Universidad Central “Marta Abreu” de Las Villas.
Facultad Matemática, Física y Computación
Licenciatura en Ciencia de la Computación

TRABAJO DE DIPLOMA

Sistema de Información para el control del trabajo operativo en Tropas Guardafronteras de Cienfuegos

Autor: Michel Romero Esquijarosa

Tutores: MSc. Lisandra Díaz De la Paz

Ing. Carlos M. Acosta Chaviano

“Año 56 de la Revolución”

Santa Clara, 2014

Hago constar que el presente trabajo fue realizado en la Universidad Central Marta Abreu de Las Villas como parte de la culminación de los estudios de la especialidad de Ciencia de la Computación, autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

Firma del autor

Los abajo firmantes, 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

Pensamiento

"… al final se sabrá
que no es cuestión de títulos
sino la utilidad de los saberes …"
Buena Fe

Agradecimientos

*A la **UCLV***

por brindarme el hogar que fue para mí durante cinco grandiosos años y por haberme dado a dos de las personas más importantes con que cuento en la vida:

Laura y Thiago.

Dedicatoria

A mi madre,
por el ejemplo constante y la dedicación infinita;
A mis dos purrys por el amor y apoyo recíprocos;
A mi familia, por la fuerza brindada en los momentos tan duros y difíciles de toda la etapa estudiantil.

Resumen

Hasta el momento, el destacamento de TGF de Cienfuegos, realiza el proceso de almacenar las informaciones proporcionadas por los Oficiales Operativos de manera manual, lo cual trae consigo pérdida e inconsistencia en los datos que allí se manejan. Por consiguiente, el presente trabajo de diploma tiene como principal objetivo desarrollar un sistema de información operacional que permita automatizar dicho proceso. Para ello se incluyen las etapas de análisis de requerimientos, diseño, implementación y prueba del sistema. Además, se propone un diseño dimensional de un mercado de datos como parte del sistema de apoyo a decisiones con vista a futuros análisis de la información histórica.

Como resultado de esta investigación se obtiene un sistema de información que permite manejar y recuperar las informaciones referentes a los sucesos que ocurren en el litoral costero o en alta mar, a partir de un diseño de una base de datos orientado a los requerimientos, ofreciendo informaciones oportunas y la confección de informes para la jefatura en apoyo a la toma de decisiones.

Abstract

So far, the task of TGF Cienfuegos, makes the process of storing the information provided by the Operating Officers manually, which entails loss of information and inconsistency in data are handled there. Therefore, this diploma work has as main objective to develop an operational information system that allows automating the process. This stage of requirements analysis, design, implementation and testing of the system are included. In addition, we propose a design of a dimensional data mart as part of decision support system with a view to future analysis of historical information. As a result of this investigation an information system that allows you to manage and retrieve information relating to events occurring in the coastline or offshore, from a design of a database oriented requirements, offering information obtained timely reporting and the preparation for leadership in supporting decision-making.

Tabla de contenidos

Lista de figuras

Lista de tablas

Introducción

Con el transcurso del tiempo y el desarrollo indetenible de las tecnologías de la información, la gestión de datos se convierte en un asunto medular. El hombre, con el nacimiento de la ciencia computacional, como actor y beneficiario principal de este proceso, desempeña un rol activo en el perfeccionamiento de métodos y herramientas que le permiten automatizar la manera de procesar la información de forma transparente.

Los sistemas de información (SI) se han convertido en el soporte que garantiza el correcto funcionamiento de la gestión corporativa. Permite que las organizaciones se inserten en la competencia, mejoren su productividad, tomen decisiones acertadas garantizando información confiable y desenvuelvan coordinadamente sus procesos (Prieto and Martínez, 2004).

En la actualidad, un gran número de empresas invierten grandes sumas de dinero en hardware, software y equipos de telecomunicaciones para los SI, además de consultoría y servicios de negocios y administrativos. Entre 1980 y 2004 las inversiones de las empresas privadas en tecnología de información crecieron de 34 a 50 por ciento del capital total invertido (Laudon, 2008).

La utilización de esta tecnología aún resulta incipiente, especialmente en los países subdesarrollados y en vías de desarrollo. Aunque Cuba se incluye en este último grupo de naciones, ya se inserta en el empleo de herramientas de informatización. Ello responde al vertiginoso crecimiento del campo tecnológico, principalmente impulsado a partir de los proyectos de la Batalla de Ideas.

A pesar de ello, todavía existen en la nación instituciones y organismos que se mantienen fieles a métodos tradicionales de almacenamiento de información. Incluso, algunos que manejan grandes cantidades de datos y documentos aún no se afilian a los mecanismos tecnológicos.

El Ministerio del Interior (MININT) es una de estas instituciones que a pesar de tener un Órgano[1] orientado, entre otras funciones, al desarrollo de aplicaciones informáticas, todavía cuenta con departamentos rezagados que no hacen uso de las poderosas

[1] Órgano de Informática Comunicaciones y Cifras (OICC).

herramientas que proporciona la computación. Entre estos se encuentran las Tropas Guardafronteras (TGF) de Cienfuegos.

El Destacamento de TGF de Cienfuegos, perteneciente a la Dirección Nacional de las TGF, vela por la tranquilidad de las costas y aguas territoriales en el centro-sur de Cuba. Este departamento consolida su labor encomiable para hacer frente principalmente a los nuevos retos impuestos por el tráfico ilegal de drogas y estupefacientes, la emigración ilegal y el tráfico de personas; además de combatir los hechos vandálicos que suceden en la costa y hacer cumplir los decretos ley aprobados por el estado cubano referente a combatir las violaciones en el mar como lo son el Decreto Ley 194 y el 164, ampliando su espectro de trabajo y trayendo consigo una mayor proporción de datos a almacenar.

El Destacamento TGF en Cienfuegos[2] comprende seis regiones que ocupan un amplio territorio. Incluye las Capitanías de Cienfuegos y Casilda, los Puntos del Castillo de Jagua, Yaguanabo y Tunas de Zaza y la zona de Ciénaga de Zapata. En estas localidades los miembros del organismo velan por el orden y disciplina en la costa.

Colaboradores en cada una de las zonas componen el personal que durante las jornadas de trabajo reúne un amplio cúmulo de datos respecto a los sucesos delictivos ocurridos. Estas informaciones son recepcionadas por los Oficiales Operativos encargados de tramitarlas al Puesto de Mando para su posterior almacenamiento, tras realizar una evaluación primaria para comprobar su veracidad.

El proceso utiliza un soporte del sistema de dirección de las TGF, conocido como Sistema Informativo; encargado de establecer objetivos, principios, requerimientos, métodos, medios y procedimientos de trabajo. Todo ello favorece la calidad y oportunidad de la información, que además debe tener continuidad y plenitud en el tiempo, destinada a la jefatura con vista a la toma de decisiones.

La presente investigación propone desarrollar un mecanismo que agilice la realización de las tareas cotidianas y el acceso a la información almacenada en las TGF. Para ello se plantea el siguiente **objetivo general**:

[2] Se encuentra ubicado en el centro–sur del país, con un frente operativo de 461 KM de litoral costero y 116 millas náuticas, abarcando el sur de las provincias de Matanzas, Cienfuegos y Sancti Spíritus. Se extiende desde la desembocadura del río Grande en Majagua, al este, hasta el canal de Bretón al oeste en Matanzas.

Desarrollar un sistema informático que gestione las informaciones del trabajo operativo en Tropas Guardafronteras en Cienfuegos y sirva de base para la toma de decisiones.

Del planteamiento anterior se desprenden los siguientes **objetivos específicos**:

1. Estudiar la dinámica del trabajo operativo en la obtención de las informaciones para obtener un diseño adecuado de la base de datos orientado a los requerimientos.
2. Implementar una aplicación con una interfaz gráfica que resulte agradable y de fácil uso para los usuarios, ofreciendo reportes que permitan el análisis de los datos por la jefatura.
3. Proponer un diseño dimensional que sirva de base para una futura implementación de un mercado de datos.
4. Validar el sistema de información propuesto mediante la creación de casos de pruebas unitarias y de integración.

Preguntas de investigación:

1. ¿Qué datos son necesarios almacenar para el control del trabajo operativo?
2. ¿De qué manera se puede implementar una aplicación con una interfaz amena que permita manipular los elementos de la base de datos?
3. ¿Qué información es necesaria para elaborar los reportes destinados a la jefatura?
4. ¿Qué estructura utilizar para proponer un diseño dimensional de un mercado de datos para las TGF en Cienfuegos?
5. ¿Cómo se puede validar el código de la aplicación y la interacción de esta con la base de datos?

Valor práctico

El empleo de un sistema de información operacional permite que el personal del Departamento de Información de las TGF en Cienfuegos desarrolle su trabajo rutinario con mayor facilidad, agilizando el almacenamiento de datos que les proporcionan los oficiales operativos de cada área y la respuesta a las solicitudes de los mandos superiores.

Contribuye a la sustitución del trabajo manual de los operarios, pues la herramienta permite confeccionar informes y reportes, además de proveer datos concretos y brindar

información estadística. El acceso rápido a contenidos almacenados también fortalece el trabajo operativo en la costa y el mar. A través del sistema de información operacional se favorece el acceso rápido a informaciones, sin la necesidad de contactar directamente con los oficiales operativos o buscar en decenas de páginas algún elemento clave.

Relevancia social

Dado el rol social de las TGF como organismo, la agilización de sus tareas rutinarias, y el acceso rápido a información almacenada beneficia el enfrentamiento a los delitos. El acceso a datos más completos de un caso facilita la toma de decisiones a la hora del enfrentamiento y el trazo de estrategias para su aplicación a largo plazo.

La presente tesis se estructura en tres capítulos.

En el **primer capítulo se** realiza una descripción del problema a resolver, caracterizando su dominio**.** Se muestra la conceptualización de las principales bases teóricas a utilizar durante la investigación acerca de los SI en general, particularizando en los operacionales y los de apoyo a la toma de decisiones. También se hace alusión a otros elementos teóricos útiles para la investigación como: la arquitectura, funcionalidades y el ciclo de vida de los SI, y se describen las principales herramientas computacionales utilizadas.

En el **segundo capítulo** se abordan aspectos esenciales sobre el análisis y el diseño del sistema. Se analizan los requerimientos detectados, se definen los casos de uso a partir de dichos requerimientos y se diseñan diagramas UML necesarios en la etapa de programación. Además, se plantea el diseño del diagrama de clases utilizando el patrón de diseño Modelo-Vista-Controlador y se propone una solución al resguardo de los datos de la base de datos.

El **tercer capítulo** se centra en la implementación del sistema y en la validación del mismo a través de pruebas unitarias, utilizando el método del camino básico para diseñar los casos de prueba, y de integración para comprobar la interacción entre el sistema y la base de datos que utiliza. En ambos casos se ejecutan los casos de prueba utilizando JUnit. Como último aspecto se propone un mercado de datos mediante un diseño dimensional.

Capítulo 1 Aspectos teóricos sobre Sistemas de Información

En el presente capítulo se analizan los datos necesarios para el desarrollo de un sistema de información para las TGF de Cienfuegos. Además, se muestran los conceptos fundamentales sobre Sistemas de Información; así como los elementos indispensables para garantizar el funcionamiento como instrumento de una entidad determinada. Igualmente el contenido que se aborda explora los diferentes tipos de Arquitectura y Modelo de Datos, con énfasis en el Modelo Entidad Relación (MER) y el patrón Modelo−Vista−Controlador. Además se analizan las características de las tecnologías a utilizar como el entorno de desarrollo integrado NetBeans, la herramienta CASE ERECASE desarrollada en nuestra Universidad y el gestor de base de datos PostgreSQL.

1.1 Antecedentes del proyecto de investigación

En el año 2000 se implantó en el departamento de Información de TGF de Cienfuegos un sistema denominado Sistema de Información Operativo (SIO) que trabaja sobre una base de datos diseñada en Microsoft Access. La funcionalidad que brinda es el almacenamiento de los datos referente solamente a los hechos[3] que enfrentan los Servicios Operativos Terrestres o Navales. A partir de la emisión de la Orden Ministerial 23/2007 en el año 2007, que contempla los métodos de recogida de los datos, quedaron minimizadas las funcionalidades del SIO. Actualmente, este sistema tiene como desventaja la imposibilidad de guardar el historial de informaciones relacionadas con un caso, lo que obstaculiza hacer un seguimiento detallado a partir de las informaciones proporcionadas por las fuentes con vistas a trazar metas o acciones para el enfrentamiento oportuno e impide el almacenamiento de datos gráficos como la imagen. A la par de esta herramienta la operaria del Departamento de Información recepciona las informaciones que proveen los Oficiales Operativos y las plasma en una libreta o un documento Microsoft Word para confeccionar un resumen de los principales datos recopilados de un caso.

[3] Eventos que ocurrieron y se detectaron por el servicio operativo de las TGF.

1.2 Análisis de la problemática

El flujo informativo de las TGF correspondiente al problema de investigación se puede dividir en cuatro procesos, como se muestra en la figura 1.1.

Figura 1.1: Flujo informativo en las Tropas Guardafronteras.

