Universidad Central "Marta Abreu" de Las Villas.
Facultad Matemática, Física y Computación
Departamento de Ciencia de la Computación

Tesis presentada en opción al grado científico de Licenciado en Ciencia de la Computación

Servicios Web para la visualización de las denuncias en el Sistema SAJO

Autor:
YAIDEL ABREU GARCÍA

Tutores:
ING. ERNESTO M. RODRÍGUEZ RODRÍGUEZ
LC. ANAILYS HERNÁNDEZ SEIJO

Santa Clara
01 de julio de 2011
"Año 53 de la Revolución

DICTAMEN

Hago constar que el presente Trabajo de Diploma ha sido realizado en la facultad de Matemática, Física y Computación de la Universidad Central “Marta Abreu” de Las Villas (UCLV) como parte de la culminación de los estudios de Licenciatura en Ciencia de la Computación, autorizando a que el mismo sea utilizado por la institución para los fines que estime conveniente, tanto de forma total como parcial y que además no podrá ser presentado en eventos ni publicado sin la previa autorización de la UCLV.

_____

Firma del Autor

Los firmantes a continuación, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del tutor

Firma del jefe del Laboratorio

DEDICATORIA

"A mis padres y mi abuelita

Que siempre están ahí cuando los necesito

Impulsándome para alcanzar mis logros"

AGRADECIMIENTOS

Les agradezco primeramente a mis padres por darme la vida y por que sin ellos no fuera quien soy.

A mi abuelita que me vio crecer y me guio por el buen camino.

A mi Milgre y José que siempre nos estamos jodiendo mutuamente.

A mi hermano que siempre ha sido un guía a seguir.

A Lisi mi querida novia que también fue autora de esta investigación ya que sin ella no hubiera podido ser posible.

A Liamcito que no ha dado sus primeros pasos pero ya empieza a ser una parte importante en mi vida.

A Marianela y Orlando que ya también son parte de mi familia.

A mis tutores Ernesto y Ana por ser parte importante y fundamental en este trabajo.

A todos y cada uno de los profesores pertenecientes a la Facultad de Matemática, Física y Computación de la Universidad Central de la Villas, quienes muy profesionalmente dedicaron su tiempo y esfuerzo a compartir sus conocimientos para hacernos menores profesionales.

Al Departamento de Informática y Comunicaciones del MININT por darme su apoyo incondicional día a día sin importar el nivel de trabajo que tuvieran.

A los Licenciados en CiberDominó Sergio, Addel, Carlos, Oscar, Luis que empezamos juntos esta batalla y demostramos que si se podía aunque diéramos varios tropezones para lograrlo.

RESUMEN

El Sistema Automatizado Jurídico Operativo (SAJO) constituye un sistema operacional para el registro y control de la denuncia utilizado por la policía y por los demás órganos operativos del MININT, sin embargo la información que proporciona se obtiene actualmente de forma disgregada y con una visión no integradora, dificultándose su utilización por parte de otras aplicaciones. La presente investigación fue concebida para desarrollar servicios Web que permitan su integración con los sistemas que lo requieran.

Para lograr este objetivo se diseñó una aplicación Web utilizando el lenguaje unificado de modelado (UML), con el apoyo de la herramienta Rational Rose Enterprise Edition. En la implementación del sistema se emplea el framework ADF, como servidor de mapas MapViewer, como entorno de desarrollo 0racle Jdeveloper y como gestor de base de datos 0racle.

Con su diseño y posterior implementación, se logra la utilización del sistema por otras aplicaciones, mostrando una ficha detallada y útil de una denuncia en particular, ubicándola geográficamente y reportando estadísticas referentes a denuncias relacionadas con el criterio de búsqueda. Además de ser consultado a través de servicios Web, contribuyendo a la agilización de la labor de la persona encargada de trabajar con el levantamiento de las denuncias y el sistema policial en general.

ABSTRACT

The Operative Law Automated System (SALO) is an operational system for the registration and control of the accusation used by the police and for the other operative organs of the Ministery of Internal Affairs, however the information that and provides is olated and with a non integrative vision, which hinderes its use on the part of other applications. The present research work was conceived to develop Web services that allow its integration with required systems.

To fulfill this objetive e Web using the unified language of modeling (UML), with the support of the tool Rational Rose Enterprise Edition for their design. In the implementation of the system the ADF framework is used, as maps servers Mapviewer, as development environment 0racle jdeveloper and a data base manager 0racle.

With their design and implementation, the system is used by other applications, showing a detailed and useful record of an accusation in particular, locating it geographically and reporting statistics of accusations related with the search criteria. It is consulted through Web services, contributing to facilitate the person's work in change of dealing whith the accusations wich benefits the Police System in general.

TABLA DE CONTENIDOS

LISTA DE FIGURAS

LISTA DE TABLAS

INTRODUCCIÓN

Con el desarrollo tecnológico desplegado en la actualidad cada día es mayor la producción de información y la rapidez con la que se propaga la misma. Ello constituye un reto para todas las organizaciones a nivel mundial, las cuales deben mantenerse actualizadas para no perder una enorme cantidad de tiempo y esfuerzo en la búsqueda de la información necesaria para la toma de decisiones. De esta manera, con la introducción de las nuevas tecnologías, la rapidez en el intercambio de datos y la necesidad de mejorar su circulación en los diferentes niveles de la empresa, los sistemas informáticos se colocan en una posición clave dentro de las organizaciones.

En cualquier entidad donde se ejecuten labores que generen información que requiera ser almacenada y procesada, resulta fundamental el uso de alguna herramienta que facilite estos trabajos, siendo los sistemas informáticos las alternativas viables más adecuadas para ayudar en la manipulación y procesamiento de la misma (Marques, 2000).

Cuba, aún siendo un país que se encuentra en vía de desarrollo, ha incursionado en el avance de las Tecnologías de la Información y las Comunicaciones (TIC). En este sentido el país se encuentra inmerso en un proceso de crecimiento y extensión, que ha repercutido en el desarrollo paulatino de sistemas informáticos en las diferentes áreas de la nación.

Particularmente, el Ministerio del Interior ha ido transformando su estructura, adaptándose a los cambios de la sociedad, sin modificar su línea directriz. Para lograr con mayor eficiencia el propósito por el cual fue creado este organismo, fue necesario que el ministerio no se enajenara del desarrollo de las (TIC), razón por la cual las incorporó a su desempeño cotidiano.

La especialidad de Informática y Comunicaciones del Ministerio del Interior durante muchos años se ha visto enfrascado en el desarrollo de una serie de sistemas para facilitar el trabajo de sus miembros en los diferentes órganos que lo componen, además de extender y modernizar la amplia red que enlaza todas

las entidades del país y de la provincia de Villa Clara específicamente (Fernández, 2011).

Existen diferentes aplicaciones informáticas que automatizan procesos de trabajo de las líneas operativas, dentro de estas se encuentra el Sistema Automatizado Jurídico Operativo (SAJO) el cual constituye un sistema operacional para el registro y control de la denuncia.

El procesamiento y análisis de esta información posibilita viabilizar el trabajo de las líneas operativas en cuanto al esclarecimiento de los hechos, así como la obtención de parámetros que permiten evaluar la eficiencia y eficacia del trabajo de enfrentamiento.

El SAJO es utilizado por la policía y también por los demás órganos operativos, a partir del mismo se registra todo el contenido de la denuncia, como el hecho, los procesados (autores o sospechosos), no procesados(denunciantes, testigos y víctimas) y objetos(objetos sustraídos en el caso de robos); sin embargo esta información de interés para el trabajo del MININT se obtiene en el sistema actual de forma disgregada y con una visión no integradora, dificultándose su utilización por parte de otras aplicaciones que lo requieren, siendo esta la situación problémica de la presente investigación.

Por ello se plantea la necesidad de contar con una herramienta(aplicación) que permita visualizar el contenido de las denuncias después de haberlas introducido por el sistema que hoy se encarga del registro de las denuncias: SAJO.

Teniendo en cuenta la innegable capacidad de los sistemas Web para manipular, procesar información y mostrar los resultados de manera que se facilite su visualización e interpretación, se considera que esta tecnología constituye una promisoria línea de trabajo. Definida la arista a seguir se pretenderá crear una aplicación Web en la presente investigación provista de un ambiente o interfaz de trabajo de fácil comprensión y manejo para el usuario, que cumpla con los estándares Web actuales y con las exigencias de la entidad a la cual pertenece. Este Sistema Automatizado confeccionado

proyectará a poder ser utilizado por otras aplicaciones, mostrando una ficha detallada y útil de una denuncia en particular, ubicándola geográficamente y reportando estadísticas referentes a denuncias relacionadas con el criterio de búsqueda. Además podrá ser consultado a través de servicios Web.

Como resultado de esta aplicación se contribuiría a la agilización la labor de la persona encargada de trabajar con el levantamiento de las denuncias (el carpeta) además de beneficiarse el Sistema Policial en general. Debido a que la seguridad Pública es un bien que afecta a todos los integrantes de la comunidad organizada, se puede reconocer como beneficiarios indirectos a todo el conjunto social.

A partir de lo antes expuesto se puede definir como **problema científico** a resolver en la presente investigación**:**

¿Como desarrollar servicios Web que permitan la integración del SAJO con los sistemas que lo requieran?

Para ofrecer respuesta al mismo se plantea como objetivo general:

Desarrollar servicios Web que permitan la integración del SAJO con los sistemas que lo requieran.

Objetivos Específicos:

1. Definir los requerimientos necesarios para el diseño del sistema propuesto a partir de un análisis de la situación actual del SAJO.
2. Realizar el diseño de los servicios que permitan la integración del SAJO con los sistemas que lo requieran.
3. Implementar los servicios que permitan la integración del SAJO.
4. Garantizar la interoperabilidad de los servicios con las aplicaciones existentes.
5. Diseñar y aplicar las pruebas que validen la propuesta de solución.

Preguntas de investigación:

1. ¿Cuáles son los requerimientos necesarios que se deben considerar para el correcto diseño de la aplicación Web?
2. ¿Qué técnicas y procedimientos son los más efectivos a utilizar para el correcto diseño e implementación de la aplicación Web?
3. ¿Cómo garantizar la interoperabilidad de los servicios con las aplicaciones existentes?
4. ¿Cómo diseñar las pruebas necesarias para validar la propuesta de solución?

Justificación de la investigación

Es fundamental para el logro de un desempeño laboral más eficiente por parte del personal del MININT encargado del registro de denuncias, una visualización integradora de todo su contenido, incluyendo: el hecho, los procesados (autores o sospechosos), no procesados (denunciantes, testigos y victimas) y objetos (objetos sustraídos en el caso de robos). El Sistema propuesto posee un gran impacto para la institución, pues brinda la posibilidad de visualizar el contenido de las denuncias después de haber sido introducidas por el sistema que hoy se encarga del registro de la denuncia: SAJO, garantizando además que la aplicación pueda ser empleada en otras aplicaciones que necesiten mostrar su contenido.

A continuación se muestran los **métodos científicos** que se emplearán:

Métodos teóricos:

Analítico-Sintético*:* A través de este método se definieron los principales elementos que caracterizan al sistema SAJO, así como las herramientas utilizadas en el desarrollo de este sistema.

Modelación: Esta se llevará a cabo mediante el análisis y diseño del proceso de planificación a través de las diferentes herramientas que se seleccionen.

Métodos empíricos:

Entrevista: Se realizaron entrevistas a los compañeros que controlan y supervisan los Sistemas Informáticos en explotación en el MININT, con el objetivo de seleccionar los principales requisitos que debe cumplir el Sistema.

Estructura de la Tesis:

El presente trabajo estará estructurado en 4 capítulos. A continuación se muestra una breve descripción de cada uno de ellos.

Capítulo 1 "Fundamentación Teórica": se incluye un estado del arte de la investigación, así como las metodologías, técnicas de programación y herramientas utilizadas para el desarrollo de la aplicación.

Capítulo 2 "Características del Sistema": se brinda una visión del Sistema, se definen los requisitos funcionales y no funcionales a los que se debe dar cumplimiento y se ofrece además la propuesta del Sistema a desarrollar.

Capítulo 3 "Diseño del Sistema": se realiza el diseño y descripción de la arquitectura del Sistema, se presentan además los diagramas de clases y el diseño de la base de datos.

Capítulo 4 "Implementación y Pruebas del Sistema": se realizan los diagramas de despliegue y componentes como resultado de la implementación del Sistema, además de las pruebas realizadas una vez concluido el desarrollo de la aplicación.

CAPITULO I. FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se detallan importantes temas a tener en cuenta para el desarrollo del trabajo. Se especifican conceptos como el delito, análisis del delito y denuncia los cuales están directamente vinculados con el tema a investigar. Se presentará un estudio de las herramientas existentes que se encargan actualmente del registro y control de la denuncia. Además de seleccionar las metodologías, herramientas y lenguajes necesarios para el desarrollo del Sistema.

1.2 Delito

El delito, en sentido estricto, es definido como una conducta, acción u omisión típica (tipificada por la ley), antijurídica (contraria a Derecho), culpable y punible. Supone una infracción del Derecho penal, es decir, una acción u omisión tipificada y penada por la ley. La palabra delito deriva del verbo latino “delinquere”, que significa abandonar, apartarse del buen camino, alejarse del sendero señalado por la ley. La definición de delito ha diferido y difiere todavía hoy entre escuelas criminológicas. Alguna vez, especialmente en la tradición, se intentó establecer a través del concepto de derecho natural, creando por tanto el delito natural. Hoy esa acepción se ha dejado de lado, y se acepta más una reducción a ciertos tipos de comportamiento que una sociedad, en un determinado momento, decide punir. Así se pretende liberar de paradojas y diferencias culturales que dificultan una definición universal(Fuenteseca, 1997).

1.3 Análisis del Delito

De acuerdo con Steven Gottlieb, en su libro "Análisis del Delito: desde el primer reporte al arresto final", el análisis del delito es definido como:

"Conjunto sistemático de procesos analíticos orientados a proveer información pertinente en tiempo y forma relativa a patrones y correspondientes tendencias delictivas para asistir al personal operacional y administrativo en la planificación de la distribución de los recursos orientados a la prevención y supresión de la

actividad criminal, ayudando al proceso investigativo e incrementando las aprehensiones y la clarificación de los casos."

1.4 Denuncia

La denuncia constituye un derecho de petición donde el denunciante no forma parte del procedimiento penal. Desde el momento en que interpone la denuncia da comienzo a un procedimiento que los órganos competentes llevarán a cabo de oficio. Esta no obliga a las autoridades a comenzar un proceso judicial, aunque pueden incurrir en infracciones administrativas o penales si no lo investigan con la debida diligencia sin un motivo fundado. Como características de este proceso de denuncia se tiene que el denunciante no tiene que aportar ninguna prueba a su denuncia. Sin embargo con el fin de que el órgano competente decida que realmente existen indicios que hacen necesario seguir investigando. Con la denuncia no se exige prestación de fianza en ningún caso(Fuenteseca, 1997).

En el MININT de la Provincia de Villa Clara el control de la denuncia se hace a través de las estaciones de policía de cada localidad, donde la misma es introducida en el SAJO mediante el carpeta.

