una parte del código ha sido modificada 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.

1.2.8 Arquitectura técnica

Con el objetivo de dar cumplimiento a los requisitos funcionales y no funcionales del Portal WAP se tiene como propuesta de solución un Portal WAP siguiendo el patrón arquitectónico MVC (Modelo, Vista, Controlador), basado en tecnología Java, al que se conectarán los funcionarios del CICPC vía GPRS, permitiendo de esta forma, realizar consultas a la base de datos del SIIPOL.

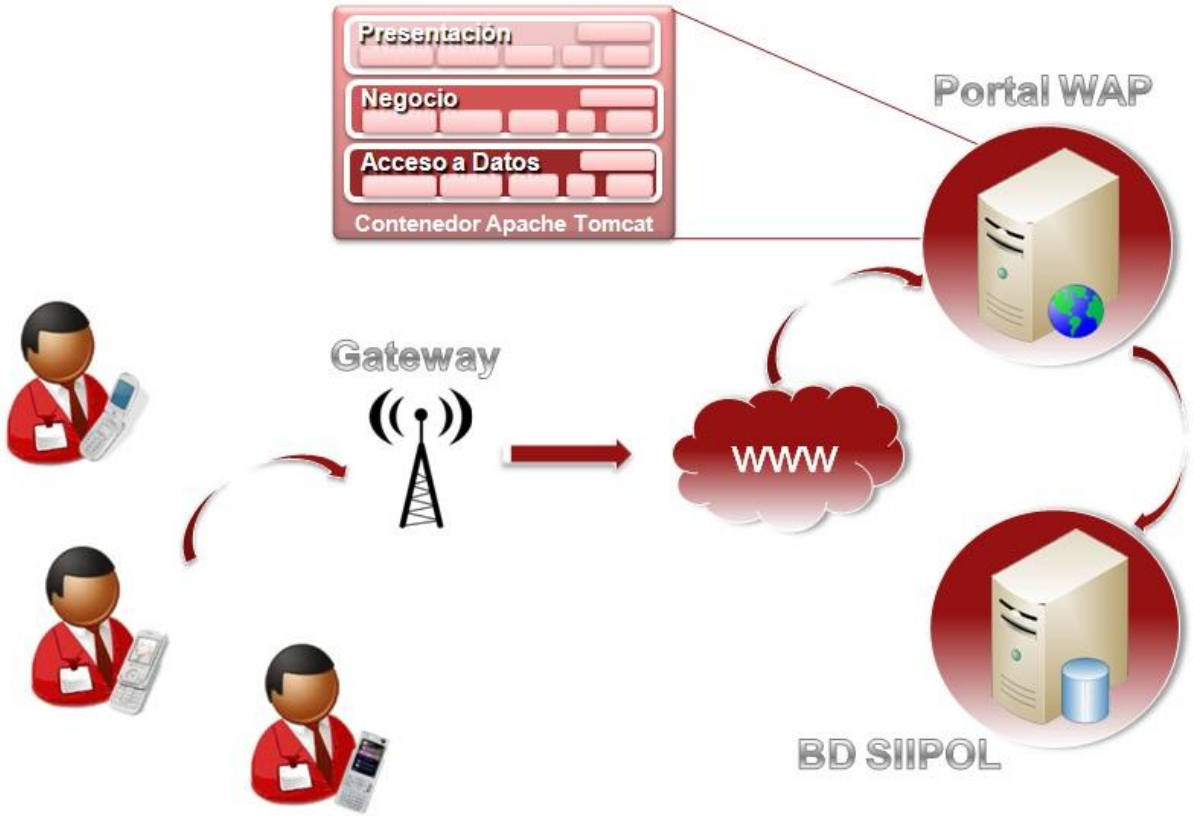


Figura 5. Propuesta de solución

1.3 Conclusiones

A partir del estudio realizado se determinó utilizar RUP como metodología de desarrollo de software. Se describieron las herramientas y frameworks a utilizar en la construcción del Portal WAP. Se selecciona Visual Paradigm para el modelado del sistema, utilizando además SpringMVC como framework para la capa de presentación, Spring para la lógica de negocio, Hibernate para el acceso a datos y JUnit para la realización de las pruebas al sistema.

Capítulo 2

Análisis, diseño e implementación de la propuesta de solución

2.1 Introducción

El objetivo principal de este capítulo consiste en realizar la descripción de los procesos que se llevan a cabo en la disciplina de análisis y diseño, obteniendo como resultado los artefactos más importantes para modelar el sistema, teniendo en cuenta los patrones de diseño que aportan soluciones concretas a problemas específicos para lograr un diseño eficaz en el software. Se describe además, el modelo de implementación utilizado y se muestran los diagramas de componente y de despliegue.

Se seleccionaron los casos de uso significativos por la funcionalidad que describen dentro de los procesos del Portal WAP, los mismos se relacionan a continuación:

- ✓ Consultar Personas con Registros Policiales.
- ✓ Ver Detalles de Personas con Registros Policiales.
- ✓ Registrar Traza de Consulta Sobre Entidad Operativa.

Los restantes casos de uso no se abordan en este capítulo debido a que poseen funcionalidades similares a los antes mencionados.

2.2 Descripción del Sistema

La ingeniería de requisitos ayuda a los ingenieros de software a entender mejor el problema en cuya solución trabajarán. Incluye el conjunto de tareas que conducen a comprender cuál será el impacto del software sobre el negocio, qué es lo que el cliente quiere y cómo interactuarán los usuarios finales con el producto final.

Un requisito es una condición o capacidad que tiene que ser alcanzada por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente.

2.2.1 Requisitos Funcionales (RF)

Los requisitos funcionales de un sistema describen lo que el sistema debe hacer, responden a las necesidades del cliente, se mantienen invariables sin importar que propiedades o cualidades se relacionen y son un punto de partida para especificar qué es lo que debe hacer el sistema a desarrollar. Los requisitos funcionales asociados al Portal WAP son:

RF 1. Iniciar Sesión en la Aplicación WAP.

RF 2. Cerrar Sesión en la Aplicación WAP.

RF 3. Consultar Personas con Registros Policiales.

RF 4. Ver Detalles de Personas con Registros Policiales.

RF 5. Consultar Vehículos con Registros Policiales.

RF 6. Ver Detalles de Vehículos con Registros Policiales.

RF 7. Consultar Armas con Registros Policiales.

RF 8. Ver Detalles de Armas con Registros Policiales.

RF 9. Consultar Objetos con Registros Policiales.

RF 10. Ver Detalles de Objetos con Registros Policiales.

RF 11. Consultar Persona Natural.

RF 12. Ver Detalles de Persona Natural.

RF 13. Registrar Traza de Consulta Sobre Entidad Operativa.

2.2.2 Requisitos No Funcionales (RNF)

Son propiedades o cualidades que el producto debe tener, es decir, aquellas características que hacen al producto atractivo, usable, rápido y confiable.

Los requisitos no funcionales más significativos del Portal WAP son (19):

Funcionalidad

RNF 1. El sistema mostrará los errores en forma de mensajes.

- ✓ Todos los mensajes de error del sistema deberán incluir una descripción textual del error.

RNF 2. El sistema adaptará los contenidos en correspondencia con las características específicas de cada dispositivo.

- ✓ El sistema transformará las imágenes que se deban visualizar en el móvil a un formato y resolución adecuada.
- ✓ La fuente de datos desde la cual se extraerán las características de los teléfonos será el fichero wurfl.xml mantenido por el proyecto WURFL.

Usabilidad

RNF 3. Los formularios serán estandarizados, por tanto:

- ✓ Los campos de texto tendrán un tamaño estándar de acuerdo con el espacio que se tenga en el área de la página y en la medida que se llene esa área primaria agregar la barra de desplazamiento vertical.
- ✓ No se utilizarán textos extensos para las etiquetas de la interfaz de usuario.

RNF 4. Todas las páginas tendrán un menú de navegación.

RNF 5. Los resultados de las consultas que tengan más de 5 coincidencias serán paginados.

- ✓ Se mostrará en la parte superior de la página el total de elementos encontrados.
- ✓ Se mostrará en la parte inferior enlaces de navegación: ir hacia delante, hacia atrás o ir al inicio de los resultados mostrados.

RNF 6. Las fotos se mostrarán siempre que el dispositivo destino las soporte y sus dimensiones y formatos se cambiarán a los adecuados para su mejor visualización en los navegadores.

Fiabilidad

RNF 7. El sistema estará disponible durante 24 horas, los 7 días de la semana, todos los días del año.

RNF 8. El sistema usará igualmente la potencialidad del EVA 8000 para realizar balanceo de carga entre los servidores de aplicación permitiendo acortar los tiempos de respuesta.

Seguridad

RNF 9. El sistema concederá acceso a partir de un nombre de usuario y una contraseña.

RNF 10. El sistema concederá el acceso a cada usuario autenticado solamente a las funciones que le estén permitidas, de acuerdo a su perfil.

RNF 11. El sistema solicitará el re-acceso del usuario después de 5 minutos de inactividad.

RNF 12. El sistema implementará el uso de campos obligatorios y validaciones para garantizar la integridad de la información que se introduce por el usuario.

RNF 13. Las conexiones desde el móvil serán a través del protocolo HTTPS.

RFN 14. El sistema inhabilitará los accesos desde un número de teléfono determinado si desde éste se realizan 5 o más intentos fallidos de acceso al sistema.

Rendimiento

RNF 15. El sistema identificará por cada caso de uso, aquellas operaciones que impliquen un elevado nivel de procesamiento en la base de datos y usará procedimientos almacenados para manejarlas.

RNF 16. El sistema procesará una transacción en un tiempo no mayor a 1 segundo, a partir de que la petición se recibe en el servidor, es decir, esta cifra no incluye los retardos por concepto de tráfico de red.

Restricciones de diseño

RNF 17. El sistema se implementará usando la plataforma J2EE.

RNF 18. El sistema estará basado en un estilo arquitectónico en capas.

RNF 19. Para la transformación de imágenes el sistema utilizará el API Alembik.

Interfaz de Usuario

RNF 20. Todos los textos y mensajes en pantalla aparecerán en idioma español. Los errores serán visibles al usuario y en lo posible incluirán sugerencias de las posibles soluciones.

RNF 21. El sistema presentará los términos capitalizados, es decir, tendrán su primera letra en mayúsculas.

Interfaz de hardware

RNF 22. Los teléfonos móviles serán capaces de establecer conexiones GPRS al servidor donde esté desplegado el sistema.

Interfaz de comunicación

RNF 23. Los teléfonos móviles se conectarán a la aplicación a través de la red GPRS.

2.2.3 Descripción de la seguridad

La seguridad es una característica de cualquier aplicación informática, que indica que está libre de todo peligro, daño o riesgo y que es en cierta manera infalible. Mantener un sistema seguro consiste básicamente en garantizar tres aspectos: confidencialidad, integridad y disponibilidad. La confidencialidad indica que la información de una aplicación ha de ser accedida únicamente por quien está autorizado y de forma controlada; la integridad significa que los datos solamente pueden ser modificados por las personas autorizadas y la disponibilidad se refiere a que la información tiene que estar disponible cuando se necesite.

La seguridad del Portal WAP se maneja a través de Spring Security, el cual es un módulo del framework Spring, encargado de controlar los procesos de autenticación y autorización.

Spring Security provee una implementación segura y además es independiente de cualquier mecanismo de seguridad provisto por un servidor, lo cual permite cambiar el servidor de aplicación sin hacer ninguna modificación en el esquema de seguridad de la aplicación. Por otra parte, aprovecha las ventajas de la

inversión de control de Spring, lo que posibilita cambiar los parámetros de autenticación y autorización en caso de que se realice algún cambio en el modelo de seguridad utilizado.

Para autenticarse en el portal se comprueba el nombre del usuario y la contraseña en la base de datos, luego se chequea que el mismo se pueda conectar a través del teléfono desde el cual intenta acceder al sistema, además de pasar la prueba de CAPTCHA (Prueba Turing Pública Completamente Automatizada para diferenciar las Computadoras de los Humanos). En caso de que el usuario no haya cambiado su contraseña en un período de 30 días, al autenticarse en la aplicación se le brinda la opción de cambiarla antes de que pueda realizar cualquier acción.

Una vez autenticado, a través de Spring Security se verifica qué roles tiene asignado el usuario y en dependencia de los permisos que tenga, podrá ejecutar o no las posibles acciones en la aplicación. Por otra parte, el sistema inhabilita una cuenta si se han realizado 5 intentos fallidos de acceso al sistema e imposibilita el acceso del número de teléfono desde el cual se intenta acceder al portal.

2.2.4 Diagrama de Casos de Uso

El Portal WAP cuenta con un total de 13 CU, los cuales agrupan las funcionalidades referidas en el epígrafe anterior. A continuación se presenta el diagrama de Casos de Uso de la aplicación.

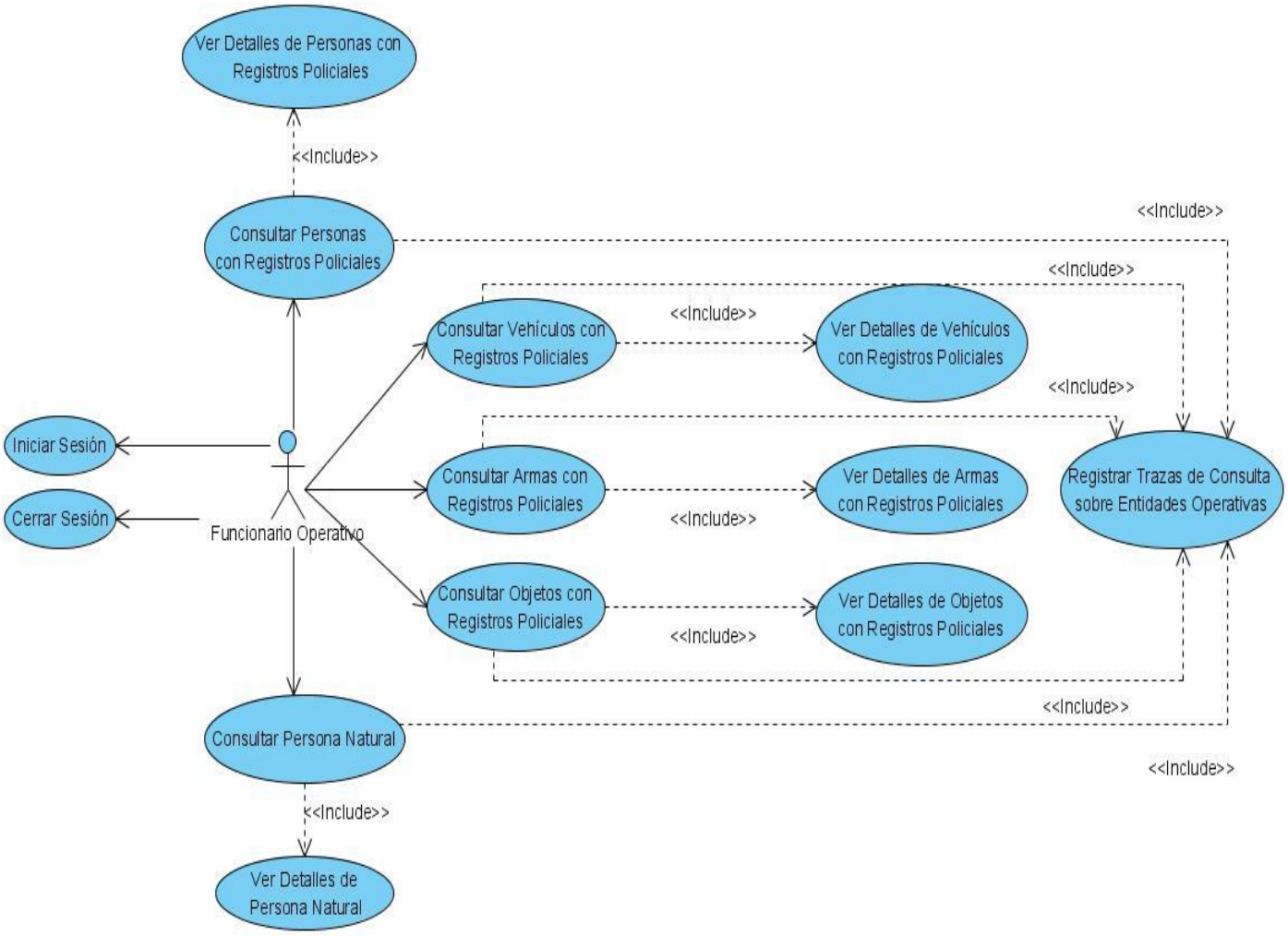


Figura 6. Diagrama de Casos de Uso

2.2.5 Descripción de los Casos de Uso del Sistema

Las descripciones se realizan a través de casos de uso a nivel resumen y su objetivo es proponer de manera general las prestaciones del módulo y garantizar la construcción de un software que cumpla con las expectativas y necesidades del usuario. A continuación se presenta un resumen de los casos de uso seleccionados.

Tabla 1. Descripción CU Consultar Personas con Registros Policiales

Nombre del CU	Consultar Personas con Registros Policiales
Propósito	Consultar las Personas con Registros Policiales.
Actor (es)	Funcionario Operativo (Inicia): Consulta las Personas con Registros Policiales.
Descripción	El caso de uso inicia cuando el Funcionario Operativo accede a Consultar las Personas con Registros Policiales. El sistema solicita los criterios por los que el Funcionario Operativo puede realizar la búsqueda, se validan los datos proporcionados, si son correctos realiza la búsqueda, devolviendo una lista de coincidencias, de lo contrario se muestra un mensaje de Error. Termina el caso de uso.
Precondiciones	Debe haberse autenticado en la aplicación WAP y el usuario debe tener permiso para realizar consultas sobre personas con registros policiales.
Poscondiciones	Se consultaron las Personas con Registros Policiales.
Complejidad	Media

Tabla 2. Descripción CU Ver Detalles de Personas con Registros Policiales

Nombre del CU	Ver Detalles de Personas con Registros Policiales.
Propósito	Ver Detalles de las Personas con Registros Policiales.
Actor (es)	Funcionario Operativo (Inicia): Ve los detalles de las Personas con Registros Policiales.

Descripción	El caso de uso inicia cuando el Funcionario Operativo accede a ver los Detalles de las Personas con Registros Policiales, el sistema muestra solo los datos que aparezcan en el sistema. Termina el caso de uso.
Precondiciones	Debe haberse realizado la consulta sobre una persona.
Poscondiciones	Se mostraron los detalles de las Personas con Registros Policiales.
Complejidad	Media.

Tabla 3. Descripción CU Registrar Traza de Consulta Sobre Entidad Operativa

Nombre del CU	Registrar Traza de Consulta Sobre Entidad Operativa.
Propósito	Registrar las trazas de las consultas que se realizan sobre las entidades operativas.
Actor (es)	Caso de Uso Base.
Descripción	El caso de uso se inicia cuando se detecta que el Funcionario Operativo ha realizado una búsqueda sobre alguna Entidad Operativa. Al realizarse dicha consulta el sistema registra una Traza con los datos de la búsqueda realizada. Termina el caso de uso.
Precondiciones	Debe haberse realizado una consulta sobre una entidad operativa.
Poscondiciones	Se registra una Traza de Consulta sobre una entidad operativa.
Complejidad	Baja.

2.3 Modelo de Análisis

Las clases de análisis se centran en los requisitos funcionales y son evidentes en el dominio del problema, porque representan conceptos y relaciones del dominio. Se inserta además la realización de casos de uso del análisis que describe cómo se lleva a cabo y se ejecuta un caso de uso determinado en término de las clases de análisis y de sus objetos en interacción. Contiene además los paquetes de análisis que organizan los artefactos en piezas manejables, así como la descripción de la arquitectura (vista del modelo de análisis) que muestra los artefactos significativos para el sistema propuesto.

El objetivo principal del análisis es transformar los requerimientos a una especificación que describa cómo implementar el sistema, consiste fundamentalmente en obtener una visión que se preocupa por ver que hace el sistema a desarrollar, por tal motivo este se interesa por los requerimientos funcionales.

Los objetivos específicos del análisis son:

- ✓ Transformar los requerimientos al diseño del futuro sistema.
- ✓ Desarrollar una arquitectura para el sistema.

2.3.1 Diagrama de Clases de Análisis

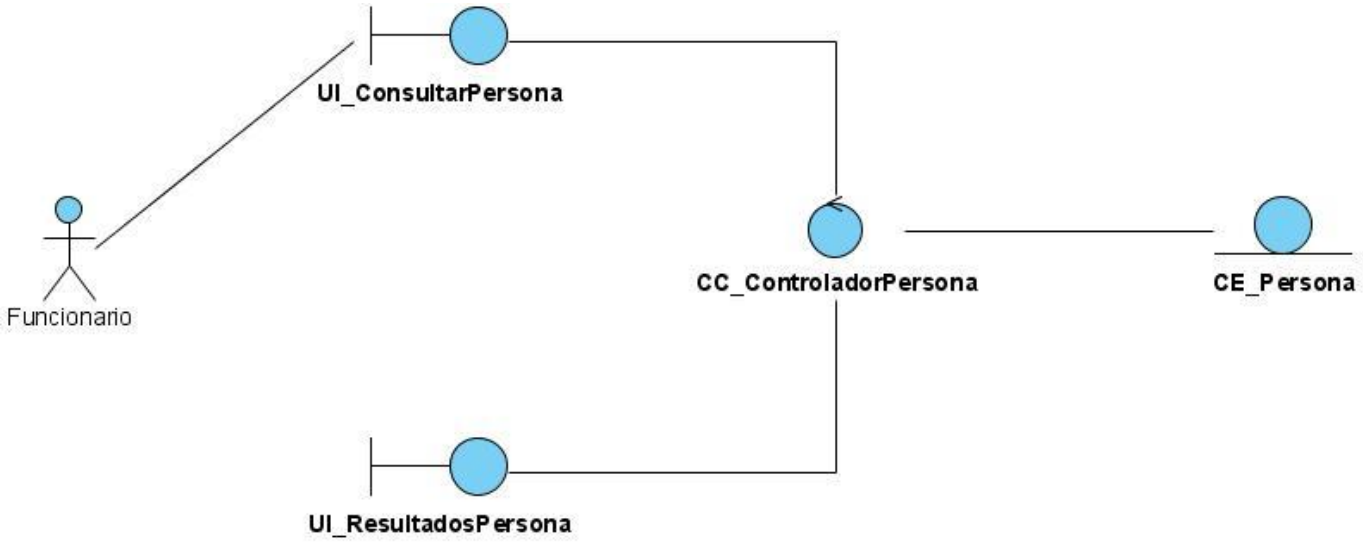


Figura 7. Diagrama de Clases de Análisis CU Consultar Personas con Registros Policiales

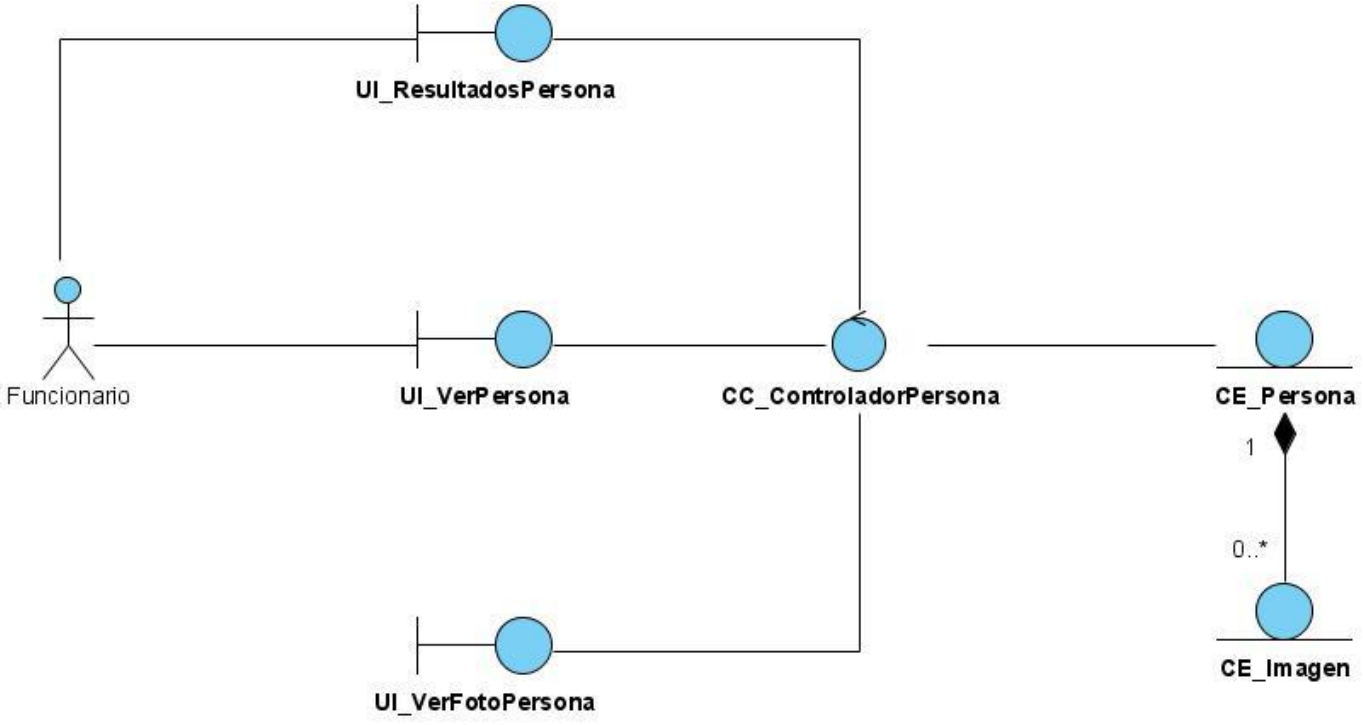


Figura 8. Diagrama de Clases de Análisis CU Ver Detalles de Personas con Registros Policiales

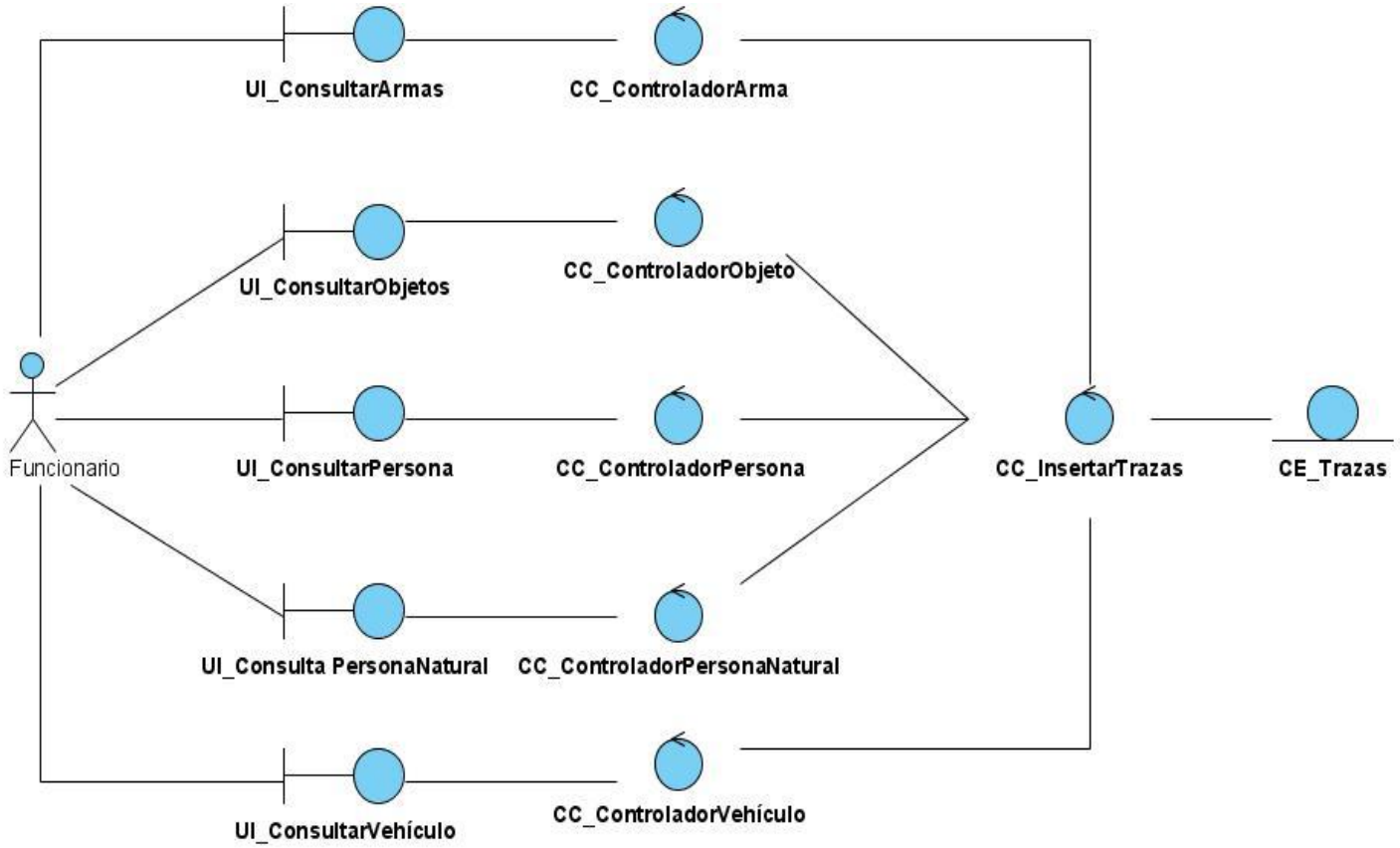


Figura 9. Diagrama de Clases de Análisis CU Registrar Traza Sobre Consulta de Entidad Operativa

Los Diagramas de Clases de Análisis restantes se pueden consultar en el Anexo 1.