La región en la que opera las TGF de Cienfuegos se divide en seis áreas con sus respectivos Oficiales Operativos que recogen las informaciones proporcionadas por las diversas fuentes, trabajando de conjunto con la población asentada en cada una de esas zonas. La captación de los elementos que conforman las informaciones son recogidas por la Oficina Nacional de Inspección Pesquera (ONIP) −que trabaja de conjunto con las TGF−, otros Órganos pertenecientes al MININT, informantes espontáneos que no son miembros del organismo pero que accidentalmente obtuvieron datos al estar presentes en el lugar de desarrollo del suceso y desean transmitirlos al Oficial Operativo. También

cuentan con los Servicios Operativos terrestres y navales, las Fuerzas Armadas Revolucionarias (FAR), los colaboradores activos, funcionarios honorarios y personas de confianza. Según el Subsistema Informativo de las informaciones es necesario registrar:

- Hora, fecha y lugar de detección.
- Hora y fecha en la que el Oficial Operativo tramita la información al Puesto de Mando.
- Amplia descripción de la información.
- Fuente que proporcionó los datos.
- Grado de relevancia[4] que posee.

Estas informaciones hacen referencia a sucesos que están en pleno desarrollo o sus autores piensan ejecutar en las próximas horas o días a partir de la obtención de los primeros datos. Las informaciones recogen aspectos como las embarcaciones y personas involucradas, drogas, objetos y armamento encontrados.

Las investigaciones son entregadas periódicamente al Puesto de Mando para su almacenamiento. Dicho período varía en dependencia del flujo de datos recopilados, el cual puede ser semanal, diario o varios en un mismo día teniendo en cuenta la relevancia que estos poseen de acuerdo al efecto que podrían tener sobre un individuo, medio físico del estado o conjunto de personas o animales, almacenándose su orden de llegada.

Todas las informaciones que poseen datos de un mismo objetivo a operar se relacionan con un caso el cual pertenece a una de las seis áreas donde se identifica con un número. Los casos pertenecen a una categoría y subcategoría y son dirigidos por el Oficial Operativo de esa zona. Si el objetivo de los implicados fue cumplido es necesario conocer la hora, el lugar y la fecha en que se consumó el hecho. Adjunto al caso, luego de su cierre, se almacena una descripción general de este, el *modus operandi* de los implicados, las deficiencias del sistema de enfrentamiento al momento de operar, las medidas tomadas, otros datos de interés y la fecha de cierre. Es de importancia también almacenar si el caso es frustrado por recalo, para actividades de narcotráfico, o por alguna otra causa en la que no interviene el sistema de TGF. Si el caso está relacionado

[4] La información se puede clasificar en Muy Relevante, Relevante y Poco Relevante de acuerdo a la importancia que esta posea.

con algún hallazgo de droga, objeto o armamento, se almacena el tiempo aproximado de permanencia en el lugar encontrado.

Según sea la importancia de una información llegada al Puesto de Mando se abre un expediente relacionado con el caso. Los expedientes tienen un número que se designa de acuerdo al orden de expedientes abiertos en el año, un nombre, la fecha de apertura y la fecha de cierre que coincide con la fecha de cierre del caso.

Los datos pueden estar relacionados con embarcaciones, objetos, drogas, armamentos y personas involucradas.

Las embarcaciones pueden ser nacionales o extranjeras, en los dos casos hay que especificar la nacionalidad. Tienen además:

- Folio (con el que se identifican).
- Nombre
- Tipo[5]
- Tipo de motor y potencia.
- Tipo de combustible.
- Matrícula.
- Eslora, manga, puntal y calado.
- Medios que existen a bordo.
- En caso de realizarle sondeo, su resultado.
- País de procedencia y destino.
- Motivo de la entrada a Cuba en caso de embarcaciones extranjeras.
- Tipo de carga.
- Estado técnico.
- Cantidad de combustible.

Las embarcaciones pueden estar relacionadas con averías en casos como accidentes navales, arribos forzosos de naves o aeronaves y solicitudes de auxilio en donde se almacena la fecha, el tipo de avería y la posible solución; lo que requiere de la agilidad del flujo de información de TGF para la toma urgente de medidas de auxilio. También

[5] Se clasifican de acuerdo a sus dimensiones y objetivos en Lista I, Lista II, Lista III, Lista IV, Lista V y Lista VI.

los medios navales, de acuerdo con las medidas tomadas pueden ser ocupados por las TGF en una fecha determinada y en una localización.

De los objetos se almacena el tipo, el color, sus dimensiones, el estado, la cantidad, el peso total, el lugar de depósito, una imagen y las posibles inscripciones que posean. Del armamento se recoge el tipo, la marca, la nacionalidad, el propietario, el estado en el que se encuentra, una imagen y la cantidad.

Dada la importancia del enfrentamiento a las actividades de narcotráfico se necesita almacenar de las drogas su tipo, cantidad, el tipo de envoltura, las dimensiones, peso total, si tiene o no indicios de escamoteo, resultados del laboratorio, una imagen y las posibles inscripciones que tenga.

De las personas involucradas se registra su carnet de identidad o pasaporte, nacionalidad, nombre y apellidos, procedencia, ocupación y una foto. En el caso de realizarle un interrogatorio se registra el resultado.

Los casos pueden tener las categorías siguientes con sus respectivas subcategorías:

- Actividad Enemiga
  - Infiltración
  - Exfiltración
  - Acción terrorista
  - Sabotaje
  - Provocación
  - Propaganda Contra/Revolucionaria
  - Hallazgo de Armamento
- Actividad de Narcotráfico
  - Bombardeo de Droga
  - Trasbordo de drogas
  - Captura de personas dedicadas a la búsqueda de recalos
  - Embarcación con drogas
  - Transmisiones radiales
  - Embarcaciones y personas sospechosas de narcotráfico
  - Hallazgo de drogas

- Recalo de Drogas
- Escamoteo de drogas
- Hallazgo de envoltura vacía

- Actividades sospechosas
  - Movimiento sospechoso de extranjeros en la costa
  - Movimiento sospechoso de nacionales en la costa
  - Avistamiento de posibles hombres rana
  - Avistamiento de naves
  - Otras actividades sospechosas
- Entradas y salidas ilegales del país
  - Salidas ilegales del país
  - Entradas ilegales al país
  - Preparativos de SIP[6]
  - Plan de SIP
  - Arribo de comunitarios
  - Devolución de emigrantes ilegales cubanos
- Actividades violatorias
  - Violación marítima
  - Violación del espacio aéreo
  - Violación de la zona restringida
  - Infracción del Decreto Ley 194
  - Violación de embarcaciones con emigrantes extranjeros
  - Violación del horario de despacho
  - Contaminación de las aguas
  - Violación de la flora y la fauna
  - Otras actividades violatorias
- Otros hechos de interés
  - Accidente naval

[6] Salidas ilegales del país.

- Accidente aéreo
- Hallazgo
- Arribo forzoso de naves o aeronaves
- Robo de embarcaciones
- Pérdida de embarcaciones
- Otros avistamientos
- Náufragos
- Solicitud de auxilio por embarcaciones
- Otros hechos de interés

1.3 Principales conceptos y tipos de Sistemas de Información

Los datos representan realidades concretas sobre hechos ocurridos en las organizaciones o en el entorno físico. Según lo expresado en (Rodríguez, 2006), los datos aisladamente no poseen significado alguno. Sólo cuando un conjunto de datos se estructuran, organizan o examinan conjuntamente a la luz de un enfoque, hipótesis o teoría se puede apreciar la información contenida en dichos datos. En sentido general, la información es un conjunto organizado de datos procesados, que constituyen un mensaje que cambia el estado de conocimiento del sujeto o sistema que recibe dicho mensaje.

Los SI constituyen una herramienta computacional para almacenar y gestionar datos que satisfacen una necesidad de información. Contienen un grupo de elementos que mantienen relación entre sí (Laudon, 2008). Los mismos, de acuerdo con el propio autor, "recolectan (o recuperan), procesan, almacenan y distribuyen información". Todo ello para apoyar la labor con la información de una entidad, elevar el nivel de conocimiento y viabilizar el apoyo a la toma de decisiones y el desarrollo de acciones.

El recurso material resulta parte integrante de los SI. Comprende el hardware de la computadora, y el recurso humano como el personal operario para la explotación de las funcionalidades del mismo.

Los SI cumplen con una amplia gama de propósitos en la vida práctica, en concordancia, sus diversos tipos interesan tanto a los operarios de una empresa como a sus directivos. Los SI se clasifican de acuerdo a su estructura y funcionamiento.

Según (Montilva, 1999) existe la presencia de dos tipos de SI. El primero, catalogado como formal, se fundamenta "en un conjunto de normas, estándares y procedimientos que permiten que la información se genere y llegue a quién la necesite en el momento deseado". Mientras, la segunda clasificación se refiere al tipo informal, que se basa "en la comunicación no formalizada, ni predefinida entre las personas de la organización".

A su vez, los SI se ubican en un segundo nivel de clasificación, ahora en correspondencia con el elemento principal de proceso de la información conforme al criterio de (Ayala, 2006). Cuando el hombre, auxiliado por cierto equipo (máquinas de escribir, calculadoras, archivos, etc.), realiza las principales funciones de recopilación, registro, almacenamiento, cálculo y generación de información, se entienden como manuales. Contrariamente, se les llama mecanizados, cuando las principales funciones de procesamiento las ejecuta una maquinaria.

Los sistemas manuales son adecuados en procesos sencillos, que manejan pequeños volúmenes de datos, sin efectuar cálculos complejos y en los que mantener actualizada la información no resulta problemático. En cambio, los mecanizados tienden a sistematizar aquellas actividades complejas, que requieren manipular altos volúmenes de datos en un tiempo corto de respuesta.

Un solo tipo de SI no puede proveer a la organización toda la información que necesita debido a la existencia de diferentes intereses, especialidades y niveles de organización. En este sentido se conciben diferentes tipos de SI.

De acuerdo con (Karen and Lares, 2000), para unificar las funciones comunes los SI se distribuyen en tres niveles de jerarquía organizativa: el transaccional u operacional, el de apoyo a las decisiones y el estratégico. El primero se dirige a los procesos operacionales para automatizarlos como actividad de un negocio. Por tal motivo, comúnmente se seleccionan como los primeros a implantar en una organización, pues se conforman como la base de la información productiva utilizada posteriormente.

Por otra parte, los sistemas de apoyo a las decisiones constituyen la plataforma de información e intentan integrar los sistemas transaccionales más relevantes de la empresa; así como generar información para ser utilizada por los mandos intermedios y la alta administración (toma de decisiones).

Relativos a la interacción con el entorno para responder a los cambios, los sistemas estratégicos se emplean con el propósito de buscar superioridad competitiva, o bien para reducir la ventaja de los competidores. Según (Karen and Lares, 2000), los desarrolla el personal de la organización.

En la figura 1.2 se muestra la tipología expuesta por (Laudon, 1996) similar a la planteada por (Karen and Lares, 2000) al clasificar los SI en nivel operativo, nivel gerencial y nivel estratégico.

Figura 1.2: Relación de los SI con los niveles organizacionales de una empresa.

1.3.1 Sistemas de información operacionales

De acuerdo con (Laudon et al., 2008) estos sistemas también conocidos como Sistema de Procesamiento de Transacciones (TPS, por sus siglas en inglés), son recolectores de gran cantidad de información para realizar sus operaciones, que generan grandes sumas de datos, ahorran en gran medida el trabajo manual, y sus cálculos y procesos suelen ser simples y poco sofisticados. Consiste en un sistema computarizado que ejecuta y registra las transacciones ordinarias, cotidianas que se requieren para la conducción de la empresa como: captura de órdenes de ventas, reservaciones hoteleras, nóminas o mantenimiento de los registros de empleados.

El objetivo primordial de estos sistemas es responder las preguntas rutinarias y dar seguimiento al flujo de transacciones en la organización. Los gerentes necesitan de estos

sistemas para supervisar el estado de las operaciones internas y las relaciones de la empresa con el entorno externo además de ser productores importantes de información para los demás tipos de sistemas.

1.3.2 Sistemas de apoyo a la toma de decisiones

Los sistemas de apoyo a la toma de decisiones, o DSS por sus siglas en inglés, permiten realizar análisis de diferentes cuestiones de una organización para decidir qué rumbo tomar. Se enfocan en extraer grandes cantidades de datos para apoyar a la gerencia intermedia en la toma de decisiones. Suelen implantarse a partir de sistemas operacionales previamente establecidos en la empresa, ya que estos últimos constituyen las fuentes de datos, además de utilizar otras fuentes externas. Son interactivos y amigables gracias a su completo diseño gráfico y visual, utilizando varios modelos para el análisis de datos. Ofrecen acceso a bases de datos y herramientas de análisis permitiendo la creación de reportes, pronósticos, gráficos y pueden apoyar el intercambio de información dentro de la organización (Laudon, 2008).

Algunos DSS son orientados a datos y utilizan procesamiento analítico en línea (OLAP) y minería de datos para analizar tendencias y patrones de comportamiento en grandes concentraciones de datos.

Sus componentes son las bases de datos utilizadas para consultas y análisis, un sistema de software con modelos, minería de datos y otras herramientas analíticas, así como una interfaz de usuario, como se observa en la figura 1.3.