1.5 Sistema SAJO

El Sistema Automatizado Jurídico Operativo constituye un sistema operacional para el registro y control de la denuncia utilizado por los oficiales de carpeta de las unidades de la Policía Nacional Revolucionaria (PNR). Este sistema se encarga de almacenar toda la información referente a una denuncia, las personas procesadas y las personas no procesadas, los modos de operar, entre otros datos de importancia para más tarde proceder con el esclarecimiento del delito cometido. Mediante el mismo se facilita, optimiza y se contribuye a la realización del trabajo que realizan los miembros del Ministerio del Interior encargados de esta labor de forma más eficiente. Sin embargo este Sistema no permite utilizar la información que se almacena en el mismo por otras aplicaciones, para consulta de usuarios que pertenecen a otras líneas de

enfrentamiento del Ministerio del Interior. Ello dificulta la toma de decisiones, en el análisis, y el estudio de determinados hechos delictivos, siendo necesaria la existencia de una aplicación Web que posibilite las acciones antes mencionadas.

1.6 Herramientas existentes

1.6.1 Aplicación de escritorio

Las aplicaciones de escritorio se caracterizan por llevar a cabo gran parte de las operaciones de negocios. Entre sus características resulta importante destacar que poseen gran seguridad debido a que el equipo donde se encuentra instalada puede contar con diferentes mecanismos de defensa frente a virus, troyanos, software espía, etc. Su rendimiento y velocidad de procesamiento son también un punto a favor, dado que de forma local aprovecha los recursos de la máquina de forma eficiente y eficaz. (Wikipedia en inglés, 2012).

1.6.2 Aplicación Web

Las aplicaciones Web son aquellas que los usuarios pueden utilizar accediendo a un servidor Web a través de internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores Web en la que se confía la ejecución al navegador.

Entre las ventajas que proporcionan las aplicaciones Web respecto a las aplicaciones de escritorio se pueden mencionar la administración centralizada y una mayor rapidez en su despliegue. También es válido destacar que sigue el paradigma de multiplataforma, ya que a través de un único navegador es capaz de acceder a la aplicación como tal y llevar operaciones como si se tratará de un programa más instalado en el equipo local. Otro de sus beneficios es que no dependen de ningún sistema operativo ni configuración de hardware específica, para su ejecución simplemente basta con teclear su dirección URL en cualquier navegador Web. Además, sus actualizaciones se hacen de una manera muy sencilla, sin necesidad de hacer descargas, instalaciones o comprar físicamente el producto (Wikipedia en inglés, 2012).

Esta flexibilidad ha sido uno de los principales motivos por los que cada vez son más utilizadas para una creciente diversidad de tareas y han sido seleccionadas para el desarrollo del presente estudio.

1.7 Servidores Web

Todo ordenador tiene un servidor o máquina dedicada, la cual está conectada a Internet para dar disponibilidad a sitios Web, cuando son solicitadas por usuarios de la red. Este ordenador o máquina, es conocida como servidor Web, aunque la palabra servidor identifica tanto al programa como a la máquina en la que dicho programa se ejecuta. El mismo acepta las peticiones "http" del navegador Web del usuario, y entrega las páginas Web que se visualizan, las cuales suelen ser páginas Web, hipertextos o páginas HTML: textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonidos. La comunicación de estos datos entre cliente y servidor se hace por medio un protocolo, concretamente del protocolo HTTP. Con esto, un servidor Web se mantiene a la espera de peticiones HTTP, que son ejecutadas por un cliente HTTP; lo que solemos conocer como un navegador Web. (Peralta, 2006).

Básicamente, cuando se navega por Internet, se está accediendo continuamente a cientos de servidores Web, los cuales constituyen almacenes de información en forma de páginas Web, dispuestas para su rápida entrega. (DUMLER, 2005).

Los servidores Web se han convertido en una tecnología que va muy de la mano junto al desarrollo progresivo y acelerado de la llamada red de redes, internet. Ello se debe en gran medida a que mediante estos, los usuarios en todo el mundo pueden recibir una serie de servicios de manera muy fácil y veloz. Algunos tipos de servidores Web son: Internet Information Servers (IIS), Apache y WebLogic.

1.7.1 WebLogic

WebLogic Server utiliza tecnologías de la plataforma Java 2, Enterprise Edition (J2EE). Esta última es la plataforma estándar para desarrollar aplicaciones multicapa basadas en el lenguaje de programación Java. Las aplicaciones J2EE están basadas en componentes estandarizados y modulares para los cuales WebLogic Server proporciona un conjunto completo de servicios y maneja automáticamente muchos detalles del comportamiento de la aplicación, sin requerir programación. También proporciona características esenciales para desarrollar y desplegar

aplicaciones críticas de comercio electrónico a través de entornos de computación distribuidos y heterogéneos. Entre estas características es posible citar: estándares de Liderazgo-Soporte Comprensivo de Java Enterprise para facilitar la implementación y despliegue de componentes, máxima flexibilidad en el desarrollo y despliegue de cualquier aplicación, escalabilidad de comercio electrónico empresarial, seguridad lista para el comercio electrónico, administración robusta y ricas opciones de cliente (Kolakowski, 2009).

WebLogic centraliza los servicios de aplicación como funciones de servidor Web, componentes del negocio, y acceso a los sistemas "backend" de la empresa. Utiliza tecnologías como el almacenamiento en memoria inmediata y almacenes de conexiones para mejorar la utilización de recursos y el funcionamiento de la aplicación.

En esta investigación será empleado WebLogic como servidor de aplicación para la realización del sitio Web por las ventajas antes mencionadas y porque utiliza tecnología en java. Otra de las causas se debe a que es política el uso de este servidor de aplicaciones en las entidades del MININT, escenario donde se desarrolla el estudio.

1.8 Metodologías para el desarrollo del Software

Cada vez con mayor frecuencia la producción de software busca adecuarse más a las necesidades del usuario. Para lograr la productividad del software se requiere un proceso que integre las diferentes fases de su desarrollo, motivo por el cual es de vital importancia seleccionar la metodología más adecuada para alcanzar los resultados deseados.

Existen muchas metodologías que han ido naciendo con el fin de dar soporte al ciclo de desarrollo de un sistema, estas se pueden dividir en dos grandes grupos: metodologías tradicionales (también denominadas pesadas) y metodologías ágiles.
Las metodologías tradicionales constituyen la artillería de peso para afrontar la realización de cualquier sistema informático, con independencia de su complejidad y tamaño, constan de un sistema organizado de artefactos y procedimientos. Dentro de este tipo de metodologías se encuentran: MÉTRICA (Pelaez Sanches, 2008),

MERISE (Matheron, 1990), SSADM (Cutts, 1987), ADOOSI(S. y. H. Alvarez, A, 2000) y RUP(Jacobson, 2000).

Las metodologías ágiles forman parte del movimiento de desarrollo ágil de software, el cual se basa en la adaptabilidad ante cualquier cambio como medio para aumentar las posibilidades de éxito de un proyecto**.**

Algunas de las metodologías más utilizadas son Programación Extrema (XP) y el Proceso Unificado de Desarrollo de Software (RUP), siendo esta última la seleccionada para guiar el proceso de automatización del sistema.

1.8.1 Proceso unificado de desarrollo (RUP)

El proceso unificado de desarrollo (RUP) es el resultado de la evolución e integración de diferentes metodologías de desarrollo de software. Este permite sacar el máximo provecho de los conceptos asociados a la orientación a objetos y al modelado visual, disponiendo de las mejores prácticas del modelo de desarrollo de un software en particular entre las cuales se encuentran:

- Desarrollo de software de forma iterativa.
- Manejo de requerimientos.
- Utiliza arquitectura basada en componentes.
- Modela el software de forma visual, usando UML.
- Verifica la calidad del software.
- Controla los cambios.
- Dirige las tareas de cada desarrollador por separado y del equipo como un todo (Pressman, 2002).

Es uno de los procesos más generales que existe, está enfocado a cualquier tipo de proyecto aunque no sea de software, basándose en la documentación generada en cada una de sus cuatro fases de desarrollo de software:

1. Inicio: El Objetivo en esta etapa es determinar la visión del proyecto.
2. Elaboración: En esta etapa el objetivo es determinar la arquitectura óptima.

3. Construcción: En esta etapa el objetivo es obtener la capacidad operacional inicial.

4. Transición: El objetivo es llegar a obtener una versión del proyecto.

La metodología RUP agrupa sus actividades en grupos lógicos definiendo así 9 flujos de trabajo, los 6 primeros conocidos como flujos de trabajo principales, y los otros 3 de apoyo.

Esta metodología es la más adecuada para el desarrollo del Sistema Informático debido a que permite mitigar en fases tempranas posibles riesgos en el proyecto, hace uso del modelado visual y de la orientación a objetos de una manera abarcadora y completa y puede alcanzar un buen grado de certificación en el desarrollo del software.

1.9 Lenguaje de modelado.

1.9.1 Lenguaje unificado de modelado (UML)

Para llevar a cabo cualquier proyecto de ingeniería resulta necesario considerar las etapas de modelamiento que posibiliten experimentar y visualizar el sistema que se construirá. Un lenguaje de modelado, de acuerdo con Rumbauhg (2000) es considerado como una estandarización de notaciones y reglas, que permitan diagramar o graficar un sistema, o parte de él. La elección de un aceptado lenguaje de modelado tiene gran significación, pues un buen modelamiento del software posee una influencia decisiva en la comunicación entre los desarrolladores y los clientes.

UML (Lenguaje de Modelación Unificado) es un lenguaje de modelación que se dirige por casos de usos, una técnica muy eficaz para la determinación de las necesidades del sistema. Según MULLER (1997) dicha notación ha sido pensada para servir de lenguaje de modelado de objetos, independientemente del método implementado. El mismo se percibe como un elemento básico en la estrategia de grandes organizaciones, no es una herramienta propietaria, sino que es accesible para todos, y los fabricantes de herramientas, al igual que las empresas de formación, pueden usarla libremente. UML define varios modelos para la

representación de los sistemas, algunos de estos modelos son: el de clases que captura la estructura estática, el de estados que expresa el comportamiento dinámico de los objetos, el de casos de uso que describe las necesidades del usuario, el de interacción que representa los escenarios y los flujos de mensajes, el de realización que muestra las unidades de trabajo, el de despliegue que precisa el reparto de procesos(Hernández, 2004). Otra característica de este modelado visual es que es independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML se pueden implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos).

Muller (1997) plantea alguna de las ventajas que posee UML, entre ellas: mayor rigor en la especificación y posibilidad de realizar una verificación y validación del modelo efectuado. Además, posibilita automatizar determinados procesos y generar código a partir de los modelos y a la inversa, es decir, a partir del código fuente generar los modelos. Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener la visión en el diseño, de más alto nivel, de la estructura de un proyecto

1.10 Plataforma de desarrollo

1.10.1 Herramienta CASE.

Se puede definir a las Herramientas CASE como un conjunto de programas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos o algunos de los pasos del ciclo de vida de desarrollo de un producto de software. Estas constituyen la unión de las herramientas automáticas de software y las metodologías de desarrollo de software formales(ESTADÍSTICA, 1999).

La popularidad que ha ido adquiriendo el modelado de sistemas de software con diagramas UML se encuentra relacionada con la existencia de herramientas CASE para UML que faciliten su gestión, permitiendo comunicar y compartir el conocimiento conceptual, de diseño, y de implementación de una arquitectura. (Fuster, 2006).

Las herramientas CASE también permiten a los analistas tener más tiempo para el análisis y diseño y minimizar el tiempo para codificar y probar. Actualmente existen

muchas herramientas CASE libres y comerciales en el mercado. En el caso de software libre se encuentran: Use Case Maker, ObjectBuilder, BoUml, Gaphor. Entre las comerciales es posible citar ejemplos como: ArgoUML, Rational Rose, Visual UML, ObjectDomain, seCAKE, Objecteering, MagicDraw. Es difícil establecer comparaciones entre unas y otras así como determinar cuál es mejor, por lo que la selección de la herramienta CASE apropiada depende del proyecto a desarrollar.

Visual Paradigm

Visual Paradigm para UML es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: negocio, análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Presenta un entorno visual de modelado que logra facilitar el diseño visual de manera dramática y ayuda a los equipos de desarrollo de software a sobrepasar el proceso de desarrollo de software que incluye modelación-construcción-despliegue, maximizando y acelerando las contribuciones tanto del equipo de desarrollo como las individuales.

Su Generación de Código e Ingeniería Inversa soportan un conjunto de lenguajes, entre ellos: Java, C++, CORBA IDL, PHP, XML Schema, Ada y Python, además de otros como C#, VB .NET, ODL, Flash ActionScript, Delphi, Perl, Objective-C, y Ruby. La ingeniera inversa también soporta clases de Java, DLL y EXE de .NET y JDBC (VALDÉS, 2007).

Esta herramienta es una de las más usadas por desarrolladores de todo el mundo, Además es colaborativa, ya que soporta múltiples usuarios trabajando sobre el mismo proyecto, genera la documentación del proyecto automáticamente en varios formatos como Web o pdf y posibilita el control de versiones. El software de modelado UML ayuda a aumentar la rapidez de la construcción de aplicaciones de calidad a un menor costo y permite modelar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.(VALDÉS, 2007)

1.10.2 Lenguaje de programación.

Con el avance y desarrollo de las tecnologías informáticas y la difusión de las redes de computadoras, los sistemas para la gestión de la información de las diferentes

organizaciones han evolucionado, constituyendo hoy en día uno de los más utilizados, los sistemas Web.

En la actualidad existen números lenguajes de programación empleados para el desarrollo de Aplicaciones Web en el servidor, entre los cuales podemos mencionar PHP, Perl, Ruby, Python, Java con sus tecnologías Java Servlets y Java Server Pages (JSP).

1.10.2.1 Java

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. Con respecto a la memoria, su gestión no es un problema ya que ésta es gestionada por el propio lenguaje y no por el programador. El mismo consta con tecnologías para el desarrollo de Aplicaciones Web como es el caso de JSP.

Java Server Pages

JSP es una tecnología multiplataforma, creada para desarrollarse al lado del servidor y desarrollada para la creación de aplicaciones Web potentes. Esta permite generar contenido dinámico para Web, en forma de documentos HTML, XML o de otro tipo. Entre sus características, según Valdés (2007), se encuentran:

- Código separado de la lógica del programa.
- Las páginas son compiladas en la primera petición.
- Permite separar la parte dinámica de la estática en las páginas Web.
- Los archivos se encuentran con la extensión (jsp).
- El código JSP puede ser incrustado en código HTML
- Código bien estructurado.
- La parte dinámica está escrita en Java.

Dadas estas facilidades es que se adopta este leguaje de programación Web como política de trabajo en el MININT, por lo que se decide utilizar para el desarrollo de la presente investigación.

1.10.3 Framework

Un framework o marco de trabajo, es una infraestructura que soporta un conjunto de conceptos, valores y prácticas, facilitando la construcción de aplicaciones. Provee una serie de herramientas y componentes que permiten modelar e implementar de manera natural la realidad.