2.3.2 Diagrama de Colaboración

Los diagramas de colaboración muestran una interacción organizada basándose en los objetos que toman parte en la interacción y los enlaces entre los mismos. La secuencia de los mensajes y los flujos de ejecución concurrentes se determinan explícitamente mediante números de secuencia.

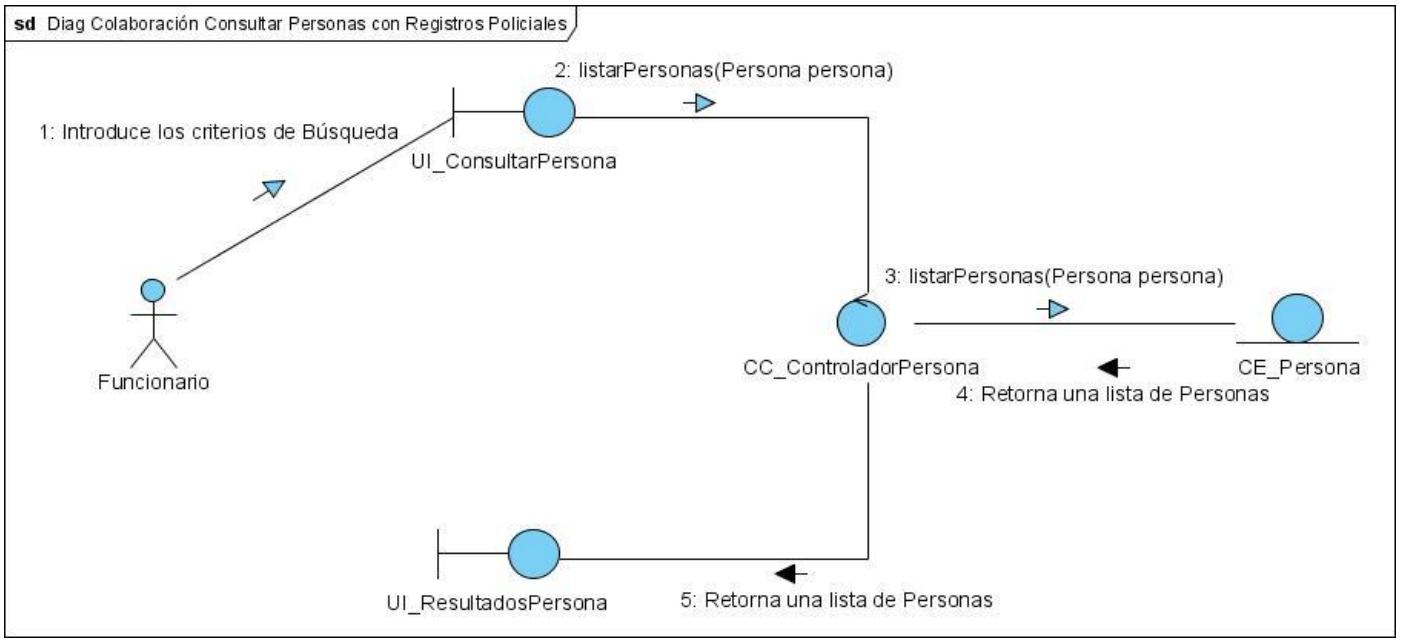


Figura 8. Diagrama de Colaboración CU Consultar Personas con Registros Policiales

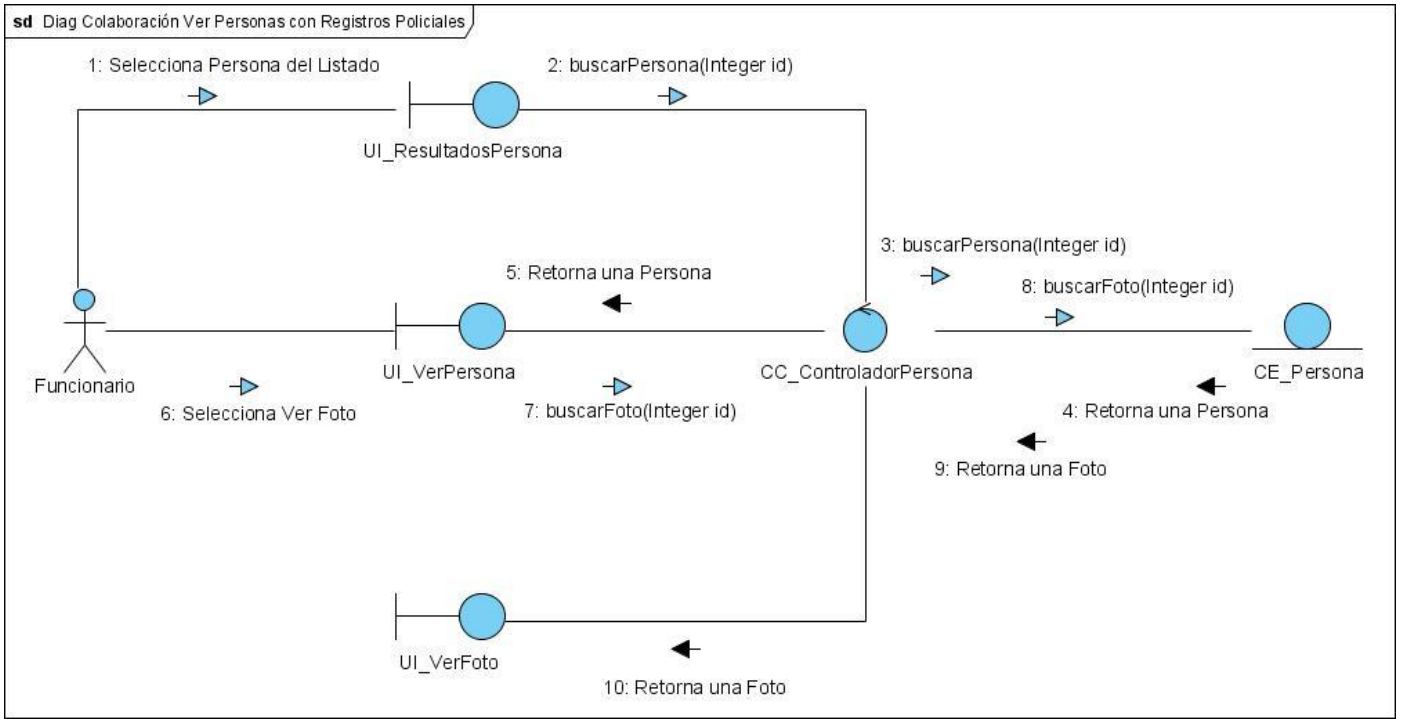


Figura 9. Diagrama de Colaboración Ver Detalles de Personas con Registros Policiales

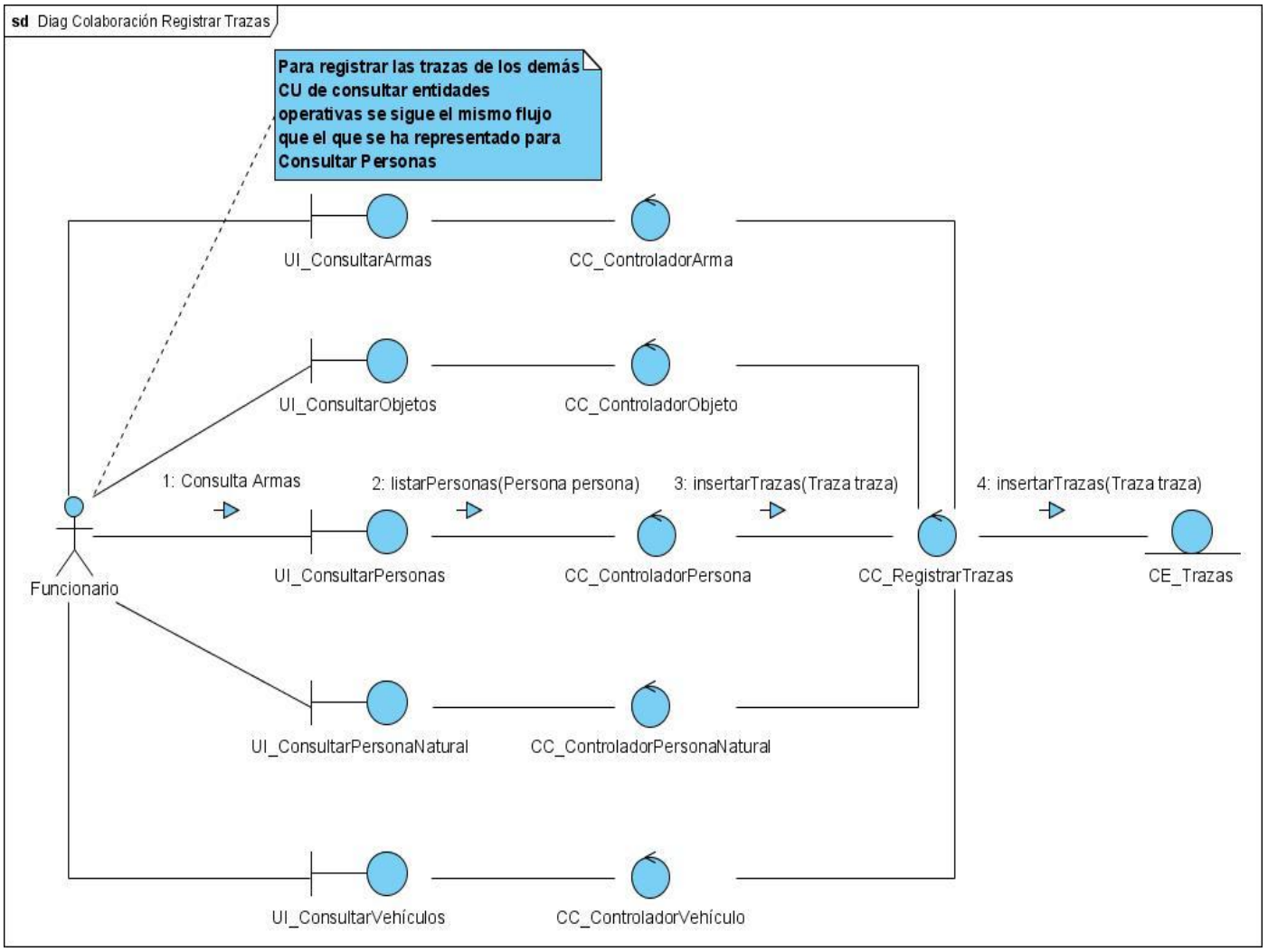


Figura 10. Diagrama de Colaboración CU Registrar Traza Sobre Consulta de Entidad Operativa

Los Diagramas de Colaboración restantes se pueden consultar en el Anexo 2.

2.4 Modelo de Diseño

El modelo de diseño, es un modelo de objetos que describe la realización física de los CU, centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. El diseño es un proceso iterativo a través del cual se traducen los requisitos en una representación del software. Se representa a un alto nivel de abstracción, un nivel que se puede seguir hasta los requisitos específicos de datos, funcionales y de comportamiento.

Con el diseño se pretende construir un sistema que:

- ✓ Satisfaga determinada especificación del sistema.
- ✓ Se ajuste a las limitaciones impuestas por el medio de destino.
- ✓ Respete los requisitos sobre forma, rendimiento, utilización de recursos, costo, etc.

2.4.1 Diagrama de Clases de Diseño

Un diagrama de clases es un tipo de diagrama de estructura estática que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema y los componentes que se encargarán del funcionamiento y la relación entre uno y otro. El diagrama de clases de diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación.

Algunos de los tipos de clases utilizadas durante el diseño son:

- ✓ Clases de servicio: son las clases donde descansa la implementación de la lógica necesaria para coordinar las acciones ejecutadas por las clases del modelo. Generalmente los métodos de las clases de servicio se corresponden con el flujo del CU que implementan.
- ✓ Clases DAO: estas clases tienen como misión encapsular toda la lógica asociada a la persistencia de objetos. Las clases DAO generalmente usan las clases del framework de persistencia para

hacer todas las operaciones sobre las clases persistentes (seleccionar, insertar, actualizar y borrar).

- ✓ Clases del dominio: estas clases son el núcleo de la capa de negocio, expresan los conceptos del negocio, las relaciones que se establecen entre los distintos elementos del negocio, así como su ciclo de vida.

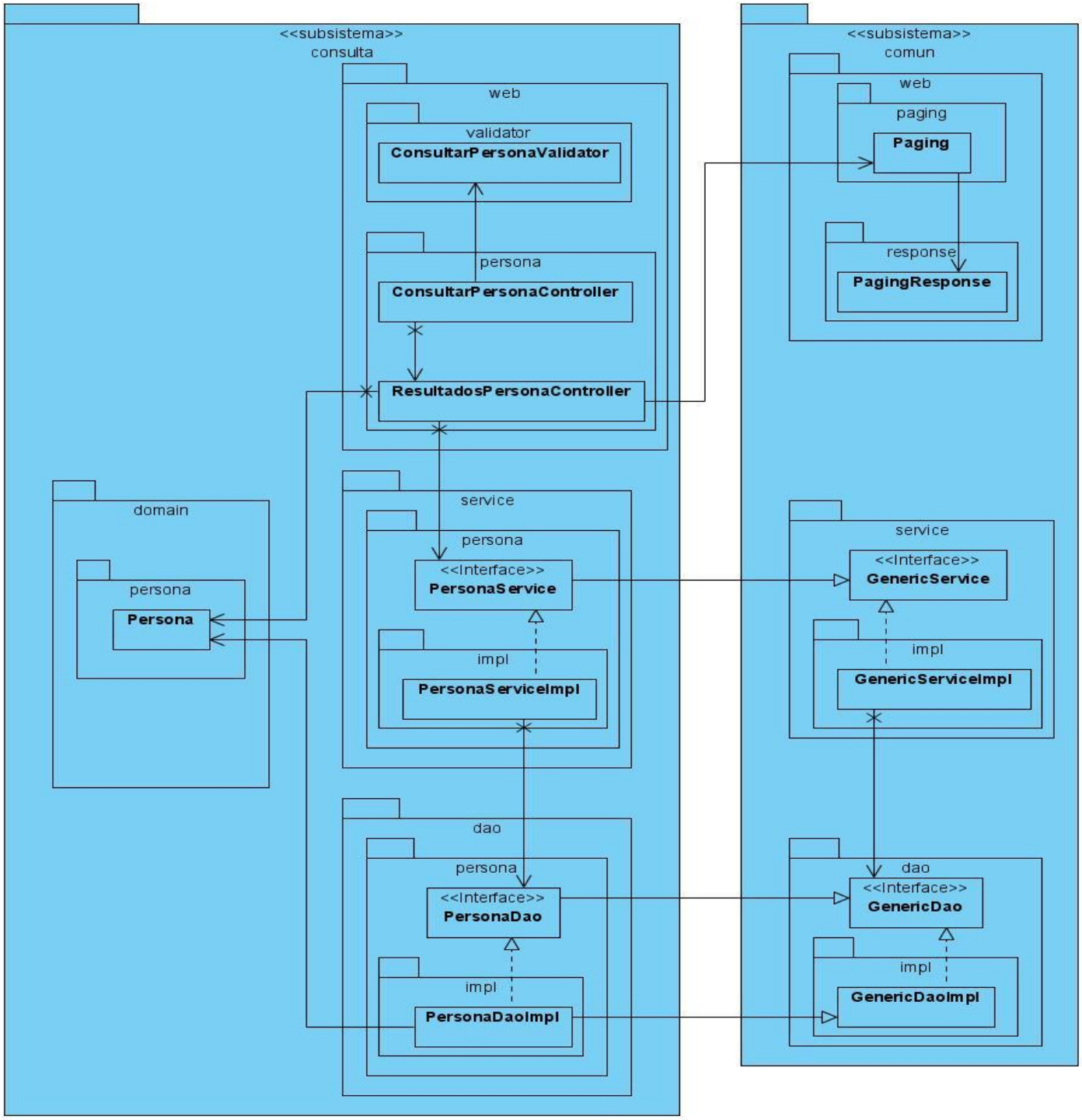


Figura 11. Diagrama de Clases del Diseño CU Consultar Personas con Registros Policiales

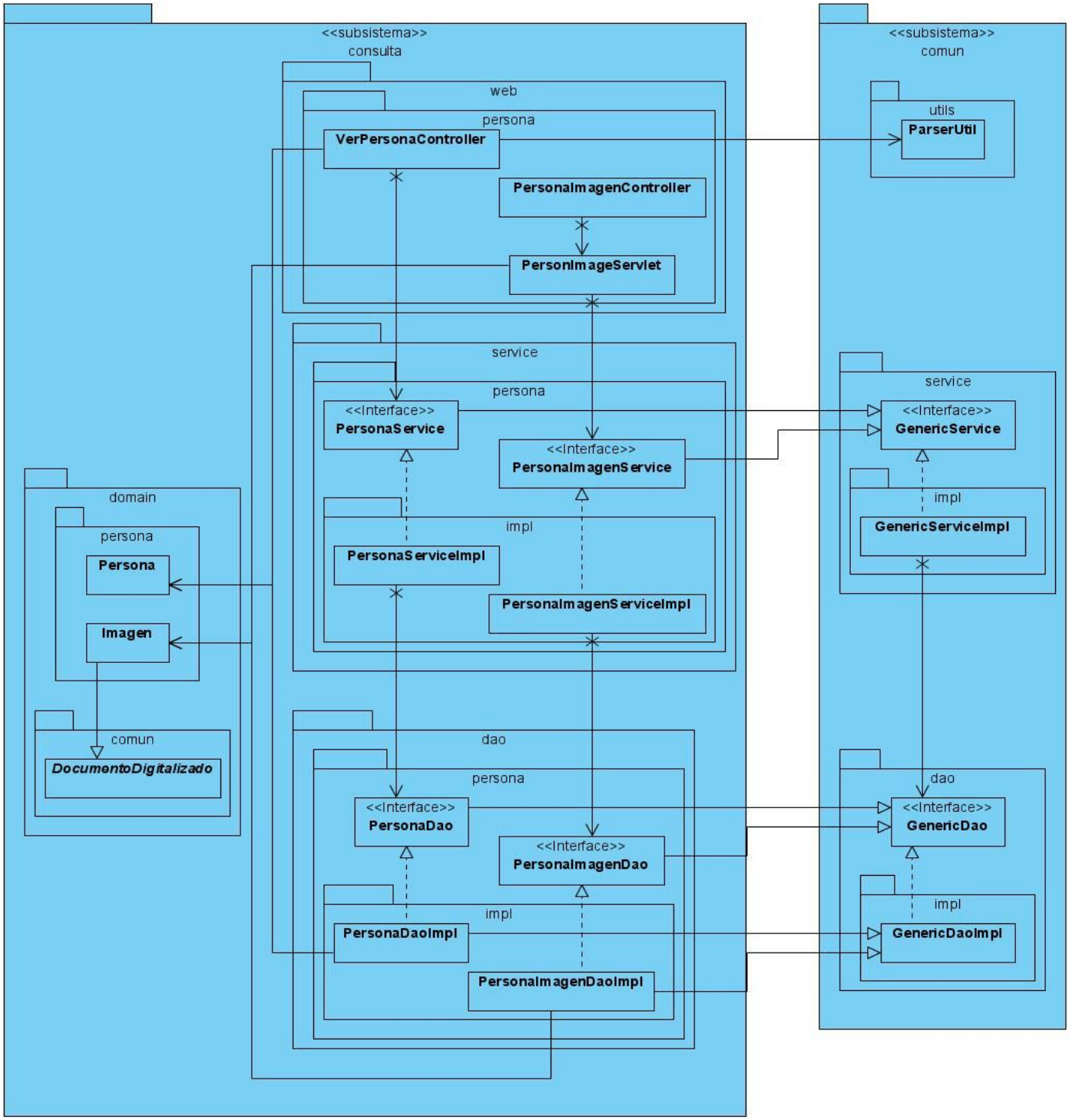


Figura 12. Diagrama de Clases del Diseño Ver Detalles de Personas con Registros Policiales

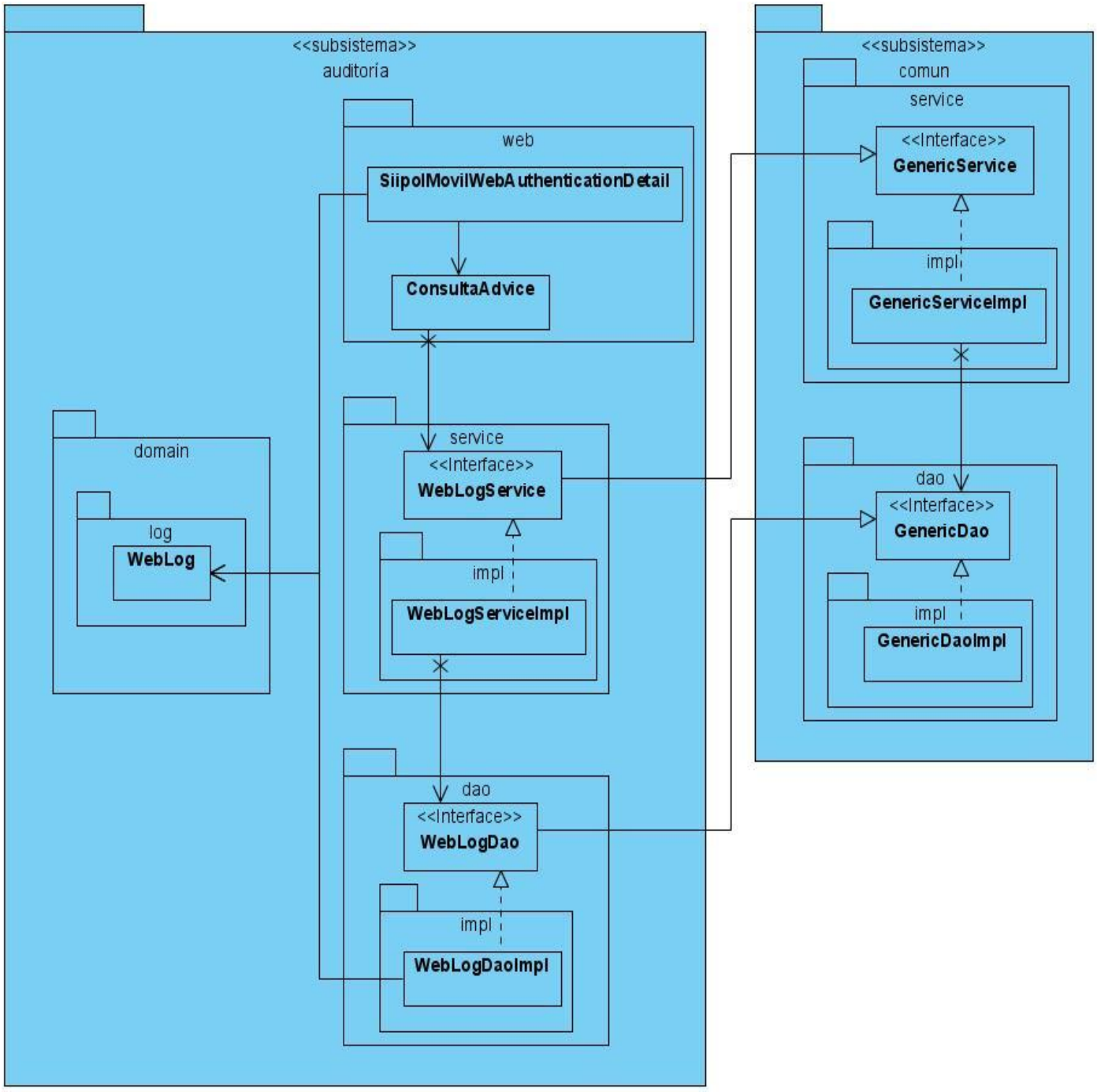


Figura 13. Diagrama de Clases del Diseño CU Registrar Traza Sobre Consulta de Entidad Operativa

Los Diagramas de Clases del Diseño restantes se pueden consultar en el Anexo 3.

2.4.2 Diagrama de Paquetes

Un diagrama de paquetes muestra cómo un sistema está dividido en agrupaciones lógicas, mostrando las dependencias entre ellas. Debido a que un paquete está pensado como un directorio, los diagramas de paquetes suministran una descomposición de la jerarquía lógica de un sistema.

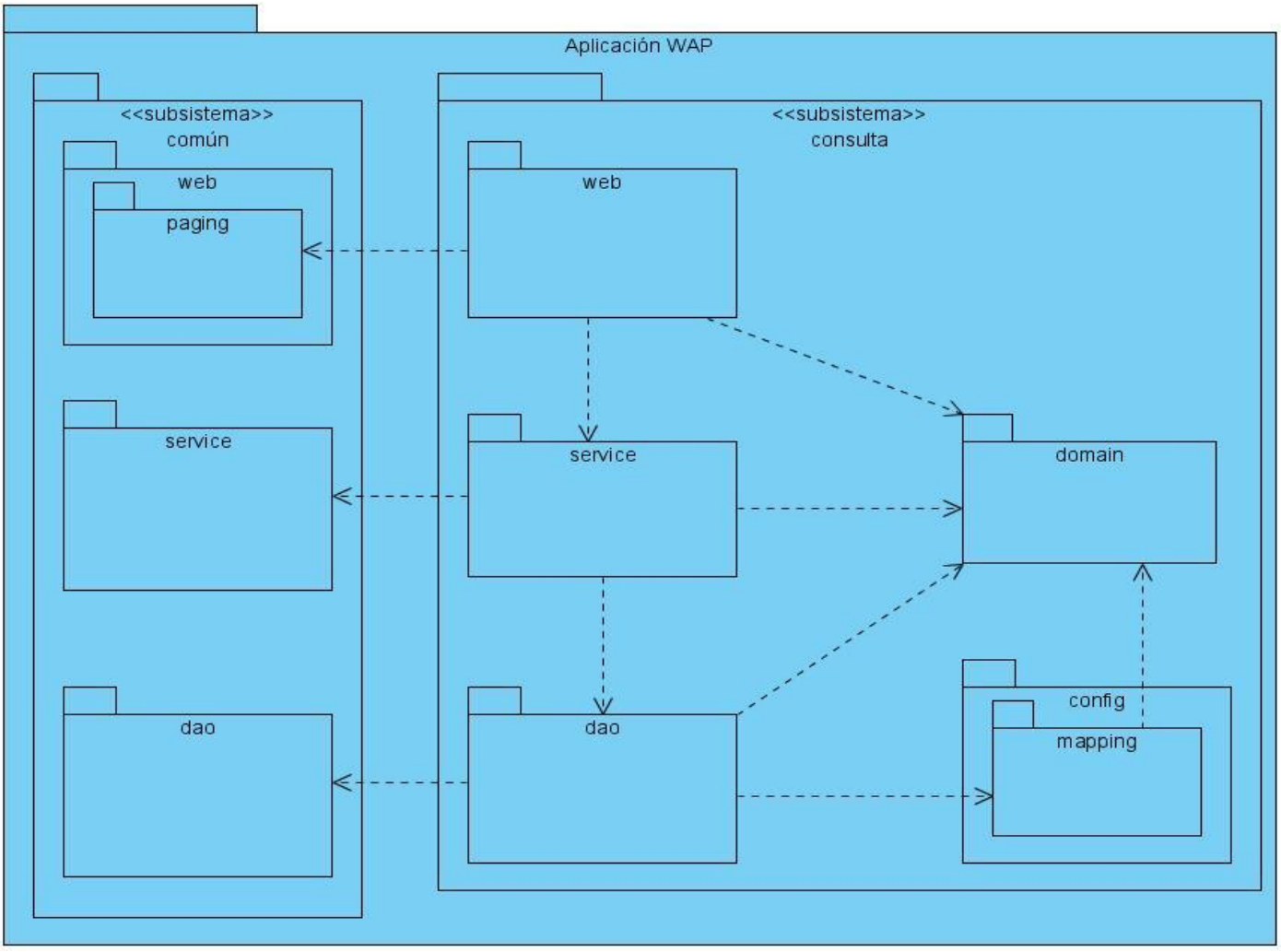


Figura 14. Diagrama de Paquetes del Subsistema de Consulta

2.4.3 Diagrama de Contrato entre Paquetes

Para representar gráficamente cómo las clases y subsistemas interactúan para llevar a cabo las funcionalidades descritas en un caso de uso, se confeccionan los 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 se hace muy engorrosa y no se llegaría a un buen entendimiento de los mismos. Por tal motivo se decide realizar Diagramas de Contrato entre Paquetes que muestren el flujo de mensajes entre paquetes.