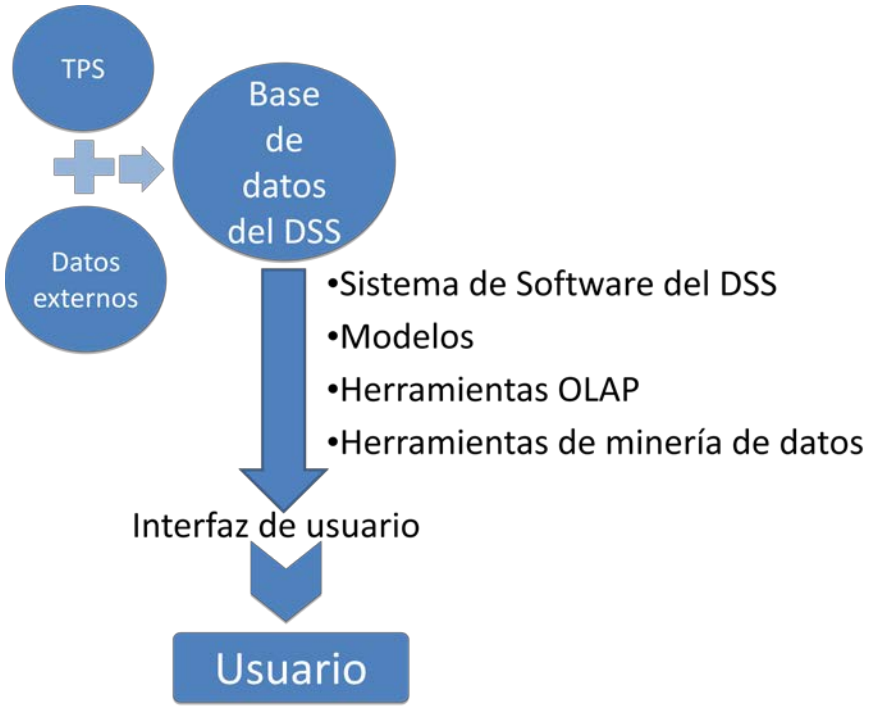

Figura 1.3: Panorama de un sistema de apoyo a la toma de decisiones (Laudon, 2008).

La base de datos de un DSS según (Laudon, 2008), es un conjunto de datos históricos y actuales de varias aplicaciones que pueden combinarse con datos externos. Alternativamente, la base de datos del DSS podría ser un enorme almacén de datos actualizado de manera continua por los sistemas fuentes operacionales que capturan las transacciones de la organización.

En la actualidad existen muchos DSS que tienen una interfaz Web para aprovechar el despliegue de gráfico, la interactividad y la facilidad de uso.

1.3.3 Almacén de datos y mercado de datos

La tecnología de los almacenes de datos (DW por sus siglas en inglés) basa sus conceptos y diferencias entre los dos tipos de sistemas de información tratados anteriormente: los sistemas técnico-operacionales y los sistemas de soporte de decisiones.

Un DW es una colección de datos transformados y separados físicamente de la fuente operacional que se usa para el soporte del proceso de toma de decisiones. El objetivo principal de esta tecnología es proporcionar una plataforma sólida, a partir de los datos

que se guardan en un amplio período de tiempo, para hacer análisis respecto a determinados indicadores de una empresa.

De acuerdo con (Inmon, 2005), un DW se define como un repositorio de datos con las siguientes propiedades:

- Orientado a temas: Los datos en la base de datos están organizados de manera que todos los elementos de datos relativos al mismo evento u objeto del mundo real queden unidos entre sí.
- Variante en el tiempo: Los cambios producidos en los datos a lo largo del tiempo quedan registrados para que los informes que se puedan generar reflejen esas variaciones.
- No volátil: La información no se modifica ni se elimina, una vez almacenado un dato, éste se convierte en información de sólo lectura y se mantiene para futuras consultas.
- Integrado: La base de datos contiene los datos de todos los sistemas operacionales de la organización y dichos datos deben ser consistentes.

El DW es un sistema que extrae, limpia, ajusta, y proporciona datos de un origen a un almacén de datos dimensional y luego apoya e implementa la consulta y análisis con el propósito de tomar decisiones (Kimball and Ross, 2011). Se puede segmentar en varios mercados de datos que son subconjuntos del DW para cierto grupo de usuarios o funciones del negocio, se construye para consultas más rápidas y menos usuarios.

El diseño de un DW se estructura mediante un modelo multidimensional, representado a través de un esquema relacional que se basa en los conceptos de dimensión y medida.

1.4 Arquitectura y funcionalidades de los SI

Los principales componentes de los sistemas de información son hardware y software, las bases de datos, las telecomunicaciones, almacenes de datos y los procedimientos gestionados por diferentes especialistas (Laudon, 2008) como se presenta en la figura 1.4.

Figura 1.4: Despliegue de la arquitectura de un SI.

Se considera al hardware como el equipamiento físico de computación que se utiliza para llevar a cabo las actividades de entrada, procesamiento y salida.

El software son las aplicaciones informáticas que se dividen en dos grandes clases: software de sistema y software de aplicación. El software del sistema principal es el sistema operativo, que gestiona el hardware, los datos y los archivos de programa, y otros recursos del sistema y proporciona un medio para el usuario de controlar el ordenador, generalmente a través de una interfaz gráfica de usuario (GUI). El software de aplicación son los programas diseñados para manejar tareas específicas para los usuarios.

Las telecomunicaciones son la transmisión electrónica de señales de comunicación que permiten a las organizaciones conectar entre sí sistemas de computación para integrar redes. Las redes sirven para enlazar las computadoras de un edificio, un país o el mundo entero, con la finalidad de establecer comunicaciones electrónicas.

Una base de datos es una colección de datos relacionados entre sí (registros), organizado para que los registros individuales o grupos de registros se puedan recuperar para satisfacer varias necesidades. Los almacenes de datos contienen los datos de los archivos, recopilados a lo largo del tiempo, que pueden ser extraídos para obtener información con el fin de desarrollar y comercializar nuevos productos, atender mejor a los clientes existentes, o llegar a nuevos clientes potenciales.

Los autores (Laudon, 2008) definen el término recurso humano como las personas calificadas, un componente vital de cualquier sistema de información. El personal técnico y de operaciones de desarrollo incluye gerentes, analistas de negocio, analistas y diseñadores de sistemas, administradores de bases de datos, programadores, especialistas en seguridad informática, y operadores de computadoras. Además, todos los trabajadores de una organización deben estar capacitados para utilizar las capacidades de los SI.

Los SI trabajan sobre cuatro funcionalidades para producir la información que las organizaciones necesitan para la toma de decisiones y el control de operaciones: la entrada, el almacenamiento, el procesamiento y la salida (Burch and Grudnitski, 1992).

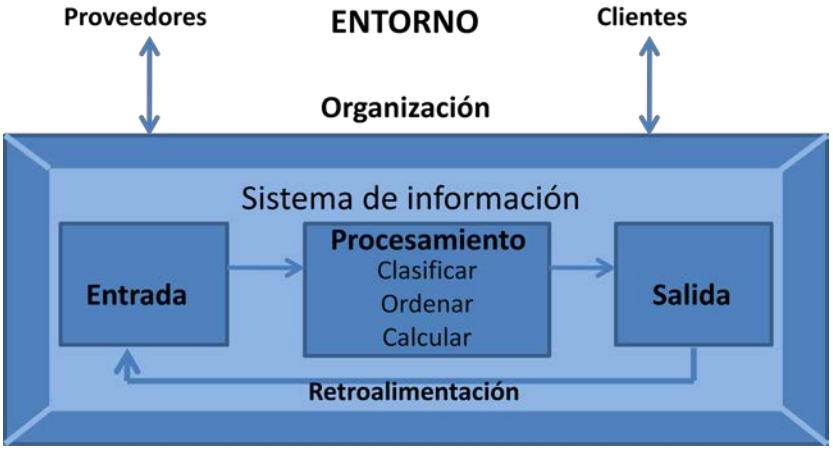

Figura 1.5: Funciones de un SI.

La entrada de información es el proceso mediante el cual los SI toman los datos que requieren para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos. Las unidades típicas de entrada de datos a las computadoras son las

terminales, las cintas magnéticas, las unidades de diskette, los códigos de barras, los escáneres, la voz, los monitores sensibles al tacto, el teclado y el mouse, entre otras.

El almacenamiento de información es una de las actividades o capacidades más importantes que tiene una computadora, según (Burch and Grudnitski), ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos. La unidad típica de almacenamiento son los discos magnéticos o discos duros, los discos flexibles y los discos compactos (CD-ROM).

El procesamiento de información es la capacidad de los SI para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida (Burch and Grudnitski, 1992). Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que estén almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada para la toma de decisiones.

La salida de información es la capacidad de los SI para obtener la información procesada o bien datos de entrada al exterior. Las unidades típicas de salida son las impresoras, terminales, diskettes, cintas magnéticas, la voz y los graficadores, entre otros. Es importante aclarar que la salida de un SI puede constituir la entrada a otro SI o módulo. En este caso, también existe una interfaz automática de salida. Existe una retroalimentación en la salida que se devuelve al usuario para ayudarle a evaluar o corregir la etapa de entrada.

La arquitectura a implementar en la propuesta de solución figura en la siguiente imagen, atendiendo a la estructura tecnológica desplegada en las TGF de Cienfuegos:

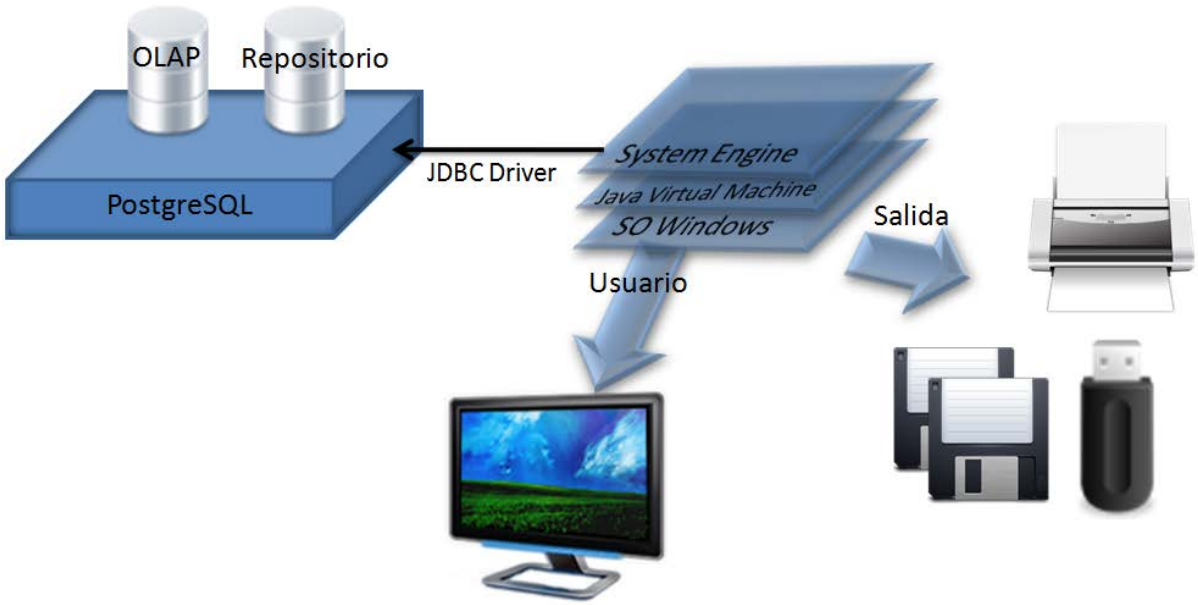

Figura 1.6: Arquitectura de la propuesta de solución.

1.4.1 Ciclo de vida de los SI

Los SI requieren para su desarrollo la integración de varios factores, a la vez que se construyen sobre una secuencia de pasos que fortalecen su diseño. Cuando la organización decide desarrollar un sistema reviste gran importancia la concepción inicial del mismo, por ello es preciso consultar al capital humano que posee el conocimiento de sus actividades, de los recursos disponibles y las necesidades de información.

Las opiniones de (Laudon et al., 2008) y (Ayala, 2006) coinciden en cuanto al proceso de desarrollo de los SI como muestra la figura 1.7.

Figura 1.7: Proceso de desarrollo de sistemas.

- **Análisis de la problemática**

Consiste en definir el problema, identificar sus causas, especificar la solución e identificar los requerimientos de información que debe satisfacer una solución de sistemas, además de elaborar un panorama general de la estructura organizacional y los sistemas existentes, identificando a los principales responsables y usuarios de datos junto con el hardware y software existentes.

Mediante el examen de documentos de trabajo, reportes y procedimientos, la observación de operaciones de sistemas y la entrevista a usuarios clave, el analista puede identificar las áreas con problemas, los principales caminos de acción a seguir y los objetivos que tendría que cumplir una solución que con frecuencia es construir un nuevo sistema de información o mejorar el existente. Además de identificar los recursos humanos, materiales y económicos a participar.