En el desarrollo de un software, un framework es definido como una infraestructura de soporte en la cual un proyecto de software puede ser organizado y desarrollado. Regularmente, puede incluir soporte de programas, librerías y código prescrito para ayudar a desarrollar y elaborar los diferentes componentes del proyecto de software que se construye con dicho framework (Rivas, 2007).

Los componentes de software que posee un marco de trabajo para desarrollo de aplicaciones están diseñados para ayudar a la construcción y modularización de una aplicación, de forma sencilla y automática. Un marco de trabajo provee componentes definidos para ocupar lugares específicos en el proceso de desarrollo de soluciones de software, entre ellos: consultas, lógica del negocio y validaciones y manejo de la persistencia de datos.

Dichos componentes de software deben ser inteligentes y deben integrarse con todos los componentes del marco de trabajo, permitiendo la personalización de su comportamiento de forma simple para adecuarse a necesidades específicas. El marco de trabajo debe manejar también la mayoría de las tareas comunes que tengan determinada incidencia en el desarrollo de aplicaciones y que sean comportamientos estándares.

1.10.3.1 ADF

Oracle Application Development Framework, generalmente llamado Oracle ADF, es un framework comercial de Java para la creación de aplicaciones empresariales. Es una herramienta del tipo RAD que se basa en patrones de diseño listos para usar. Provee un enfoque visual y declarativo para el desarrollo de aplicaciones J2EE (Rivas, 2007).

Oracle ADF esta basado en cuatro capas: la capa de servicios del negocio (provee de acceso a datos desde diferentes fuentes y maneja la lógica de la aplicación), la capa del modelo (permite que la capa de la vista y la capa del controlador trabajen con diferentes implementaciones de los servicios del negocio de forma consistente), la capa del controlador (provee mecanismos para controlar el flujo de la aplicación) y la capa de la vista (provee la interfaz del usuario de la aplicación).

Sus principales características, que lo distinguen en comparación con otros frameworks de desarrollo J2EE, son las siguientes:

- Ambiente de desarrollo: es una herramienta poderosa para la construcción de aplicaciones utilizando este marco de trabajo, de forma visual y declarativa, por lo tanto reduce la necesidad de escribir código.
- Elección de tecnología: Los desarrolladores puede elegir entre una diversidad de componentes para implementar cada una de las capas de la aplicación.
- Oracle ADF es una solución para una capa. Provee soluciones completas, para cada una de las capas que J2EE específica y para todas las fases del ciclo de vida del desarrollo de software, desde el diseño hasta la construcción.
- Cualquier cliente que tenga una licencia de cualquiera de los servidores de Aplicaciones de Oracle, tiene incluida la licencia de ADF sin ningún tipo de costo adicional.

1.10.4 Entorno de desarrollo integrado.

Un entorno de desarrollo integrado o IDE, es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse de forma exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios.

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDE pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. El lenguaje Visual Basic, por ejemplo, puede ser usado dentro de

las aplicaciones de Microsoft Office, lo que hace posible escribir sentencias Visual Basic en forma de macros para Microsoft Word.

Los IDE proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, C#, Delphi, Visual Basic, JSP etc. Es posible que un mismo IDE pueda funcionar con varios lenguajes de programación. Este es el caso de Eclipse, al que mediante plugins se le puede añadir soporte de lenguajes adicionales(Bell & Mike).

1.10.4.1 Jdeveloper

Según Muench (2010) Jdeveloper es un (IDE) para la construcción de aplicaciones usando java, xml, sql, plsql, html, java script, etc. Este IDE permite el ciclo de vida de todo el desarrollo de software con herramientas integradas para modelar, programar, depurar, realizar pruebas, afinar y desplegar las aplicaciones. Es la plataforma principal de desarrollo de toda la plataforma de productos de la capa media de Oracle. Es además un IDE multiplataforma que funciona tanto en Windows, Linux, Mac OS X y otros sistemas operativos. Proporciona un enfoque de desarrollo visual y declarativo y trabaja de conjunto con la tecnología Oracle ADF para simplificar el desarrollo de aplicaciones.

1.10.5 Gestor de base de datos

Un sistema gestor de base de datos es concebido como el conjunto de programas que administran y gestionan la información contenida en una base de datos. De acuerdo con Álvarez (2007) permite realizar acciones como: definición y mantenimiento de la integridad de los datos dentro de la base de datos, además del control de la seguridad y privacidad de los mismos y su manipulación.

Sus principales componentes son: control de autorización, control de la integridad, optimizador de consultas, gestor de transacciones, planificador, gestor de recuperación y gestor de buffers.

Posee varios módulos, cada uno de los cuales realiza una función específica, entre ellos están:

- El procesador de consultas que transforma las mismas en un conjunto de instrucciones de bajo nivel que se dirigen al gestor de la base de datos.
- El gestor de la base de datos es el interface con los programas de aplicación y las consultas de los usuarios. Este acepta consultas y examina los esquemas externo y conceptual para determinar qué registros se requieren para satisfacer la petición.
- El gestor de ficheros maneja los ficheros en disco en donde se almacena la base de datos. Este gestor establece y mantiene la lista de estructuras e índices definidos en el esquema interno.
- El preprocesador del LMD convierte las sentencias del LMD embebidas en los programas de aplicación, en llamadas a funciones estándar escritas en el lenguaje anfitrión. Este debe trabajar con el procesador de consultas para generar el código apropiado.
- El compilador del LDD convierte las sentencias del LDD en un conjunto de tablas que contienen metadatos.
- El gestor del diccionario controla los accesos al diccionario de datos y se encarga de mantenerlo(MARQUÉS, 2001).

Existen varios de estos sistemas a nivel mundial que han sido de gran aceptación por parte de las diferentes comunidades de desarrollo, entre ellos se encuentran: PostgreSQL, MySQL, Microsoft SQL Server, Oracle

1.10.5.1 Oracle

Oracle es un sistema gestor de base de datos relacional extremadamente potente y flexible, dotado de cierta complejidad. Para poder diseñar aplicaciones útiles basadas en Oracle es necesario entender cómo este manipula los datos almacenados en el Sistema. El mismo se encuentra orientado al acceso remoto y a Internet, se asienta en diferentes arquitecturas en cuanto a procesadores (Intel, Alpha, Sparc, RISC) y puede ser implementado en diferentes plataformas: las

distintas versiones Windows, Linux, UNIX, siendo esta última para la cual fue diseñado inicialmente.

Como base de datos, Oracle ha acaparado casi todo el mercado de su sector, pues ofrece una solución integral para resolver cualquier cuestión empresarial, convirtiéndose en el software que casi todas las empresas están empleando para el asesoramiento a su propia toma de decisiones. Contiene una avanzada seguridad, los clientes pueden introducir de manera transparente todos los datos de sus aplicaciones o solo algunas columnas específicas de la base de datos, posibilitando un rendimiento mucho mayor que cualquier otra plataforma de Base de Datos. Es posible tener en todo momento control del crecimiento y rendimiento de los distintos esquemas que componen la base debido a que permite la asignación de zonas de memoria propias a sus datos y cualidades (*Características Oracle, 2009*).

Entre sus principales facilidades se encuentran: su transportabilidad (funciona sobre decenas de plataformas), la potencia de sus instrumentos de desarrollo de aplicaciones, la riqueza de su diccionario de datos, los mecanismos encargados de la seguridad y la confidencialidad y una experiencia probada sobre el terreno y una buena presencia a nivel de formación, consejo y soporte técnico.

Producto de las características antes mencionadas se ha decidido utilizar Oracle como sistema gestor de base de datos para el desarrollo del Sistema Informático. Además el Sistema Automatizado Jurídico Operativo (SAJO) se encuentra desarrollado en este Gestor de Base de Datos.

1.10.6 Servidor de Mapas

1.10.6.1 Oracle MapViewer

Mapviewer es una de las aplicaciones que compone la Capa Media de Oracle, contiene un motor de interpretación cartográfica que se encarga de establecer la conexión con la Base de Datos Oracle a través del JDBC; se comunica con el cliente a través de un explorador Web o una aplicación utilizando el protocolo HTTP; la base de datos incluye un paquete llamado Oracle Spatial para el trabajo con los

metadatos espaciales y gestionar todas las operaciones entre ellos(Rodríguez & Valdés, 2011).

1.11 Protocolo de Comunicación

1.11.1 HTTP

El protocolo de transferencia de hipertexto (HTTP) es un protocolo del nivel de aplicación usado para la transferencia de información entre sistemas, de forma clara y rápida. Este protocolo ha sido usado por el WWW (World Wide Web) desde 1990.

Este protocolo permite usar una serie de métodos para indicar la finalidad de la petición. Se basa en otros conceptos y estándares como Uniform Resourse Identifier (URI), Uniform Resouse Location (URL), para indicar el recurso al que hace referencia la petición. Los mensajes se pasan con un formato similar al usado por el Internet Mail y el Multipurpose Internet Mail Extensions (MIME).Generalmente es el cliente el que inicia la comunicación HTTP, consistente en la petición de un recurso del servidor que puede hacerse de forma directa o a través de intermediarios**.**

1.12 SOA

La Arquitectura SOA (Arquitectura Orientada a Servicios) establece un marco de diseño para la integración de aplicaciones independientes de manera que desde la red pueda accederse a sus funcionalidades, las cuales se ofrecen como servicios. La forma más habitual de implementarla es mediante Servicios Web, una tecnología basada en estándares e independiente de la plataforma, con la que SOA puede descomponer aplicaciones monolíticas en un conjunto de servicios e implementar esta funcionalidad en forma modular. Para una mejor comprensión del término es necesario establecer qué se entiende por servicio. Este se define como una funcionalidad concreta que puede ser descubierta en la red y que describe tanto lo que puede hacer como el modo de interactuar con ella.

1.12.1 Servicios Web

Los servicios Web son la forma más habitual de implementar SOA, estos son aplicaciones que utilizan estándares para el transporte, codificación y protocolo de

intercambio de información, permitiendo la intercomunicación entre sistemas de cualquier plataforma. Se basan en un conjunto de estándares de comunicación, como son XML para la representación de datos, SOAP (Simple Object Access Protocol) para el intercambio de datos y el lenguaje WSDL (WebServices Description Language) para describir las funcionalidades de un servicio Web (Microsoft, 2006).

1.13 Sistema de control de versiones

1.13.1 Subversión (SVN)

De acuerdo con GALÍNDEZ (2008), SVN es un sistema de control de versiones creado en el año 2000 el cual surge con el objetivo de ser una alternativa real y mejorada a CVS entre sus principales características podemos mencionar:

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.
- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente.
- Se envían sólo las diferencias en ambas direcciones.
- Permite selectivamente el bloqueo de archivos.

1.14 Conclusiones Parciales

En este capítulo se realizó una descripción panorámica general de la herramienta que se encarga del proceso de registro y control de la denuncia en la provincia de Villa Clara, además de haberse tratado los principales conceptos relacionados con el tema que permiten una mejor compresión de la problemática a tratar. Quedó reflejado además el estudio y análisis de la selección de las herramientas, metodologías y lenguajes necesarios para el desarrollo de la implementación del sistema.

CAPITULO II. PROPUESTA DEL SISTEMA

2.1 Introducción

En el presente capítulo se describen las características del sistema a desarrollar, se expone el resultado del proceso de captura de requisitos funcionales y no funcionales a los que se debe dar cumplimiento. Se identifican los actores que intervienen en el sistema y se modela el diagrama de casos de uso. Finalmente, se describirán todos los casos de uso para una mejor comprensión de las funcionalidades de la plataforma.

2.2 Objeto de estudio

2.2.1 Problema y situación problemática

El Sistema Automatizado Jurídico Operativo (SAJO) constituye un sistema operacional para el registro y control de la denuncia utilizado por la policía y también por los demás órganos operativos. La información que este ofrece resulta de interés para el trabajo del MININT, pero se obtiene de forma disgregada y con una visión no integradora, dificultándose su utilización por parte de otras aplicaciones que lo requieren.

2.2.2 Objeto de automatización

A partir de la situación problémica planteada con anterioridad se puede visualizar la necesidad de contar con una herramienta(aplicación) que permita visualizar el contenido de las denuncias después de haberlas introducido por el sistema que hoy se encarga del registro de las denuncias: SAJO.

2.2.3 Información que se maneja

Se maneja información de las Base de Datos SAJO y Mapinfovcl las cuales poseen todos los requerimientos que se necesitan de la denuncia objeto de interés, así como datos de otras denuncias relacionadas con la misma y estadísticas.

2.3 Propuesta del sistema

En la presente investigación se pretende crear una aplicación Web provista de un ambiente o interfaz de trabajo de fácil comprensión y manejo para el usuario, que cumpla con los estándares Web actuales y con las exigencias de

la entidad a la cual pertenece. Este sistema automatizado tiene el objetivo de ser utilizado por otras aplicaciones, mostrando una ficha detallada y útil de una denuncia en particular, ubicándola geográficamente y reportando estadísticas referentes a denuncias relacionadas con el criterio de búsqueda. Además podrá ser consultado a través de servicios Web.

2.4 Modelo del negocio

Para modelar el negocio se utilizó el modelo de dominio, ya que no se definen concretamente los procesos del mismo, se enfoca al uso de herramientas y tecnologías informáticas, por lo que se decide mostrar los principales conceptos y objetos que se manejan en el dominio de la aplicación. Este modelo contribuye posteriormente a identificar algunas clases que se utilizarán en el sistema, permitiendo comprender el contexto en que se enmarca.

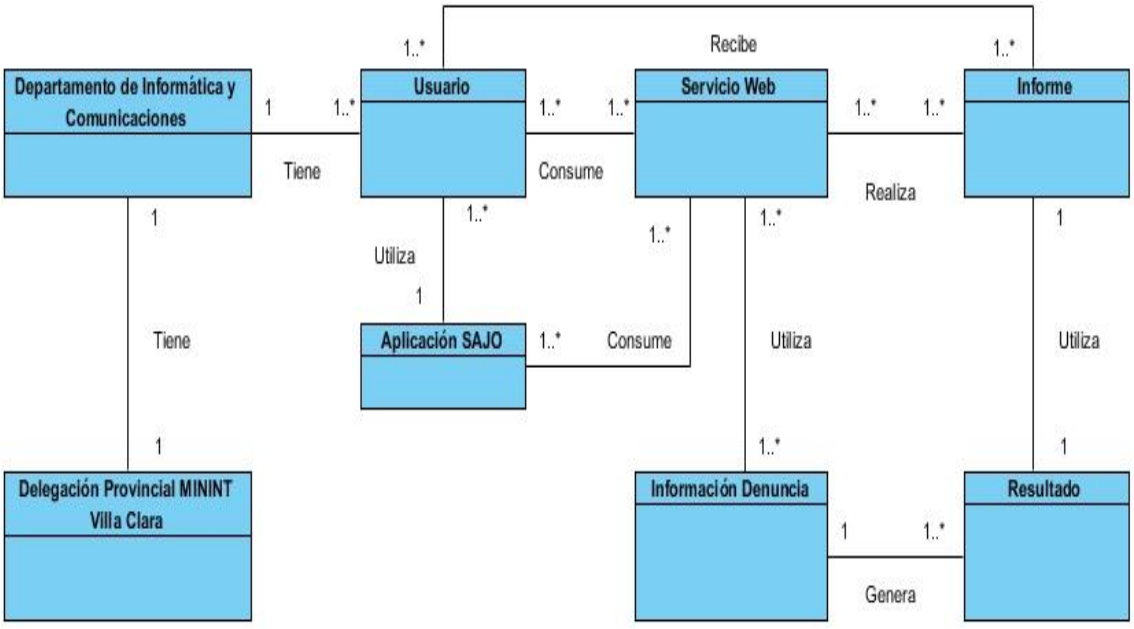


Ilustración 1: Modelo del Dominio

Definiciones

La definición de cada uno de los conceptos del modelo de dominio se muestra a continuación:

Delegación Provincial del MININT Villa Clara: representación de todos los departamentos u órganos

Departamento de Informática y Comunicaciones: grupo de personas que desarrollan software.