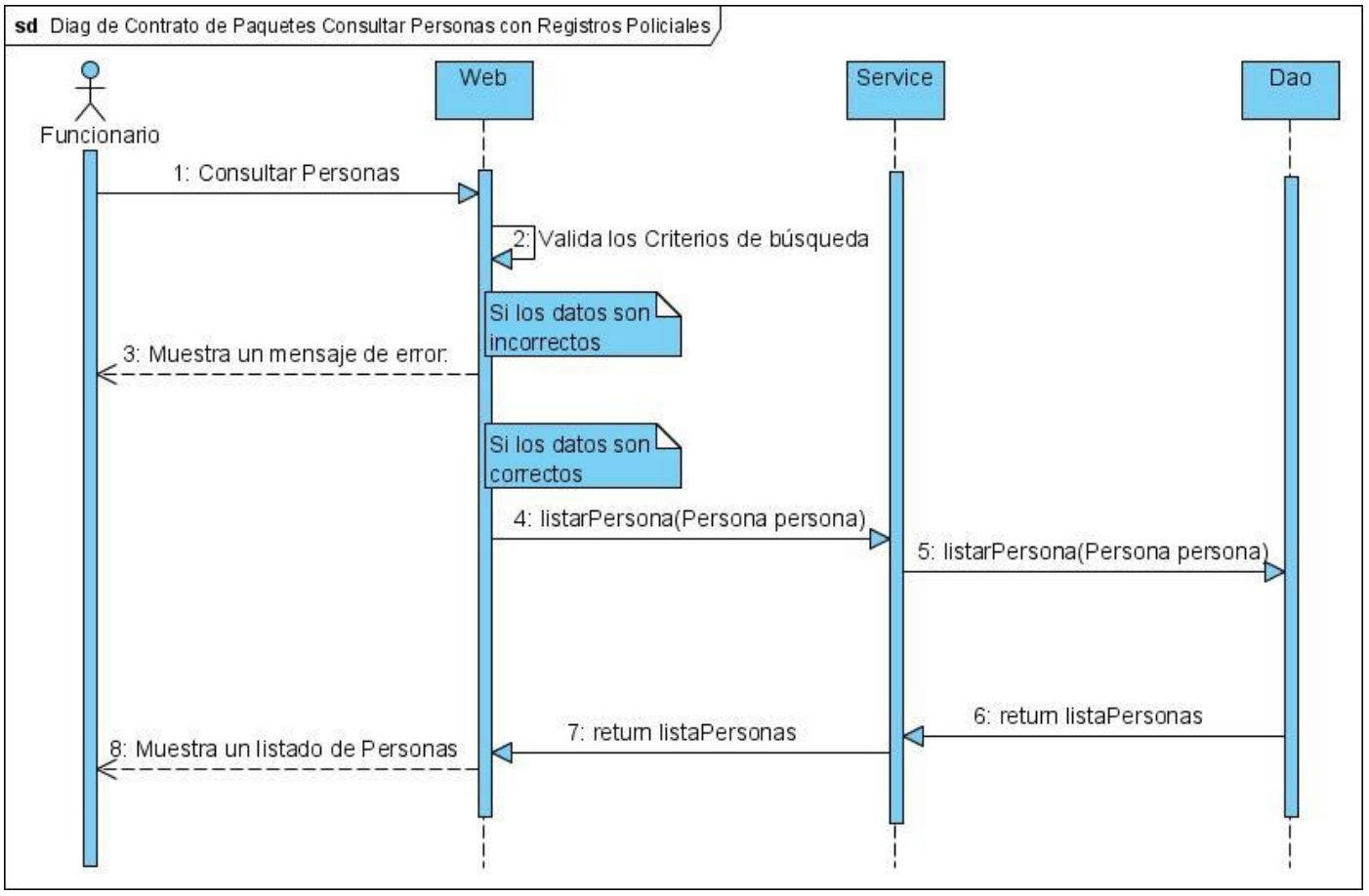


Figura 15. Diagrama de Contrato entre Paquetes CU Consultar Personas con Registros Policiales

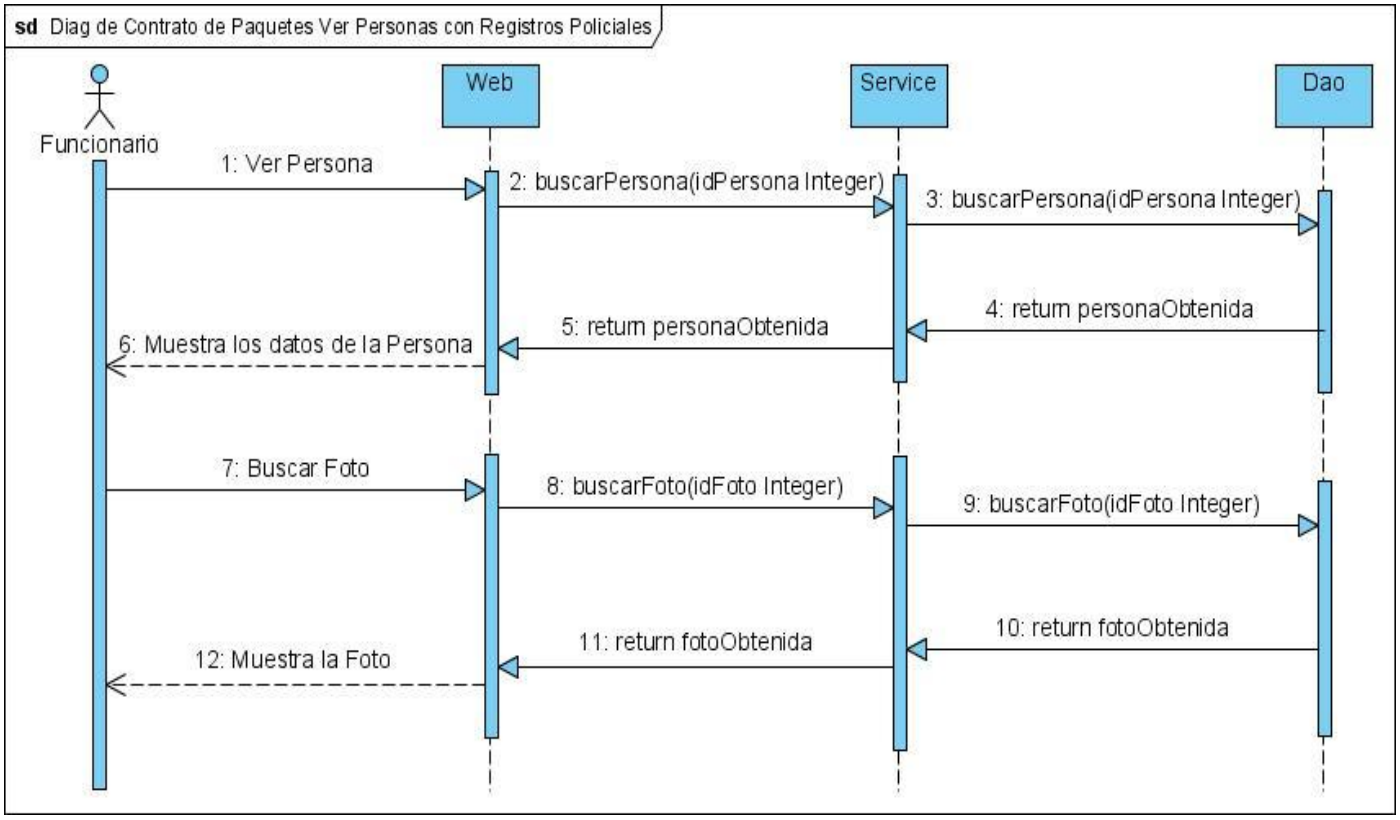


Figura 16. Diagrama de Contrato entre Paquetes CU Ver Detalles de Personas con Registros Policiales

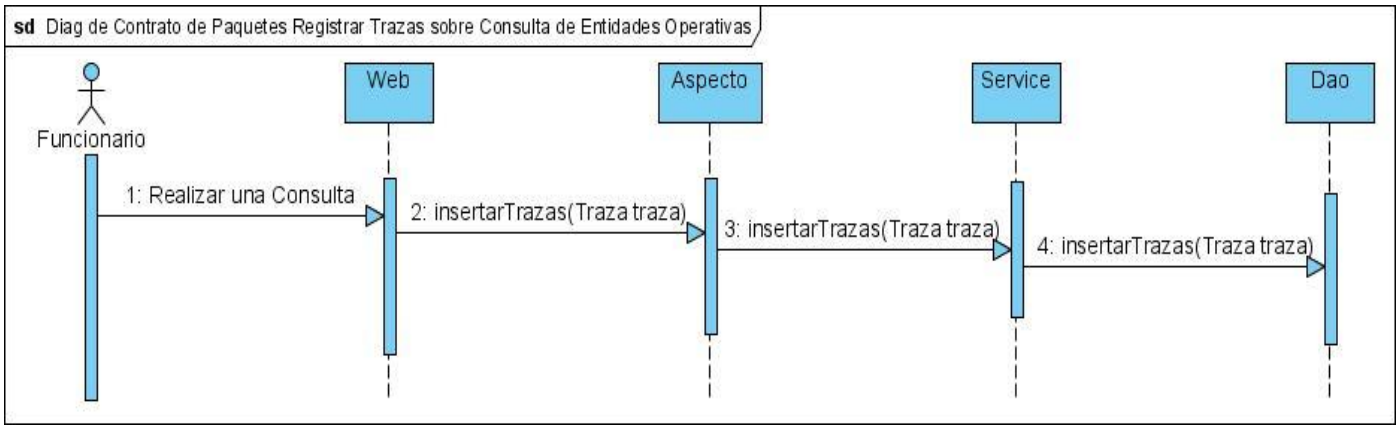


Figura 17. Diagrama de Contrato entre Paquetes CU Registrar Traza Sobre Consulta de Entidad Operativa

Los Diagramas de Contrato entre Paquetes restantes se pueden consultar en el Anexo 4.

2.4.4 Patrones de Diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces, constituyendo una solución a un problema de diseño.

Los patrones de diseño se dividen en tres grupos principales (18):

- ✓ Patrones Creacionales: inicialización y configuración de objetos.
- ✓ Patrones Estructurales: separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan para formar estructuras más grandes.
- ✓ Patrones de Comportamiento: describen la comunicación entre objetos o clases.

Para dar cumplimiento exitosamente a los requerimientos del Portal WAP, se emplearon los siguientes patrones para garantizar un diseño claro y lo más simple posible.

Bajo Acoplamiento

Este patrón le da respuesta a la problemática de soportar bajas dependencias, disminuyendo el impacto del cambio e incrementando la reutilización. El acoplamiento es una medida de la fuerza con que un elemento está conectado, tiene conocimiento y confía en otros elementos. Un elemento con bajo (o débil) acoplamiento no depende de muchos otros. Estos pueden ser clases, subsistemas, entre otros, de ahí la importancia de llevar a cabo el desempeño de este patrón, obteniendo de esta manera una aplicación lo más flexible posible.

Controlador

El patrón Controlador aumenta el potencial de reutilización y asegura que la lógica de la aplicación no se maneje en la capa de presentación. Un controlador es un objeto que no pertenece a la interfaz de usuario y es responsable de recibir o manejar un evento del sistema.

El desempeño de este patrón consiste en 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.

Experto

Consiste en asignar una responsabilidad al experto en información, o sea, a la clase que tiene la información necesaria para realizar la responsabilidad. Un modelo de diseño podría definir miles de clases y una aplicación podría requerir que se realicen gran cantidad de responsabilidades. Si se aplican de la forma correcta, los sistemas tienden a ser más fáciles de entender, mantener y ampliar, existiendo 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 se ha mencionado, da lugar a sistemas más robustos y más fáciles de mantener. Además, se distribuye el comportamiento entre las clases que contienen la información requerida, por tanto, se estimula la definición de clases más cohesivas y ligeras.

Fachada

Utilizado para proporcionar interfaces simples para subsistemas complejos. 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, lo cual trae como consecuencia que al separar al cliente de los componentes del subsistema, se reduzca el número de objetos con los que el cliente trata, facilitando así el uso del subsistema.

El patrón Fachada promueve además un débil acoplamiento entre el subsistema y sus clientes, eliminándose o reduciéndose las dependencias.

Polimorfismo

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

Su aplicación trae como beneficio añadir fácilmente las extensiones necesarias para nuevas variaciones, permitiendo añadir nuevas implementaciones sin afectar a los clientes.

2.4.5 Prototipo de Interfaz

Un prototipo es una visión preliminar del sistema futuro, es un modelo operable, fácilmente ampliable y modificable, que tiene todas las características que hasta el momento debe tener el sistema.

Las ventajas principales de los prototipos son:

- ✓ Posibilidad de cambiar el modelo.
- ✓ Oportunidad para suspender el desarrollo del modelo si no es funcional.
- ✓ Posibilidad de crear un nuevo modelo que se ajuste mejor a las necesidades y expectativas del usuario.

Para el desarrollo de la aplicación se ha decidido crear una interfaz de usuario amigable, sencilla y con un apropiado patrón de navegación para dispositivos móviles. Los prototipos propuestos se rigen por un conjunto de patrones de diseño web conocidos como Buenas Prácticas para el diseño web Móvil, especificado por el W3C.

Para lograr una aplicación que satisfaga los requerimientos del cliente, se tuvo en cuenta siempre colocar al final de cada página un menú básico de navegación, donde el usuario pueda volver cada vez que desee a la página de inicio o salir de la aplicación. Los resultados de las consultas que presenten más de 5 coincidencias serán paginados, mostrando en la parte superior de la página el total de elementos encontrados y en la parte inferior los enlaces de navegación: ir hacia delante, hacia atrás o ir al inicio de

los resultados mostrados. Las fotos de las personas se mostrarán siempre que el dispositivo destino las soporte y sus dimensiones y formatos se cambiarán a los adecuados para su mejor visualización en los navegadores.

A continuación se muestran los prototipos de las interfaces de los casos de uso Consultar Personas con Registros Policiales y Ver Detalles de Personas con Registros Policiales. El resto de las interfaces se pueden encontrar en el Anexo 5.

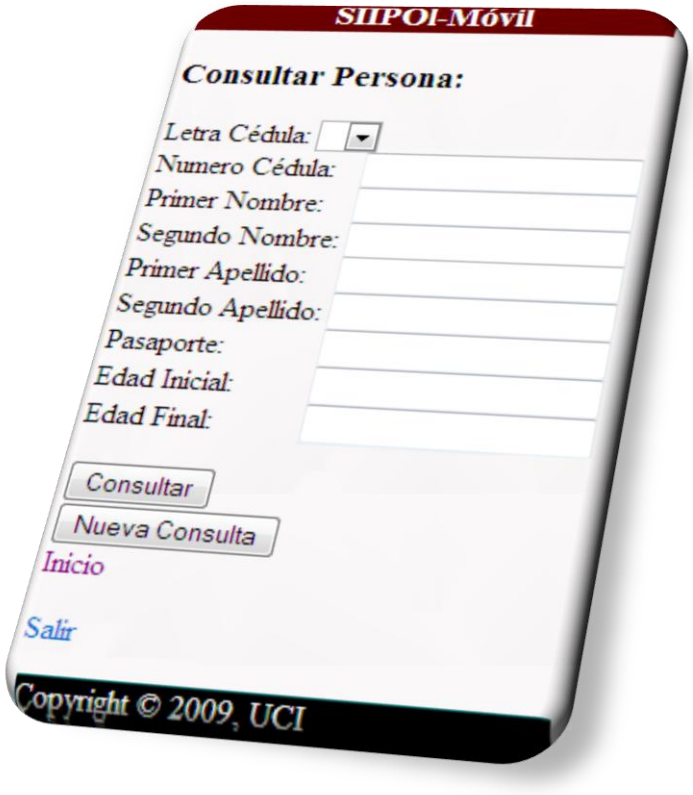


Figura 18. Consultar Personas con Registros Policiales

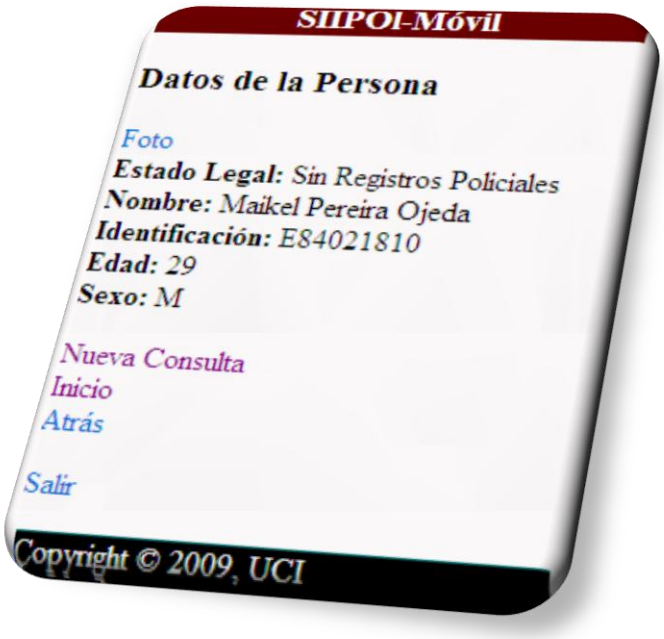


Figura 19. Ver Detalles de Personas con Registros Policiales

2.5 Modelo de Despliegue

Un modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. El modelo de despliegue se utiliza como entrada fundamental en las actividades de diseño e implementación.

2.5.1 Diagrama de Despliegue

Es un tipo de diagrama UML que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes.

Elementos que lo componen:

- ✓ Procesadores: nodos que tienen capacidad de procesamiento (computadoras por lo general).

- ✓ Dispositivos: nodos que no tienen capacidad de procesamiento.
- ✓ Protocolos: estándares que deben estar implementados en la red, para efectuar la comunicación entre los diferentes elementos.

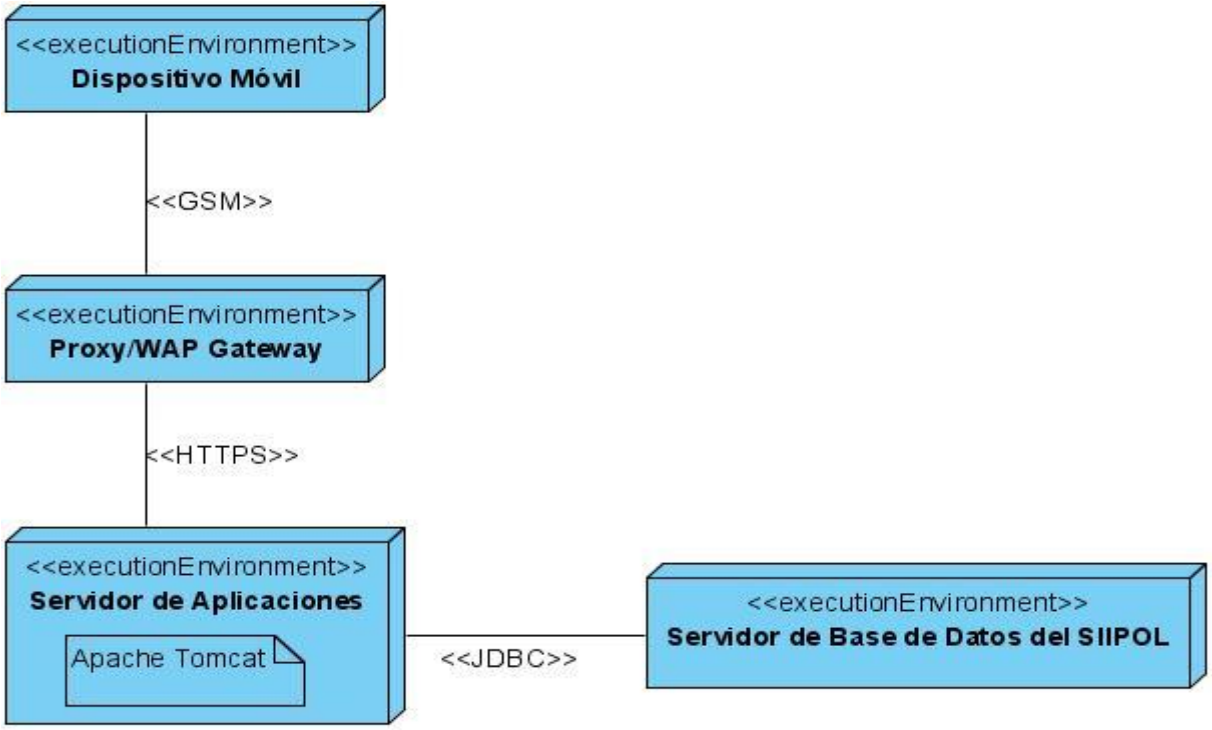


Figura 20. Diagrama de Despliegue

2.6 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, permiten además, describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan esos elementos entre sí.

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.

2.6.1 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. Generalmente las clases persistentes coinciden con los conceptos de información que se usan en el sistema. También se les denomina en ocasiones: clases de dominio.

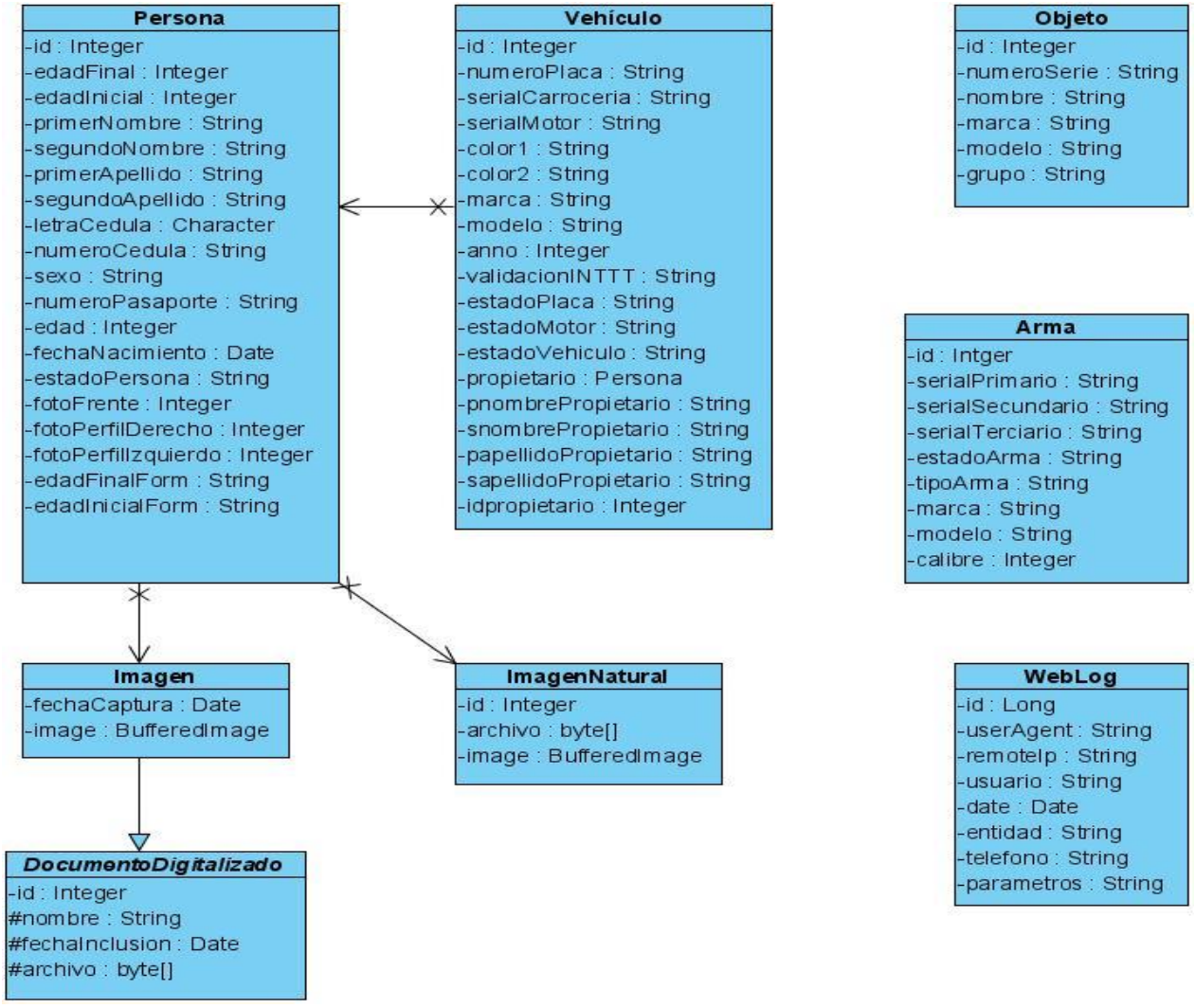


Figura 21. Diagrama de Clases Persistentes

El Diagrama de tablas del Modelo Relacional del Portal WAP se puede encontrar en el Anexo 6.

2.7 Modelo de Implementación

Consiste en una visión general de lo que tiene que ser implementado. En el modelo de implementación se generan una serie de artefactos que constituyen la composición física de la implementación del sistema como son los diagramas de subsistemas y componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares, donde se detalla esencialmente la relación que existe desde las clases y componentes del modelo de diseño a subsistemas y componentes físicos. En este modelo se ajustan los subsistemas formados por los elementos de implementación y se definen las dependencias entre subsistemas.

2.7.1 Diagrama de Subsistemas de Implementación

Organiza los artefactos del modelo de implementación en partes manejables y describe cómo se implementan los componentes, agrupándolos en subsistemas organizados en capas y jerarquías, señalando las dependencias entre éstos.

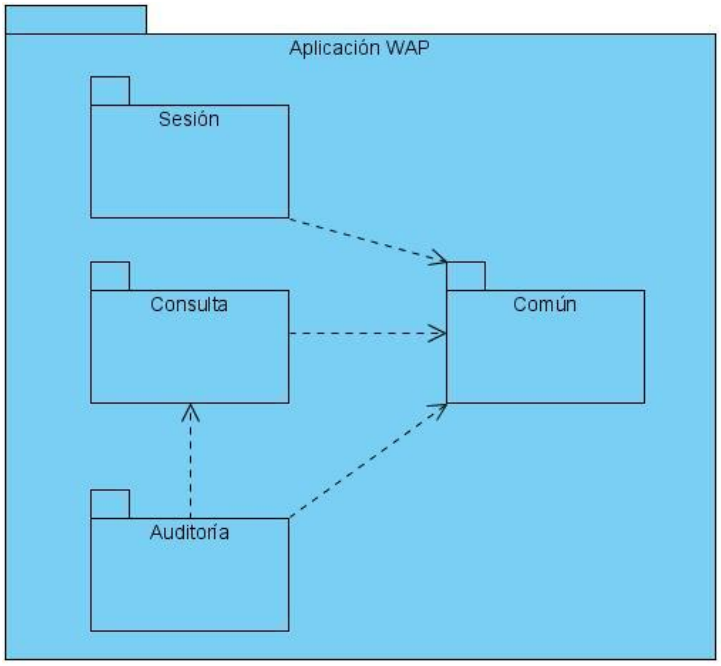


Figura 22. Diagrama de Subsistemas del Portal WAP

2.7.2 Diagrama de Componentes

Un diagrama de componentes ilustra los fragmentos de software y los controladores embebidos que conforman un sistema, permiten editar de forma segura todo el contenido que llevarán los archivos de datos de la aplicación. Tienen un nivel de abstracción más elevado que un diagrama de clases. Usualmente un componente se implementa por una o más clases (u objetos) en tiempo de ejecución. Estos diagramas se utilizan para describir la vista de implementación estática de un sistema.

A continuación se representan los diagramas de componentes referentes al subsistema Consulta, se elaboran dos diagramas diferentes, uno para representar los archivos Java, interfaces, XML de configuración y propiedades que son utilizadas para la programación del lado del servidor, y el segundo para representar la dependencia entre los archivos o páginas JSP.