- **Diseño de sistemas**

El diseño de sistemas resulta la piedra angular en la vida de un SI. Constituye el modelo o plan general donde se detalla las especificaciones del sistema que producirán las funciones identificadas durante el análisis de sistemas. Tiene el propósito de establecer los aspectos lógicos y físicos de las salidas, modelos de

organización y representación de datos, entradas y procesos que componen el sistema. En las entradas se establecen los medios (documentos, parámetros, etc.) que alimentan y actualizan los archivos que integran el sistema, determinado el contenido (datos, cifras de control, etc.) y medio adecuados. En la definición de procesos se establece la arquitectura del sistema, los principales procesos que se llevan a cabo y sus interrelaciones mediante las entradas y salidas que manejan.

En esta etapa se define la base de datos, determinando el conjunto de datos a manejar con sus características físicas. Los usuarios deben tener suficiente control sobre el proceso del diseño para asegurar que el sistema refleje sus prioridades de negocio y necesidades de información.

- **Programación**

En esta etapa las características técnicas del sistema que son preparadas durante la etapa de diseño se traducen en código de programa. Se definen las actividades de programación a realizar, se formulan estrategias donde se eligen las técnicas y estándares a utilizar y los lenguajes y paquetes de programación.

- **Prueba**

Se deben realizar pruebas exhaustivas y completas para determinar si el sistema produce los resultados correctos. Los datos de las pruebas se deben preparar cuidadosamente, repasar los resultados y hacer las correcciones en el sistema. Las pruebas de un SI pueden dividirse en tres tipos (Ayala, 2006):

- **Pruebas unitarias:** consisten en comprobar por separado cada programa del sistema.
- **Pruebas del sistema:** comprueban el funcionamiento en conjunto del SI determinando si los módulos independientes funcionan juntos.
- **Prueba de aceptación:** proporciona la certificación final de que el sistema está listo para usarse en una situación de producción. Son evaluadas por los usuarios y revisadas por la administración.

- **Conversión**

Es el proceso de cambiar del sistema antiguo al sistema nuevo (Laudon et al., 2008). Esta etapa requiere que se capacite a los usuarios finales para que puedan usar el nuevo sistema. Pueden emplearse cuatro estrategias principales de conversión:

- **Estrategia en paralelo:** los dos sistemas, el nuevo y el antiguo, se ejecutan juntos durante un tiempo hasta que todos en la organización estén seguros de que el nuevo funciona correctamente.
- **Conversión directa:** reemplaza totalmente el sistema anterior con el nuevo en un día designado. Éste es un método muy arriesgado, que puede ser potencialmente más costoso que el anterior si se encuentran problemas graves en el nuevo sistema, pues no hay ningún otro sistema al cual acudir.
- **Estudio piloto:** presenta el sistema a sólo un área limitada de la organización, como un sólo departamento o unidad operativa. Cuando esta versión piloto esté completa y trabajando sin problemas, se instala en toda la organización, ya sea simultáneamente o por etapas.
- **Enfoque por fases:** introduce el nuevo sistema en etapas, ya sea por funciones o por unidades organizacionales.

- **Producción y mantenimiento**

Durante esta etapa, los usuarios y los especialistas técnicos revisan el sistema para determinar vienen qué grado se ha cumplido con los objetivos originales y decidir si se requiere alguna revisión o modificación. Para un adecuado uso del sistema se cuenta con la documentación necesaria y debidamente actualizada. Es preciso emplear medidas de seguridad en el acceso al sistema en el que solamente personal autorizado lo puede gestionar.

Se deben realizar los respaldos necesarios al sistema después de cada modificación o actualización procurando conservar el número adecuado de versiones.

El mantenimiento se encarga de corregir las fallas detectadas durante la operación de un SI, así como el de realizar las modificaciones pertinentes a los nuevos requerimientos que se van presentando para mejorar la eficacia del proceso. Las principales funciones a realizar son detectar la falla o planteamiento del nuevo requerimiento, definir ajustes a realizar, ejecución de las modificaciones necesarias

para satisfacer los requerimientos planteados, probar las modificaciones y actualización de la documentación y adiestrar al personal involucrado en la operación del sistema.

Es importante en esta etapa hacer un respaldo del sistema antes de comenzar a realizar las modificaciones, así como respetar las normas, estándares y procedimientos que han respaldado su construcción.

1.5 Herramientas utilizadas

Para la conformación del presente trabajo se utilizan varias herramientas que facilitan el trabajo del desarrollador. Haciendo uso de estas herramientas, se puede diseñar bases de datos que modelen correctamente el problema, utilizando una interfaz gráfica de fácil uso, y la estructura de la programación a seguir, contando con entornos de diseño y desarrollo amenos para el diseñador y el programador respectivamente.

1.5.1 ERECASE

La ingeniería de software asistida por computadora (CASE, por sus siglas en inglés) es la automatización de metodologías paso a paso para el desarrollo de software y de sistemas para reducir la cantidad de trabajo repetitivo que el diseñador necesita hacer.

Cumpliendo con este propósito en el Centro de Estudios de Informática el autor Carlos García (2009) implementa la herramienta ERECASE para el diseño de bases de datos utilizando como modelo conceptual el modelo Entidad−Relación−Extendido (ERE). A través de la interfaz de usuario de la herramienta ERECASE, el diseñador construye los diagramas ERE, los que son primeramente transformados en esquemas relacionales y luego en sentencias SQL que permiten crear la base de datos.

Según plantea (García, 2009), la herramienta se caracteriza por la inclusión de un conjunto de construcciones del modelo ERE para lograr una mejor expresividad en el diagrama. Comprende “entidades fuertes y débiles; interrelaciones de asociación recursivas, binarias y ternarias; interrelaciones débiles; interrelaciones de subconjuntos (ISA) y jerarquías de generalización/especialización”, así como agregación y la interrelación de asociación con dependencia de existencia.

Dicho autor considera la realización de validaciones estructurales en un esquema en la etapa de diseño, para evitar inconsistencias como ciclos formados por los tipos de entidades e interrelaciones cuyas restricciones de cardinalidad son inconsistentes.
Además se suma la transformación de un esquema al modelo relacional, la validación estructural del diagrama ERE, sobre la base de las cardinalidades máximas y mínimas de las relaciones y la generación de código SQL para la creación de la Base de Datos física en un Sistema Gestor de Bases de Datos (SGBD) determinado.

1.5.2 PostgreSQL

PostgreSQL es un sistema gestor de bases de datos objeto-relacional de código abierto. Según (B., 2008), sus principales características son:

1. Implementación del estándar SQL92/SQL99.
2. Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, cadenas de bits, etc. También permite la creación de tipos propios.
3. Incorpora una estructura de arreglos de datos.
4. Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
5. Permite la declaración de funciones propias, así como la definición de disparadores.
6. Soporta el uso de índices, reglas y vistas.
7. Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
8. Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

1.5.3 NetBeans

Esta herramienta es un entorno de desarrollo integrado (IDE por sus siglas en ingles) en el que se puede programar en Java y C++. Algunos de los beneficios técnicos de este IDE incluyen (Myatt, 2008):

- Interfaz gráfica de usuario (GUI).

- Agrupación de código fuente y los archivos de configuración en el concepto de un proyecto.
- Integración estrecha con el compilador.
- Acoplamiento con un repositorio de código fuente.
- Capacidad para utilizar plugins y herramientas de terceros.
- Habilidad para depurar código, ejecutando una línea a la vez.
- Reduce el tiempo de ciclo de desarrollo.
- Aumenta la calidad y la fiabilidad de su código.
- Estandariza los procesos de desarrollo de software.

1.5.4 JUnit

JUnit es un conjunto de clases (framework) que son utilizadas en programación para hacer pruebas unitarias de aplicaciones Java. Permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera (Tahchiev et al., 2010).

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

1.6 Conclusiones parciales

Durante este capítulo se analizó la problemática existente en las TGF de Cienfuegos identificando sus antecedentes y la manera que actualmente utilizan para obtener las informaciones. Se describieron las características de los SI, junto a su arquitectura y sus funcionalidades, particularizando en los operacionales y los de apoyo a la toma de decisiones. Además, se detallaron las herramientas a utilizar en las etapas de diseño, implementación y prueba de un sistema informático.

Capítulo 2 Análisis y diseño del sistema

El presente capítulo aborda aspectos esenciales sobre el análisis y el diseño del sistema. Se especifican los requisitos tanto funcionales como no funcionales del sistema para tener un amplio conocimiento del caso de estudio. Además, se definen los casos de uso a partir de dichos requerimientos, se diseñan los diagramas de actividad y componentes del sistema y la base de datos correspondiente. Otro tópico que se trata en el capítulo es el patrón arquitectónico a seguir haciendo uso del diagrama de clases y se plantea el modo de resguardar los datos de la base de datos.

2.1 Requerimientos del sistema

Los requerimientos del sistema establecen en detalle los servicios, restricciones y metas que debe cumplir el sistema y se definen a partir de las consultas con los usuarios. Se pueden especificar en dos grupos, los requerimientos funcionales y los no funcionales.

En (Sommerville, 2005) y (Pressman, 2010) se precisa que los requerimientos funcionales son declaraciones de los servicios que proveerá el sistema, de la manera en que este reaccionará a entradas particulares y de cómo se comportará en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también declaran explícitamente lo que el sistema no debe hacer.

Estos autores afirman que los requerimientos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema. Precisan también que existen disímiles restricciones de los servicios, aunque en el presente trabajo se tienen en cuenta las relacionadas con las de seguridad, eficiencia, espacio y estándares a seguir.

Requerimientos funcionales:

1. Autentificarse.
2. Almacenar datos sobre los casos y sus informaciones.
3. Almacenar imágenes relacionadas con las personas involucradas, drogas, armamentos, embarcaciones y objetos.
4. Guardar datos relacionados con averías y ocupaciones de las embarcaciones.
5. Guardar datos de los expedientes relacionados con los casos.

6. Visualizar las informaciones ordenadas por orden de llegada de un caso específico.
7. Obtener reportes asociados a los casos con sus informaciones en una fecha dada.
8. Visualizar estadísticas relacionadas con los grados de relevancia de las informaciones proporcionadas por las fuentes humanas, dado un período de tiempo.
9. Obtener reportes asociados a los casos de Actividades de narcotráfico en un período de tiempo dado.

Requerimientos no funcionales:

1. Memoria RAM de la máquina hospedera debe ser 512 megabytes o superior.
2. Espacio en disco para el código fuente debe ser mayor o igual que 120 megabytes.
3. Espacio en disco igual a 10 megabytes o superior para el repositorio de la base de datos.
4. Sistema Operativo hospedero debe ser Windows XP o superior.
5. Hacer una copia de resguardo de la base de datos cada vez que se cierre el sistema.

2.2 Diagrama de casos de uso

El modelo de casos de uso se utiliza para conseguir un acuerdo con los usuarios y clientes sobre qué debe hacer el sistema para ellos. Proporcionan un medio sistemático e intuitivo de capturar requisitos funcionales dirigiendo todo el proceso de desarrollo debido a que la mayoría de las actividades como el análisis, diseño y prueba se llevan a cabo partiendo de los casos de uso.

Grady Booch (2009) define formalmente un caso de uso como: *"… una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto"*.

El modelo de casos de uso se encarga también de delimitar el sistema definiendo las funciones que debe cumplir para un usuario específico como se observa en la siguiente figura. El actor Usuario permite realizar acciones como gestionar expediente, caso e información, entre otras que se aprecian en la figura 2.1.