Usuario: persona o aplicación que necesita información sobre el estado actual de una denuncia.

Servicio Web: es un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

Datos Denuncia: conjunto de datos almacenados sistemáticamente para su posterior uso por diferentes aplicaciones,

Resultado: archivo generado por la aplicación con la descripción del hecho cometido.

Informe: documento que se genera al interactuar los servicios Web con los datos de la denuncia.

2.5 Especificación de los requisitos de software

Los Requisitos Software es una descripción completa del comportamiento del sistema que se va a desarrollar. Incluye un conjunto de casos de uso que describe todas las interacciones que tendrán los usuarios con el software. Los casos de uso también son conocidos como requisitos funcionales. Además de los requisitos funcionales, también contiene requisitos no funcionales que imponen restricciones en el diseño o la implementación.

2.5.1 Requisitos funcionales

Los requerimientos funcionales definen el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica. A continuación se presenta el listado de los requisitos funcionales en el sistema que se modela:

RF1. Recibir notificación de nueva solicitud de búsqueda.

Consiste en recibir la notificación de la solicitud de búsqueda de datos de una denuncia por parte del usuario o aplicación que lo necesite.

RF2. Obtener datos de la denuncia a analizar en la base de datos.

Permite que el servidor obtenga la información de la denuncia que se desea para que pueda ser mostrada al usuario.

RF3. Mostrar datos generales de la denuncia.

Permite que el usuario visualice mediante la interfaz los datos generales de la denuncia.

RF4. Mostrar datos de las personas procesadas en la denuncia.

Permite que el usuario visualice datos personales y datos de interés de las personas procesadas.

RF5. Mostrar datos de las personas no procesadas en la denuncia.

Permite que el usuario visualice datos personales y datos de interés de las personas no procesadas.

RF6. Mostrar datos de los objetos sustraídos en caso de robo.

Permite que el usuario pueda visualizar las características de los objetos sustraídos en caso de que la denuncia sea un robo.

RF7. Mostrar datos sobre el modo de operar, las causas y las condiciones que propiciaron el hecho.

Permite que el usuario pueda visualizar el modo de operar, las causas y las condiciones que conllevaron al individuo a cometer el delito.

RF8. Mostrar denuncias que posean la misma radicación final.

Permite visualizar información sobre delitos que posean la misma radicación final de la cual se esta investigando.

RF9. Mostrar denuncias que posean al mismo procesado.

Permite visualizar información sobre los delitos en que ha participado el procesado de la denuncia que se esta investigando.

RF10. Ubicar geográficamente la denuncia.

Permite visualizar la localización geográfica de la denuncia que se esta investigando.

RF11. Generar estadísticas relacionadas con las denuncias.

Permite generar estadísticas de interés sobre el control de la denuncia las cuales posibilitan un mejor trabajo preventivo para el control del delito.

2.5.2 Requisitos no funcionales

Los requerimientos no funcionales hacen relación a las características del sistema que aplican de manera general como un todo, más que a rasgos particulares del mismo.

Requisitos de Software

RNF1. Las computadoras que utilizaran el software deben tener instalado: Windows 2003 Server, Windows 2000 NT, Windows XP Professional o GNU/LINUX en cualquiera de sus distribuciones.

RNF2. Navegador Web Mozilla Firefox, Opera o Safari.

RNF3. El nodo o servidor que alojara la aplicación deberá tener instalado como servidor de Aplicaciones Java como el WebLogic o el Tomcat.

RNF4. El nodo o servidor que alojara la base de datos deberá tener instalado como gestor de base de datos ORACLE en su versión 11g o superior.

RNF5. Se necesita la instalación de Oracle Jdeveloper como IDE de programación con framework ADF**.**

RNF6. Se necesita un servidor de mapas Oracle MapViewer 11g.

Requisitos de hardware

RNF7. Las computadoras que utilizarán el software a desarrollar deberán tener 512MB de Memoria tipo RAM y 1.0 GHz de velocidad del microprocesador.

Requisitos en el diseño y la implementación

RNF8. El lenguaje de programación a utilizar es Java.

RNF9. La herramienta de desarrollo a utilizar es Jdeveloper.

Requisitos de seguridad

RNF10. La Base de Datos se le realiza salvas en otro servidor espejo.

RNF11. Se hacen salvas diarias de la información (exportada y consistente).

RNF12. Se usa una red interna en la cual no hay contacto con ninguna red externa.

Requisitos de Rendimiento

RNF13. Debe ser eficiente a la hora de gestionar las solicitudes, permitiendo alcanzar los resultados deseados sin hacer un uso extensivo de la navegación por el sitio.

RNF14. Debe estar disponible las 24 horas del día.

Requisitos de Soporte

RNF15. El sistema debe contar con Manual de Usuario (documentación) para evitar problemas en caso de fallas.

RNF16. Debe ser lo más extensible posible. Puede ser usado por otros servidores.

RNF17. Fácil para el mantenimiento, de configuración sencilla y factible para los clientes.

Aspectos Legales

RNF18. La aplicación y la documentación pertenecen al Centro de Informática y Comunicaciones del Ministerio del Interior en Villa Clara

2.6 Definición de los casos de uso

2.6.1 Definición de los actores

Los actores del sistema representan el rol que desempeña una o varias personas, un equipo o un sistema automatizado. Son parte del sistema y pueden intercambiar información con él o ser recipientes pasivos de información. Estos fueron trabajadores del negocio que interactúan con el sistema, en este caso los actores que interactúan con el sistema se definen a continuación:

Actores	Justificación
Usuario	Actor que se origina como parte del evento de solicitud de información de la denuncia.

Tabla 1: Definición de actores del sistema

2.6.2 Diagrama de casos de uso

El diagrama de casos de uso es el modelo de la secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. A continuación se muestra el diagrama de casos de uso para el desarrollo del sistema.

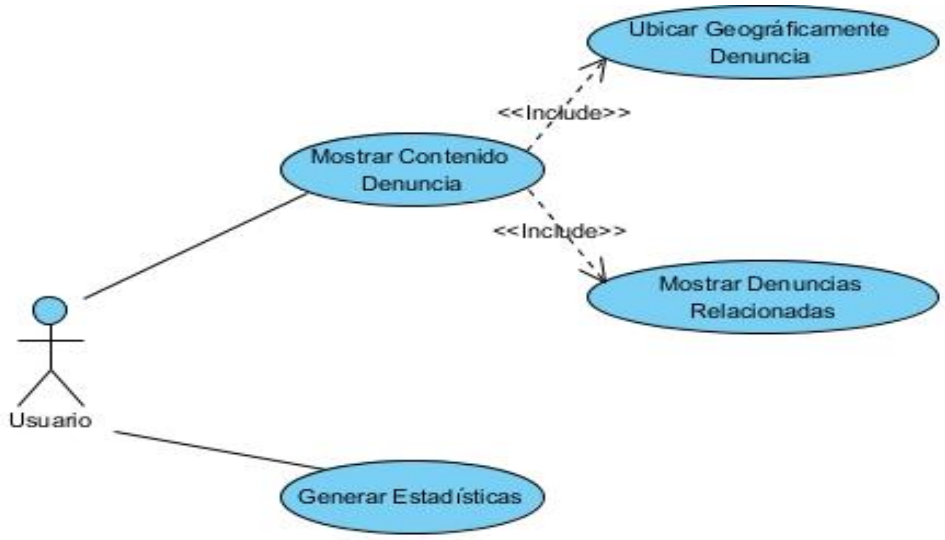


Ilustración 2: Diagrama de casos de uso

2.6.3 Casos de uso expandido

A través de la expansión de los casos de uso se realiza una descripción detallada de la secuencia de eventos que se ejecutan dentro del caso de uso, en colaboración con sus actores correspondientes. De ésta forma en el sistema se podrán implementar todas las funcionalidades necesarias para su funcionamiento. Para lograr un entendimiento de la descripción de los casos de uso se necesita conocer algunos términos relacionados con esta:

Definiciones:

Servidor de base de datos: sistema gestor encargado de guardar la información de todos los delitos cometidos necesarios para el desarrollo del sistema.

Servidor MapViewer: sistema gestor encardo de guardar la información geográfica de los delitos.

A continuación se presentan las descripciones de los casos de uso del sistema.

Caso de Uso	
CU-1	Mostrar contenido de la denuncia
Propósitos	Procesar la solicitud para mostrar la información de la denuncia deseada.
Actores	Usuario (inicia)
Resumen	El Caso Uso del Sistema se inicia cuando el usuario envía una solicitud y culmina cuando el mismo puede observar la información deseada.
Referencias	**RF1, RF2, RF3, RF4, RF5, RF6, RF7**
Precondiciones	La base de datos debe tener almacenado la información de la denuncia solicitada y debe de estar en funcionamiento.
Acción del Actor	**Respuesta del Sistema**
1.El usuario envía la solicitud de información con los siguientes datos: *número:* indica el identificador de la denuncia que se desea analizar. *año:* indica el año de la denuncia que se desea analizar.	2. El sistema recibe la notificación de la nueva solicitud de análisis.
	3. El sistema se conecta a la base de datos.
	4. Invoca al caso de uso Ubicar Geográficamente Denuncia.

	5. Invoca al caso de uso Buscar Denuncias Relacionadas.
	6. Invoca al caso de uso Generar Estadísticas.
	4. El sistema muestra la información relacionada a la denuncia solicitada.
Flujo Alterno 3a "Conexión fallida a la base de datos"	
	3a.1 Envía un mensaje de error al Usuario **"**Error en conexión a la base de datos**"**, intente enviar nuevamente la solicitud.
	3a.2 El sistema espera que el Usuario envíe nuevamente la solicitud.
	3a.3 Ir a la acción 1.

Tabla 2: Descripción del caso de uso Mostrar Contenido de la Denuncia

Caso de Uso	
CU-2	Mostrar Denuncias Relacionadas.
Propósitos	Procesar la solicitud para mostrar la información de las denuncias relacionadas con la misma.
Actores	Usuario(inicia)
Resumen	El Caso Uso del Sistema se inicia cuando el usuario envía una solicitud por URL y culmina cuando el mismo puede observar la información deseada.
Referencias	**RF8, RF9**
Acción del Actor	**Respuesta del Sistema**
	1. El sistema se conecta a la base de datos.
	2. El sistema muestra la información relacionada a la denuncia solicitada.
Flujo Alterno 1a "Conexión fallida"	

	1a.1 El sistema muestra un mensaje de error **"**Error en conexión a acceso a datos**"**.
	1a. 2 El sistema espera que se restablezcan las conexiones.
	1a.3 El sistema accede a la base de datos y obtiene el fichero pendiente.
	1a.4 Ir a la acción 2 del caso de uso **"Mostrar Denuncias Relacionadas"**

Tabla 3: Descripción del caso de uso Mostrar Denuncias Relacionadas

Caso de Uso	
CU-3	Ubicar Geográficamente Denuncia.
Propósitos	Procesar la solicitud para ubicar geográficamente la denuncia en la provincia de Villa Clara.
Actores	Usuario (Inicia).
Resumen	El Caso Uso del Sistema se inicia cuando el usuario envía una solicitud y culmina cuando el mismo puede observar la localización geográfica de la denuncia en la provincia de Villa Clara.
Referencias	**RF10**
Acción del Actor	**Respuesta del Sistema**
	1. El sistema se conecta a la base de datos.
	2. El sistema localiza geográficamente la denuncia.
Flujo Alterno 1a "Conexión fallida con el servidor MapViewer"	
	1a.1 El sistema muestra un mensaje de error **"**Error en conexión a acceso a datos**"**.

	1a. 2 El sistema espera que se restablezcan las conexiones.
	1a.3 El sistema accede a la base de datos y obtiene la información pendiente.
	1a.4 Ir a la acción 2 del caso de uso **"**Ubicar Geográficamente Denuncia**"**

Tabla 4: Descripción del caso de uso Ubicar Geográficamente Denuncia

Caso de Uso	
CU-4	Generar Estadísticas.
Propósitos	Procesar la solicitud para generar estadísticas relacionadas con la denuncia en la provincia de Villa Clara.
Actores	Usuario (Inicia).
Resumen	El Caso Uso del Sistema se inicia cuando el usuario envía una solicitud por URL y culmina cuando el mismo puede observar las estadísticas relacionadas con la denuncia en la provincia de Villa Clara.
Referencias	**RF11**
Acción del Actor	**Respuesta del Sistema**
	1. El sistema se conecta a la base de datos.
	2. El sistema genera estadísticas relacionadas con la denuncia.
Flujo Alterno 1a "Conexión fallida"	
	1a.1 El sistema muestra un mensaje de error **"**Error en conexión a acceso a datos**"**.
	1a. 2 El sistema espera que se restablezcan las conexiones.

	1a.3 El sistema accede a la base de datos y obtiene la información pendiente.
	1a.4 Ir a la acción 2 del caso de uso **"Generar Estadísticas".**

Tabla 5: Descripción del caso de uso Generar Estadísticas

2.7 Conclusiones parciales

Durante la realización del capítulo transcurrido fue posible la elaboración del modelo del dominio, a partir del mismo se pudo comprender el funcionamiento del sistema de acuerdo a sus conceptos y relaciones. Fue efectuada una descripción detallada de los actores del mismo, así como de sus responsabilidades. También fue posible identificar los requisitos funcionales, estos resultan de gran utilidad para su despliegue, constituyendo la funcionalidad del mismo. Además se modeló el diagrama de casos de uso, el cual representa un esbozo de lo que resulta necesario para la implementación. También fueron identificados los requisitos no funcionales y se detallaron todos los casos de uso, ello resulta de suma importancia para la edificación del sistema que se propone. El haber finalizado esta etapa de la investigación, permite la entrada a la siguiente fase de diseño del sistema.

CAPITULO III. DISEÑO DEL SISTEMA

3.1 Introducción

En el presente capítulo se realiza el diseño del sistema. Se diseña la arquitectura a utilizar, así como los patrones empleados. El propósito de este capítulo es especificar una solución que pueda ser fácilmente convertida en código, construyendo así una arquitectura simple y fácilmente extensible.

3.2 Diseño del sistema

El Diseño de sistema es el arte de definir la arquitectura de hardware y software, componentes, módulos y datos de un sistema computacional para satisfacer los requisitos definidos con anterioridad.