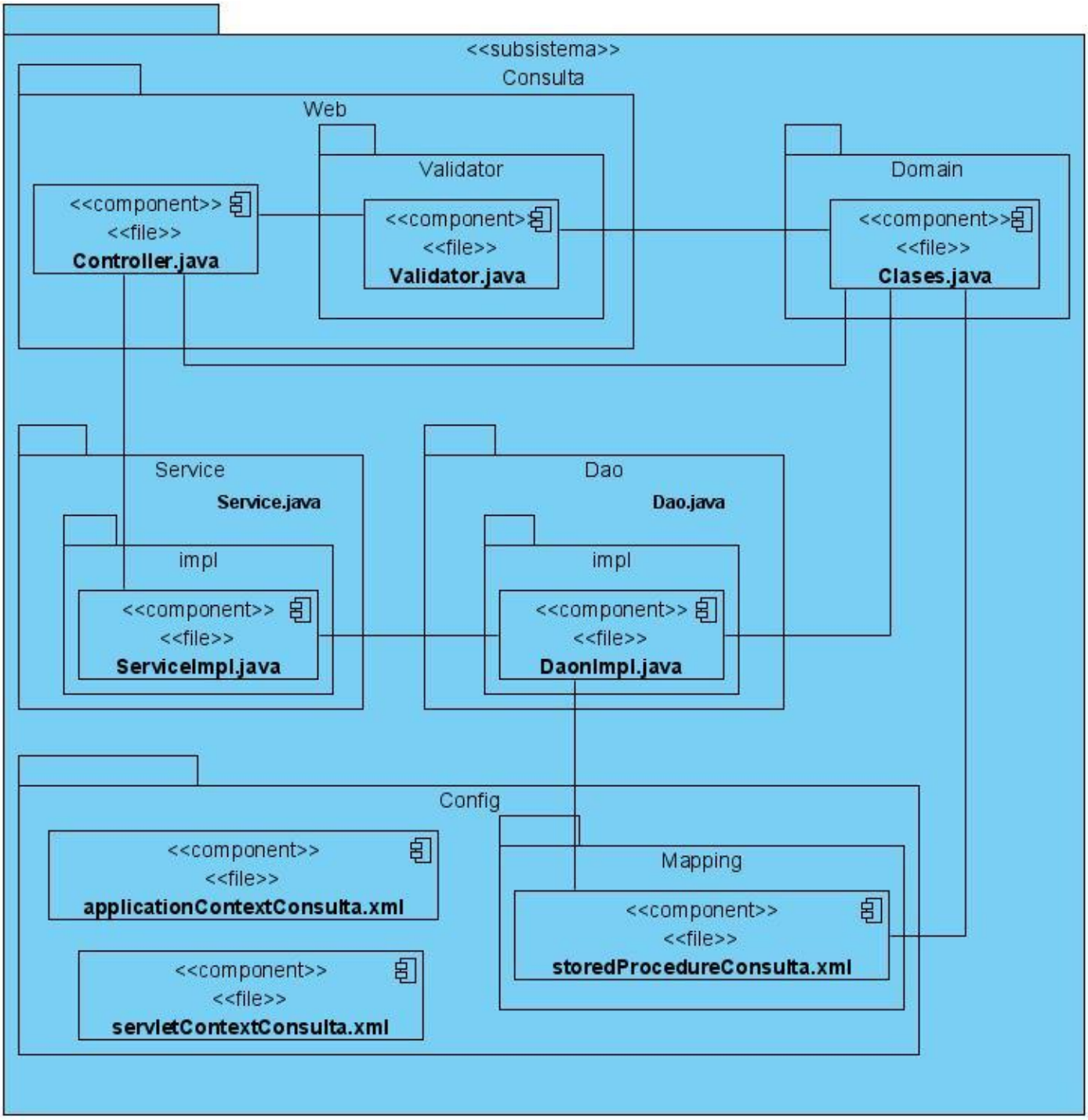


Figura 23. Diagrama de Componentes 1

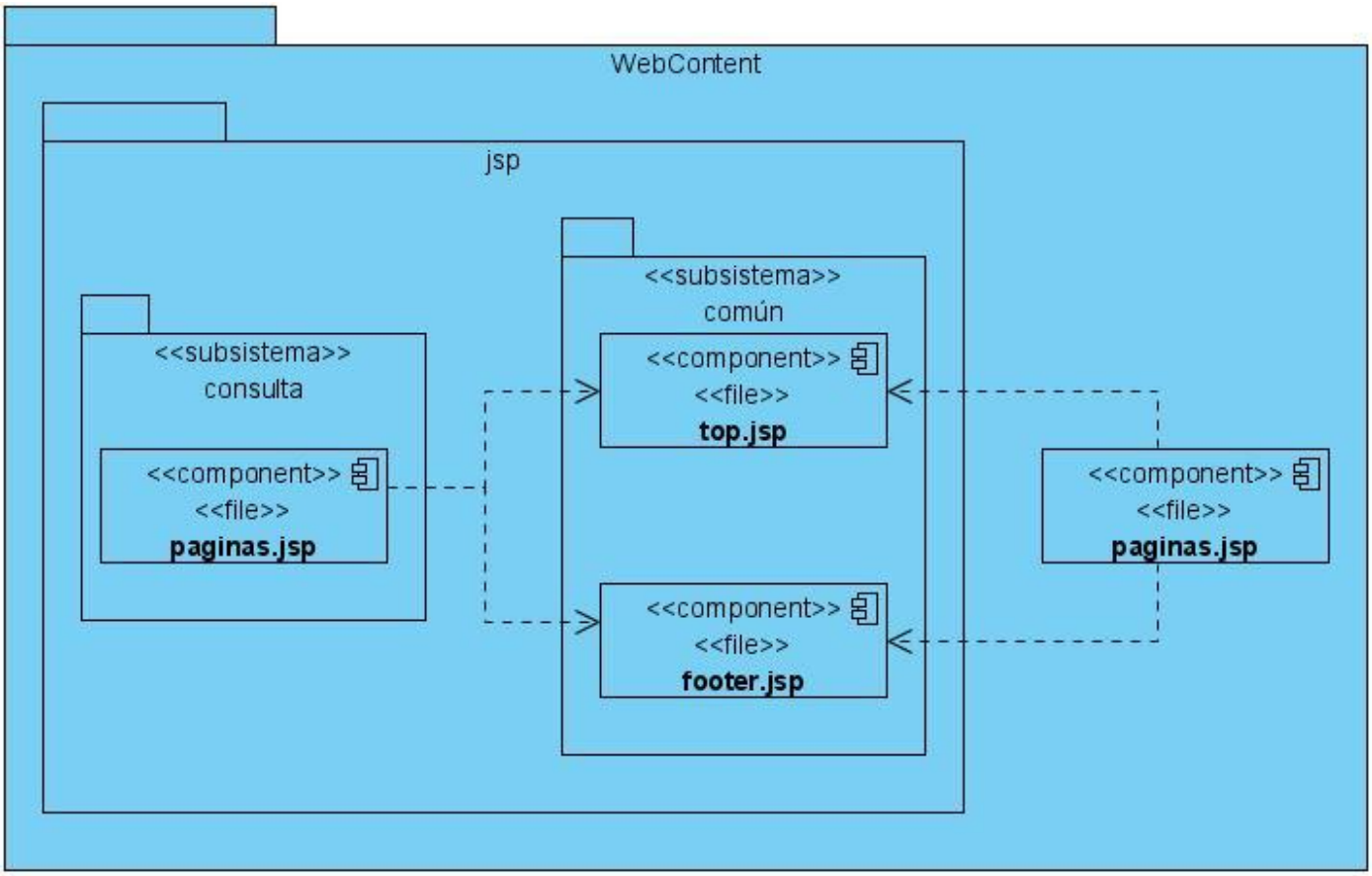


Figura 24. Diagrama de Componentes 2

2.8 Estándares de Codificación

Teniendo en cuenta el lenguaje de programación y los frameworks utilizados se hace referencia a algunas pautas significativas que se tienen en cuenta en la solución del problema, el resto de los patrones a seguir se incluyen en el documento de estilo de código para el proyecto CICPC (21).

Convenciones de nombres

Todo lo referido a los nombres se rige por la convención de código de Java. Como principio todos los nombres serán en español exceptuando aquellos que correspondan al nombre de un patrón conocido, ejemplo: ServiceLocator, Facade, Builder, etc.

Clases

Los nombres de las clases deben ser sustantivos, en el caso de ser compuestos tendrán la primera letra de cada palabra que lo forme en mayúscula. Los nombres de las clases deben ser simples y descriptivos. Además, se deben usar palabras completas, evitar acrónimos y abreviaturas (exceptuando los casos en los que la abreviatura sea mucho más conocida que el nombre completo, como URL o HTML).

Métodos

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

Ejemplo: consultarPersona().

2.9 Conclusiones

En este capítulo se han expuesto los principales artefactos del análisis, diseño e implementación del sistema propuesto. Además, se han descrito los patrones de diseño utilizados y los prototipos de interfaz de usuario, llegando a la conclusión de que tanto el análisis como el diseño realizados con la calidad requerida, proporcionan rapidez y precisión en el momento de la implementación.

Capítulo 3

Validación de la propuesta de solución

3.1 Introducción

En el presente capítulo se realiza un estudio de los diferentes métodos y tipos de pruebas, con el objetivo de llevar a cabo la ejecución de las mismas, garantizando de esta forma la calidad del sistema y el total cumplimiento de los requisitos establecidos con el cliente.

El proceso de pruebas es clave a la hora de detectar errores o fallas. Conceptos como estabilidad, escalabilidad, eficiencia y seguridad se relacionan con la calidad de un producto bien desarrollado. Las aplicaciones de software han crecido en complejidad y tamaño y por consiguiente en costos, por lo que se hace necesario que el proceso de pruebas se lleve a cabo con calidad y eficiencia, logrando la aceptación del producto por parte del usuario final.

3.2 Métodos de prueba

La prueba es un elemento crítico para la garantía de la calidad del software, es el proceso que permite verificar y revelar la calidad de un producto de software. Son utilizadas para identificar posibles fallos de implementación, calidad o usabilidad de un programa. Básicamente es una fase en el desarrollo de software consistente en probar las aplicaciones construidas. La etapa de pruebas implica:

- ✓ Verificar la interacción de componentes.
- ✓ Comprobar la integración adecuada de los componentes.
- ✓ Verificar que todos los requisitos se han implementado correctamente.
- ✓ Identificar y asegurar que los defectos encontrados se han corregido antes de entregar el software al cliente.

- ✓ Diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo.

Las pruebas de software consisten en una serie de actividades en las que un sistema o componente es ejecutado bajo determinadas condiciones o requerimientos, los resultados son observados y registrados, realizando una evaluación del sistema o del componente probado. Su principal objetivo es evaluar o valorar la calidad del producto.

Las pruebas se enfocan sobre la lógica interna del software y las funciones externas, para lo cual se definen los siguientes métodos:

Prueba de Caja Blanca

Las pruebas de caja blanca realizan un seguimiento del código fuente según se van ejecutando los casos de prueba, de manera que se determinan de manera concreta las instrucciones en las que existen errores. Requieren del conocimiento de la estructura interna del programa y son derivadas a partir de las especificaciones internas del código.

Prueba de Caja Negra

Pruebas que se llevan a cabo sobre la interfaz de usuario, se centran principalmente en los requisitos funcionales del software. Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. Examinan aspectos del modelo, principalmente del sistema, sin tener en cuenta la estructura interna del software.

3.3 Niveles de pruebas

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.

Para lograr una mayor efectividad, las pruebas de software se realizan a diferentes niveles:

Pruebas de Unidad

La mayoría de las arquitecturas de software modernas incorporan la modularidad como uno de sus elementos esenciales, lo cual posibilita el trabajo de varios programadores en distintos módulos sin que esto signifique un problema. De la misma forma es posible que los programadores ejecuten pruebas locales sobre sus módulos, a fin de comprobar el buen funcionamiento (autónomo) de lo que han realizado hasta el momento. A este tipo de prueba se le conoce como pruebas de unidad, las cuales centran su proceso de verificación en la menor unidad de software, el módulo.

El objetivo de estas pruebas es aislar cada parte del programa y mostrar que las partes individuales son correctas. Estas pruebas permiten llegar a la siguiente fase con un alto grado de seguridad de que el código está funcionando correctamente, facilitando de esta forma las pruebas de integración.

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 las pruebas de todos los elementos unitarios que componen un proceso, hechas en conjunto, de una sola vez. Consisten en realizar pruebas para verificar el funcionamiento en conjunto de diferentes partes del software.

Conjuntamente con el proceso de integración se aplican las pruebas de regresión, para asegurar que los módulos causantes de fallas fueron efectivamente corregidos y que no se introdujeron nuevos errores al tratar de solucionar alguno. Estas pruebas de regresión consisten en aplicar el mismo proceso de pruebas planificado originalmente, repitiéndose cada vez que se identifica un resultado erróneo y se le de solución.

3.4 Resultados de las pruebas

Para evaluar la solución desarrollada se planificaron 2 iteraciones de pruebas en las cuales se probó la aplicación con un alto grado de detalle. Se abarcaron los métodos y niveles de prueba expuestos anteriormente en cada una de las iteraciones, arrojando resultados visibles, los cuales validan la calidad del producto construido.

Durante el proceso de desarrollo de la aplicación se llevaron a cabo pruebas unitarias para ir comprobando el correcto funcionamiento del avance de la solución, las cuales fueron dirigidas por los mismos desarrolladores. La automatización de las pruebas de unidad fue un aspecto fundamental para agilizar el proceso, para lo cual fue utilizado el framework JUnit. Estas pruebas no se planificaron ni se registraron sus resultados, se fueron haciendo a medida que la solución se desarrollaba.

Durante las pruebas se documentaron todas las No Conformidades surgidas, dándole la categoría de Alta, Media o Baja según su impacto en el negocio y el sistema. A continuación se muestra el resultado de las pruebas en cada una de las iteraciones.

Pruebas Internas

Pruebas realizadas al sistema por el equipo de calidad interna del proyecto. Se realizaron con el fin de entregar un producto libre de errores al tercero encargado de validar la factibilidad del sistema. Se centraron en el cumplimiento de las funcionalidades descritas en el listado de requisitos y de casos de uso.

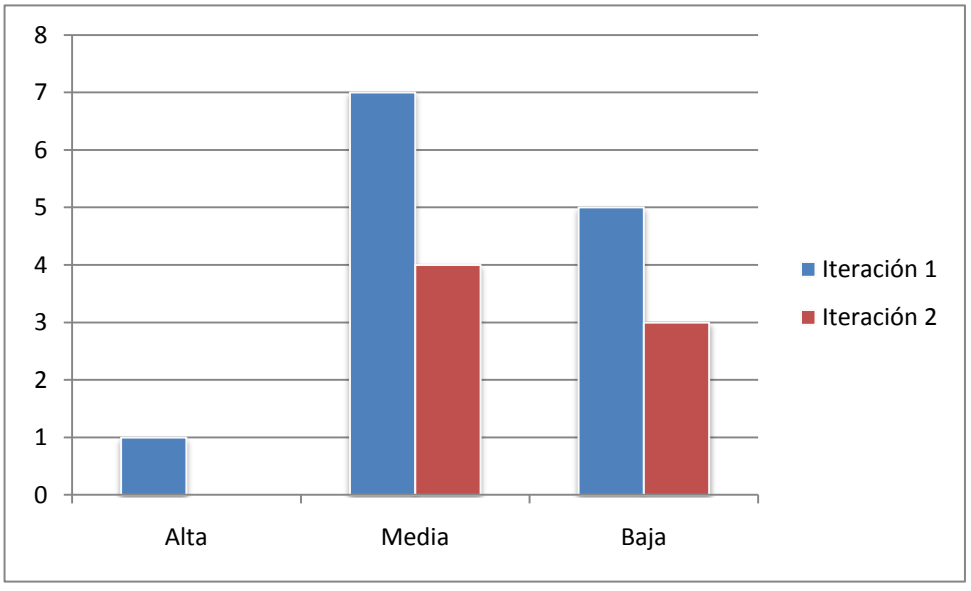


Figura 25. No Conformidades detectadas durante las Pruebas Internas

Pruebas Cruzadas

Realizadas al sistema por el resto de los equipos de desarrollo del proyecto. Se realizaron con el fin de encontrar la mayor cantidad de errores posibles en término de validaciones, pautas definidas por arquitectura de información, formato de los campos, entre otras.

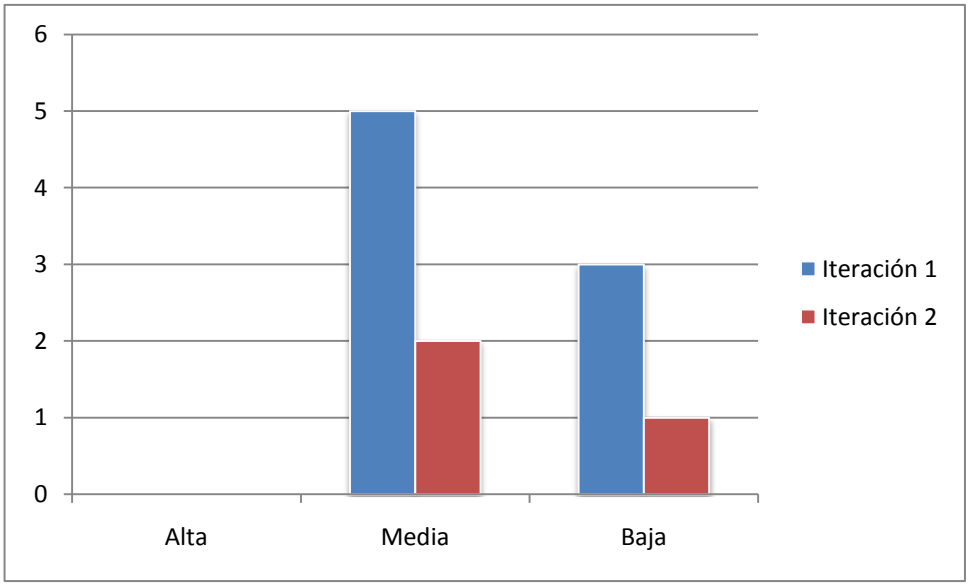


Figura 26. No Conformidades detectadas durante las Pruebas cruzadas

Luego de concluida cada iteración de pruebas se analizaron, por parte del equipo de desarrollo, las No Conformidades encontradas para determinar cuáles realmente constituyeron defectos del sistema o presentaban alguna variación según la descripción de los casos de uso y necesitaban ser solucionadas. Las pruebas se realizaron de forma iterativa e incremental, comprobando en cada iteración que hubiesen sido corregidos los errores detectados en la iteración anterior, lo que contribuyó a mejorar la calidad y eficiencia del software, promoviendo así la satisfacción del cliente sobre el producto.

A continuación se muestra un gráfico resumen del total de No Conformidades encontradas en cada iteración de pruebas, evidenciándose la correcta evolución del software en cuanto a la reducción del número de defectos detectados.

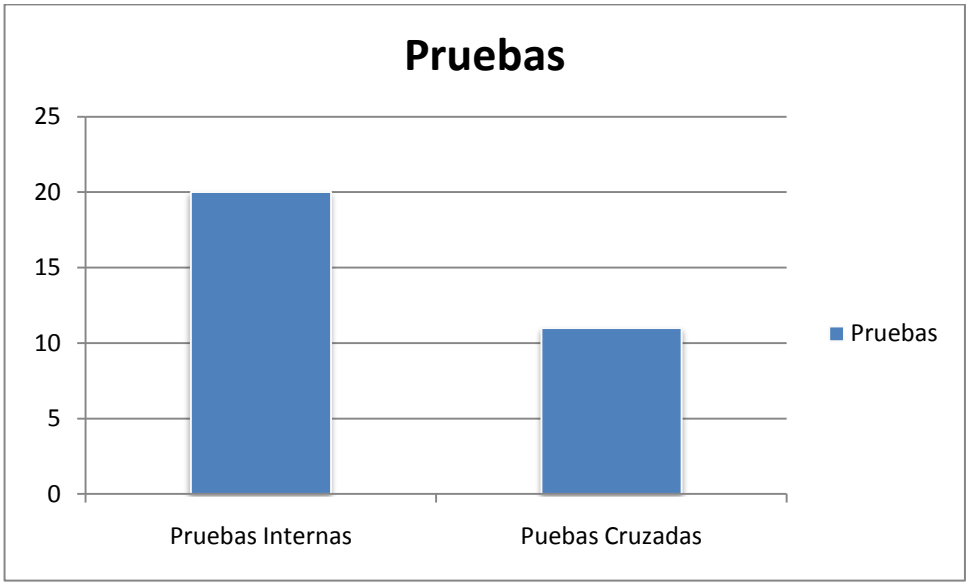


Figura 27. Resumen general de las pruebas

3.5 Conclusiones

En el capítulo se realizó un estudio de los métodos y niveles de pruebas comúnmente aplicados cuando se va a probar un determinado producto de software, con el objetivo de lograr una aplicación libre de errores y con la calidad requerida por el cliente.

Tras el análisis de los resultados expuestos en este capítulo, se demuestra la solidez de la propuesta, basada en el paso por 2 iteraciones de pruebas las cuales denotaron un número decreciente de No Conformidades encontradas, lo cual se corresponde con la calidad de la aplicación construida.

Conclusiones

Con el propósito de darle cumplimiento al objetivo general y a la problemática planteada en el presente trabajo, se han llevado a cabo satisfactoriamente cada una de las tareas que fueron trazadas al comienzo del mismo.

Durante el desarrollo de la investigación, se profundizó en el conocimiento de las herramientas utilizadas para la creación de portales para dispositivos móviles, así como en la metodología utilizada para guiar el proceso de desarrollo del software, con la cual se obtuvo una abundante documentación y se generaron los artefactos necesarios para validar la solución con el cliente.

El diseño y la implementación se rigieron por los estándares internacionales establecidos para la elaboración de portales con capacidad de ser visualizados de manera correcta en dispositivos móviles. Se puede concluir por tanto, que se ha cumplido satisfactoriamente el objetivo trazado para este trabajo, enfatizando en los siguientes puntos:

- ✓ Se eligieron las tecnologías adecuadas para desarrollar el sistema, cumpliendo con los requisitos planteados por el cliente.
- ✓ Se implementaron las funcionalidades para la consulta de entidades operativas, lo cual garantizará el acceso a información desde lugares donde no se pueda acceder al SIIPOL.
- ✓ Se logró adaptar los contenidos de manera eficiente, alcanzando su correcta visualización mediante el Portal WAP.

Teniendo en cuenta los resultados obtenidos, se concluye que los objetivos presentados al inicio de la investigación han sido cumplidos en su totalidad, garantizando de esta forma una mayor eficiencia en las respuestas operativas de los funcionarios del Cuerpo de Investigaciones Científicas, Penales y Criminalísticas de Venezuela.

Recomendaciones

- ✓ Adicionar al portal la funcionalidad de consultar los vehículos en INTTT (Instituto Nacional de Tránsito y Transporte Terrestre).
- ✓ Mantener actualizado el repositorio de descripción de dispositivos móviles, debido a la constante evolución de los teléfonos celulares y el surgimiento de nuevos modelos o adición de nuevas características a modelos ya existentes.
- ✓ Estudiar la posibilidad de realizar un sistema similar para los órganos del MININT en Cuba.
- ✓ Poner este trabajo a disposición de la comunidad universitaria como referencia para proyectos similares, teniendo en cuenta las reglas de confidencialidad establecidas.

Bibliografía

1. **López, Jorge Blanco.** Cnice. [En línea] 12 de junio de 2006. http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=354.

2. Teléfonos-móviles. [En línea] 21 de mayo de 2003. http://www.telefonos-moviles.com/articles/item.asp?ID=20.

3. Masadelante. [En línea] http://www.masadelante.com/faqs/wap.

4. Protocolo Inalàmbrico de Aplicaciones. [En línea] http://www.electronguia.com.ar/Comunicacion/Informacion/wap/protocoloWAP.htm.

5. WebEstilo. [En línea] http://www.webestilo.com/wml/.

6. W3C. [En línea] 8 de abril de 2008. http://www.w3c.es/Noticias/2008/04/08/ltima-convocatoria-api-simple-para-repositorio-de-descripcin-de-dispositivo/.

7. AccNera. [En línea] http://www.accnera.com/webmovil/webmovil-adaptacion.html.

8. **Pressman, Roger.** *Ingeniería de Software un Enfoque Práctico.* 2006.

9. **García, Joaquin.** IngenieroSoftware. [En línea] 7 de mayo de 2005. http://www.ingenierosoftware.com/analisisydiseno/uml.php.

10. eleZeta . [En línea] 27 de agosto de 2004. http://elezeta.net/2004/08/27/extreme-programming-xp/.

11. Proyectosagiles. [En línea] http://www.proyectosagiles.org/que-es-scrum.

12. **Ecuador, Pontificada Universidad Católica del.** *Herramientas Case.*

13. LuAuf. [En línea] 13 de mayo de 2008. http://luauf.com/2008/05/13/entornos-de-desarrollo-integrado-para-java/.

14. Educación. [En línea] 17 de enero de 2008. http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=548.

15. Elwebmaster. [En línea] 7 de noviembre de 2007. http://www.elwebmaster.com/referencia/api-interface-de-programacion-de-aplicaciones.

16. Codebox. [En línea] http://www.codebox.es/glosario.

17. **Agüero, Martín.** *Introducción a Spring Framework.* 2007.

18. **Herrera, Cristhian.** Informática Profesional. [En línea] 16 de agosto de 2007. http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=hibernateVSEJB3.

19. **Albet, S.A.** *Especificaciones Suplementarias, SIIPOL Móvil.* 2009.

20. **Tedeschi, Nicolás.** Patrones de Diseño. [En línea] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

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

22. **Martínez, Evelio.** Veliux. [En línea] http://www.eveliux.com/mx/la-evolucion-de-la-telefonia-movil.php.

23. **Tejeda, Luis.** Babo's. [En línea] 16 de junio de 2007. http://babotejada.wordpress.com/2007/06/16/proceso-unificado-de-rational/.

24. La tecla de Escape. [En línea] 25 de enero de 2009. http://latecladeescape.com/w0/ingenieria-del-software/metodologias-de-desarrollo-del-software.html.

25. **Guerrero, Luis A.** *Taller de UML.* s.l. : Universidad de Chile Departamento de Ciencias de la Computación.

26. **Gutiérrez, Gustavo A.** [En línea] junio de 2003. http://www.enterate.unam.mx/Articulos/2003/junio/j2ee.htm.

27. **Nájera, Gilberto.** Mailxmail. [En línea] 14 de julio de 2005. http://www.mailxmail.com/curso-desarrollo-aplicaciones-dispositivos-inalambricos-j2me/lenguaje-programacion-java.

28. Lenguajes de programación. [En línea] 2009. http://www.lenguajes-de-programacion.com/programacion-java.shtml.

29. *El lenguaje de Programación Java.*

30. **Software, Grupo de Investigación en Ingeniería de.** *Experiencia y retos en la adopción de frameworks para el desarrollo acelerado de aplicaciones.* 2005.

31. Marco de Desarrollo de la Junta de Andalucía. [En línea] http://www.juntadeandalucia.es/xwiki/bin/view/MADEJA/Hibernate.

32. **López, Jorge Blanco.** Educación. [En línea] 12 de junio de 2006. http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=354.

33. Eclipse. [En línea] [Citado el: 30 de enero de 2010.] http://www.eclipse.org/.

34. JUnit. [En línea] [Citado el: 29 de enero de 2010.] http://www.junit.org/.

35. Sitio Oficial de IBM. [En línea] [Citado el: 29 de enero de 2010.] http://www-01.ibm.com/software/awdtools/rup/.

36. **Craig Walls, Ryan Breidenbach.** *Spring in Action 2nd Edición.* s.l. : Manning, 2007.

37. —. *Spring in Action.* s.l. : Manning, 2005.

38. TortoiseSVN. [En línea] [Citado el: 30 de enero de 2010.] http://tortoisesvn.net/.

39. Subversion. [En línea] 2008. [Citado el: 29 de enero de 2010.] http://subversion.tigris.org.

40. Alembik. [En línea] [Citado el: 4 de febrero de 2010.] http://alembik.sourceforge.net/overview.html.

41. Métodos de Prueba de Software. [En línea] 13 de octubre de 2005. http://pruebasoftware.blogcindario.com/2005/10/00001-metodos-de-prueba-del-software.html.

42. **Walls, Craig y Breidenbach, Ryan.** *Spring in Action.* s.l. : Manning, 2005.

Referencias Bibliográficas

1. **López, Jorge Blanco.** Cnice. [En línea] 12 de junio de 2006. http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=354.

2. Teléfonos-móviles. [En línea] 21 de mayo de 2003. http://www.telefonos-moviles.com/articles/item.asp?ID=20.

3. Masadelante. [En línea] http://www.masadelante.com/faqs/wap.

4. Protocolo Inalàmbrico de Aplicaciones. [En línea] http://www.electronguia.com.ar/Comunicacion/Informacion/wap/protocoloWAP.htm.

5. WebEstilo. [En línea] http://www.webestilo.com/wml/.

6. W3C. [En línea] 8 de abril de 2008. http://www.w3c.es/Noticias/2008/04/08/ltima-convocatoria-api-simple-para-repositorio-de-descripcin-de-dispositivo/.

7. AccNera. [En línea] http://www.accnera.com/webmovil/webmovil-adaptacion.html.

8. **Pressman, Roger.** *Ingeniería de Software un Enfoque Práctico.* 2006.

9. **García, Joaquin.** IngenieroSoftware. [En línea] 7 de mayo de 2005. http://www.ingenierosoftware.com/analisisydiseno/uml.php.

10. eleZeta . [En línea] 27 de agosto de 2004. http://elezeta.net/2004/08/27/extreme-programming-xp/.

11. Proyectosagiles. [En línea] http://www.proyectosagiles.org/que-es-scrum.

12. **Ecuador, Pontificada Universidad Católica del.** *Herramientas Case.*

13. LuAuf. [En línea] 13 de mayo de 2008. http://luauf.com/2008/05/13/entornos-de-desarrollo-integrado-para-java/.

14. Educación. [En línea] 17 de enero de 2008. http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=548.

15. Elwebmaster. [En línea] 7 de noviembre de 2007. http://www.elwebmaster.com/referencia/api-interface-de-programacion-de-aplicaciones.

16. Codebox. [En línea] http://www.codebox.es/glosario.

17. **Agüero, Martín.** *Introducción a Spring Framework.* 2007.

18. **Herrera, Cristhian.** Informática Profesional. [En línea] 16 de agosto de 2007. http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=hibernateVSEJB3.

19. **Albet, S.A.** *Especificaciones Suplementarias, SIIPOL Móvil.* 2009.

20. **Tedeschi, Nicolás.** Patrones de Diseño. [En línea] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

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

Glosario de términos

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

CAPTCHA: se trata de una prueba desafío-respuesta utilizada en computación para determinar cuándo el usuario es humano o no. La prueba consiste en que el usuario introduzca un conjunto de caracteres que se muestran en una imagen distorsionada que aparece en pantalla.

Código abierto: práctica de desarrollo de software que promueve el acceso al código fuente de los sistemas computacionales.

Código fuente: es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar un programa.

CVS: es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros que forman un proyecto y permite que distintas personas colaboren.

DAO: Data Access Object, traducido al español como Objeto de Acceso a Datos, es un patrón de diseño de clases que permite a quien lo aplique, suministrar una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos.

Datagramas: Es un fragmento de paquete que es enviado con la suficiente información como para que la red pueda simplemente encaminar el fragmento hacia el equipo terminal, de manera independiente a los fragmentos restantes.