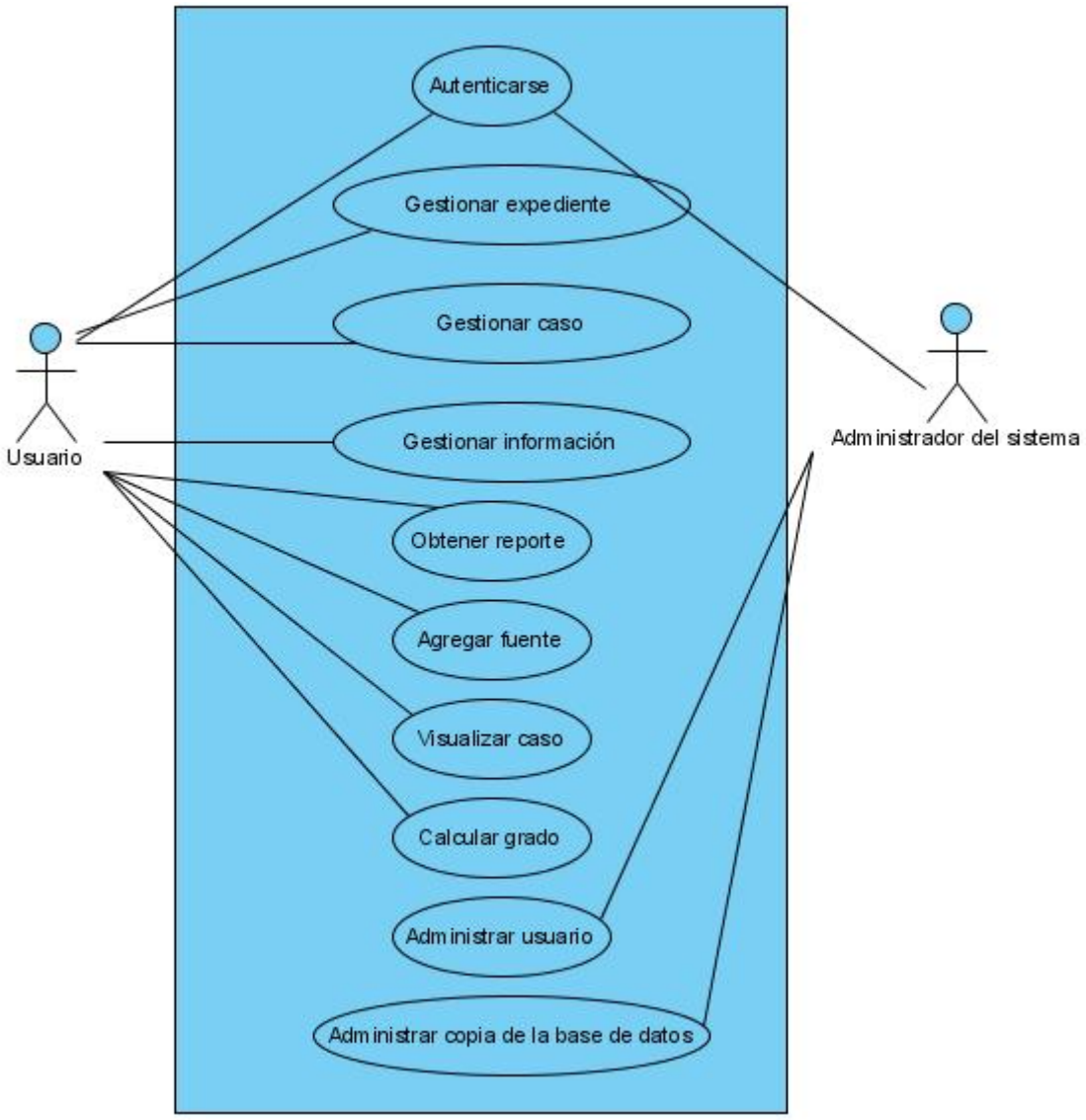

Figura 2.1: Diagrama de casos de uso para los actores Usuario y Administrador.

Para agregar el volumen de datos que maneja el Departamento de Información el sistema cuenta con la funcionalidad de "Gestionar las informaciones" [véase la figura 2.2]. A continuación se describe detalladamente en qué consiste el caso de uso.

Caso de uso: Gestionar información

Precondiciones: Usuario autenticado

Propósito: Agregar datos a un caso a través de los datos de las informaciones.

Resumen: El usuario al seleccionar el área y número del caso prosigue a introducir las informaciones con sus datos, prescindiendo o no de agregar una fuente nueva.

Postcondiciones: Datos actualizados de un caso.

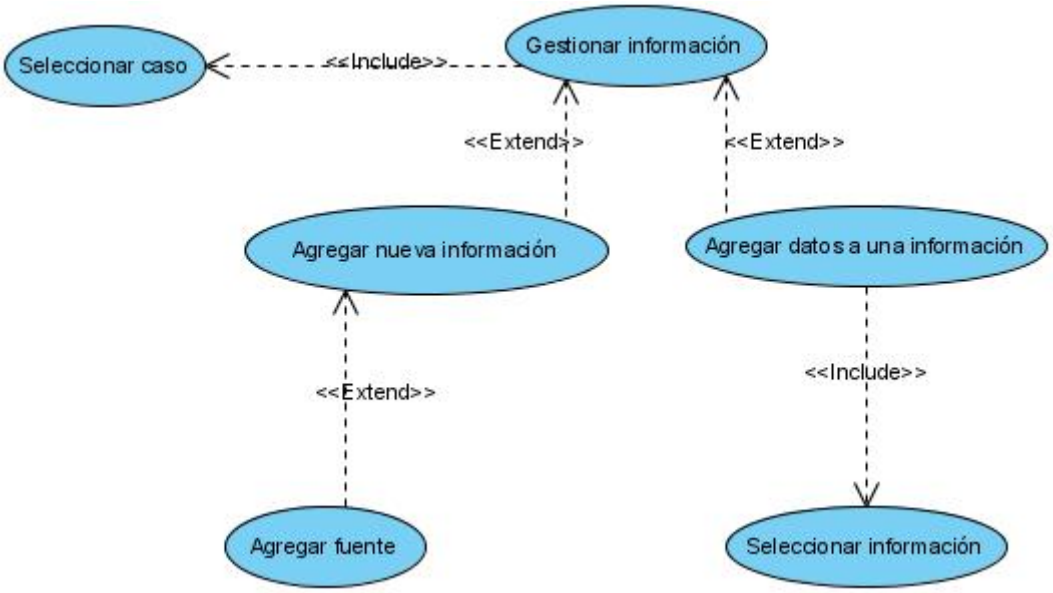

Figura 2.2: Diagrama del caso de uso Gestionar información (extensión).

El diagrama que se esboza en la figura 2.3 corresponde al caso de uso "Agregar datos a una información" encargado de granular las informaciones que llegan al Puesto de Mando.

Caso de uso: Agregar datos a una información

Precondiciones: Usuario autenticado

Propósito: Agregar datos a una información proporcionados por una fuente.

Resumen: El usuario selecciona el número de la información para agregar los datos relacionados con personas involucradas, embarcaciones, drogas, armamentos u objetos para actualizar la información y el caso.

Postcondiciones: Datos actualizados de una información.

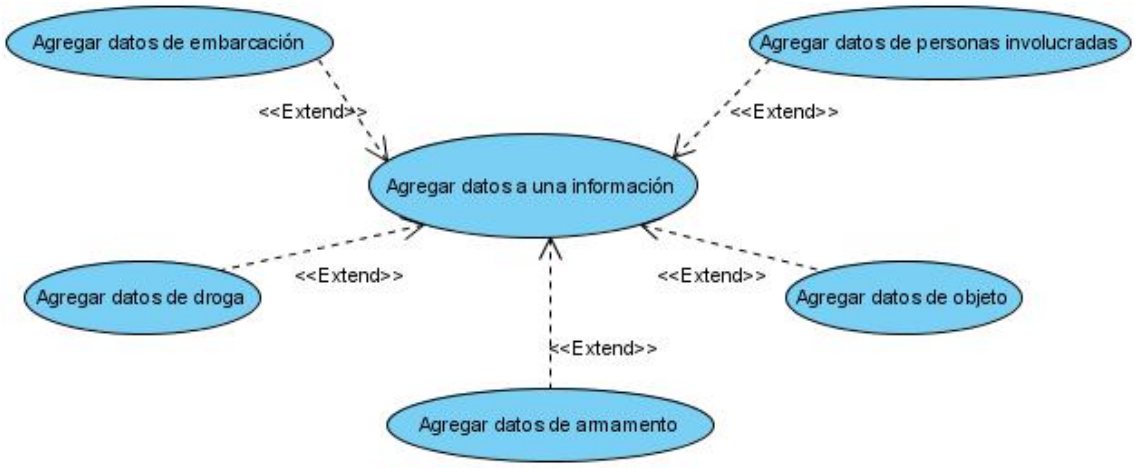

Figura 2.3: Diagrama del caso de uso Agregar datos a una información.

Los diagramas de caso de uso restantes se encuentran en el Anexo I.

2.3 Diagrama de actividad

El diagrama de actividad gráficamente es una colección de nodos y arcos que se utilizan para el modelado de los aspectos dinámicos de los sistemas y es fundamentalmente un diagrama que muestra el flujo de control entre actividades (Grady Booch, 2009).

El diagrama de actividades para el caso de uso de la figura 2.3, que se muestra a continuación, describe el proceso que se debe seguir para agregar elementos a una información ya almacenada. En el diagrama se aprecian los pasos a transitar para verificar los datos introducidos.

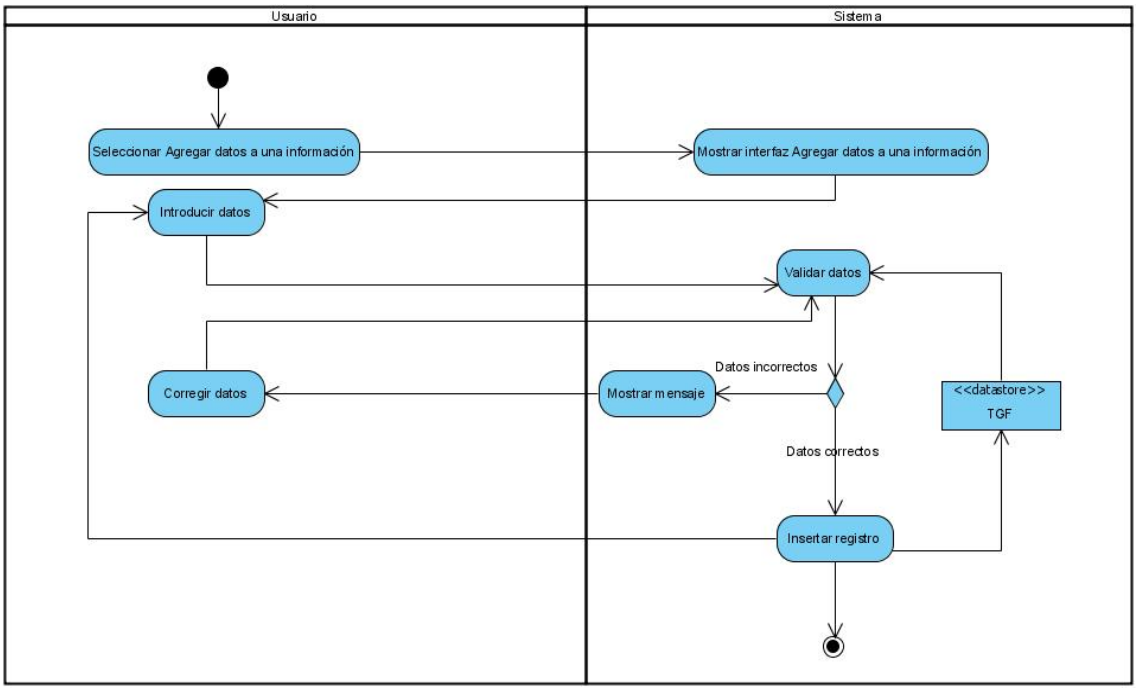

Figura 2.4: Diagrama de Actividades para el Caso de Uso "Agregar datos a una información" (extensión del Caso de Uso "Gestionar información").

Para fortalecer la seguridad del sistema es de obligatoriedad autentificarse como se aprecia en la figura 2.5. Con la acción "Validar usuario y contraseña" se verifica si el usuario es correcto o no, mostrándose para este último caso un mensaje informativo.

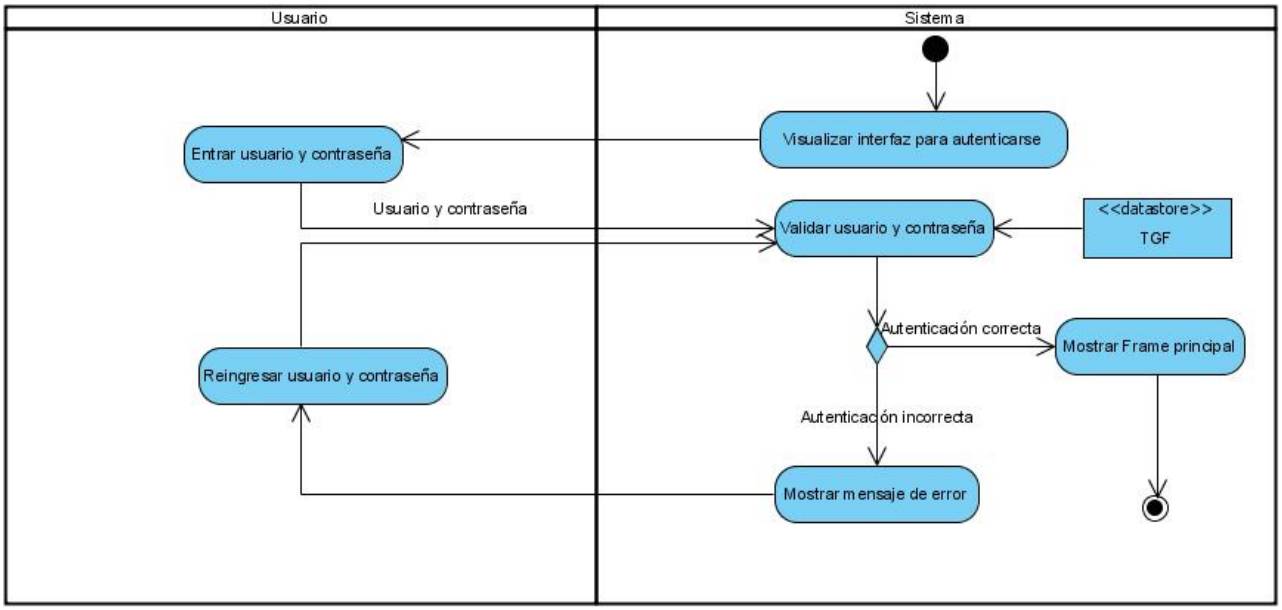

Figura 2.5: Diagrama de Actividades para el Caso de Uso "Autenticación".

En el Anexo II se encuentran los restantes diagramas de actividad.