Por lo tanto, en esta etapa se investigará que datos es necesario almacenar y cómo se van a almacenar, qué procesos se van a implementar y cómo se van a implementar y que interfaces se requieren diseñar y cómo se van a diseñar. Un buen diseño del sistema permite que el mismo pueda ser implementado sin ambigüedades.

3.2.1 Patrones arquitectónicos

Los patrones arquitectónicos describen los principios fundamentales de la arquitectura de un sistema de software. Identifica los subsistemas, define sus responsabilidades y establece las reglas y guías para organizar las relaciones entre ellos.

Uno de estos patrones es el “Modelo-Vista-Controlador” el cual se adapta a la aplicación aquí diseñada. Este patrón propone la separación en distintos componentes o vistas de la interfaz de usuario, el modelo de negocio y la lógica de control. Una vista es una instantánea del modelo en un determinado momento. Un control recibe un evento disparado por el usuario a través de la interfaz, accede al modelo de manera adecuada a la acción realizada, y presenta en una nueva vista el resultado de dicha acción. Por otro lado, el modelo consiste en el conjunto de objetos que modelan los procesos de negocio que se realizan a través del sistema(Ceballos, 2008).

En una aplicación Web, las vistas serían las páginas HTML que el usuario visualiza en el navegador. El usuario interactúa con la aplicación a través de estas páginas, enviando eventos al servidor mediante peticiones HTTP. En el servidor se encuentra el código de control para estos eventos, que en función de un evento concreto actúa sobre el modelo convenientemente. Los resultados de la acción se devuelven al usuario en forma de página HTML mediante la respuesta HTTP. La clave está en la separación entre vista y modelo. El modelo es más estable y menos sujeto a variaciones mientras que las vistas pueden cambiar con frecuencia. Con esta separación las vistas pueden cambiar sin afectar al modelo y viceversa. Los controladores son los encargados de hacer de puente entre ambos, determinando el flujo de salida de la aplicación.

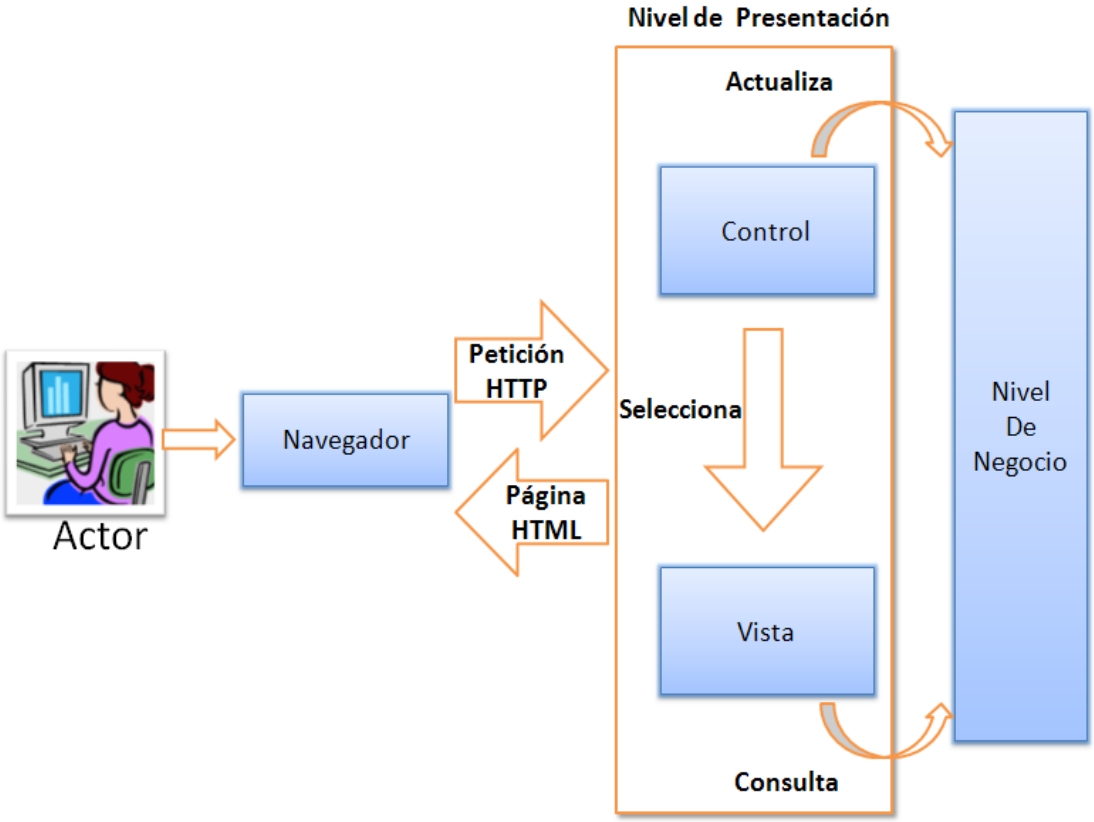


Ilustración 3: Arquitectura del patrón Modelo Vista Controlador

3.2.2 *Arquitectura del sistema*

La arquitectura que se presenta se basa en el Modelo Vista Controlador, donde cada uno de los componentes posee una funcionalidad específica, de la cual es responsable.

***Front_End*:** aplicación externa encargada de visualizar la información recopilada cuando se realiza una petición al servidor y brindar servicios a otras aplicaciones que necesiten de la misma.

Map Viewer: es una de las aplicaciones que compone la Capa Media de Oracle, contiene un motor de interpretación cartográfica que se encarga de establecer la conexión con la Base de Datos Oracle a través del JDBC; se comunica con el cliente a través de un explorador Web o una aplicación utilizando el protocolo HTTP.

Administración de datos: su propósito es hacer persistente toda la información relacionada con la denuncia, en este nivel el código encargado de las actualizaciones de la BD se encuentra en el servidor y la parte restante en la BD.

Oracle JDeveloper: en este IDE se encuentran todo el código de control para los diferentes eventos que se puedan ocasionar y en función de un evento concreto actúa sobre el modelo convenientemente.

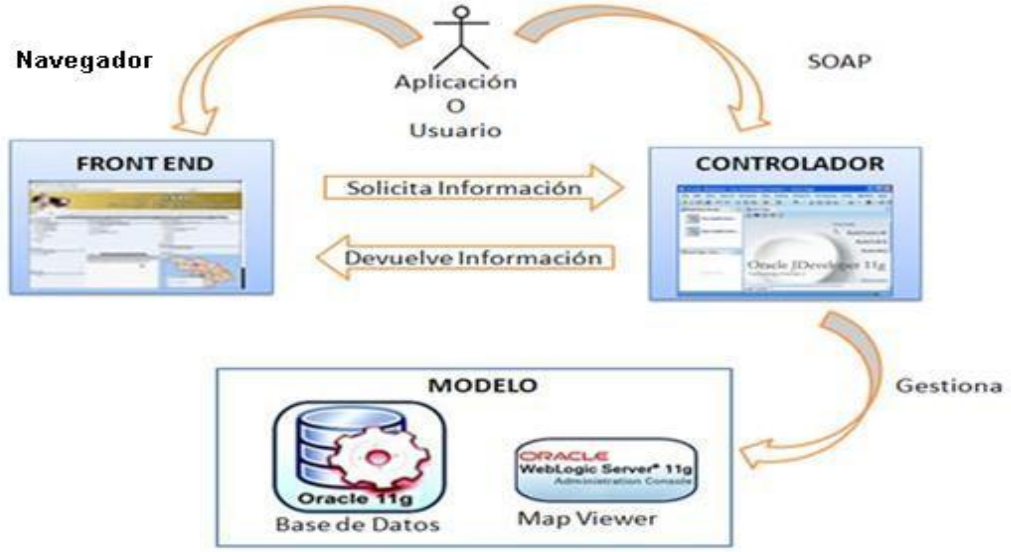


Ilustración 4: Arquitectura del Sistema

3.2.2.1 Flujo de la arquitectura del sistema

El flujo comienza cuando el actor a través de un navegador Web accede a la aplicación y efectúa una petición que es enviada por el Front End al controlador. También puede ocurrir cuando este hace una petición mediante servicios Web directamente al controlador. El controlador a partir de la petición realiza una búsqueda de los datos relacionados con la denuncia, extrayendo la información necesaria del servidor de mapas y de la Base de Datos. A partir de este momento el servidor de mapas y la Base de Datos le proporcionan los datos al controlador. Este último le envía los datos al navegador que los visualiza para un posterior análisis del actor.

3.3 Modelo de datos

El modelo de datos describe las representaciones lógicas y físicas de datos persistentes utilizados por una aplicación.

3.3.1 Modelo físico de datos (modelo de datos)

El modelo de datos es la representación física o estructura de las tablas de la base de datos. El modelo físico se desarrolló a partir de la base del conjunto de clases persistentes y sus asociaciones en el modelo de diseño.

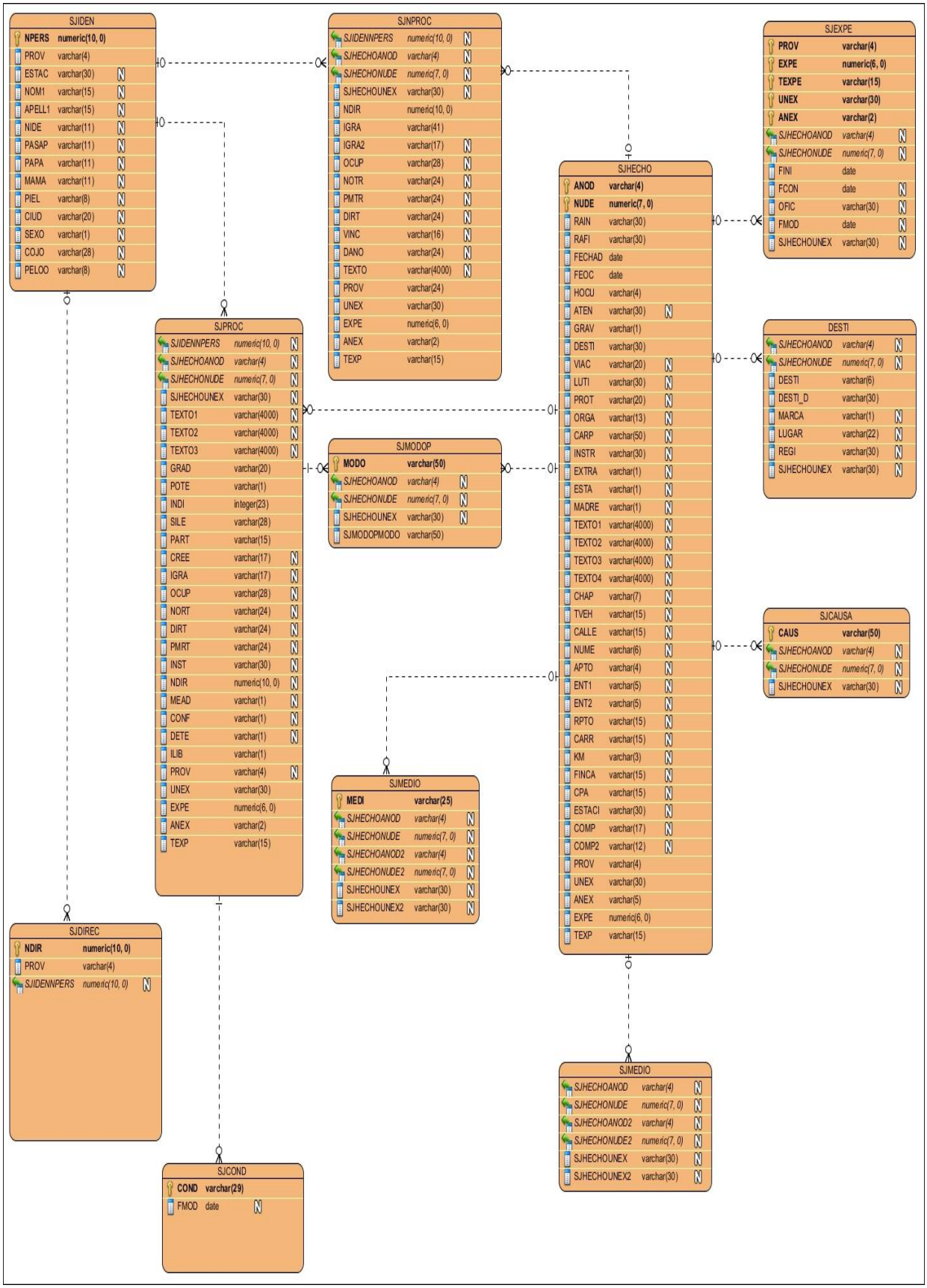


Ilustración 5: Diagrama físico de datos

3.4 Conclusiones parciales

En este capítulo se realizaron los diagramas de clases del diseño, sirviendo como base fundamental para la implementación del sistema. Se desarrolló una arquitectura basada en el modelo vista controlador, lo que permite tener una idea completa del software.

CAPITULO IV. IMPLEMENTACIÓ

4.1 Introducción

El presente capítulo tiene como objetivo desarrollar los artefactos correspondientes a la implementación del sistema, donde se tendrá como base el diseño realizado en el capítulo anterior. Se modelará el diagrama de componentes y diagrama de despliegue, conformando los modelos de implementación; en los que se obtendrá una visión general del desarrollo de la aplicación, así como los resultados del software. Una vez concluido la implementación se realizarán las pruebas al sistema.

4.2 Modelo de implementación

El modelo de implementación describe como los elementos del modelo del diseño se implementan en términos de componentes y como estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue.