Dirección IP: es una etiqueta numérica que identifica, de manera lógica y jerárquica, a una interfaz de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP, que corresponde al nivel de red del protocolo TCP/IP.

Encapsulación: es el proceso de ocultar todos los detalles de un objeto que no contribuyan a sus características esenciales.

Gateway: es un dispositivo que permite interconectar redes con protocolos y arquitecturas diferentes a todos los niveles de comunicación. Su propósito es traducir la información del protocolo utilizado en una red al protocolo usado en la red de destino.

HQL: lenguaje de consultas del framework Hibernate, similar al SQL, pero que se refiere a clases y objetos no a tablas de la base de datos.

JSP: una tecnología orientada a crear páginas web con programación en Java.

Lenguaje de script: es un pequeño lenguaje de programación cuyo código se inserta dentro de un documento HTML. Este código se ejecuta en el navegador del usuario al cargar la página, o cuando sucede algo especial como puede ser el hacer clic sobre un enlace.

IoC: es un método de programación en el que el flujo de ejecución de un programa se invierte respecto a los métodos de programación tradicionales. En la inversión de control se especifican respuestas deseadas a sucesos o solicitudes de datos concretas, dejando que algún tipo de entidad o arquitectura externa lleve a cabo las acciones de control que se requieran en el orden necesario y para el conjunto de sucesos que tengan que ocurrir.

Micronavegador: Es un navegador web diseñado para el uso en dispositivos móviles como los PDA o teléfonos móviles. Los micronavegadores están optimizados para mostrar contenido de Internet en pantallas reducidas, y utilizan tamaños de archivo reducidos para ser instalados en dispositivos con memorias de baja capacidad.

MVC: 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.

Navegador: Un navegador web o cliente HTTP, es un programa que permite interpretar la información y el código que contiene una página web y presentarla de manera legible a los usuarios.

OMA: Organización formada por las principales compañías ligadas a la telefonía móvil, tanto en hardware como en servicios y software, cuyo principal objetivo es la creación de especificaciones y estándares abiertos para el desarrollo de este sector.

ORM: Es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional.

Plugin: se trata de un pequeño programa que proporciona alguna funcionalidad específica a otra aplicación mayor o más compleja.

Polimorfismo: posibilidad de definir clases diferentes que tienen métodos o atributos denominados de forma idéntica, pero que se comportan de manera distinta.

Script: es un guión o conjunto de instrucciones. Permiten la automatización de tareas creando pequeñas utilidades. Es un fragmento de código que puede recibir argumentos y devolver un valor.

Software Libre: programas desarrollados y distribuidos de acuerdo con la filosofía de dar al usuario la libertad de ejecutar, copiar, distribuir, estudiar, cambiar y mejorar dichos programas.

Subversion: es un software de sistema de control de versiones diseñado específicamente para reemplazar al CVS.

User Agent: cuando un usuario accede a una página web, la aplicación generalmente envía una cadena de texto que identifica al agente de usuario ante el servidor. Este texto forma parte del pedido a través de HTTP, llevando como prefijo user-agent y generalmente incluye información como el nombre de la aplicación, la versión, el sistema operativo y el idioma.

Velocity: es un motor de plantillas basado en Java. Se puede utilizar para crear páginas web, SQL, PostScript y cualquier otro tipo de salida de plantillas. Además se puede utilizar como una aplicación independiente para generar código fuente y reportes, o como un componente integrado en otros sistemas.

XHTML: Lenguaje Extensible de Marcado de Hipertexto, es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web.

XHTML-MP: Lenguaje Extensible de Marcado de Hipertexto - Perfil Móvil, es el lenguaje de marcado estándar diseñado específicamente para teléfonos móviles y otros dispositivos con recursos restringidos.

XML: Lenguaje Extensible de Marcado, desarrollado por el W3C para permitir la descripción de información contenida en el www a través de estándares y formatos comunes

XSLT: es un estándar de la organización W3C que presenta una forma de transformar documentos XML en otros. Es muy usado en la edición web, generando páginas HTML o XHTML.

Anexos

Anexo 1: Diagramas de Clases de Análisis

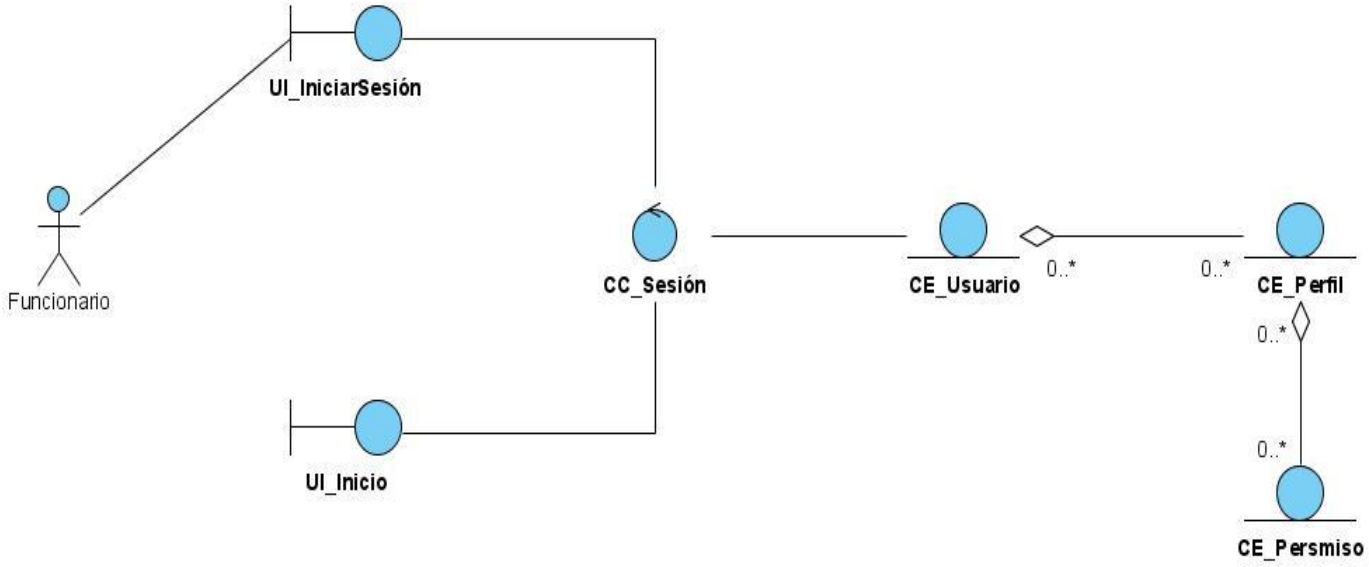


Figura 28. Diagrama de Clases de Análisis CU Abrir/Cerrar Sesión

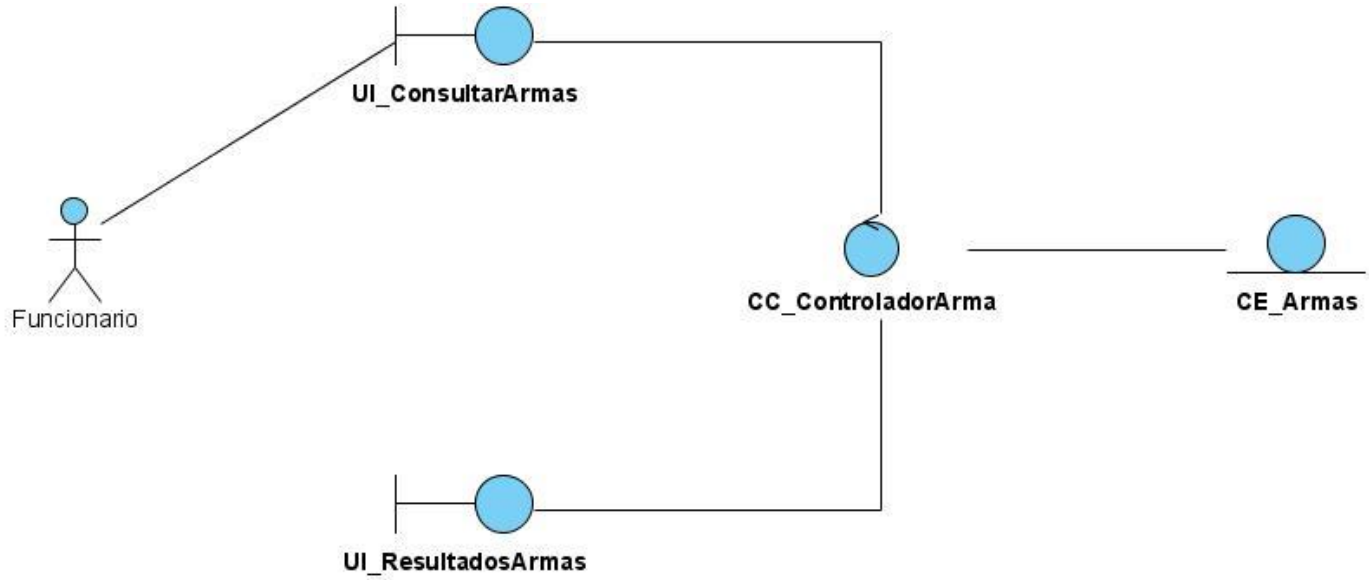


Figura 29. Diagrama de Clases de Análisis CU Consultar Armas con Registros Policiales

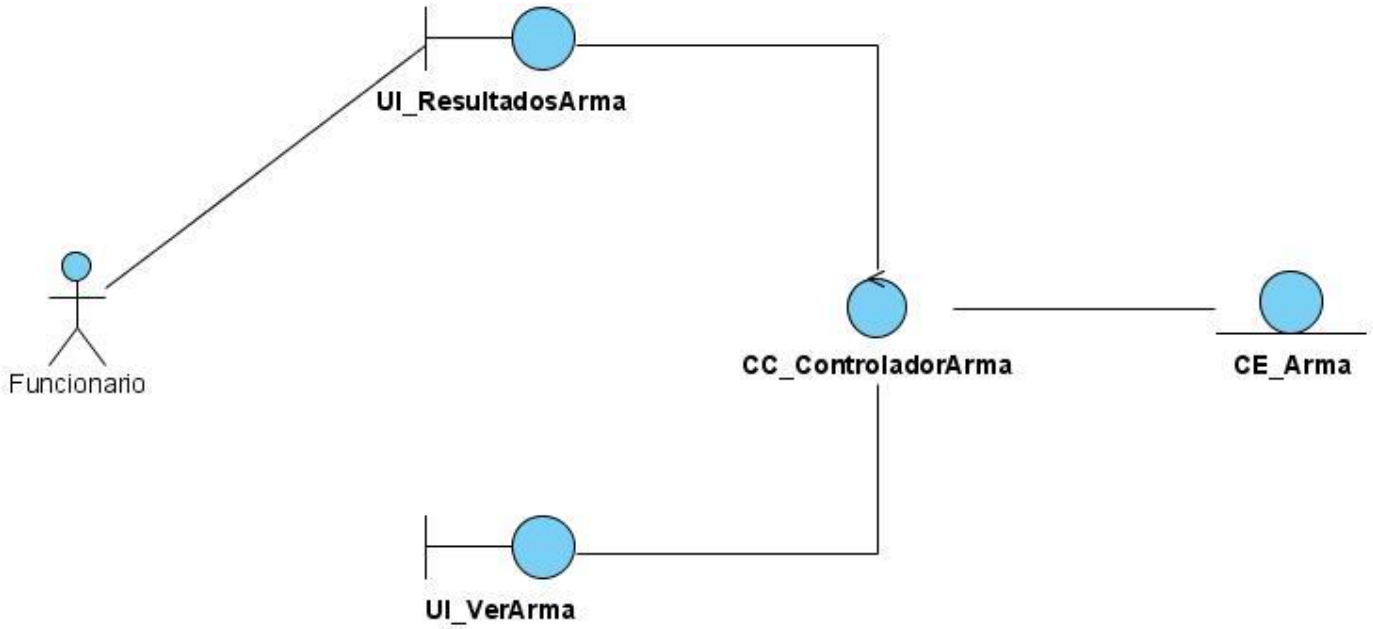


Figura 30. Diagrama de Clases de Análisis CU Ver Detalles de Armas con Registros Policiales

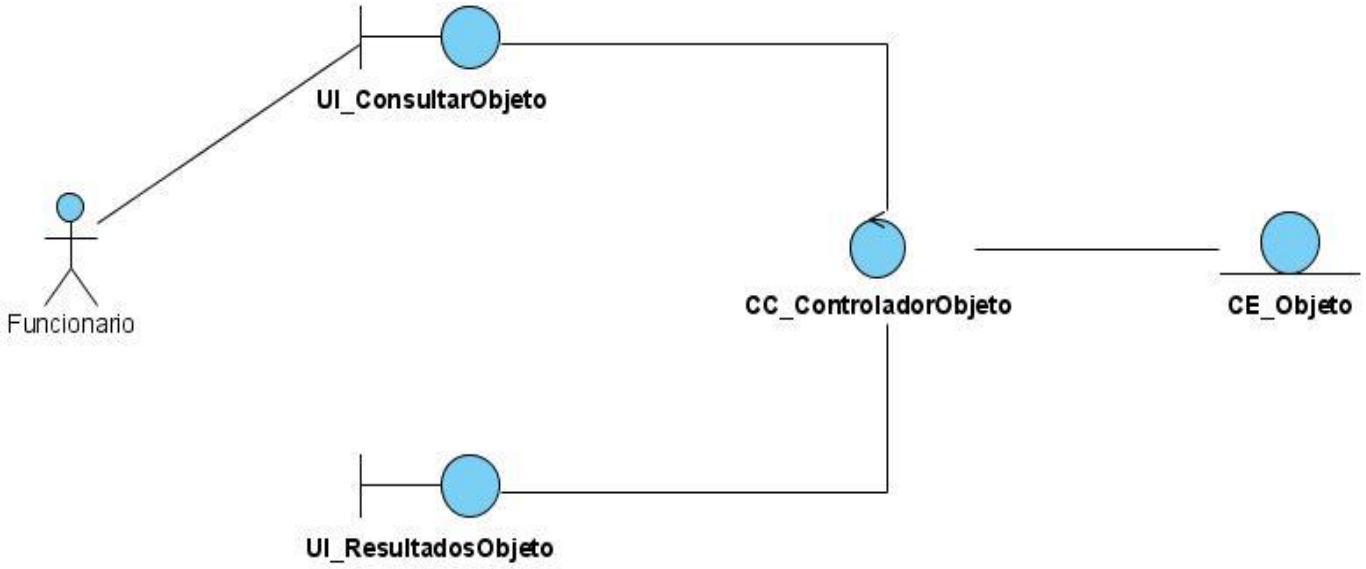


Figura 31. Diagrama de Clases de Análisis CU Consultar Objetos con Registros Policiales

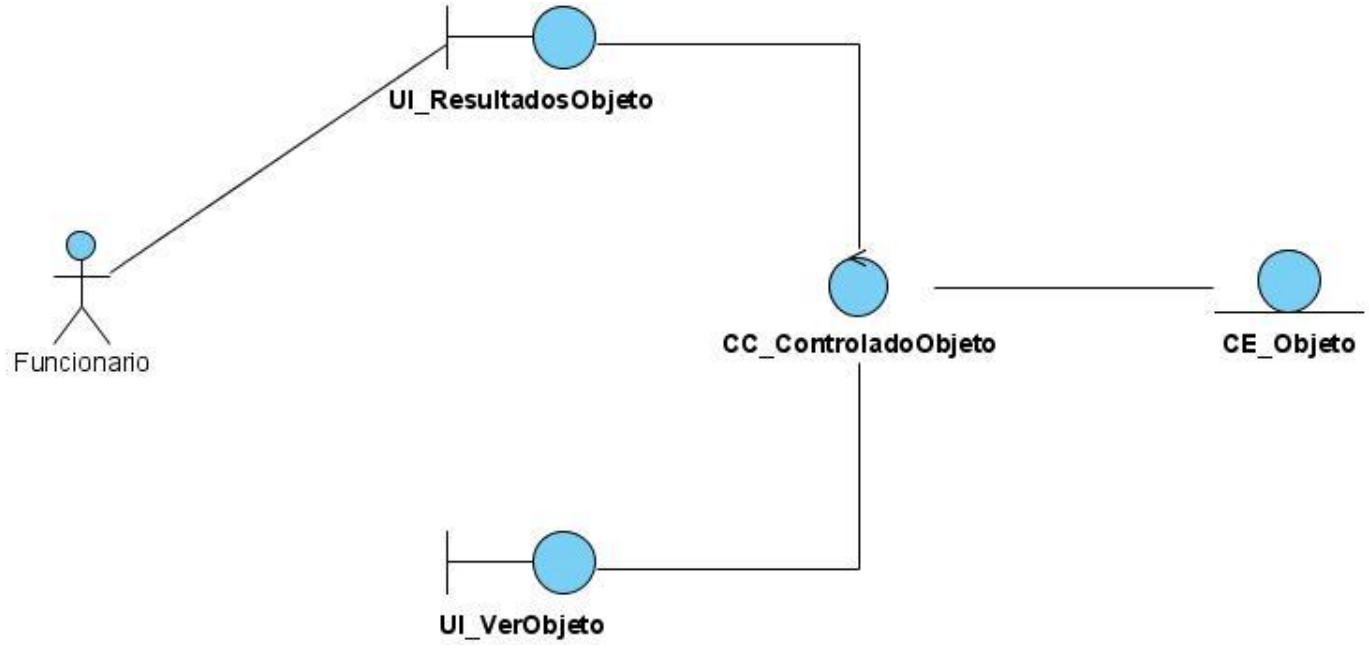


Figura 32. Diagrama de Clases de Análisis CU Ver Detalles de Objetos con Registros Policiales

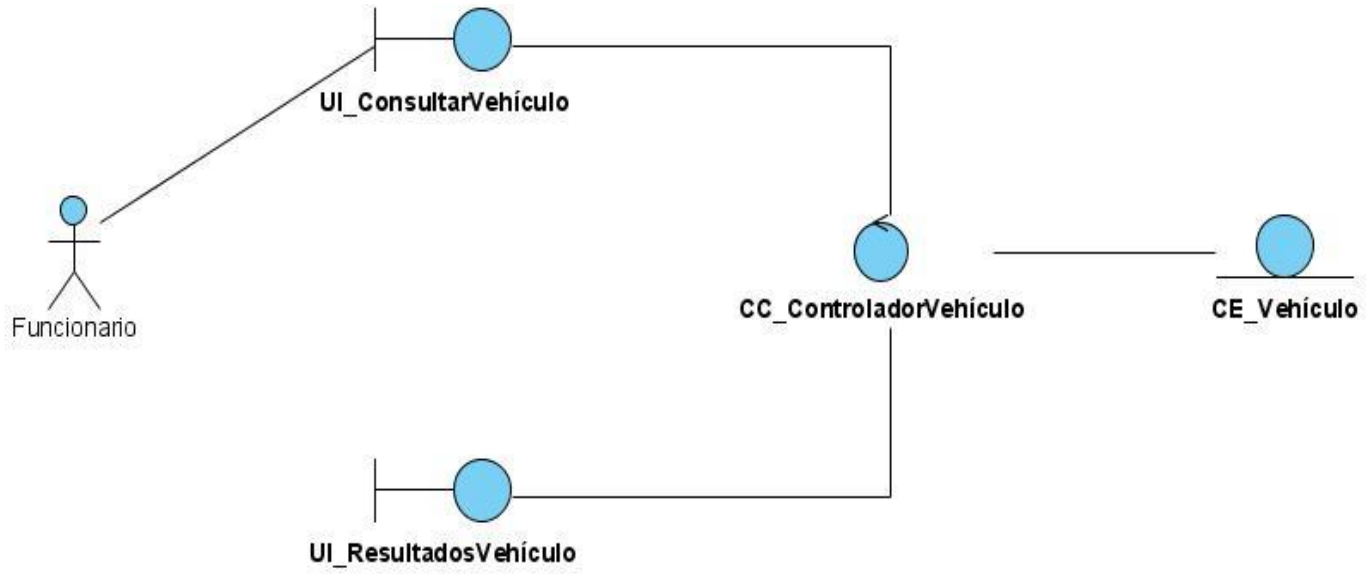


Figura 33. Diagrama de Clases de Análisis CU Consultar Vehículos con Registros Policiales

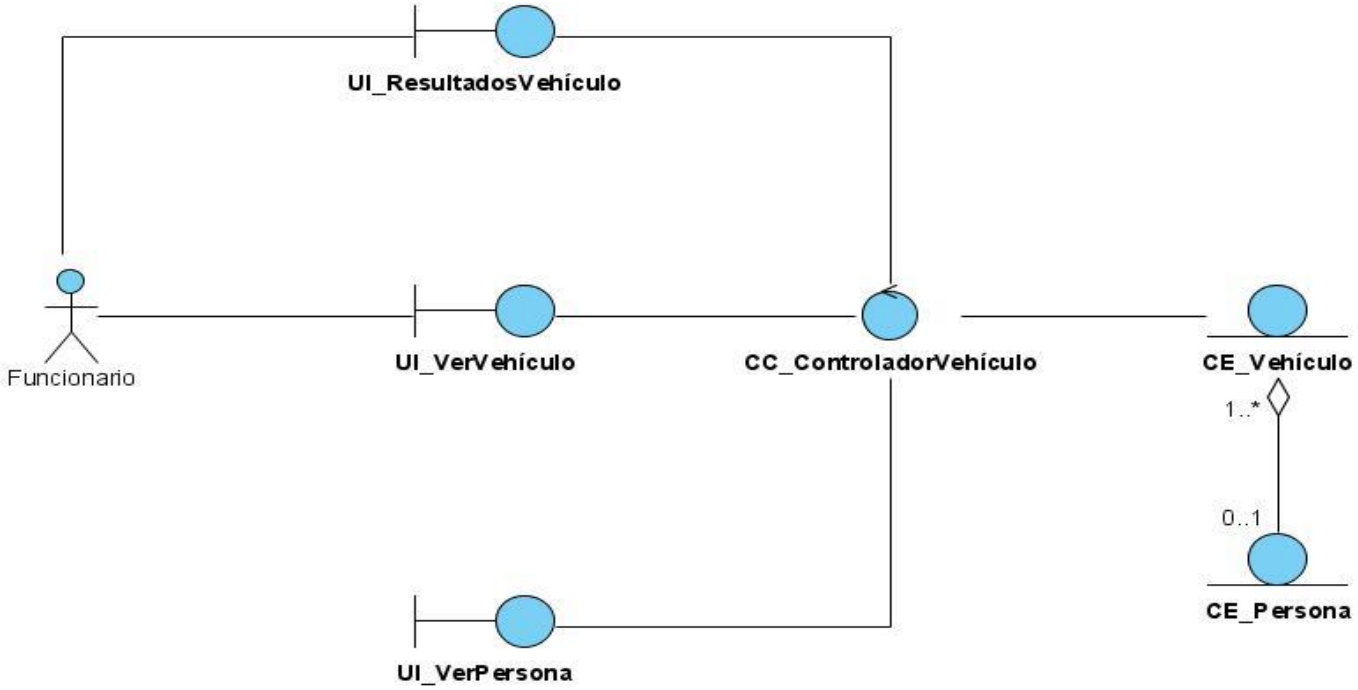


Figura 34. Diagrama de Clases de Análisis CU Ver Detalles de Vehículos con Registros Policiales