2.4 Diagrama Entidad−Relación

Peter Chen (1976) fue el primero en proponer el modelado de bases de datos mediante la técnica gráfica, brindando un modelado conceptual de la problemática a ser representada. Basada en la percepción del mundo real tiene como elementos los objetos llamados entidades, sus atributos y las relaciones existentes entre cada uno de estos objetos.

Según (García, 2009) este primer modelo tenía desventajas tales como que "sólo era posible establecer restricciones de cardinalidad máxima en las interrelaciones de asociación, lo que limitaba la capacidad de expresividad del modelo" además de la incapacidad de representar solamente interrelaciones de asociación.

Desde unas décadas atrás el modelo original sufre cambios condicionado por la necesidad de representar bases de datos más exigentes para las aplicaciones actuales. A partir de esta necesidad surge el modelo Entidad Relación Extendido (ERE).

Tomando en consideración lo expuesto por (Silberschatz et al., 2011), el modelo ER encierra un alto grado de independencia de los datos, lo cual significa que el diseñador no tiene que preocuparse por la estructura física que tome la base de datos.

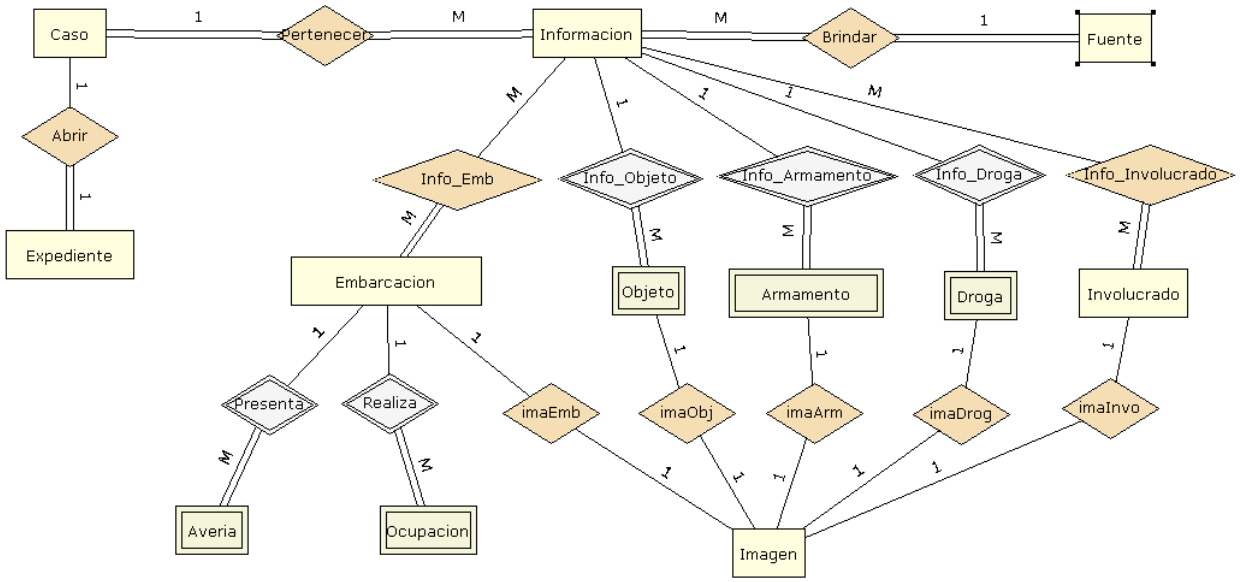

Figura 2.6: Diagrama Entidad-Relación para el caso de estudio TGF.

El diagrama que se observa en la figura 2.6, está compuesto por entidades fuertes como: Caso, Expediente, Información, Fuente, Embarcación, Involucrado e Imagen y débiles

como: Avería, Ocupación, Objeto, Armamento y Droga que no disponen de un atributo que las identifique y necesitan estar asociadas con una entidad fuerte para identificarse utilizando su(s) llave(s) primaria(s) (Silberschatz et al., 2011). En el caso de las tres últimas entidades débiles, estas se relacionan con Información, de la que obtienen la llave primaria *numero_id* para poder identificarse.

Otros aspectos que tiene en cuenta este modelo son:

1. Los casos pueden abrir un expediente o no abrir ninguno.
2. Un expediente pertenece solamente a un caso.
3. Un caso puede tener una o varias informaciones.
4. Una información pertenece a un solo caso.
5. Una fuente proporciona una o varias informaciones.
6. Una información es reportada por una sola fuente.
7. El atributo *fecha_cierre* de la entidad expediente tiene que coincidir con el atributo *fecha_cierre* de la entidad caso asociada a este (Expediente).
8. El valor almacenado en la columna *fecha_det* tiene que ser menor que el valor de la columna *fecha_info* ambos campos de la tabla información.
9. El atributo *fecha_apertura* de tipo **date** tiene que ser menor o igual que el atributo *fecha_cierre* del mismo tipo, ambos de la tabla expediente.
10. Si el valor del atributo *fecha_ocu* es diferente del valor nulo, este debe ser menor que el atributo *fecha_cierre*. Ambos de la tabla Caso.
11. Las drogas, armamentos, personas involucradas, objetos y embarcaciones pueden tener una imagen asociada o ninguna.
12. Un expediente se puede abrir cuando el caso asociado tenga al menos una información.

Para definir los tipos de fuentes se emplea el siguiente check constraint que obliga a no introducir otros tipos que no coincidan con estos:

```
alter table informacion
add constraint CK_informacion_tipo_fuente
check (tipo_fuente='Servicio Operativo Terrestre' or tipo_fuente='Servicio Operativo Naval' or tipo_fuente='ONIP' or tipo_fuente='FAR' or tipo_fuente='Fuente humana');
```

Para insertar nuevas informaciones se tiene la función ***insertar_nueva_informacion,*** la cual se encarga de registrar las informaciones de un caso y autogenerar la llave primaria (numero_id) con la utilización de la secuencia **serial**. A continuación se puede apreciar el código SQL.

```
CREATE OR REPLACE FUNCTION insertar_nueva_informacion(area character varying, numero integer, fecha_det date, hora_det time without time zone, hora_info time without time zone, fecha_info date, lugar_det character varying, descripcion character varying, grado_relevancia character varying, t_fuente character varying, fuente__ci character varying)
 RETURNS void AS
$BODY$
    INSERT INTO informacion(numero_id,fecha_det, hora_det, hora_info,fecha_info,lugar_det,descripcion,grado_relevancia,tipo_fuente,fuente__ci)
    VALUES (nextval('serial'), fecha_det, hora_det, hora_info,fecha_info,lugar_det,descripcion,grado_relevancia,t_fuente,fuente__ci);
    select insertar_datos_en_pertenecer(cast(currval('serial')as integer), area , numero);
$BODY$
```

La función ***listar_numero_de_caso_para_crear_expediente*** obtiene solamente los casos que tienen expedientes asociados y no se han cerrado. Para comprender mejor el objetivo de dicha función se plantea a continuación su código SQL.

```
CREATE OR REPLACE FUNCTION listar_numero_de_caso_para_crear_expediente(ar character varying)
 RETURNS SETOF integer AS
$BODY$
select a.numero
from(
    select area, numero
    from caso
    where area=ar and fecha_cierre is null
    except select caso__area, caso__numero
```

from expediente

) as a

$BODY$

2.5 Patrón arquitectónico Modelo−Vista−Controlador (MVC)

El MVC es un patrón de arquitectura de las aplicaciones software que nos permite mantener individualizadas, las "responsabilidades" dentro de un sistema, permitiéndonos diferenciar y aislar el diseño del sistema (modelo), de la interfaz gráfica del usuario (GUI) y su correspondiente lógica de negocios (vistas), utilizando como "conector intermediario" un objeto controlador, incrementando la reutilización y flexibilidad del código.

Por una parte se definen componentes para la representación de la información, y por otra para la interacción del usuario (Reenskaug and Coplien, 2009). Fue diseñado para reducir el esfuerzo de programación necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos (Yenisleidy F. Romero, 2012).

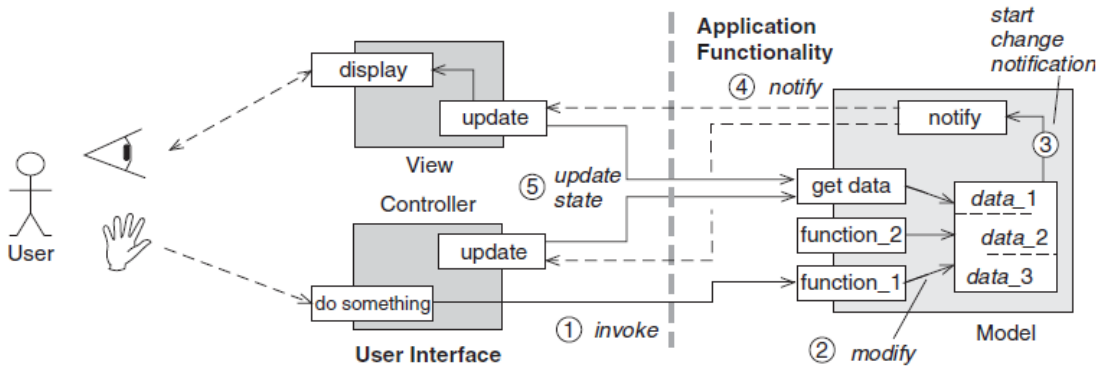

Figura 2.7: Modelo-Vista-Controlador tomado de (Buschmann et al., 2007).

Las autoras sostienen que "al incorporar el modelo de arquitectura MVC a un diseño, las piezas de un programa se pueden construir por separado y luego unirlas en tiempo de ejecución". El diseño modular permite a los diseñadores y a los desarrolladores trabajar conjuntamente y hacer cambios en una parte de la aplicación sin que las demás se vean afectadas. Desde el punto de vista del usuario, todo comienza en el controlador.

En general este modelo de arquitectura presenta varias ventajas (Catalani, 2007):

- Separación clara entre los componentes de un programa; lo cual permite su implementación por separado.
- Interfaz de Programación de Aplicaciones API (Application Programming Interface) muy bien definida; cualquiera que use el API, podrá reemplazar el Modelo, la Vista o el Controlador, sin aparente dificultad.
- Conexión entre el Modelo y sus Vistas dinámicas; se produce en tiempo de ejecución.

2.5.1 Diagrama de clases

Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema. Para (Grady Booch, 2009) un diagrama de clases es gráficamente una colección de nodos y arcos, que muestra un conjunto de clases, interfaces, colaboraciones y sus relaciones que pueden ser de dependencia, generalización o asociación. Además pueden contener paquetes o subsistemas para agrupar elementos de un modelo en partes más grandes. En la figura 2.8 se muestra el diagrama de clases propuesto.

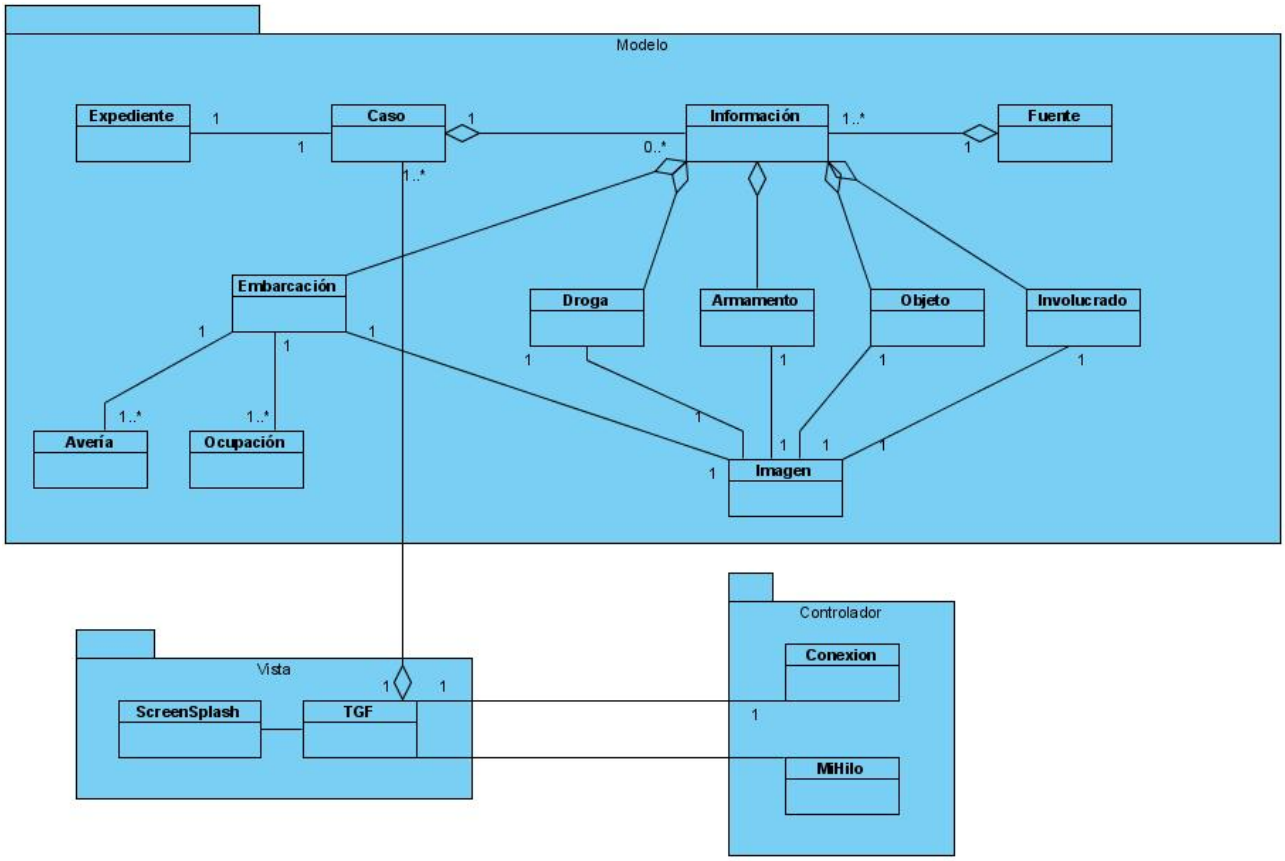

Figura 2.8: Diagrama de clases propuesto.