4.2.1 Diagrama de componentes

Los diagramas de componentes describen los elementos físicos y sus relaciones en el entorno de realización. Ilustran las piezas del software que conforman un sistema. En un diagrama de componentes se puede apreciar que conformaran un sistema. En un diagrama de componentes se puede apreciar la organización lógica de la implementación de un sistema. (Muller, 1997) A continuación se muestra el diagrama de componentes del sistema SAJO:

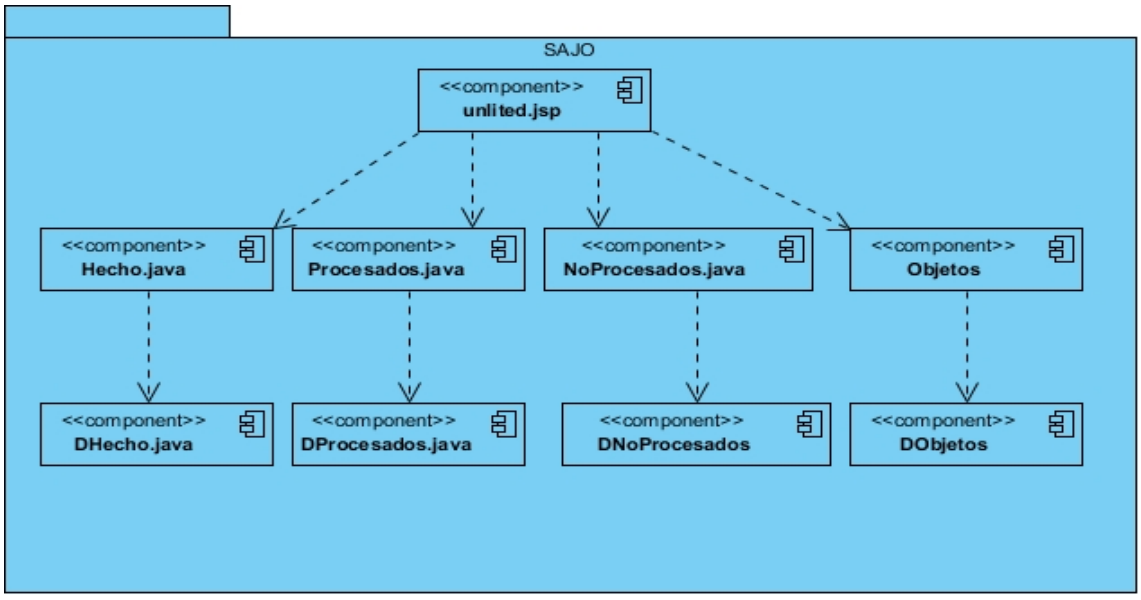


Ilustración 6: Diagrama de Componentes

4.2.1.1 Descripción de los componentes

- Unlited.jsp
- Hecho.java
- DHecho.java
- Procesados.java
- DProcesdos.java
- NoProesdos.java
- DNoProcesados.java
- Objetos.java
- DObjetos.java

4.2.2 Diagrama de despliegue

El modelo de despliegue define la arquitectura física del sistema por medio de nodos interconectados. Estos nodos son elementos hardware sobre los cuales pueden ejecutarse los elementos software. Con frecuencia conocemos cómo será la arquitectura física del sistema antes de comenzar su desarrollo. Por tanto, podemos modelar los nodos y las conexiones del modelo de despliegue tan pronto como comience el flujo de trabajo de los requisitos.

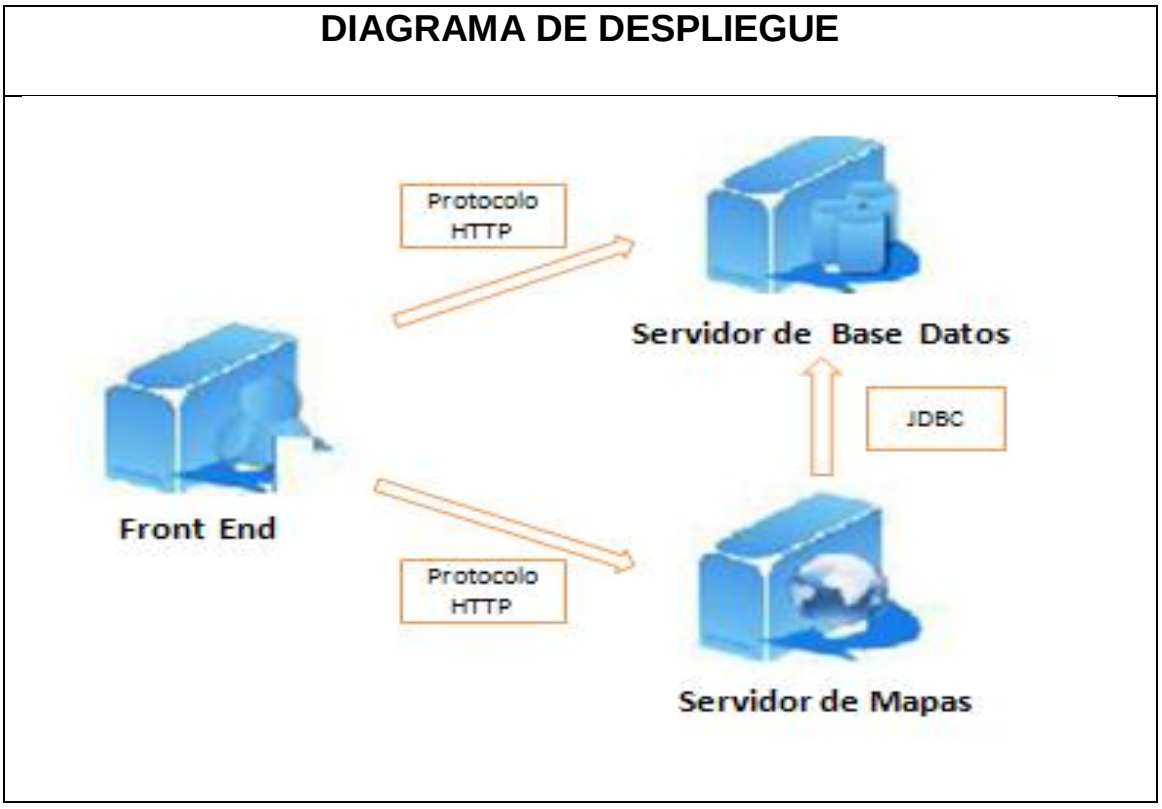


Ilustración 7: Diagrama de Despliegue

4.2.2.1 Descripción de los nodos

Nombre del Nodo	Descripción
Nodo Front End	En este nodo se modela la aplicación externa encargada de enviar solicitudes de datos de una denuncia en particular.
Nodo Servidor de Mapas	En este nodo se encuentra un motor de interpretación cartográfica que se encarga de establecer la conexión con la Base de Datos Oracle
Nodo Servidor de Base de Datos	En este nodo se encuentra la base de datos del sistema, se cuenta con el gestor Oracle para este objetivo.

Tabla 6: Descripción de los nodos del Diagrama de Despliegue

4.2.2.2 Vista del Diagrama de Despliegue

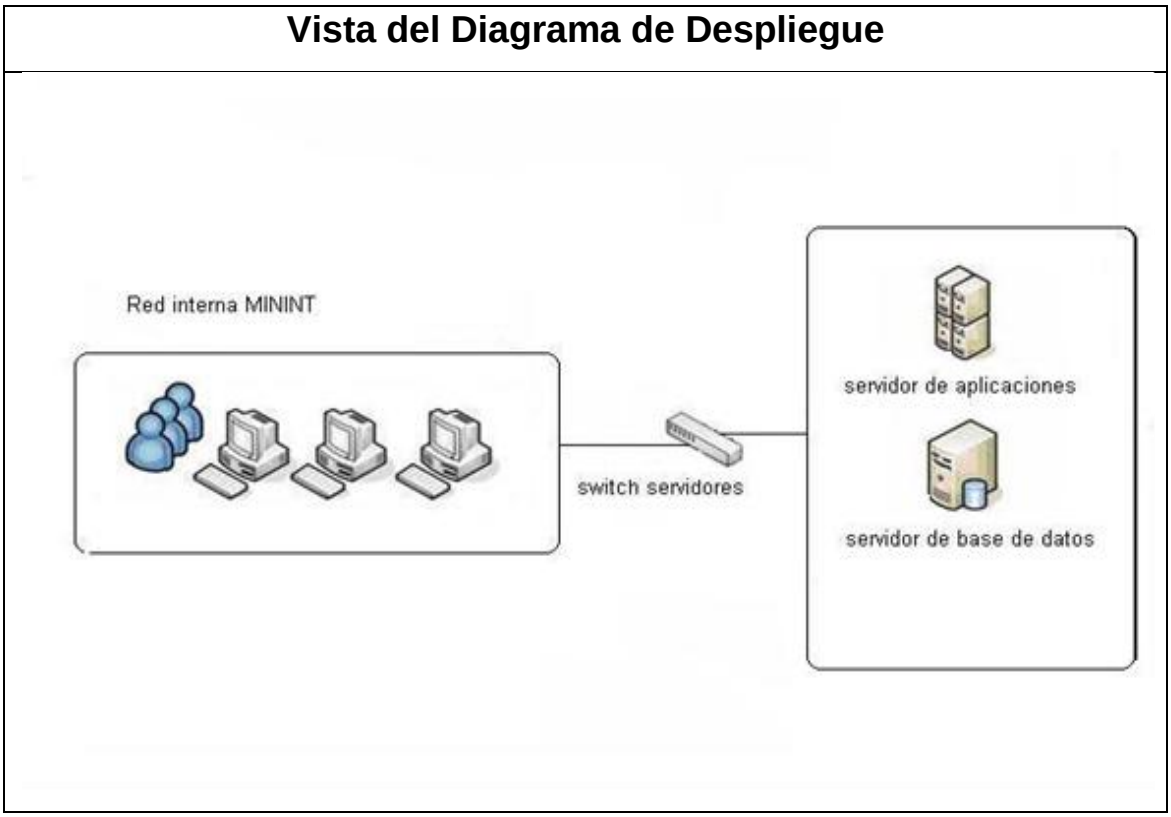


Ilustración 8: Vista del despliegue

4.3 Pruebas

Una vez generado el código fuente de la aplicación, el software debe ser probado para corregir el máximo de errores que puede existir antes de la entrega al cliente.

Las pruebas de Software no son más que una actividad en la cual un sistema o uno de sus componentes se ejecutan en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto. El objetivo de las mismas es presentar información sobre la calidad del producto a las personas responsables de este, identificando así los posibles fallos de la implementación (UCI, 2010).

A continuación se muestra un diagrama que muestra el ciclo completo de las pruebas de software:

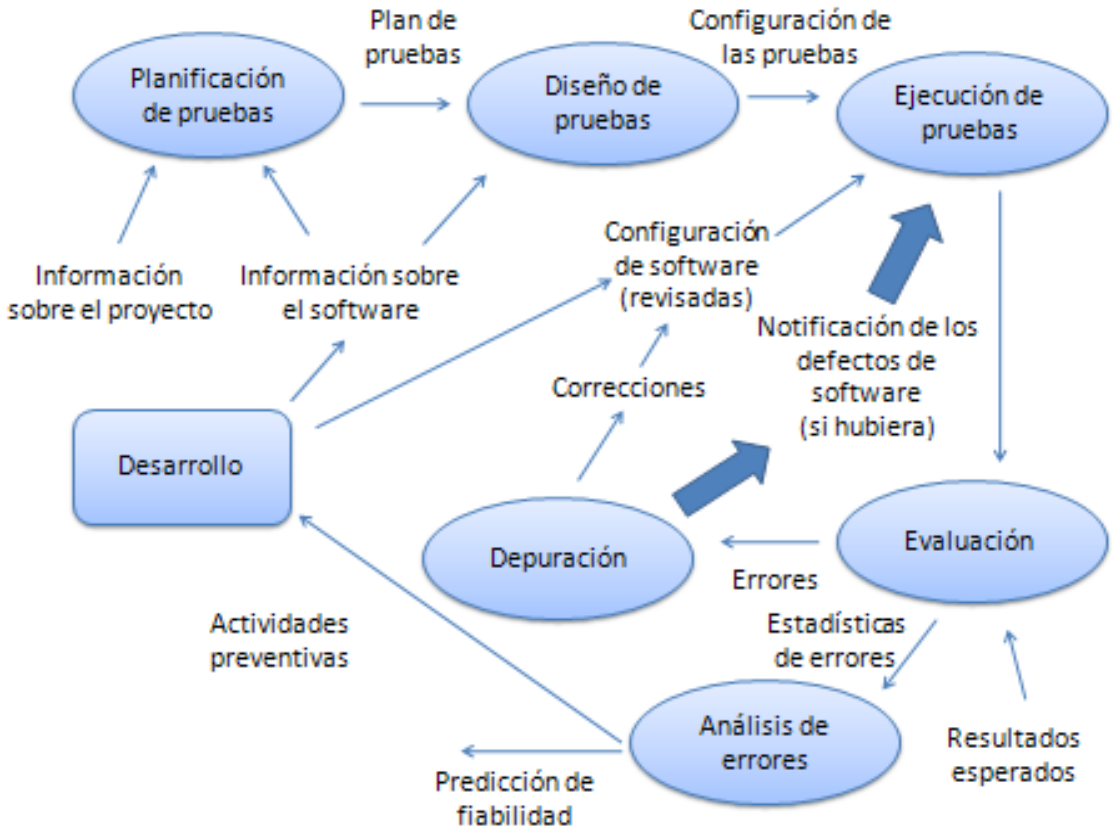


Ilustración 9: Ciclo completo de las pruebas del Software

4.3.1 Métodos de prueba

Según UCI (2010) existen tres enfoques principales para el diseño de las pruebas:

1.- El enfoque estructural o de caja blanca**.** Se centra en la estructura interna del programa (analiza los caminos de ejecución).

2.- El enfoque funcional o de caja negra. Se centra en las funciones, entradas y salidas.

3.- El enfoque aleatorio consiste en utilizar modelos (en muchas ocasiones estadísticos) que representen las posibles entradas al programa para crear a partir de ellos los casos de prueba.

4.3.1.1 Prueba de caja blanca

En la prueba de caja blanca se realiza un examen minucioso de los detalles procedimentales, comprobando los caminos lógicos del programa, comprobando los bucles y condiciones, y examinado el estado del programa en varios puntos (UCI, 2010).

4.3.1.2 Prueba de caja negra

Los casos de prueba de la caja negra pretenden demostrar que:

• Las funciones del software son operativas.

• La entrada se acepta de forma correcta.

• Se produce una salida correcta.

• La integridad de la información externa se mantiene.

Resulta relevante destacar que se hace irrealizable la tarea de someter a prueba el software confeccionado para todas las alternativas posibles. Por eso se hace necesario poseer criterios para una buena elección de pruebas para el sistema.

Se puede decir que un caso de prueba funcional es bien elegido si se cumple que:

- Reduce el número de otros casos necesarios para que la prueba sea razonable. Esto implica que el caso ejecute el máximo número de posibilidades de entrada diferentes para así reducir el total de casos.

- Cubre un conjunto extenso de otros casos posibles, es decir, no indica algo acerca de la ausencia o la presencia de defectos en el conjunto específico de entradas que prueba, así como de otros conjuntos similares.

4.3.1.3 Técnicas de caja negra

Según Juristo (2006) existen varios tipos de técnicas de la caja negra que se pueden aplicar entre las que se puede mencionar:

- La técnica de participaciones o clases de equivalencia: se caracteriza porque en ella cada caso debe cubrir el máximo número de entradas. Además, debe tratarse el dominio de valores de entrada dividido en un número finito de clases de equivalencia.
- La técnica de análisis de valores al límite (AVL): esta técnica de diseños de casos de prueba complementa las particiones de equivalencia probando los casos límites de un programa produciendo un mejor resultado para detectar.
- La técnica de conjeturas de errores: consiste en enumerar una lista de posibles equivocaciones típicas que pueden cometer los desarrolladores y de situaciones propensas a ciertos errores, entre estas podremos citar algunas como:
  - El valor cero tanto en la salida como en la entrada.
  - En la entrada de valores requeridos, cuando se deja de introducir un valor necesario o no se introduce ninguno.
  - Interpretación inadecuada del programador en la comprensión del problema.
  - Entrada errónea de datos al programa por parte del usuario.

4.3.2 Pruebas realizadas a la aplicación

Se decide utilizar las pruebas de caja negra con la técnica de conjeturas de errores debido a que están muy bien definidos los datos de entrada y de salida es decir la interfaz, además no se necesita precisar, ni conocer los detalles internos de su funcionamiento.

Caso 1: Se entran los datos requeridos el número de la denuncia y el año 222222 y 2 respectivamente y el programa devuelve un mensaje informándole al usuario que el año debe tener 4 dígitos.