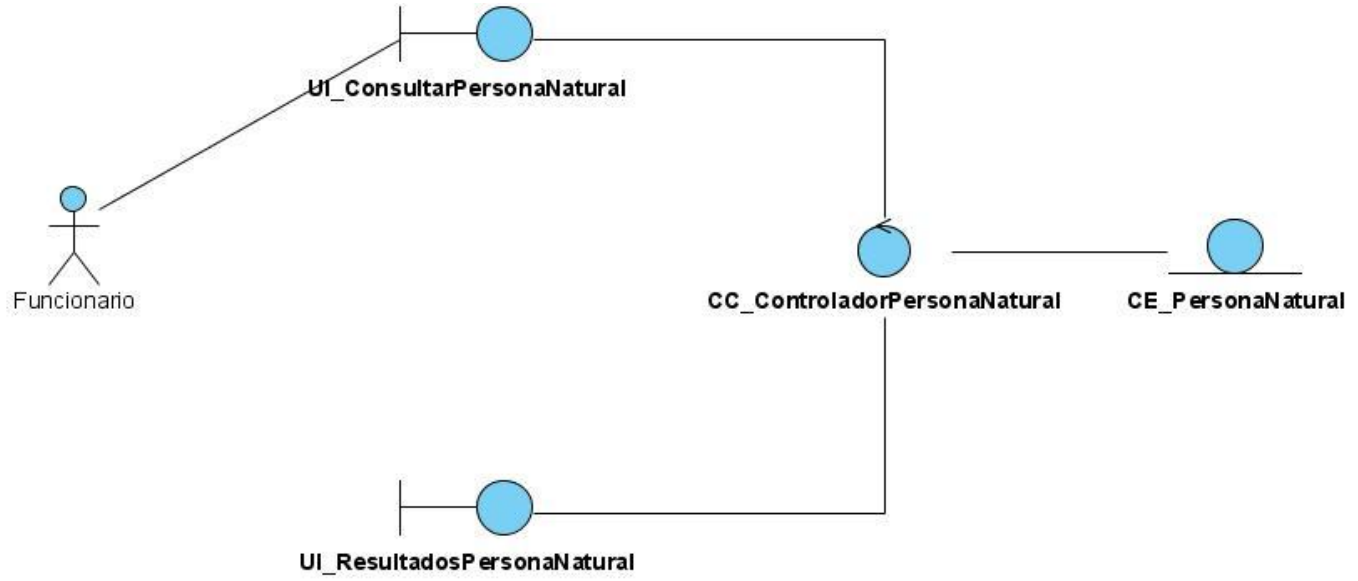


Figura 35. Diagrama de Clases de Análisis CU Consultar Persona Natural

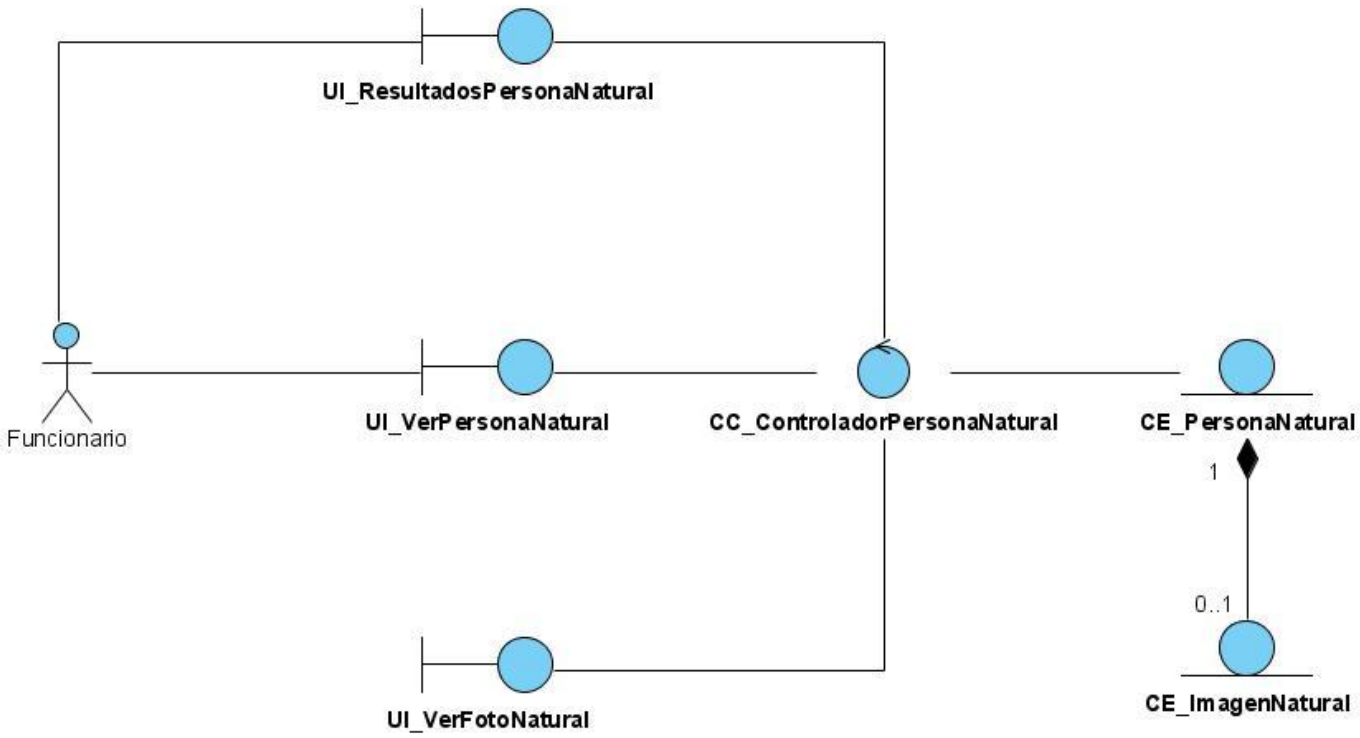


Figura 36. CU Diagrama de Clases de Análisis Ver Detalles de Persona Natural

Anexo 2: Diagramas de Colaboración

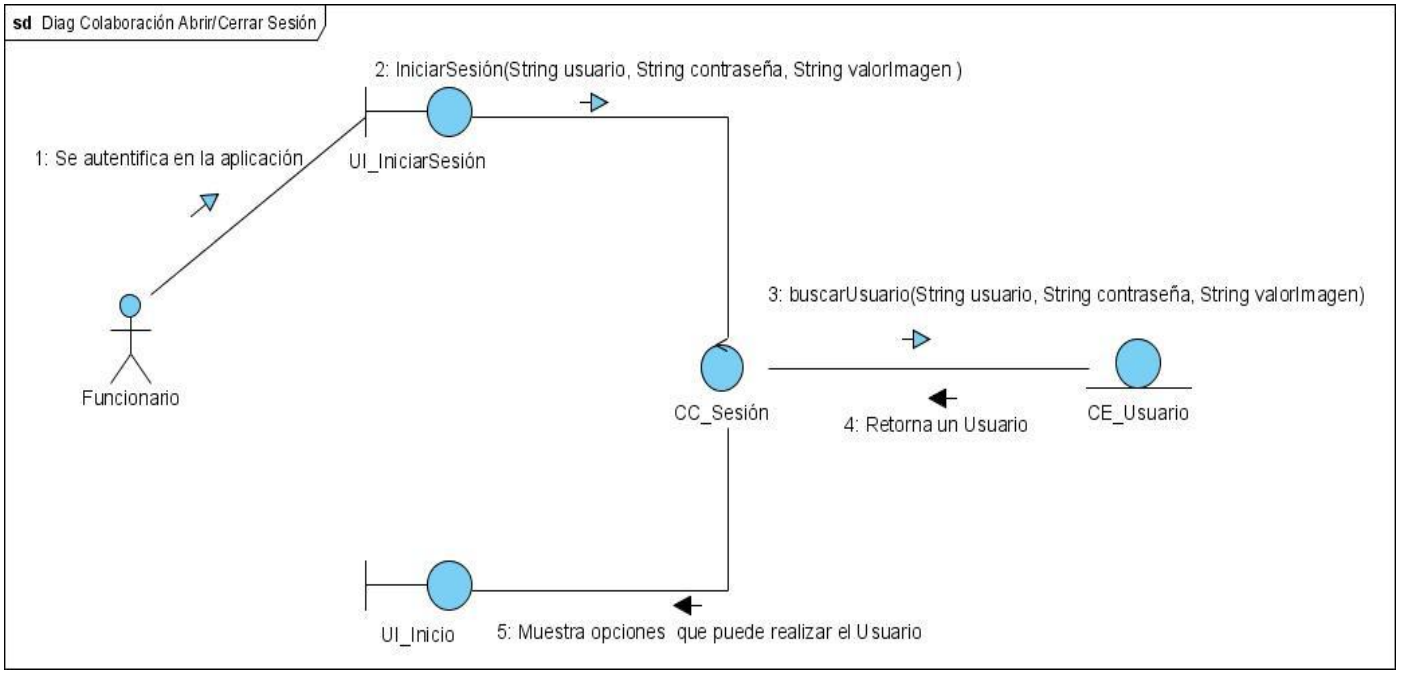


Figura 37. Diagrama de Colaboración CU Abrir/Cerrar Sesión

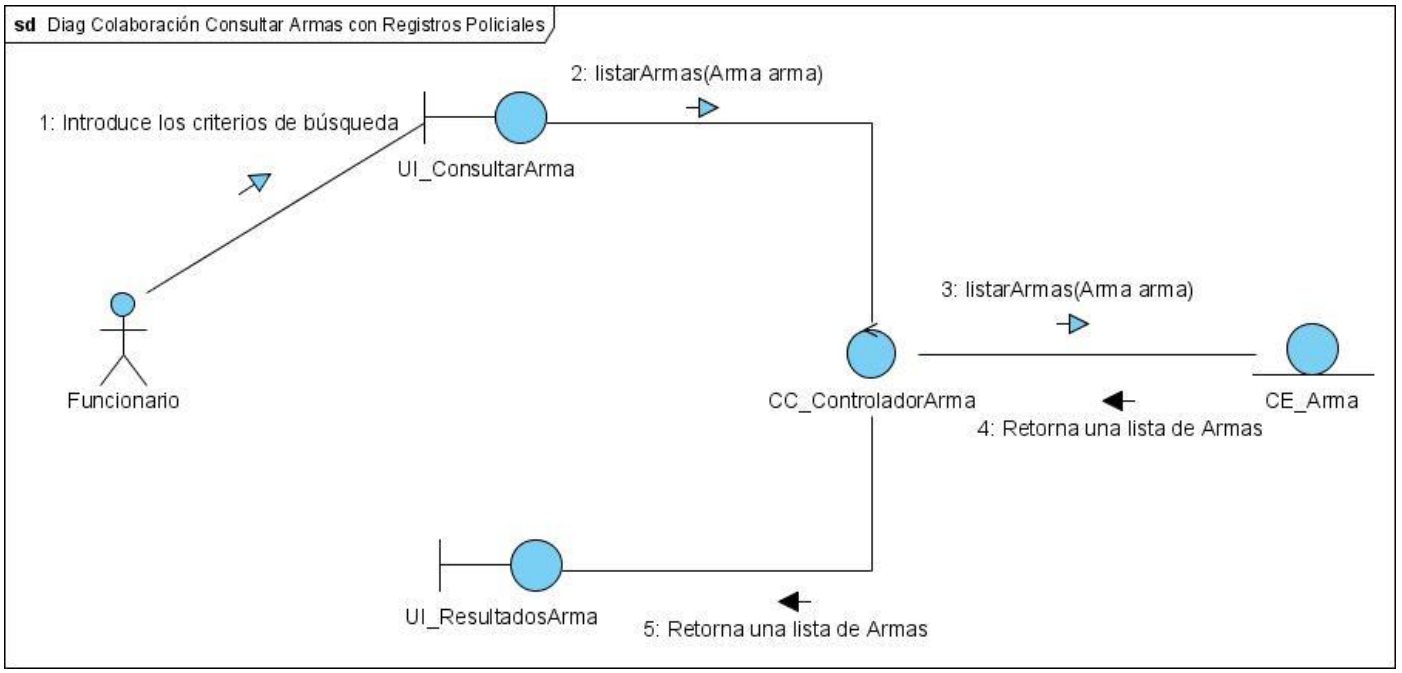


Figura 38. Diagrama de Colaboración CU Consultar Armas con Registros Policiales

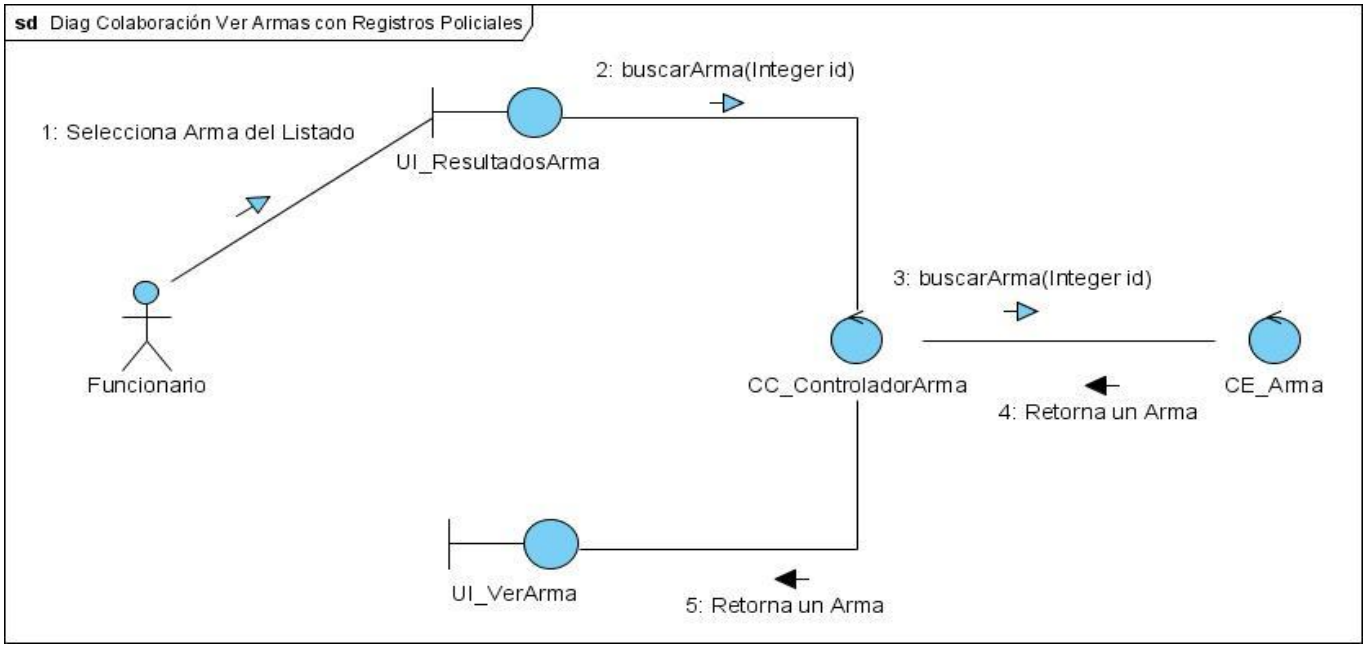


Figura 39. Diagrama de Colaboración CU Ver Detalles de Armas con Registros Policiales

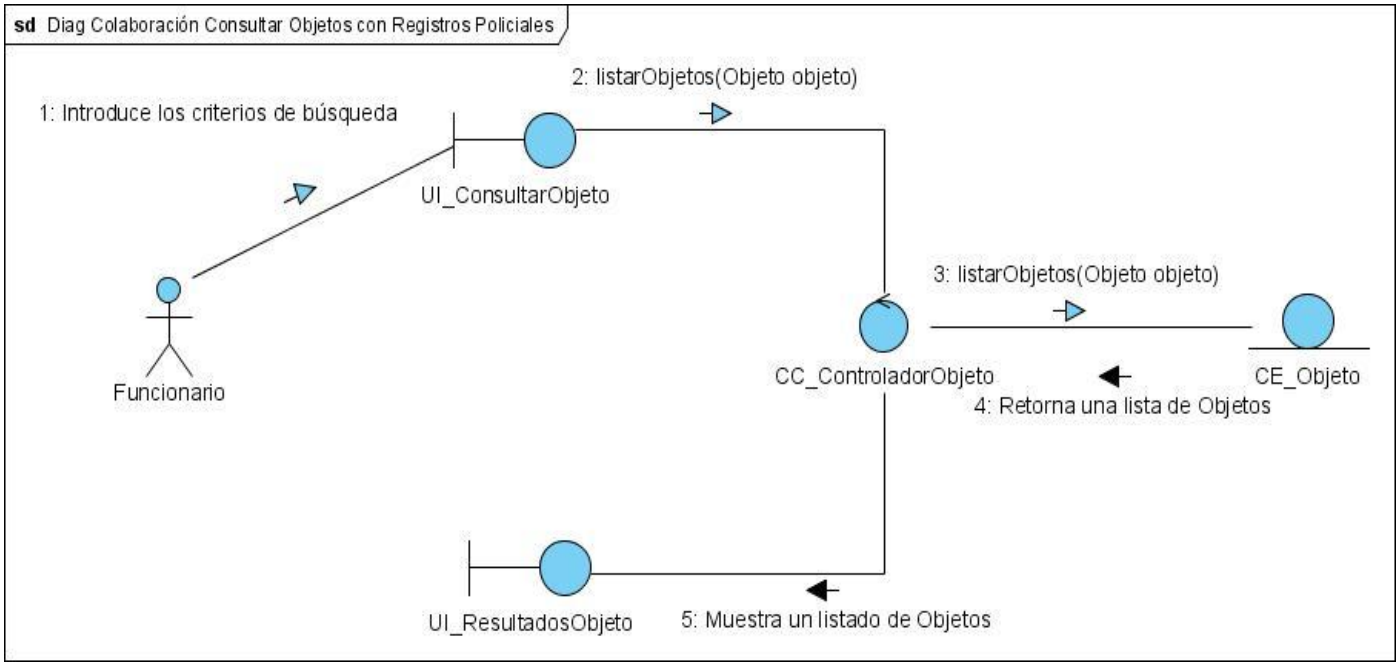


Figura 40. Diagrama de Colaboración CU Consultar Objetos con Registros Policiales

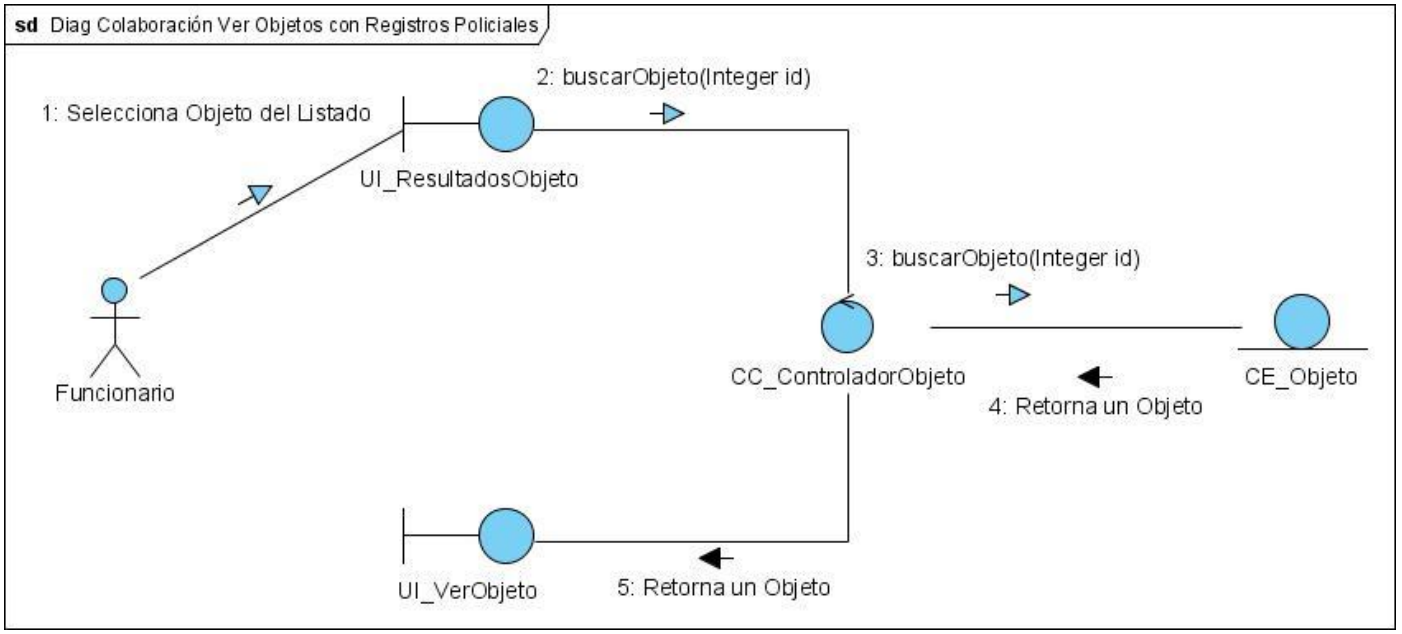


Figura 41. Diagrama de Colaboración CU Ver Detalles de Objetos con Registros Policiales

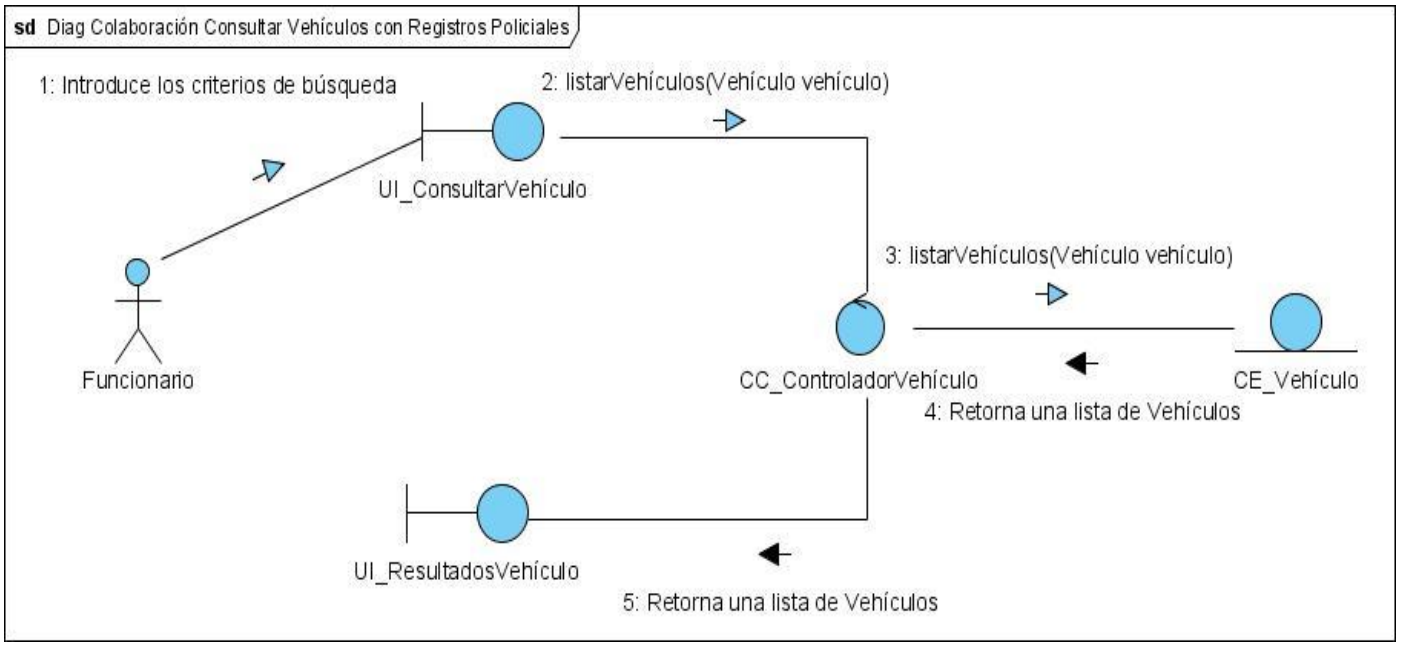


Figura 42. Diagrama de Colaboración CU Consultar Vehículos con Registros Policiales

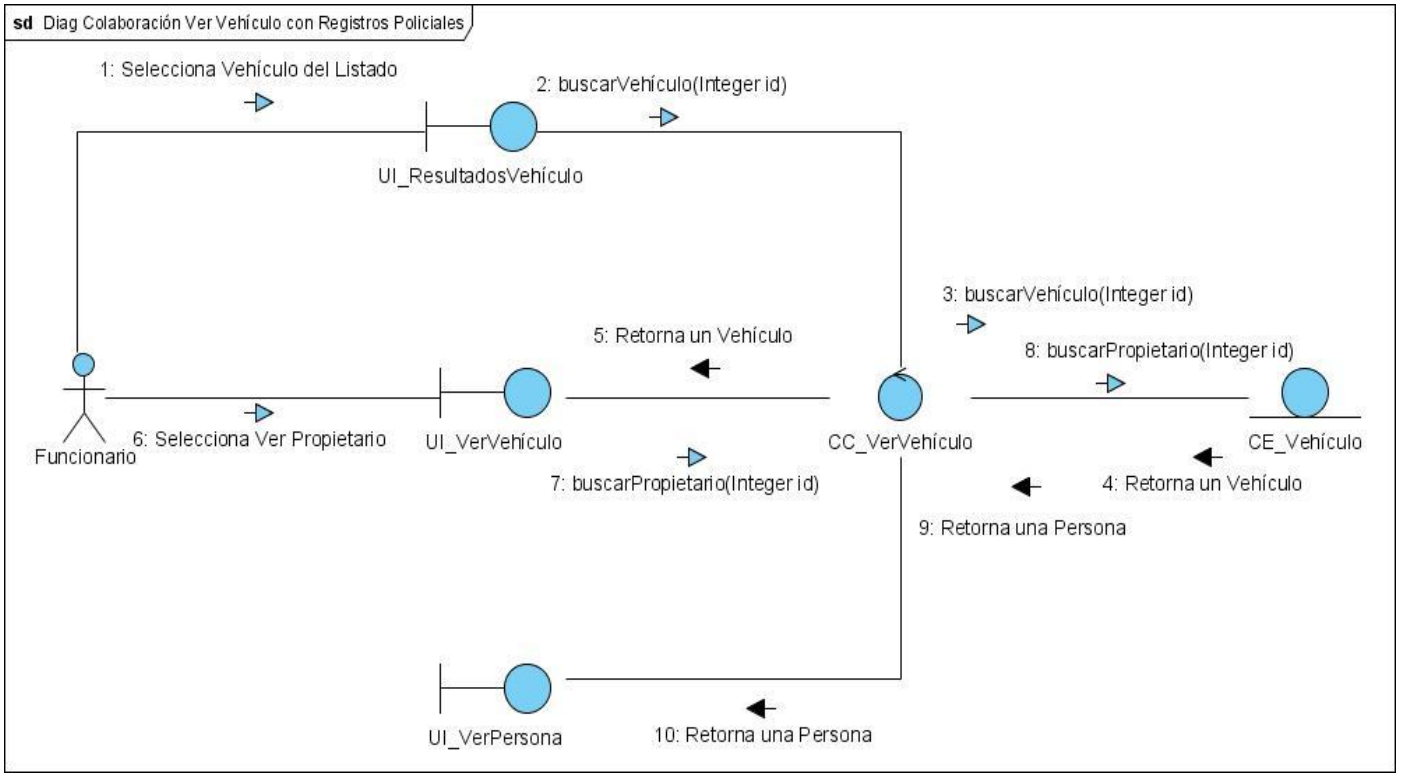


Figura 43.Diagrama de Colaboración CU Ver Detalles de Vehículos con Registros Policiales

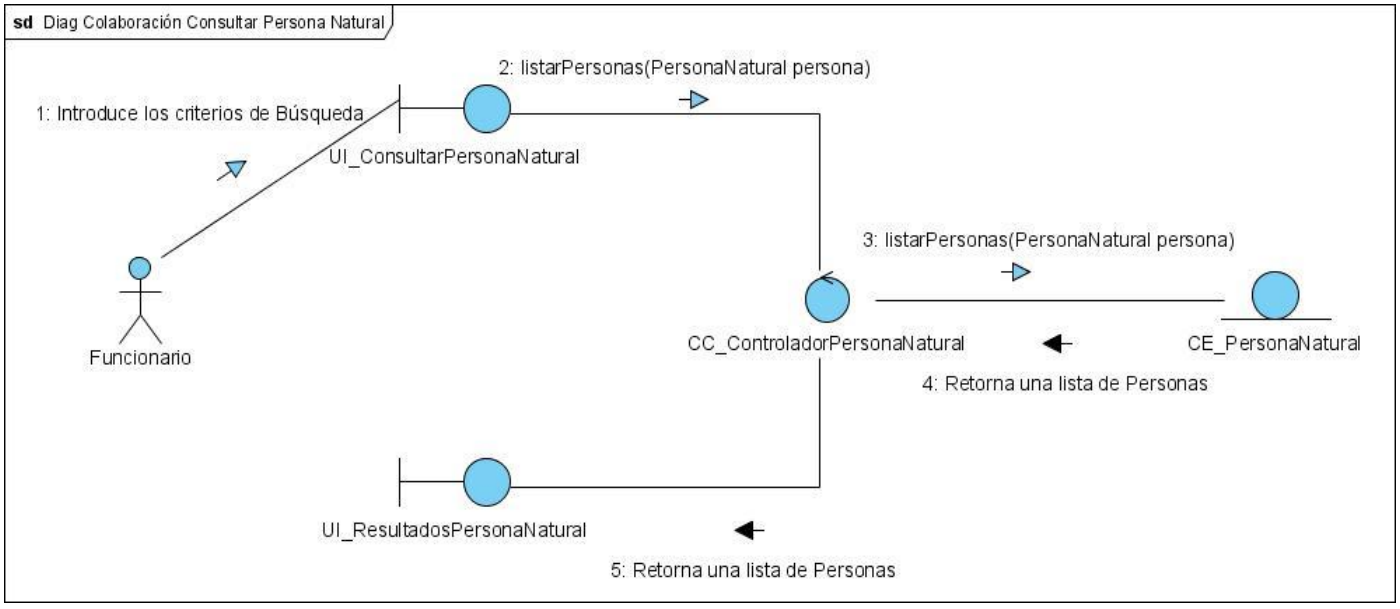


Figura 44. Diagrama de Colaboración CU Consultar Persona Natural

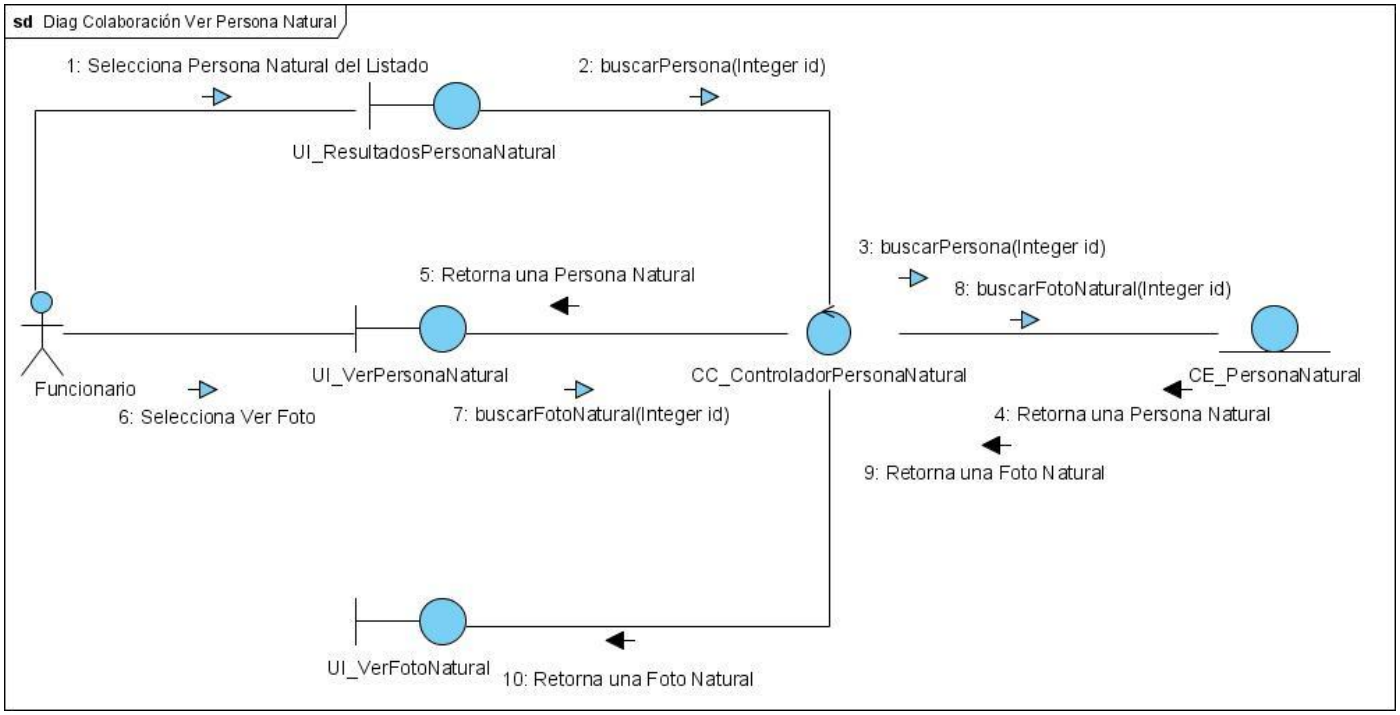


Figura 45. Diagrama de Colaboración CU Ver Detalles de Persona Natural

Anexo 3: Diagramas de Clases del Diseño

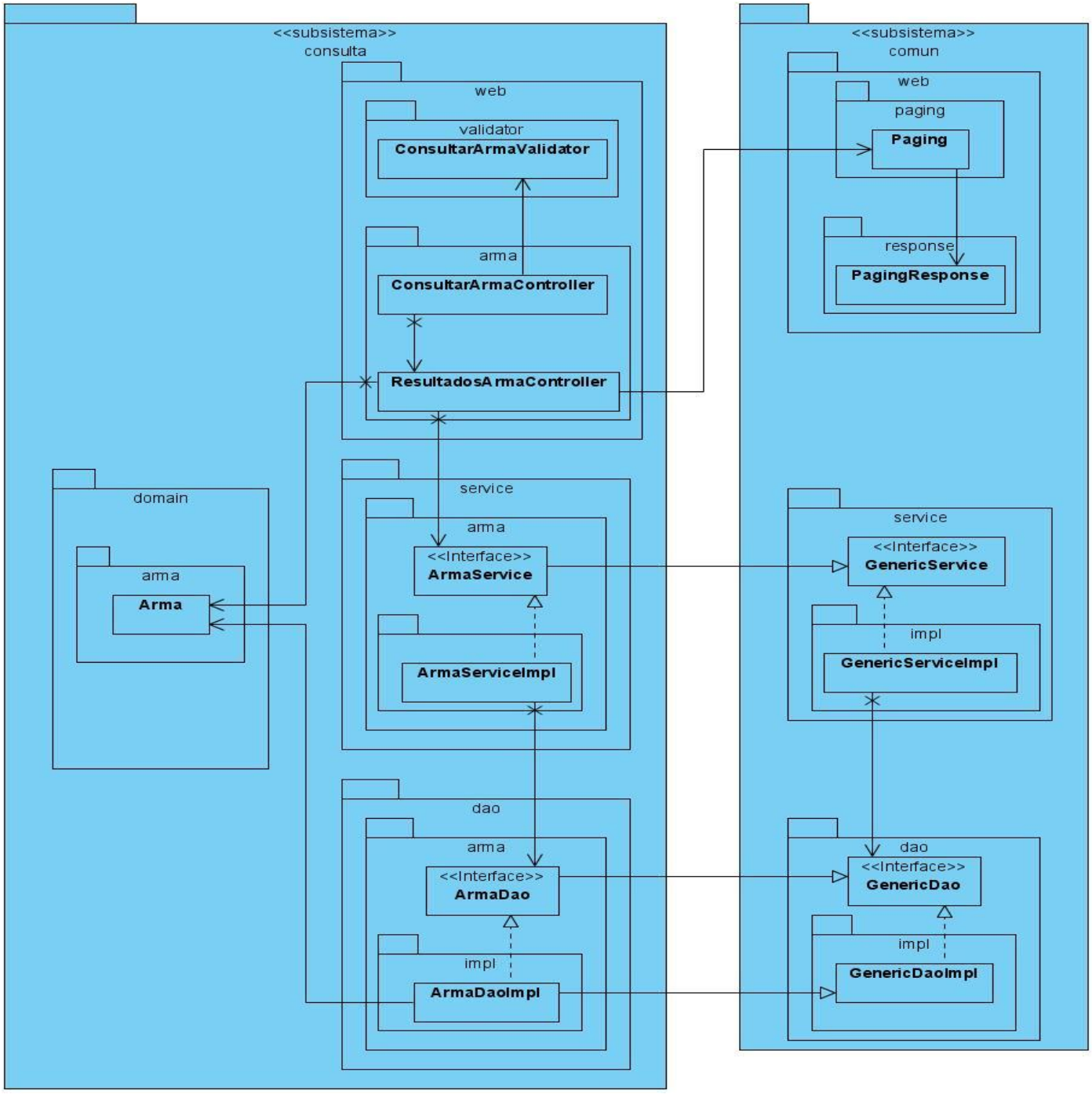


Figura 46. Diagrama de Clases del Diseño CU Consultar Armas con Registros Policiales