2.6 Diagrama de componentes

Este tipo de diagrama es uno de los que aparecen cuando se modelan aspectos físicos de los sistemas orientados a objetos. Muestra la organización y dependencias entre un conjunto de componentes y se utiliza para comunicar un aspecto de la vista de implementación estática de un sistema.

Los componentes se utilizan para modelar elementos físicos tales como ejecutables, bibliotecas, tablas, archivos y documentos.

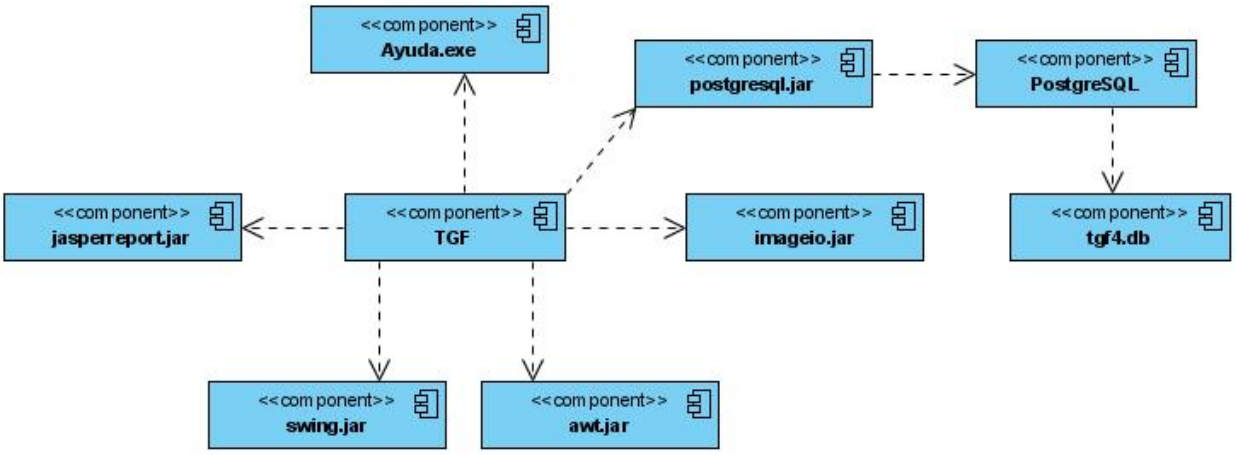

Figura 2.9: Diagrama de componentes.

2.7 Seguridad de los datos

En la presente investigación se plantea el siguiente requisito funcional de seguridad de los datos:

- Hacer una copia de resguardo de la base de datos cada vez que se cierre el sistema.

El cual tiene como objetivo mantener una copia de los registros de la base de datos en un directorio diferente en el que se encuentra originalmente. Para ello cada vez que el usuario cierra el sistema este último autogenera una copia de seguridad en una dirección especificada con anterioridad por el actor "Administrador del sistema".

```
8997    private void formWindowClosing(java.awt.event.WindowEvent evt) {
8998        // TODO add your handling code here:
            String[] s=null;
9000        s=con.leer_fichero("fichero.txt");
9001        if(s[0]!=null){
            MiHilo h=new MiHilo();
9003        h.MiHilo(s[0],s[1],s[2],s[3],s[4]);
9004        h.resguardar();}
9005    }
```

Figura 2.10: Evento ***formWindowClosing*** para la ventana principal.

Como muestra la figura 2.10 utilizando el evento ***formWindowClosing*** que se implementa en la clase **Frame**, dentro del paquete **Vista,** para cerrar la aplicación principal se lee el fichero que contiene los datos referentes al *hostname*, puerto, nombre de la base de datos, dirección donde se desea almacenar la copia y camino donde se encuentra la aplicación de PostgreSQL **pg_dump.exe**, encargada de hacer el resguardo.

Para hacer esto se llama al método **leer_fichero (dirección)** de la clase **Conexión** perteneciente al paquete **Controlador**.

Posteriormente se crea una instancia de la clase "MiHilo". Dicha clase se construye con los datos del fichero que se leyeron anteriormente y se encarga de ejecutar el proceso **pg_dump.** Utilizando el objeto creado de la clase "MiHilo" se llama al método **resguardar()** iniciando el hilo.

El hilo principal de la aplicación espera a que el hilo creado termine preguntando cada 500 milisegundos si este está vivo todavía como muestra la figura 2.11.

```
53      public void resguardar(){
54       Thread t = new Thread(new MiHilo.Hilo());
55          long startTime = System.currentTimeMillis();
56          t.start();
57          Frame f=new Frame(0);
58          f.iniciarVentanaResguardo();
59          while (t.isAlive()) {
60              try {
61
62                  t.join(500);
63              } catch (InterruptedException ex) {
64                  Logger.getLogger(MiHilo.class.getName()).log(Level.SEVERE, null, ex);
65              }
66
67          }
68          long finishTime = System.currentTimeMillis();
69          f.terminarVentanaResguardo();
70          System.out.println(finishTime-startTime);
71          System.out.println("Soy el proceso padre");
72      }
73
74  }
```

Figura 2.11: Método **resguardar()** de la clase MiHilo.

Para ello se usa la llamada al método **join()** con el que permite que un hilo pueda esperar por la terminación de otro (Oracle, 2011) véase la figura 2.12.

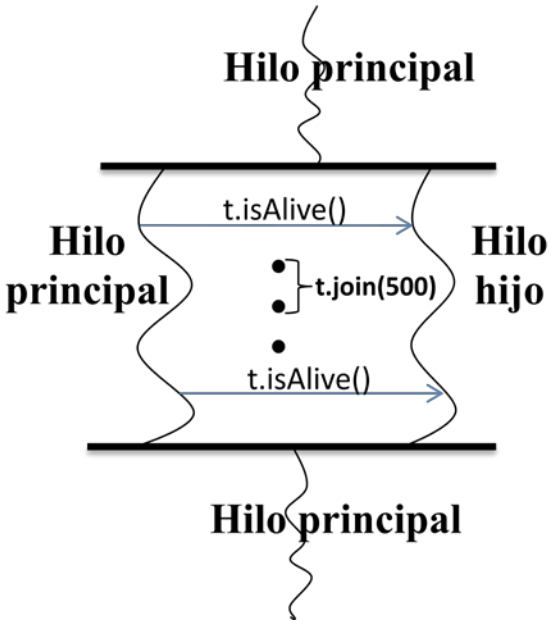

Figura 2.12: Concurrencia entre dos hilos.

2.7.1 Comando pg_dump

Según (PostgreSQL, 2012) **pg_dump** es una aplicación cliente que genera un archivo de texto con comandos SQL representando un *snapshot* de la base de datos en el momento de empezar su ejecución. Es una funcionalidad muy importante que brinda este gestor para hacer copias de las bases de datos y reconstruirlas posteriormente.

La sintaxis utilizada es:

```
pg_dump -h host –p puerto –U user --no-password –-format custom –blobs –-encoding UTF8 –verbose --file repositorio bd
```

Donde

- **-h host** especifica el hostname de la máquina sobre la cual el servidor Postgres se ejecuta.
- **-p puerto** define el puerto TCP por el que el servidor está esperando conexiones.
- **-U user** hace referencia al usuario para conectarse. Es importante destacar que esta aplicación se debe ejecutar como superusuario de PostgreSQL.
- **--no-password** evita que la aplicación solicite una contraseña para conectarse.

- **--format** establece en qué formato se crea el *backup.* Se selecciona **custom** pues Postgres crea las bases de datos en este formato por defecto y es junto a **directory** el formato de salida más flexible de entre todos.
- **-blobs** permite incluir en la copia objetos muy grandes como archivos.
- **--encoding UTF8** codifica en un formato de codificación de caracteres Unicode la salida, en este caso se utiliza UTF8.
- **--file direccion** define la dirección del archivo para la salida.
- **bd** representa el nombre de la base de datos que se desea resguardar.

2.8 Conclusiones parciales

En este capítulo se realizó un análisis de los requerimientos funcionales y no funcionales del sistema y un estudio de los casos de uso que se encargan de delimitar el sistema definiendo las funciones que debe cumplir para un usuario específico (los actores). Se obtuvo un diseño mediante el modelado de las características principales del sistema utilizando la notación UML para una mejor comprensión de la estructura. Se diseñó también la base de datos y la disposición de las clases para la etapa de programación utilizando el patrón arquitectónico de diseño MVC. Además se planteó cómo solucionar las autenticaciones de los usuarios.

Capítulo 3 Implementación y validación del sistema

En el presente capítulo se exponen aspectos relacionados con la implementación del sistema y se explican las características del software facilitando al usuario una mejor comprensión del mismo. Se realizan pruebas unitarias para la validación del código, para el diseño y evaluación de las mismas se utiliza la herramienta JUnit de Netbeans. Además, se realizan pruebas de integración entre la aplicación principal y la base de datos y se propone un diseño dimensional para la futura implementación de un mercado de datos.

3.1 Pruebas unitarias

Las pruebas unitarias, o *unit testing* en inglés, es el proceso de probar individualmente subprogramas, subrutinas, clases, métodos o procedimientos en un software. Específicamente, en lugar de probar inicialmente el programa en su conjunto, la prueba es primero centrada en los bloques de construcción más pequeños del programa (Myers et al., 2012).

Según (Gómez et al., 2013) estas pruebas están centradas en el procesamiento lógico interno y en las estructuras de datos dentro de un componente. Es aplicable a componentes representados en el modelo de implementación para verificar que los flujos de control y de datos están cubiertos, y que ellos funcionen como se espera. Dichas pruebas requieren del conocimiento de la estructura interna del programa y son derivadas a partir de las especificaciones internas de diseño o del código programado.

3.1.1 Casos de pruebas

El diseño de casos de prueba de una unidad comienza una vez que se ha desarrollado, revisado y verificado en su sintaxis el código a nivel fuente. Hay diferentes técnicas para el diseño de pruebas como son el camino básico y las pruebas de decisión/condición.

Según lo planteado en (Pressman, 2010) el método del camino básico permite al diseñador de casos de pruebas obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución.

El autor define cinco pasos para el diseño de pruebas utilizando el principio del camino básico:

1. Obtener el grafo de flujo a partir del diseño o del código programado.
2. Obtener la complejidad ciclomática del grafo de flujo.
3. Definir los caminos básicos independientes.
4. Determinar los casos de prueba.
5. Ejecutar cada uno de los casos de prueba y verificar que los resultados son los esperados.

La complejidad ciclomática se usa para determinar el número de caminos a buscar y se caracteriza por ser una medida que da la idea de la complejidad lógica de un programa. En (Gómez et al., 2013) se plantean tres ecuaciones para calcularla:

1. V(G)= A-N+2, donde A es la cantidad de aristas del grafo y N la cantidad de nodos.
2. V(G)=P+1, donde P es la cantidad de nodos predicados[7].
3. V(G)=R. donde R representa el número de regiones[8] del grafo.

3.1.2 Diseño e implementación de los casos de prueba utilizando el método del camino básico.

Una vez calculada la complejidad ciclomática, se determina un conjunto básico de caminos linealmente independientes y posteriormente se fuerza el paso por cada uno de ellos a partir de los valores de entrada. A continuación se diseñan los casos de prueba pertenecientes a los métodos ***crear_caso_Frame***, ***agregar_Objeto_Frame*** y ***leer_fichero*** respectivamente.

[7] Nodo del que emergen dos o más aristas.

[8] Las regiones son las áreas delimitadas por las aristas y nodos. También se incluye el área exterior del grafo, contando como una región más.

- Método ***crear_caso_Frame***

El diseño e implementación de la primera prueba se hace utilizando el método **crear_caso_Frame** de la clase **Frame**; el cual permite la inserción de un caso en la base de datos. Como parámetros este método recibe cinco valores enteros que representan los índices de las selecciones de los jComboBox y la longitud del campo **Oficial operativo**. A partir del código se obtiene el grafo de flujo como muestran las figuras 3.1 y 3.2.