Ilustración 10: Caso de prueba 1

Caso 2: No se entran datos y se envía una solicitud de información, y el programa devuelve un mensaje informándole al usuario que debe introducir datos o ambos datos en el caso de que introduzca uno solo.

Ilustración 11: Caso de prueba 2

Caso 3: Se entran los datos requeridos el número de la denuncia y el año 222222 y 2012 respectivamente y la página devuelve un mensaje informándole que el número de la denuncia debe poseer de 1 a 5 dígitos.

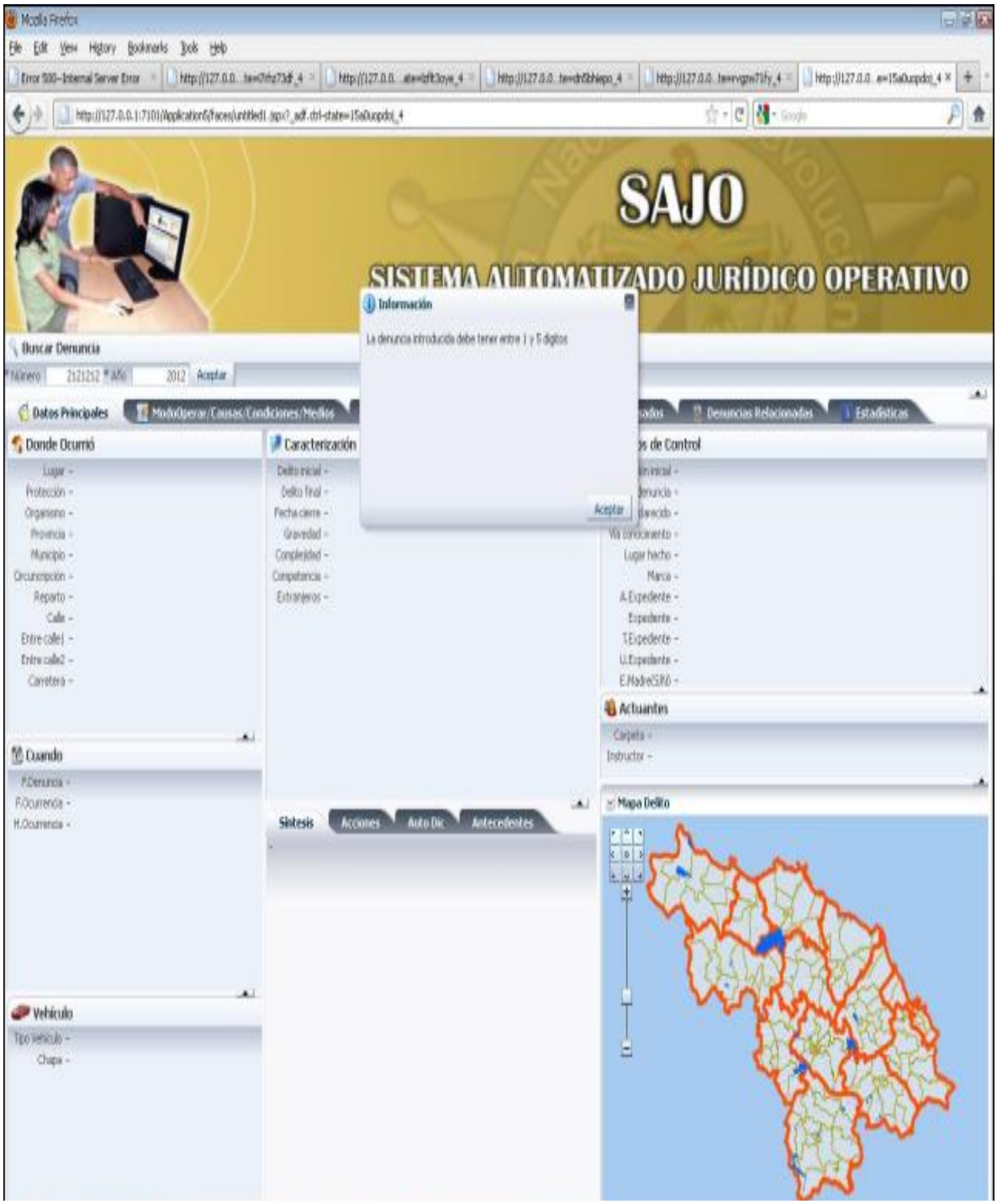


Ilustración 12: Caso de prueba 3

Caso 4: Se entran los datos requeridos el número de la denuncia y el año 12 y 1998 respectivamente y la página devuelve un mensaje informándole que el año introducido debe ser mayor que el 2000 debido a que no se posee información en años anteriores.

Ilustración 13: Caso de prueba 4

Caso 5: Se entran los datos requeridos el número de la denuncia y el año asas15 y sas2012 respectivamente y la página devuelve un mensaje informándole que el número de la denuncia y el año deben ser números.

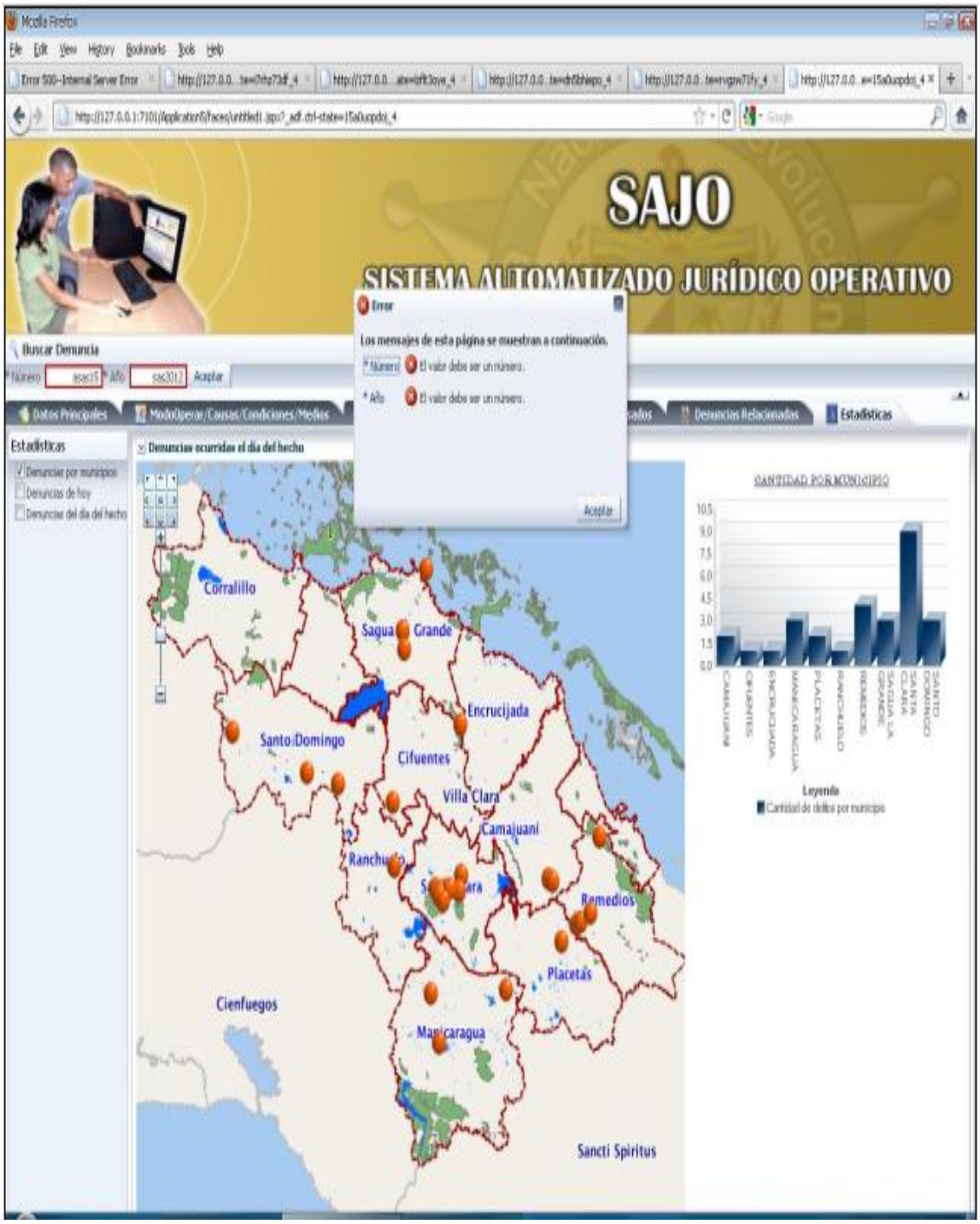


Ilustración 14: Caso de prueba 5

Caso 6: Se introduce el numero de la denuncia y el año 7, 2012 respectivamente y el programa devuelve la información solicitada satisfactoriamente

Ilustración 15: Caso de prueba 6

Caso 7: Prueba para comprobar interoperabilidad de los servicios con otras aplicaciones.

Para la realización de la siguiente prueba se implemento una aplicación de consola en otra plataforma en este caso .net con el objetivo de probar la interoperabilidad de los servicios implementados, para ello se probó el servicio que brinda la información general de la denuncia y la prueba lanzo el siguiente resultado:

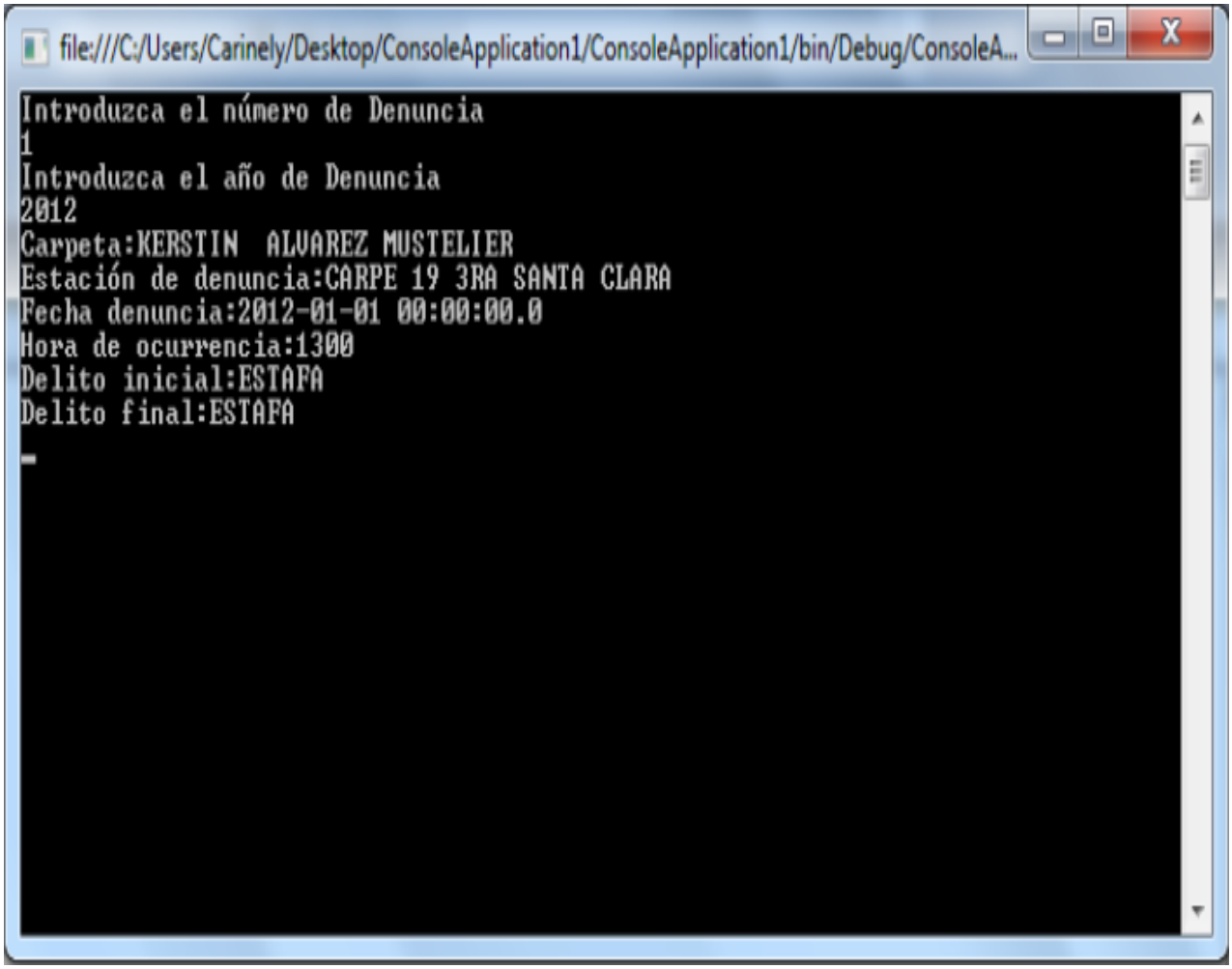


Ilustración 16: Caso de prueba 7

De esta manera se puede comprobar que los resultados fueron satisfactorios ya que se devolvieron los datos solicitados con éxito, demostrándose la funcionalidad del servicio seleccionado y con el mismo se prueba la disponibilidad de los restantes servicios para su consumo por otras aplicaciones.

4.4 Conclusiones parciales

En el presente capítulo se obtuvo como principal resultado el modelo de implementación de la solución del sistema, a través del diagrama de componentes y de despliegue.

Se expuso además el método de pruebas definido para verificar las funcionalidades de la solución, donde se realizaron las pruebas necesarias al código de la aplicación para detectar los posibles errores que pudiese tener. De esta forma se le da cumplimiento a todos los objetivos trazados inicialmente.

CONCLUSIONES

Durante el transcurso de la presente investigación fue posible desarrollar servicios Web que permiten la integración del SAJO con los sistemas que lo requieren, ello fue posible una vez que se les dio cumplimiento a las tareas de investigación planteadas:

1. El análisis minucioso de la situación actual del SAJO brindó la posibilidad de definir los requerimientos necesarios para el diseño del sistema.
2. Se diseñaron e implementaron los servicios necesarios para la integración del SAJO con los sistemas que lo requieren.
3. Se logró garantizar la interoperabilidad de los servicios que ofrece el sistema confeccionado con las aplicaciones existentes.
4. Se empleó las pruebas de caja negra con la técnica de conjeturas de errores para la validación de la propuesta de solución.

RECOMENDACIONES

- Desarrollar nuevos servicios que brinden estadísticas relacionadas con la denuncia y puedan ser utilizados por los miembros de los diferentes órganos del MININT, favoreciendo el desarrollo de una labor profiláctica con el delito.
- Implementar aplicaciones que consuman los servicios que brinda el Sistema confeccionado.
- Perfeccionar el Sistema con el objetivo de aumentar su rendimiento y velocidad a la hora de ser consumido por los diferentes usuarios.

REFERENCIAS BIBLIOGRAFICAS

2009. *Características Oracle* [En línea]. 2009. *Características Oracle* [En línea]. http://www.todoexpertos.com/categorias/tecnología-e-internet/bases-de-datos/oracle/respuestas/252736/características-oracle.