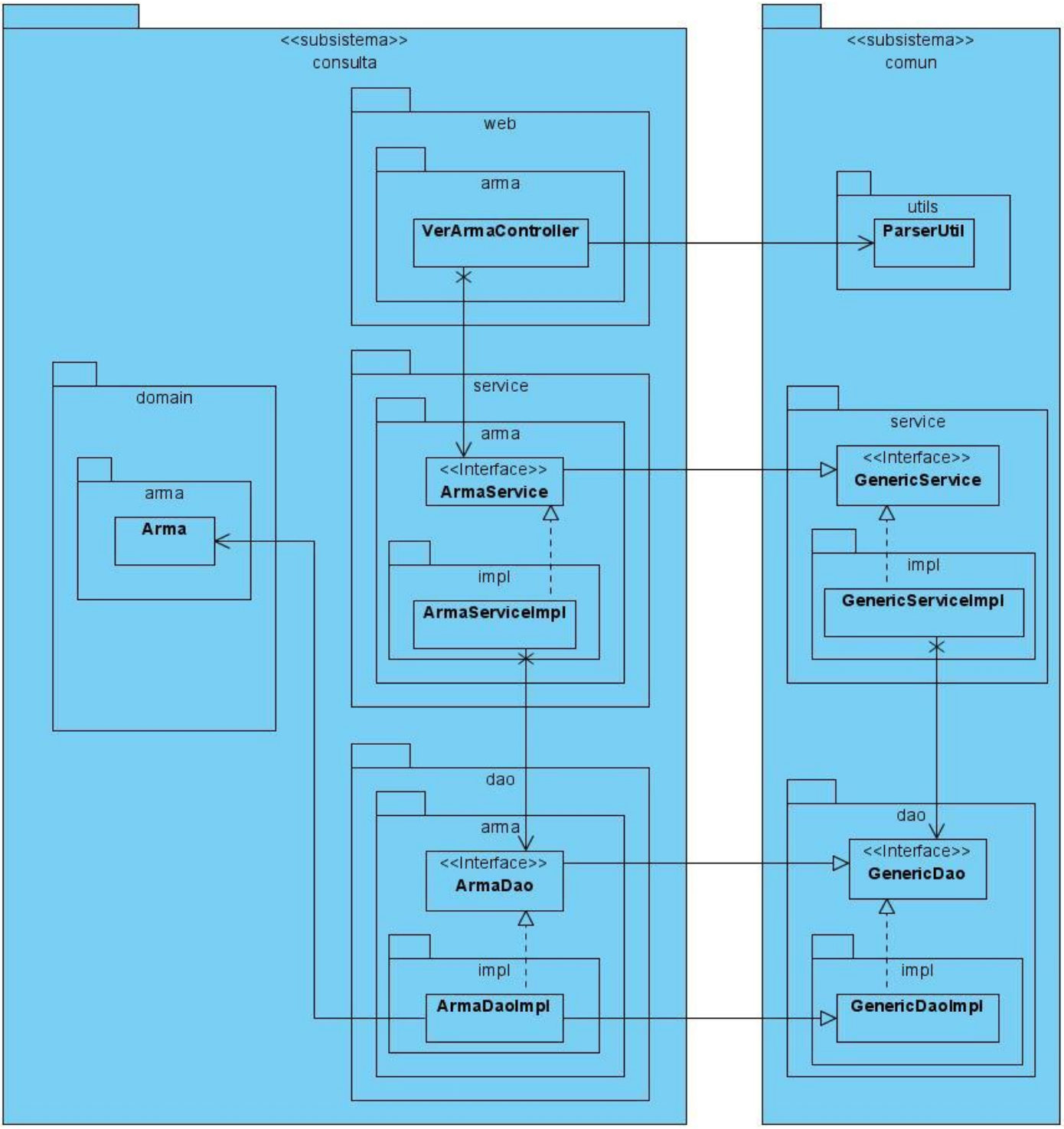


Figura 47. Diagrama de Clases del Diseño CU Ver Detalles de Armas con Registros Policiales

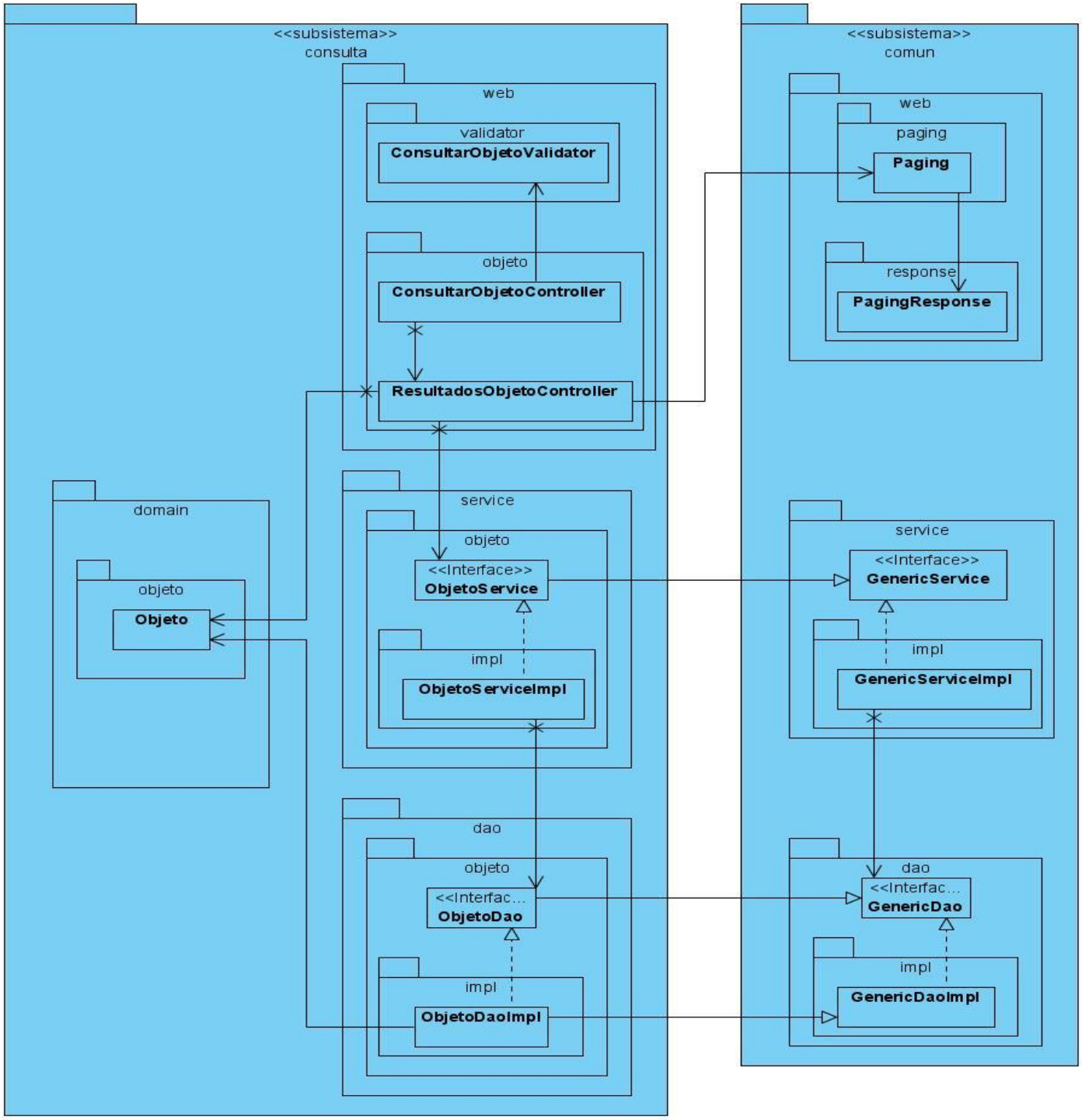


Figura 48. Diagrama de Clases del Diseño CU Consultar Objetos con Registros Policiales

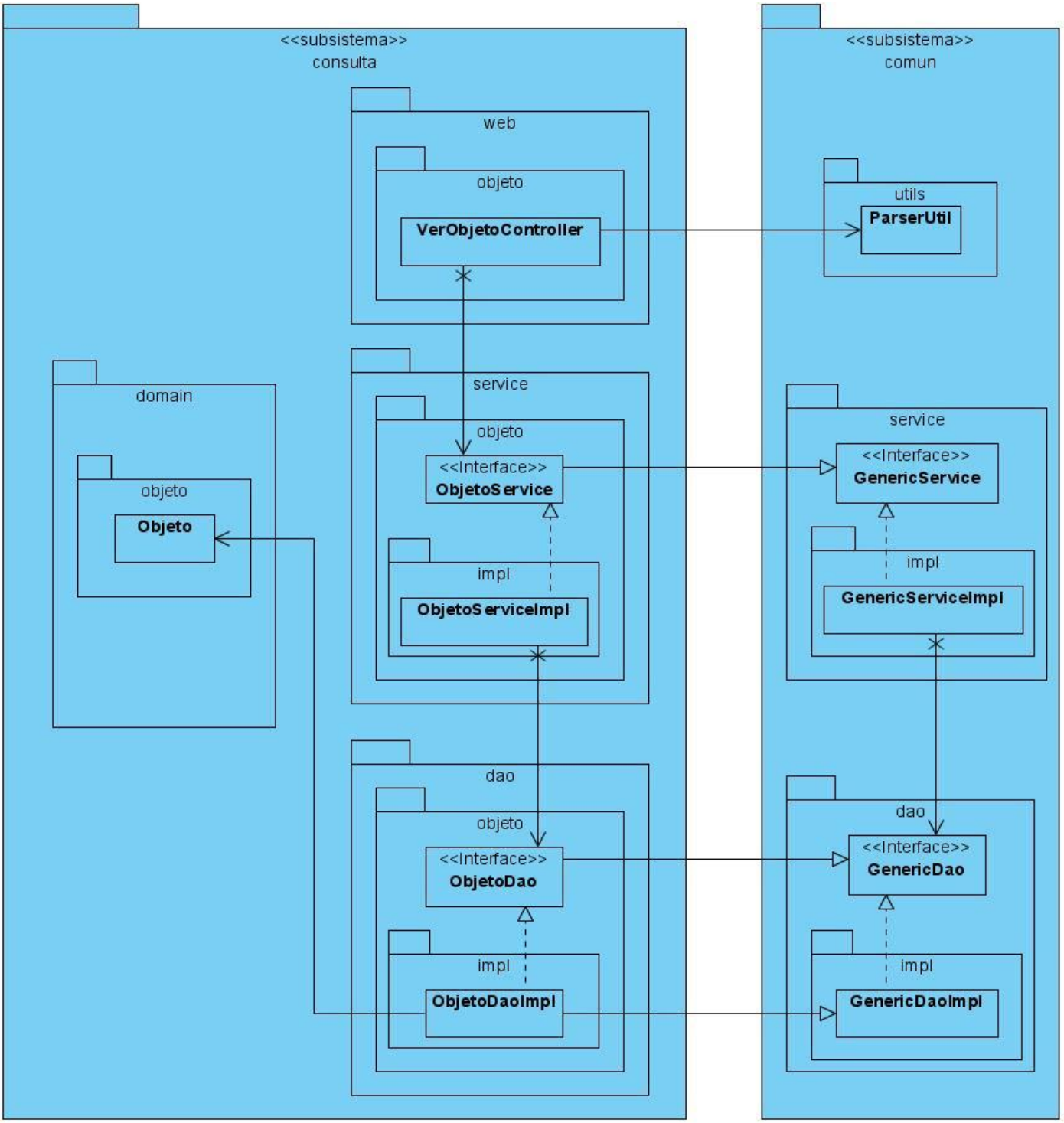


Figura 49. Diagrama de Clases del Diseño CU Ver Detalles de Objetos con Registros Policiales

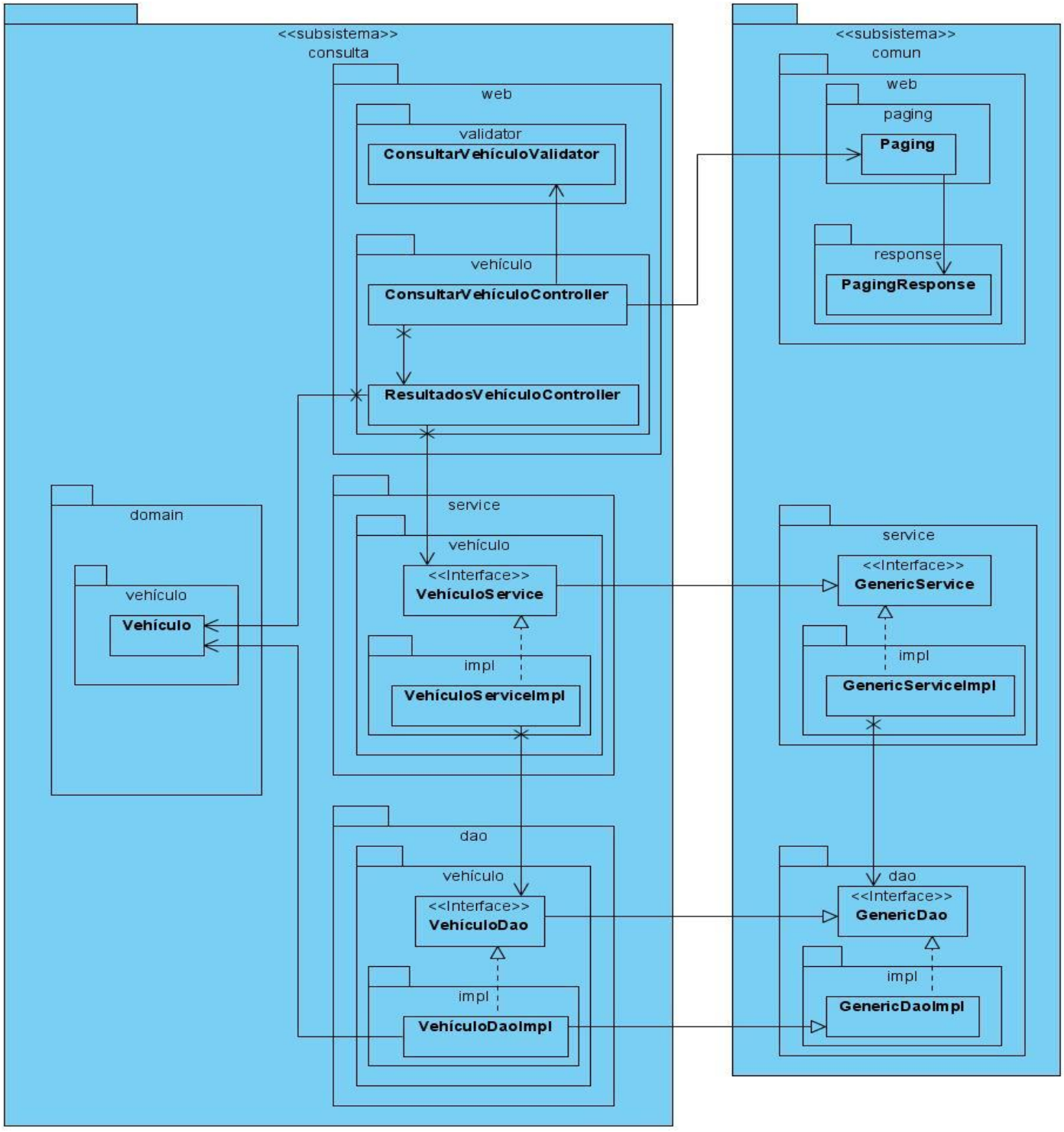


Figura 50. Diagrama de Clases del Diseño CU Consultar Vehículos con Registros Policiales

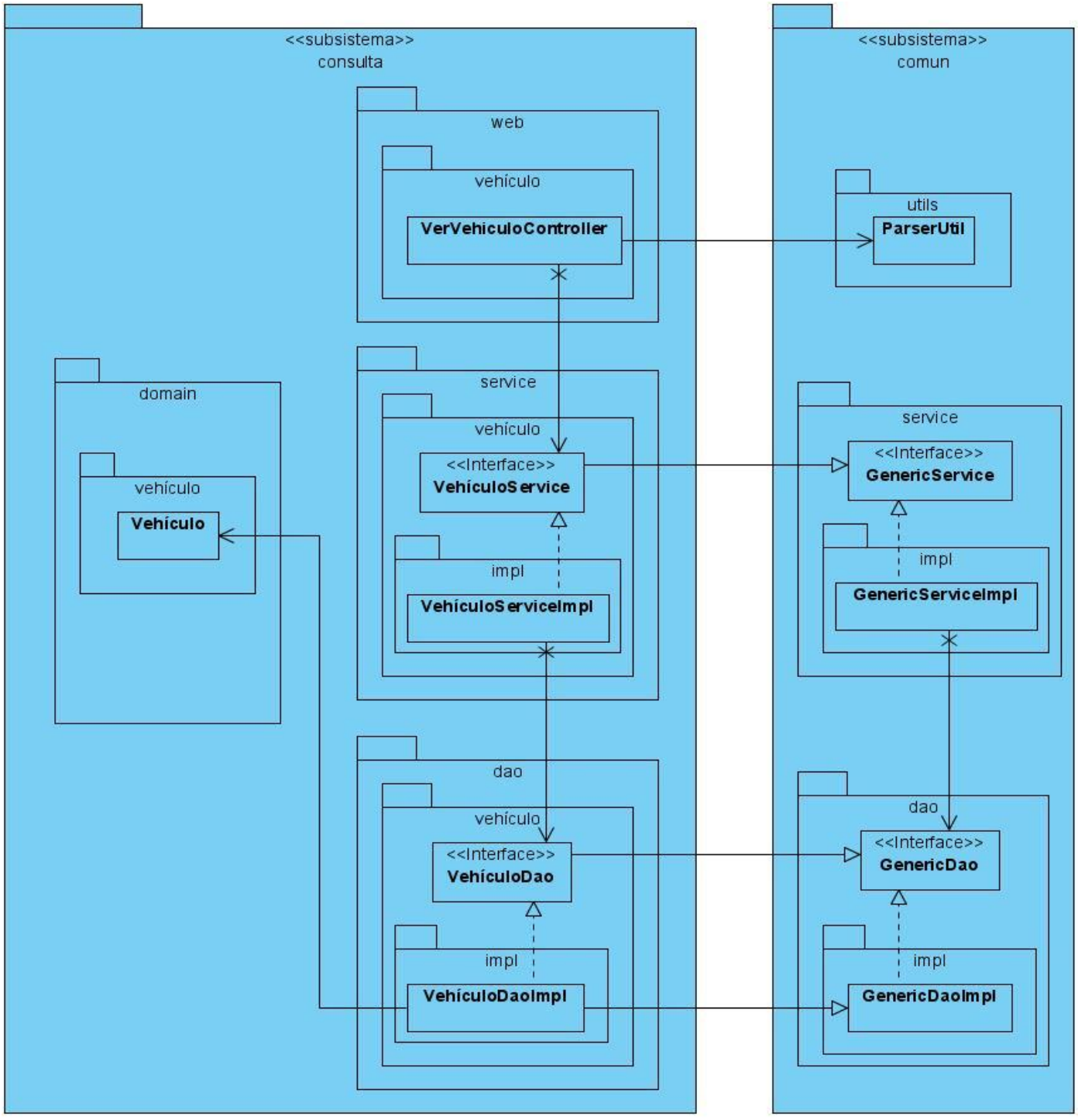


Figura 51. Diagrama de Clases del Diseño CU Ver Detalles de Vehículos con Registros Policiales

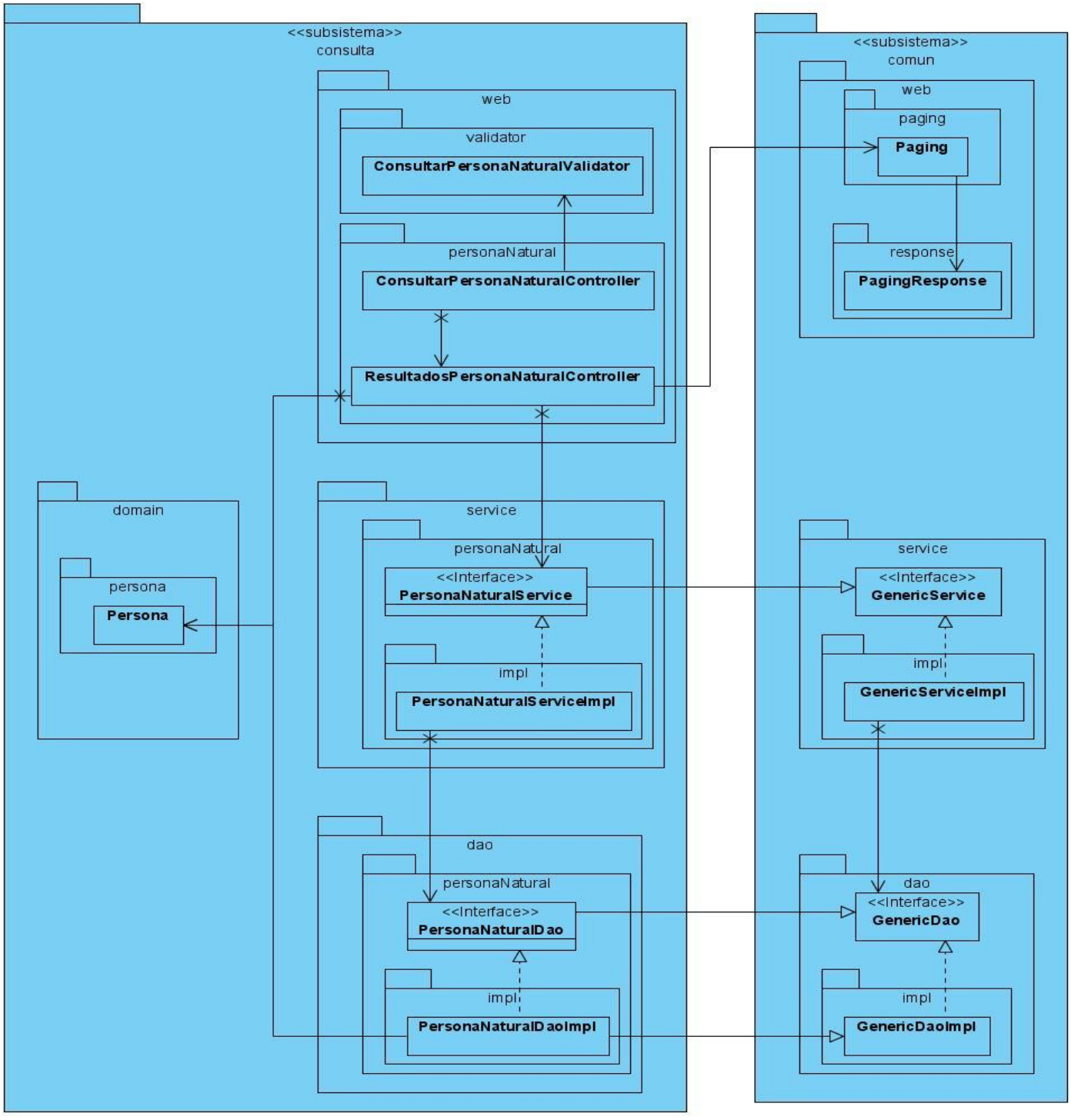


Figura 52. Diagrama de Clases del Diseño CU Consultar Persona Natural

Anexo 4: Diagramas de Contrato entre Paquetes

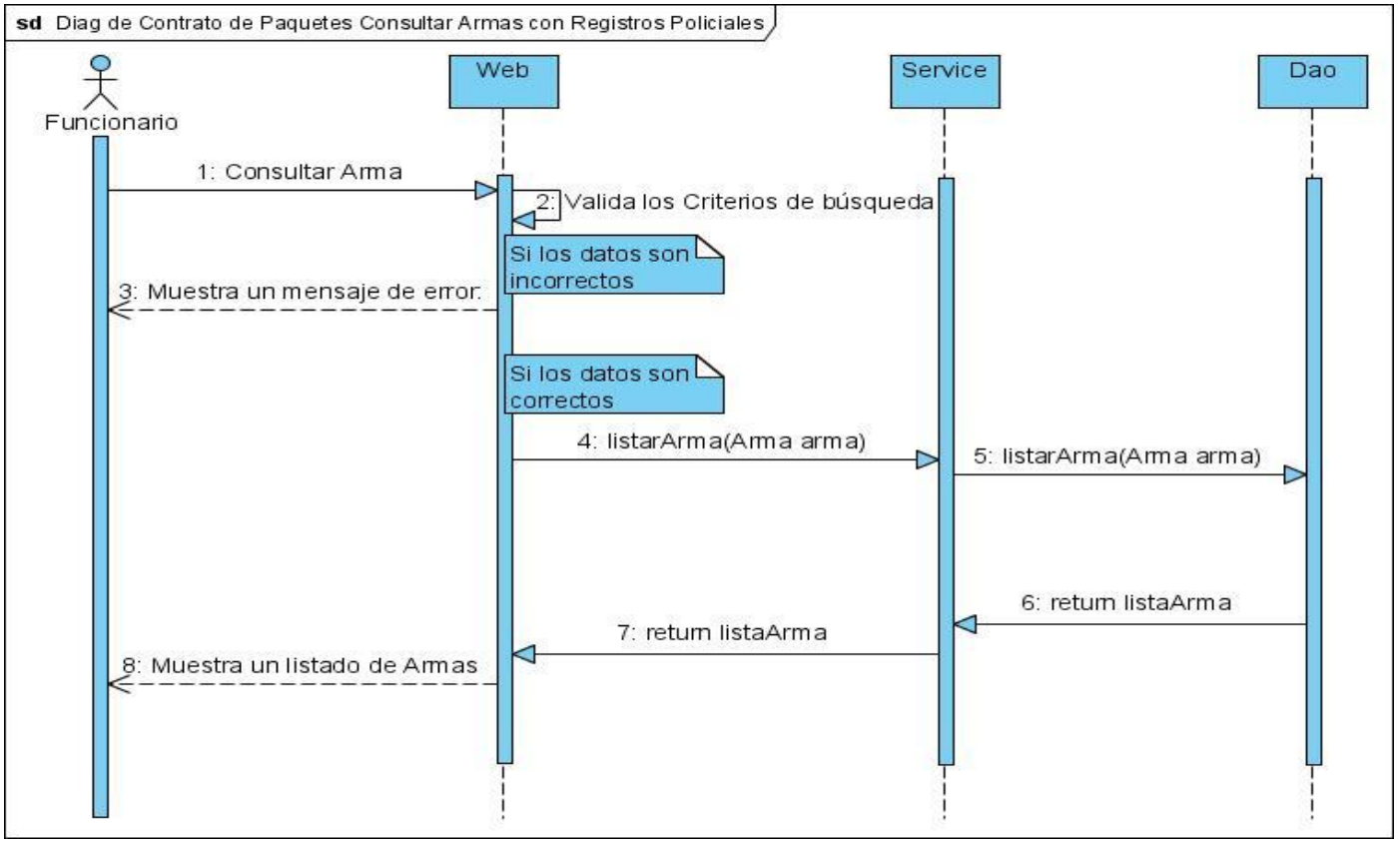


Figura 53. Diagrama de Contrato entre Paquetes CU Consultar Armas con Registros Policiales