ALVAREZ, S. 2007. *Sistemas gestores de bases de datos. Introducción a este concepto y características especiales* [En línea].http://www.desarrolloWeb.com/articulos/sistemas-gestores-bases-datos.html.

ANAILYS HERNANDEZ SEIJO, E. M. D. O. G. 2009. *Procesamiento Penal a través de un Modelado del Proceso de Negocio.* Universidad Central de las Villas.

ANDRÉS MARQUÉS, M. M. 2001. *Componentes de un sistema de gestión de bases de datos* [En línea]. http://www3.uji.es/~mmarques/f47/apun/node40.html.

BELL, D. Y. P., MIKE. . *Entorno de desarrollo integrado*.

CEBALLOS, F. 2008. *Java 2: Interfaces gráficas y aplicaciones para internet,* México, S.A.d.C.V. Alfa omega Grupo Editor.

CRUZ QUISPE VÍCTOR FABIO, G. M. E. D., MENDIVIL TORRICO LUÍS BRIAM. Year. Diagrama de Componentes. *In*, 2010.

DUMLER, M. 2005. *Microsoft SQL Server* [En línea]. www.sqlgurus.org/SQLGURUS/SS2005sodaIntroGLarriera0511.pdf.

DUVIEL RODRÍGUEZ RODRÍGUEZ & VALDÉS., L. S. P. 2011. Uso de Sistemas de Información Geográfica (SIG) en el Minint de Villa Clara.

ESTADÍSTICA, I. N. D. 1999. *Herramientas CASE*, Cultura Informática.

FUENTESECA, M. 1997. *El delito civil en Roma y en el derecho español*.

FUSTER, G. G. 2006. *Evaluación comparativa de herramientas CASE para UML desde el punto de vista notacional,* Madrid, España.

GONZÁLEZ, D. F. 2011. *Análisis y Diseño del Sistema para la Gestión de las Aplicaciones Web en la Delegación Provincial del MININT. .* Universidad Central de las Villas.

HERNÁNDEZ, A. 2004. Un método para el diseño de la base de datos a partir de un modelo orientado a objetos. Computación y Sistemas. [En línea].http://www.visual-paradigm.com/product/vpuml/.

JURISTO, N., MORENO, ANA M. Y VEGAS, SIRA. 2006. *Técnicas de Evaluación de Software.*

MARQUES, P. 2000. Las TIC y sus aportaciones a la sociedad. http://dewey.uab.es/pmarques/tic.htm.

MICROSOFT, C. 2006. La Arquitectura Orientada a Servicios (SOA) de Microsoft aplicada al mundo real.

MULLER, P.-A. 1997. *Modelado de objetos con UML,* Barcelona, España.

PERALTA, S. 2006. El servidor Web. Arquitectura y funcionamiento. http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=366.

PÉREZ, J. C. C. 2012. Service Oriented Architecture

PRESSMAN, R. S. 2002. *Ingeniería del software. Un enfoque práctico.,* Madrid, McGraw-Hill.

RIVAS, D. C. 2007. *ARQUITECTURA DE APLICACIONES J2EE BASADAS EN EL PATRÓN MVC UTILIZANDO ORACLE ADF.* Universidad de San Carlos de Guatemala.

RUMBAUHG, J., I. JACOBSON, AND G. BOOCH 2000. *El Lenguaje Unificado de Modelado*.

RUMBAUHG, J. J., I G. BOOCH 2000. *El Proceso Unificado de Desarrollo de Software,* Madrid, Pearson Educación: Madrid

UCI. 2010. *Entorno Virtual de Aprendizaje.* [en línea]. Available: http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_9/Conferencia_7/Materiales_Basicos/Sobre_la_disciplina_de_Prueba.pdf.

UCI. 2010. *Entorno Virtual de Aprendizaje.* [en línea]. Available: http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_10/Clase_Practica_6/Materiales_Basicos/Material_de_caja_b_y_caja_n.pdf.

VALDÉS, D. P. 2007. Los diferentes lenguajes de programación para la Web. [en línea]. http://www.maestrosdelWeb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-Web.

MUENCH, S. 2010. The Route to Success. *Oracle Magazine.*

KOLAKOWSKI, N. 2009. Oracle Fusion Middleware 11g Embraces Social Networking, Efficiency.

GLOSARIO

C

Caso de uso: descripción de un conjunto de secuencia de acciones que un sistema ejecuta y produce un resultado observable de interés para un actor.

F

FTP: protocolo de transferencia de archivos entre sistemas conectados a una red

H

HTTP (HyperText Transfer Protocolo): Protocolo cliente-servidor utilizado para el intercambio de páginas Web (HTML).

J

J2EE: es una plataforma para desarrollar programas multicapas basadas en el lenguaje de programación Java.

JDBC: es una API que permite la ejecución de operaciones sobre base de datos desde el lenguaje de programación Java.

L

Lenguaje de programación: conjunto de símbolos y reglas sintácticas y semánticas que definen la estructura y significado de sus elementos y expresiones. Es utilizado para controlar el comportamiento de una computadora.

M

MVC (Modelo Vista Controlador): Estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Macro: Serie de instrucciones que se almacenan par que se puedan ejecutar de forma secuencial mediante una sola llamada u orden de ejecución.

O

Oracle: es un sistema de gestión de base datos relacionales. Se considera uno de los sistemas de base de datos más completos.

R

RUP: Proceso Unificado de Desarrollo.

S

Software: conjunto de componentes lógicos (programas, procedimientos, reglas, documentación y datos) necesarios para hacer posible la realización de una tarea específica.

U

UML (Unified Modelling Language): Es el lenguaje de modelado de sistemas de software más conocido en la actualidad. Sirve para la creación de diseños para sistemas basados en computadoras.

W

Web: La palabra Web (del inglés telaraña, malla) se utiliza para denominar uno de los servicios más importantes de la red Internet. Son páginas que utilizan un lenguaje especial llamado HTML, que permite presentar en pantalla texto y gráficos en el formato deseado. Estas páginas contienen referencias o enlaces que permiten acceder a otras páginas. Existen millones de páginas Web con gran cantidad de información sobre todo tipo de temas.

WebLogic: es un servidor de aplicaciones Java EE.

ANEXO

Anexo 1: Manual de usuario

Introducción

Una parte importante dentro del proceso de desarrollo de un software lo constituye la elaboración del manual de usuario, que no es más que una guía de pasos a seguir por el usuario para interactuar con el sistema.
El manual de usuario elaborado en esta investigación persigue facilitarles el trabajo a los clientes consumidores de los servicios que brinda el sistema desarrollado.

El mismo esta dedicado a la exposición de las principales facilidades que ofrece el sistema a cada uno de sus usuarios, así como las formas de aprovechar dichas facilidades. Se tratan además los prerrequisitos necesarios para el consumo de los mismos.

Desarrollo

Prerrequisitos para el consumo de los servicios

Para consumir los servicios se necesita que la aplicación este previamente desplegada en un servidor de Aplicaciones Java como el WebLogic. Además se necesita la instalación de Oracle Jdeveloper como IDE de programación con framework ADF.

Crear el servicio proxy Web

A continuación vamos a seguir los pasos necesarios para generar un proxy de Java para llamar a un servicio Web. Una vez completado los pasos se puede crear un cliente para conectarse a ella y utilizarla.

1. Lo primero que se debe hacer es crear un nuevo proyecto.
2. Para el consumo de cada servicio se necesita la dirección de cada servicio a continuación se brinda las direcciones necesarias:

http://localhost:7101/Application5-Project1-context-root/controladoraSoap12HttpPort?WSDL Hecho

http://localhost:7101/Application5-Modo-context-root/ModoSoap12HttpPort?WSDL Modo Operar

http://localhost:7101/Application5-Causas-context-root/CausasSoap12HttpPort?WSDL Medios

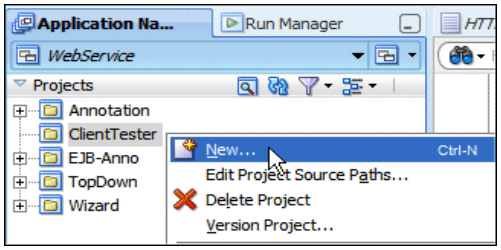
http://localhost:7101/Application5-Medios-context-root/MediosSoap12HttpPort?WSDL Causas

http://localhost:7101/Application5-NoProcesados-context-root/CNoProcesadosSoap12HttpPort?WSDL
Personas no procesadas

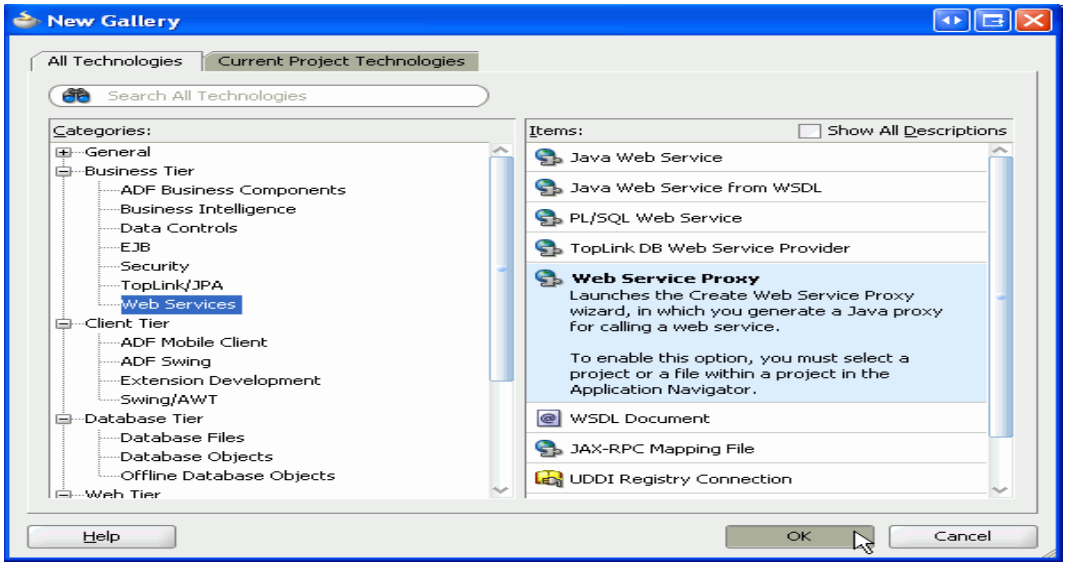
http://localhost:7101/Application5-Procesados-context-root/ControladoraProcesadosSoap12HttpPort?WS
Personas Procesadas

http://localhost:7101/Application5-Objetos-context- root/ControladoraObjetosSoap12HttpPort?WSDL
Objetos sustraídos en caso de robo

3. Para generar el **proxy, haga** clic en el proyecto **creado anteriormente** y seleccione **Nuevo.**

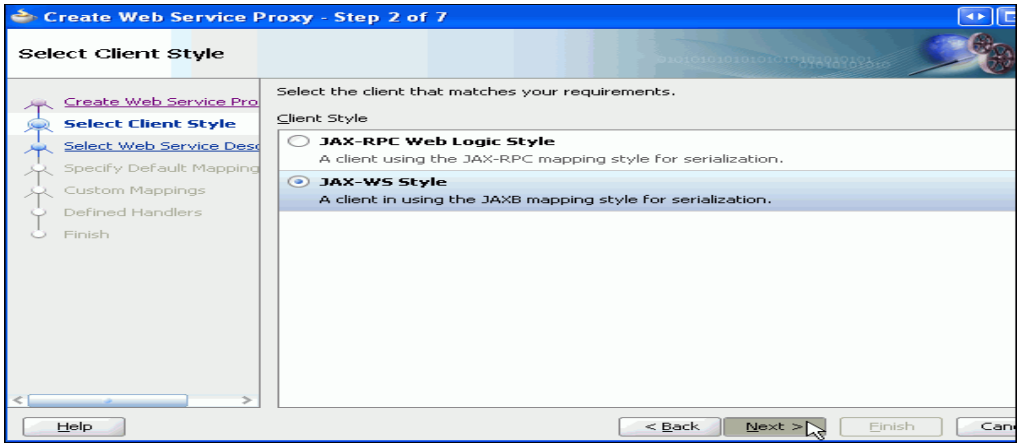


En la Nueva Galería, seleccione la ficha **todas las tecnologías.** Expanda el nodo de nivel de negocios y seleccione **Servicios Web** en la lista Categorías. Seleccione el **servicio Web Proxy** tema y haga clic **en Aceptar.**

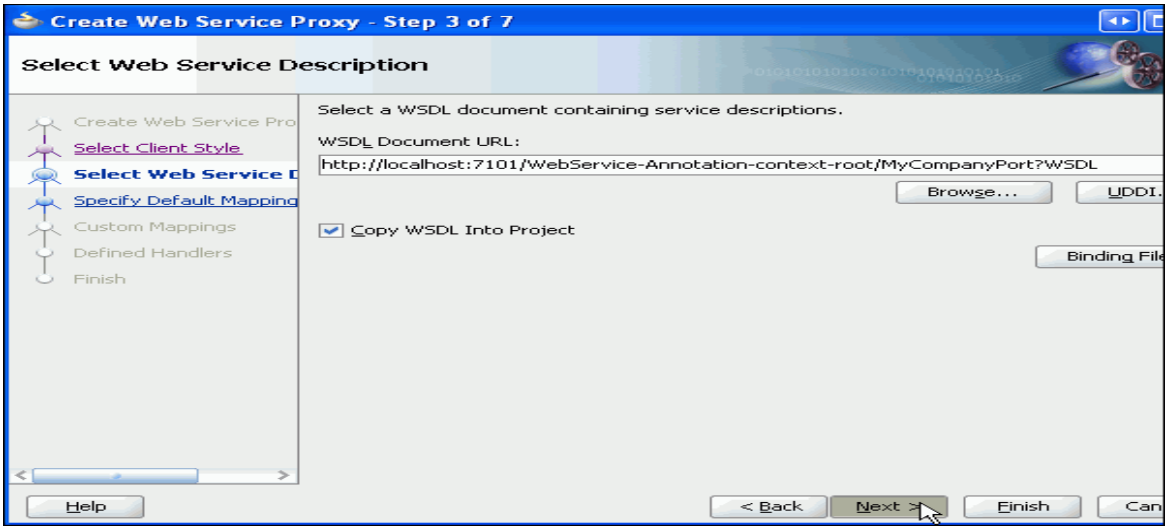


Esta acción invoca el asistente Crear servicio Web Proxy.

4. Haga clic en **Siguiente** para despedir a la página de bienvenida. En la página Seleccionar cliente de Estilo, seleccione el **estilo de JAX-WS y continúe al próximo paso.**



5. En la página Descripción del servicio Web copiar la dirección URL del servicio que se va a utilizar, dichas direcciones se proporcionan anteriormente en este manual y dar clic en continuar.



6. Para el resto de las páginas del asistente, los valores por defecto están bien. Puede hacer clic en **Siguiente** para examinar los pasos restantes, o haga clic en **Finalizar** para crear el proxy.

A partir de aquí ya se puede usar el servicio seleccionado en su nueva aplicación.