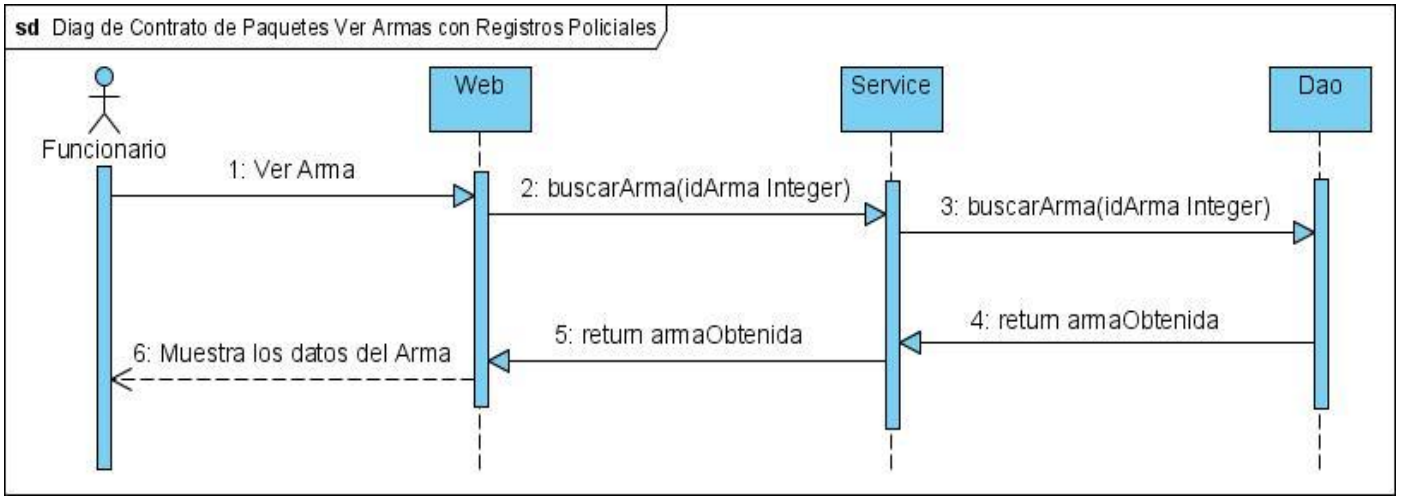


Figura 54. Diagrama de Contrato entre Paquetes CU Ver Detalles de Armas con Registros Policiales

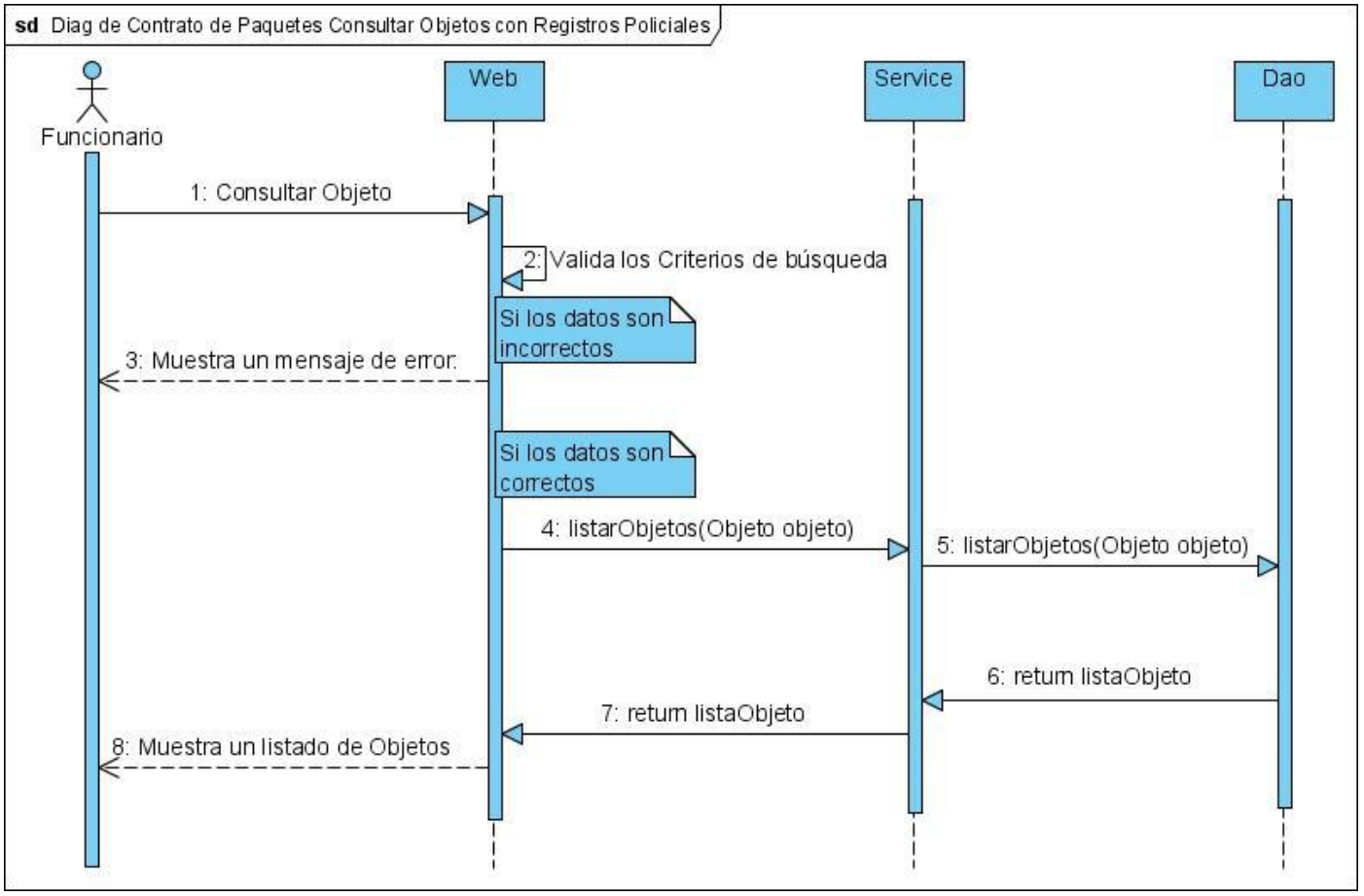


Figura 55. Diagrama de Contrato entre Paquetes CU Consultar Objetos con Registros Policiales

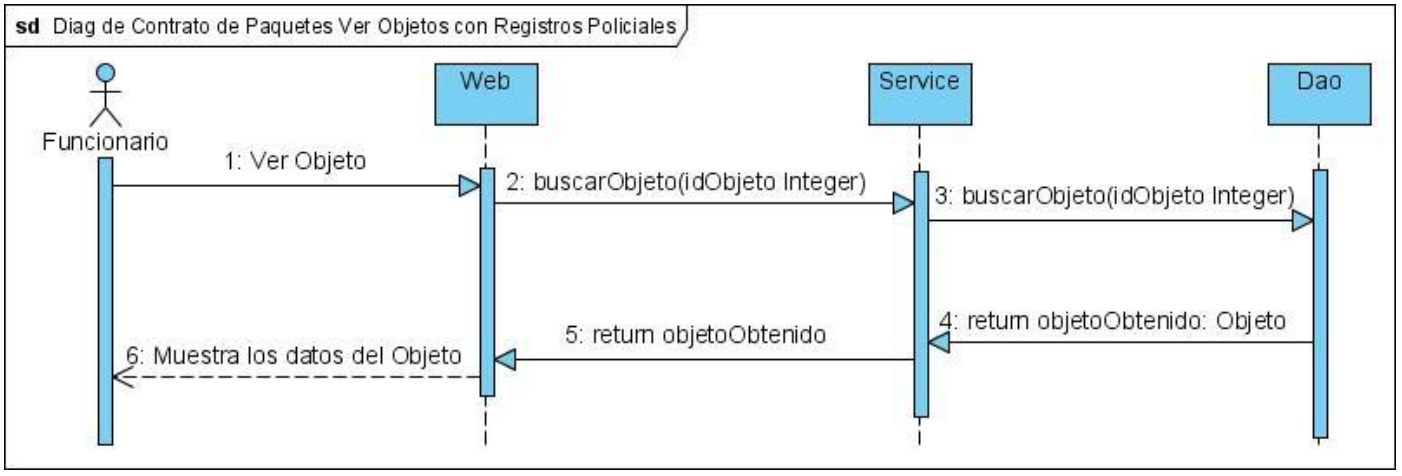


Figura 56. Diagrama de Contrato entre Paquetes CU Ver Detalles de Objetos con Registros Policiales

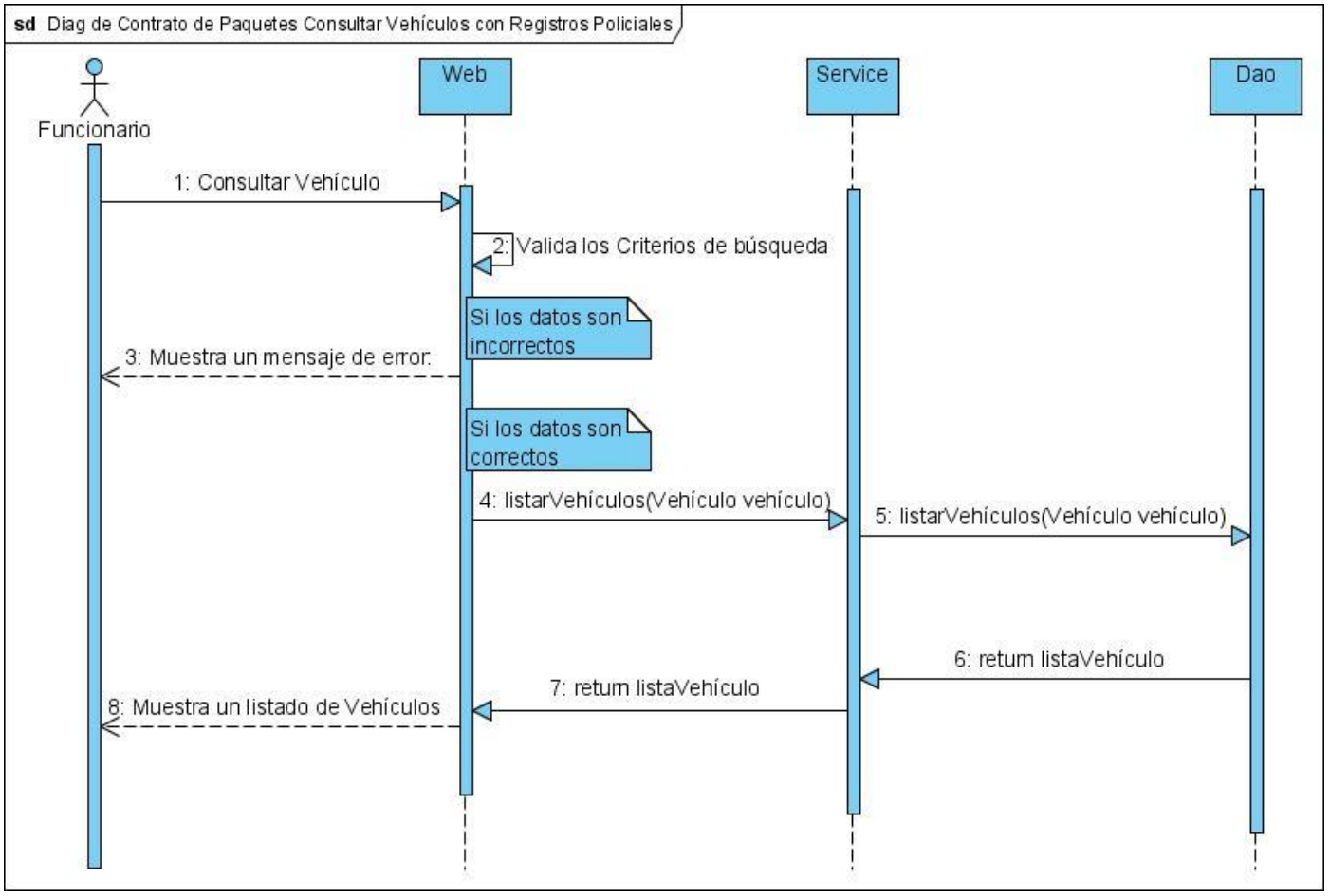


Figura 57. Diagrama de Contrato entre Paquetes CU Consultar Vehículos con Registros Policiales

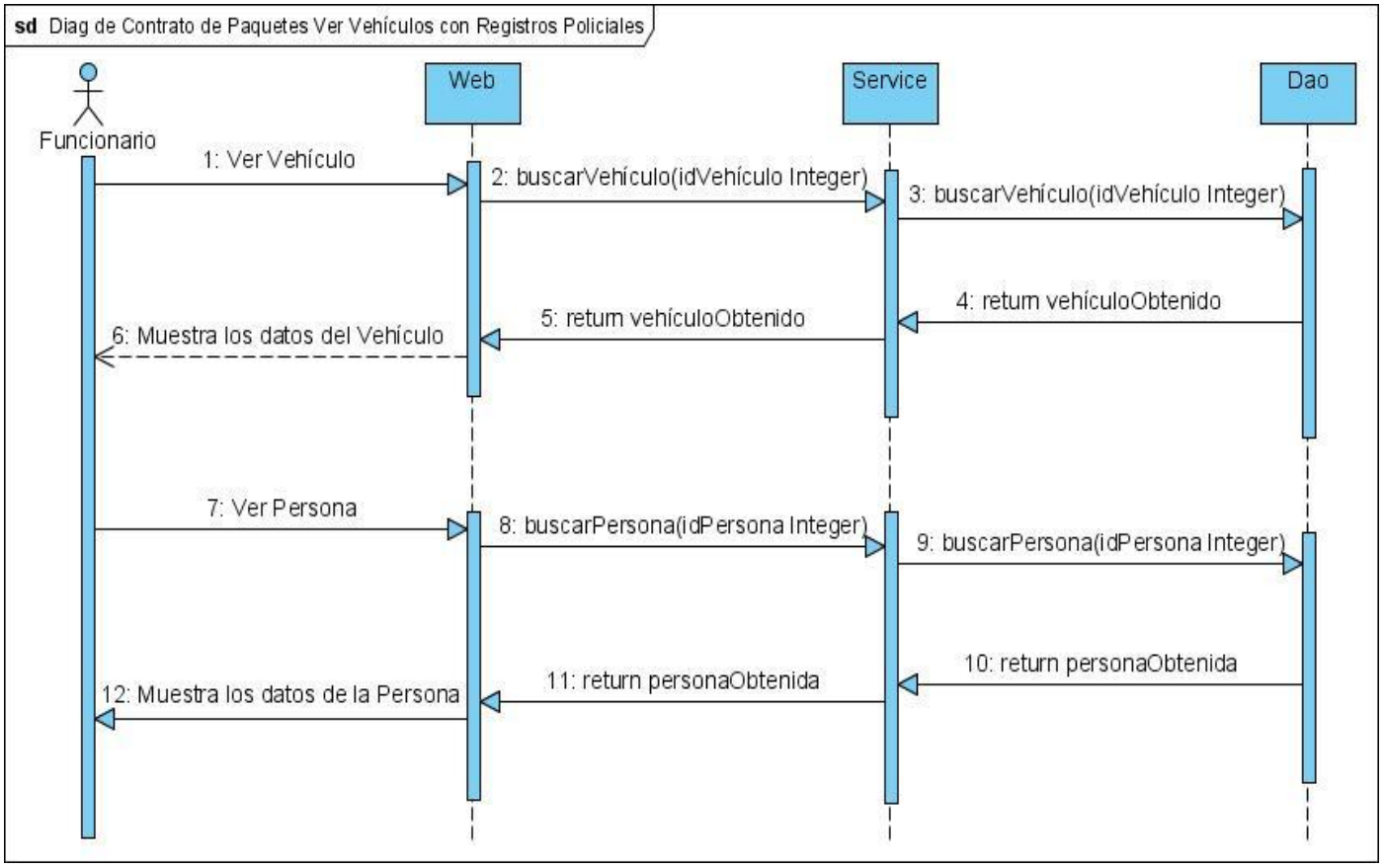


Figura 58. Diagrama de Contrato entre Paquetes CU Ver Detalles de Vehículos con Registros Policiales

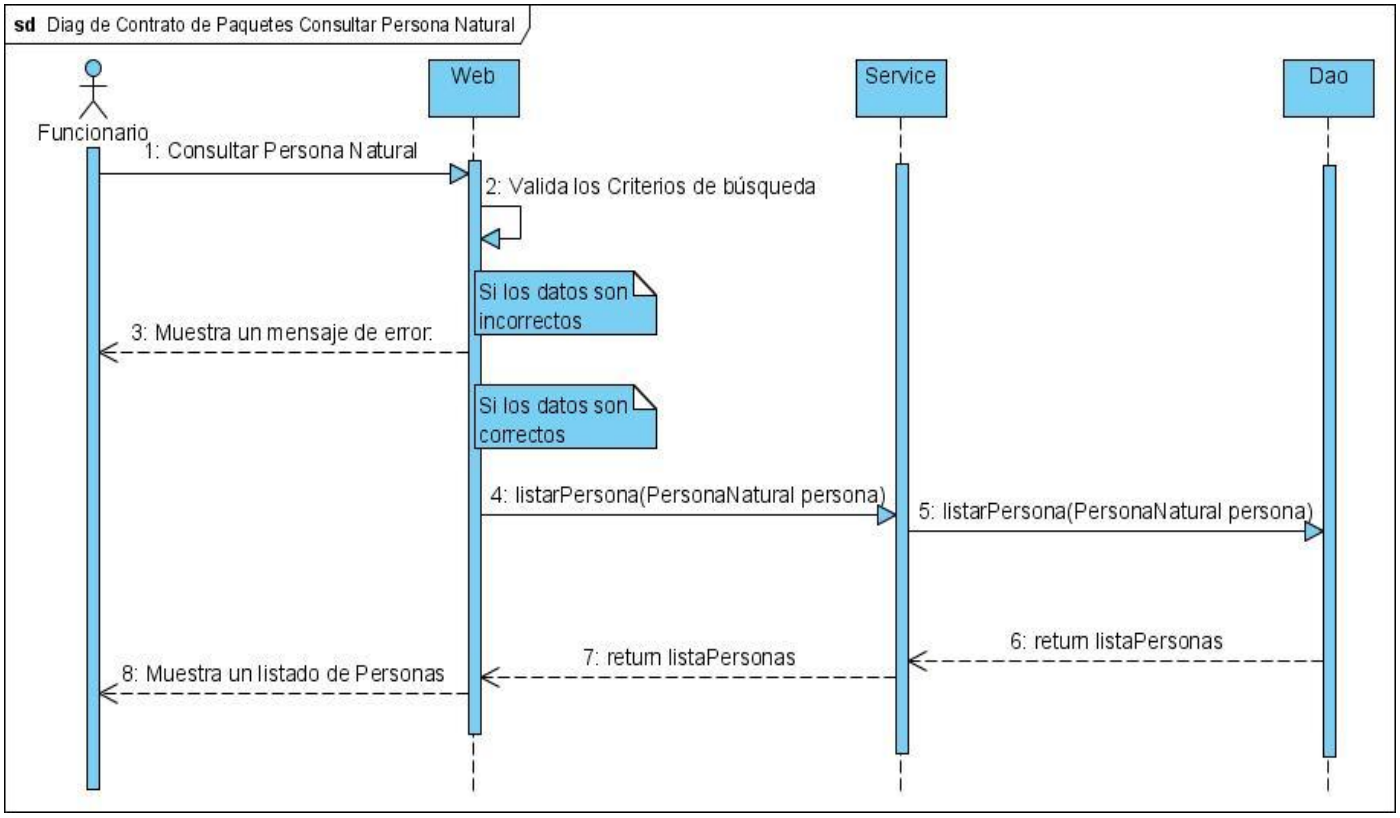


Figura 59. Diagrama de Contrato entre Paquetes CU Consultar Persona Natural

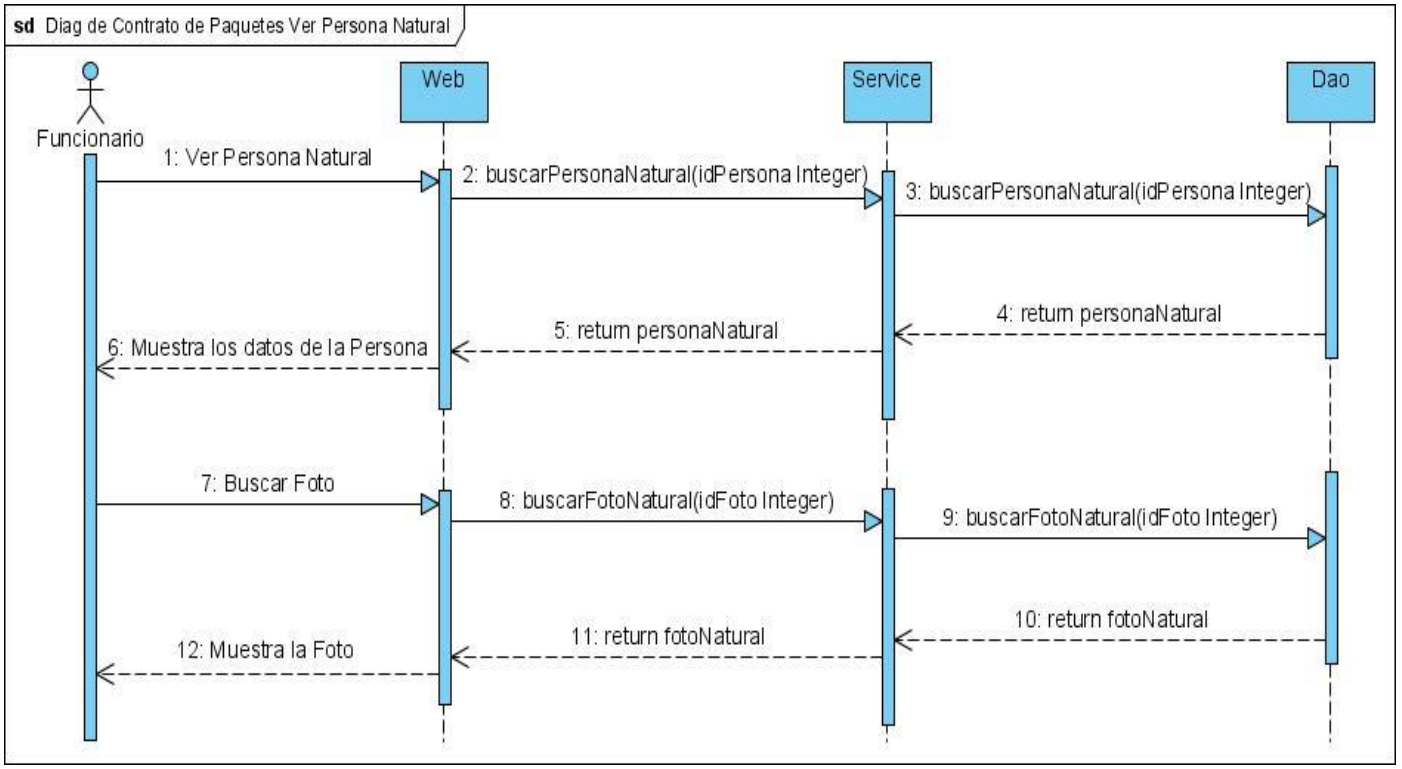


Figura 60. Diagrama de Contrato entre Paquetes CU Ver Detalles de Persona Natural

Anexo 5: Prototipos de interfaz del Portal WAP

Figura 61. Iniciar Sesión

Figura 62. Página de Inicio

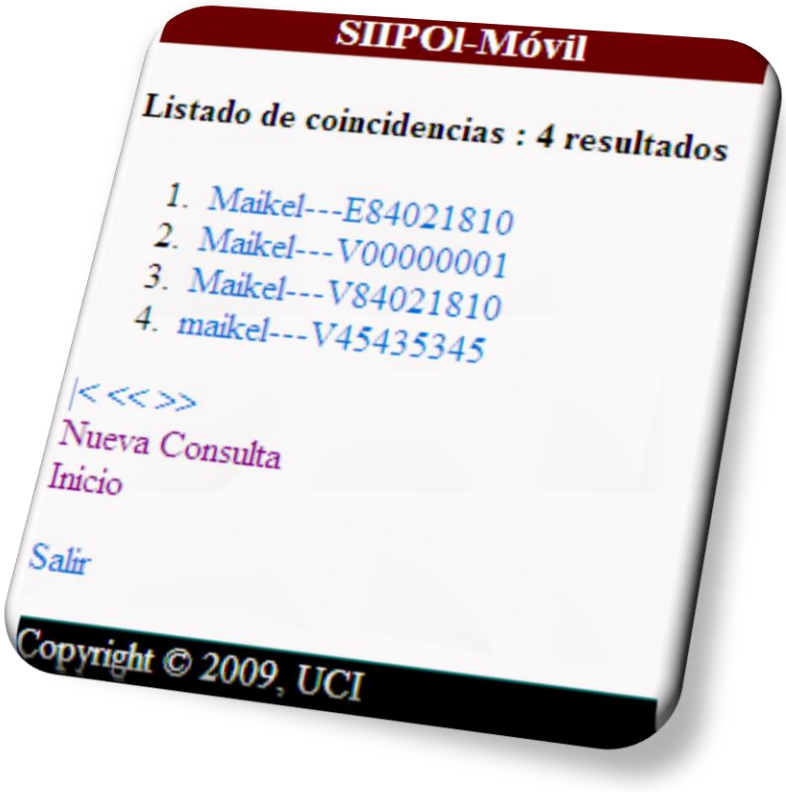


Figura 63. Listado de Resultados

Figura 64. Ver Foto de Persona

Anexo 6: Diagrama de Tablas del Modelo Relacional

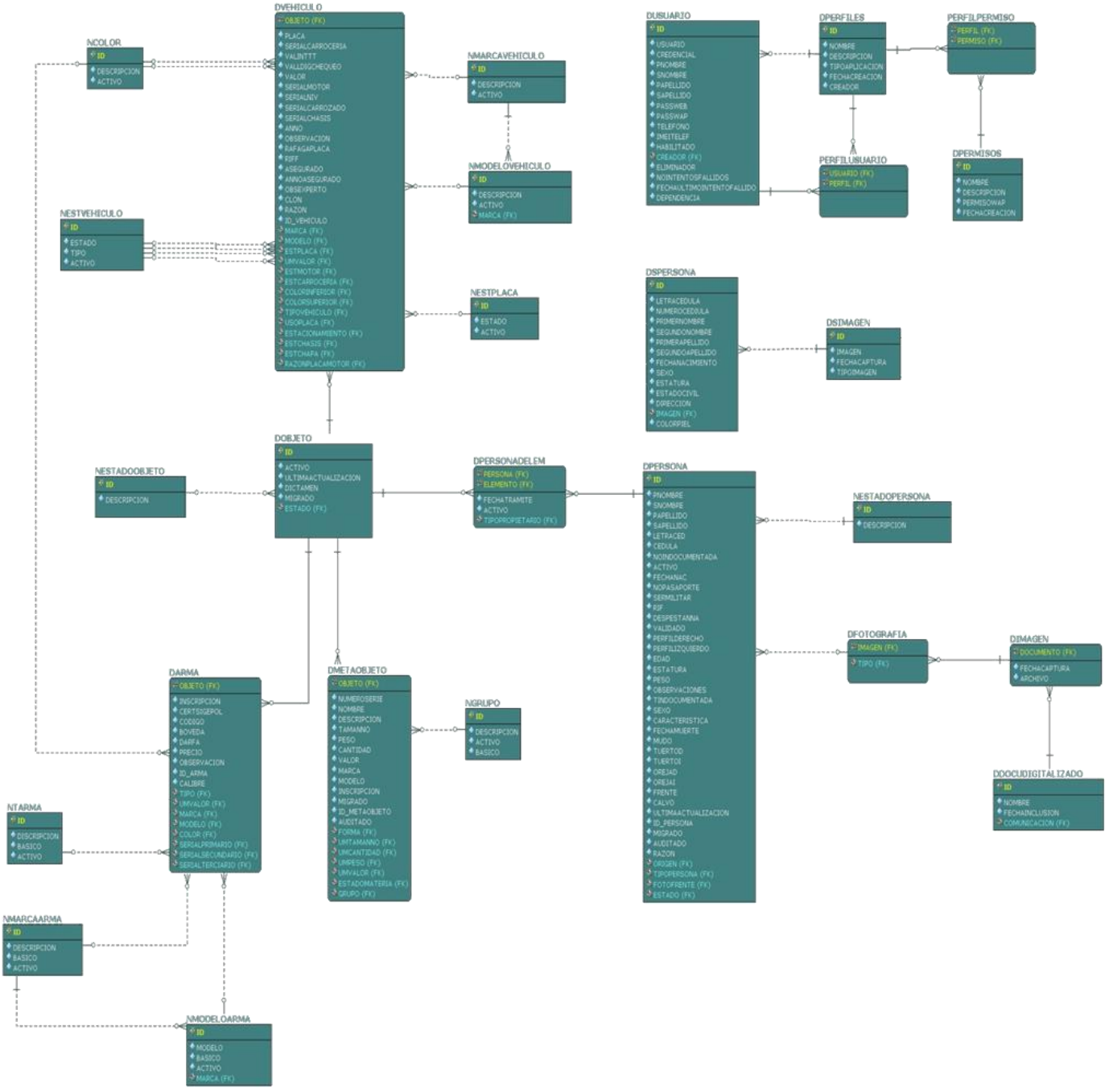


Figura 65. Modelo Relacional del Portal WAP