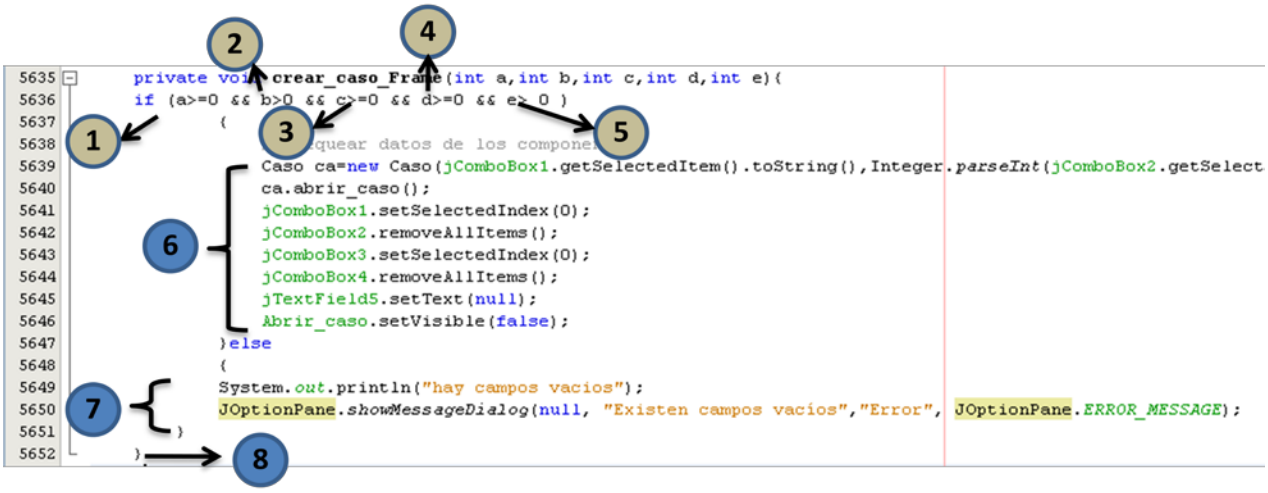

Figura 3.1: Método **crear_caso_Frame** con los nodos representados.

Los nodos 1, 2, 3, 4 y 5 representan los nodos predicados.

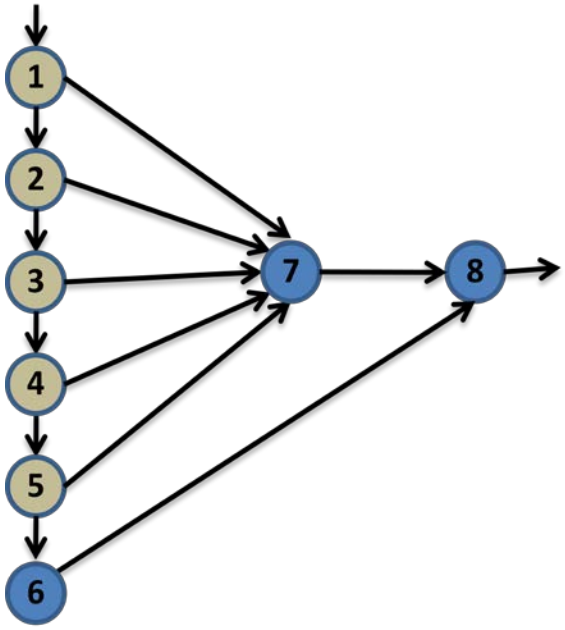

Figura 3.2: Grafo de flujo obtenido.

La complejidad ciclomática para el grafo es igual a seis por lo que es posible esa misma cantidad de caminos o casos de pruebas que se definen a continuación.

Camino1:1-7-8

Camino2:1-2-7-8

Camino3:1-2-3-7-8

Camino4:1-2-3-4-7-8

Camino5:1-2-3-4-5-7-8

Camino6:1-2-3-4-5-6-8

El método que se muestra a continuación prueba los valores de entrada para forzar el camino 1 del conjunto que se obtuvo. En este ejemplo no se selecciona ningún área del jComboBox1 por lo que la entrada es -1.

```
19      /**
20       * Test of crear_caso_Frame method, of class Frame.
21       */
22      @Test
23      public void caso_de_prueba1() {
24          System.out.println("crear_caso");
25          int a = -1;//si no hay item seleccionado
26          int b = 55;//posibles valores del 0 en adelante
27          int c = 55;//posibles valores del 0 al 6
28          int d = 55;//posibles valores del 0 en adelante
29          int e = 55;//length >> posibles valores del 1 en adelante
30          Frame instance = new Frame(a,b);
31          boolean expResult = false;//resultado esperado
32          boolean result = instance.crear_caso(a,b,c,d,e);
33          assertEquals(expResult, result);
34      }
```

Figura 3.3: Diseño de la prueba para el camino 1.

Se espera que el resultado sea *false* para los valores dados y al ejecutar la prueba el valor esperado coincide con el resultado, por lo que la prueba es exitosa.

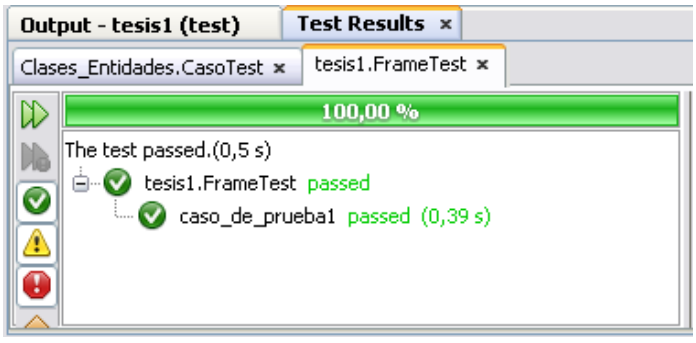

Figura 3.4: Resultado satisfactorio de la prueba *caso_de_prueba1*.

La ejecución de los restantes casos de prueba se encuentra en el Anexo III.

- Método ***agregar_Objeto_Frame***

La segunda prueba está dirigida al método **agregar_Objeto_Frame** de la clase Frame en el paquete **Vista**, que introduce un objeto en una información. Este método recibe como entrada todos los datos recogidos en la interfaz gráfica relacionados con el área y número del caso a que pertenece, además del número de la información y los datos del objeto a almacenar. La figura 3.5 muestra el método reducido y con los nodos del grafo identificados.

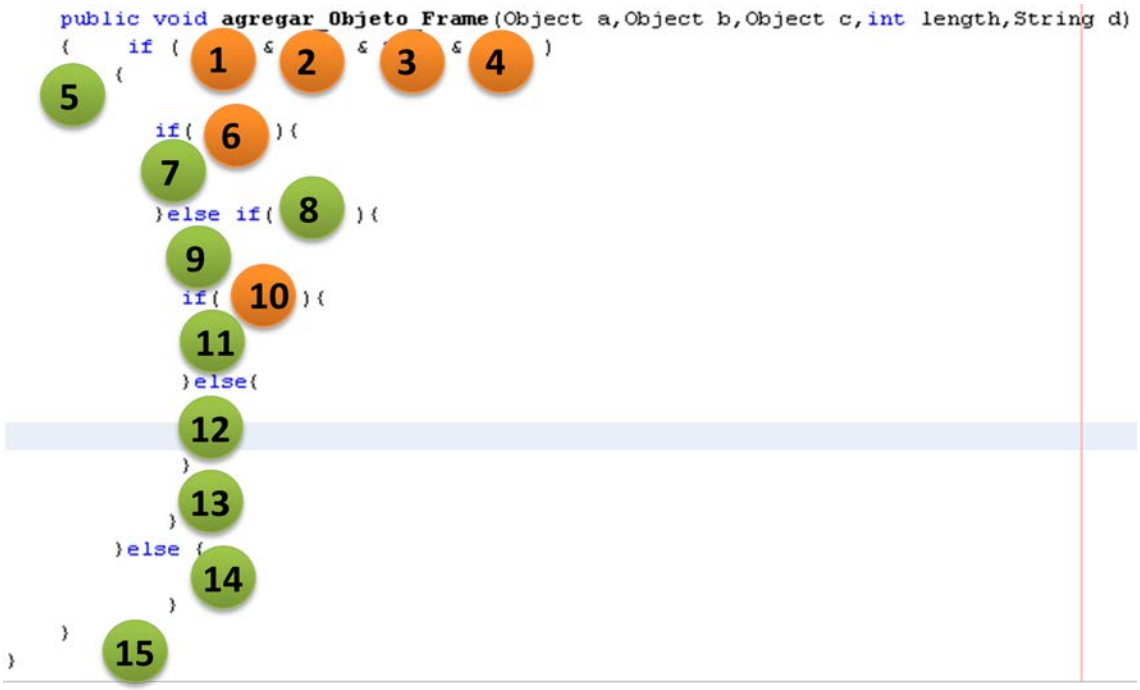

Figura 3.5: Nodos identificados en el código fuente.

El grafo de flujo es el que se muestra a continuación en la figura 3.6:

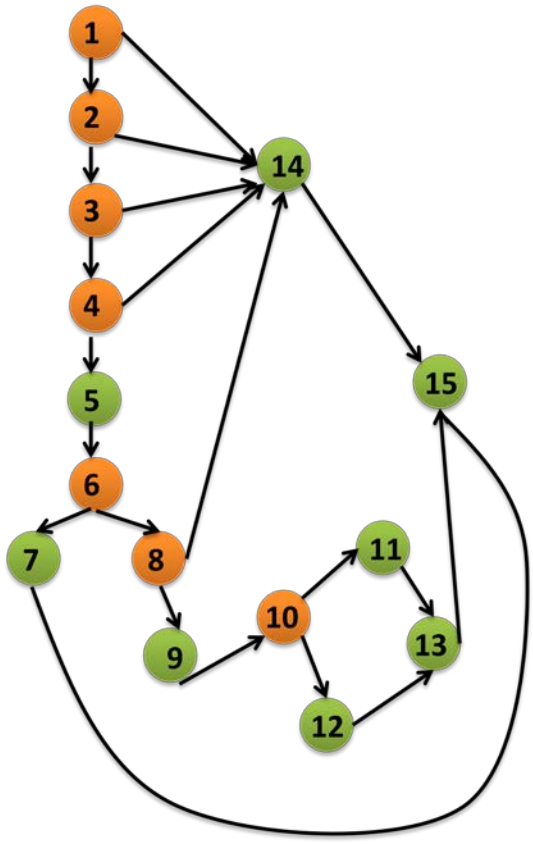

Figura 3.6: Grafo de flujo para el método ***agregar_Objeto_Frame***.

A partir del cálculo de la complejidad ciclomática se detectaron ocho caminos o casos de pruebas que se detallan en la siguiente tabla.

Casos de prueba	Descripción	Valores de entrada	Resultado esperado

1. 1-14-15	El objeto del primer parámetro deberá ser null. Los valores de los restantes parámetros no tienen importancia.	Object a = null;
		Object b = BBB";
		Object c = "CCC";
		int length = 1;
		boolean retorno= true;
		int q=0;
			String expResult = "7";

2. 1-2-14-15	El objeto del segundo parámetro deberá ser null. Los	Object a = "AAA"; Object b = null; Object c = "CCC";	String expResult = "7";

	valores de los restantes parámetros no tienen importancia.	int length = 1;
		boolean retorno= true;
		int q=0;	

3. 1-2-3-14-15	El objeto del tercer parámetro deberá ser null. Los valores de los restantes parámetros no tienen importancia.	Object a = "AAA";
		Object b = "BBB";
		Object c = null;
		int length = 1;
		boolean retorno= true;
		int q=0;
			String expResult = "7";

4. 1-2-3-4-14-15	El valor de la variable length deberá ser menor que 1.	Object a = "AAA";
		Object b = "BBB";
		Object c = "CCC";
		int length = 0;
		boolean retorno= true;
		int q=0;
			String expResult = "7";

5. 1-2-3-4-5-6-7-15	El valor de la variable q debe ser igual a 1.	Object a = "AAA";
		Object b = "BBB";
		Object c = "CCC";
		int length = 1;
		boolean retorno= true;
		int q=1;
			String expResult = "126";

6. 1-2-3-4-5-6-8-14-15	El valor de la variable q debe ser igual a 0.	Object a = "AAA";
		Object b = "BBB";
		Object c = "CCC";
		int length = 1;
		boolean retorno= true;
		int q=0;
			String expResult = "1346";

7. 1-2-3-4-5-6-8-9-10-11-13-15	El valor de la variable q debe ser 0 y la variable retorno debe ser true, los objetos a, b, c diferentes de null y la variable length >= 1.	Object a = "AAA";
		Object b = "BBB";
		Object c = "CCC";
		int length = 1;
		boolean retorno= false;
		int q=0;
			String expResult = "1356";

8. 1-2-3-4-5-6-8-9-10-12-13-15	Las variables deberán tener los mismos valores que se utilizaron en el camino 7 con la excepción de la variable retorno que debe ser false.	Object a = "AAA";
		Object b = "BBB";
		Object c = "CCC";
		int length = 1;
		boolean retorno= false;
		int q=3;
			String expResult = "16";

Tabla 3.1: Casos de prueba para el método ***agregar_Objeto_Frame***.

En la figura 3.2 se muestra la prueba ejecutada para el caso de prueba 7 que se observa en la tabla 3.1. En el Anexo III se encuentran las pruebas realizadas en JUnit a los demás casos de prueba con sus respectivas ejecuciones.

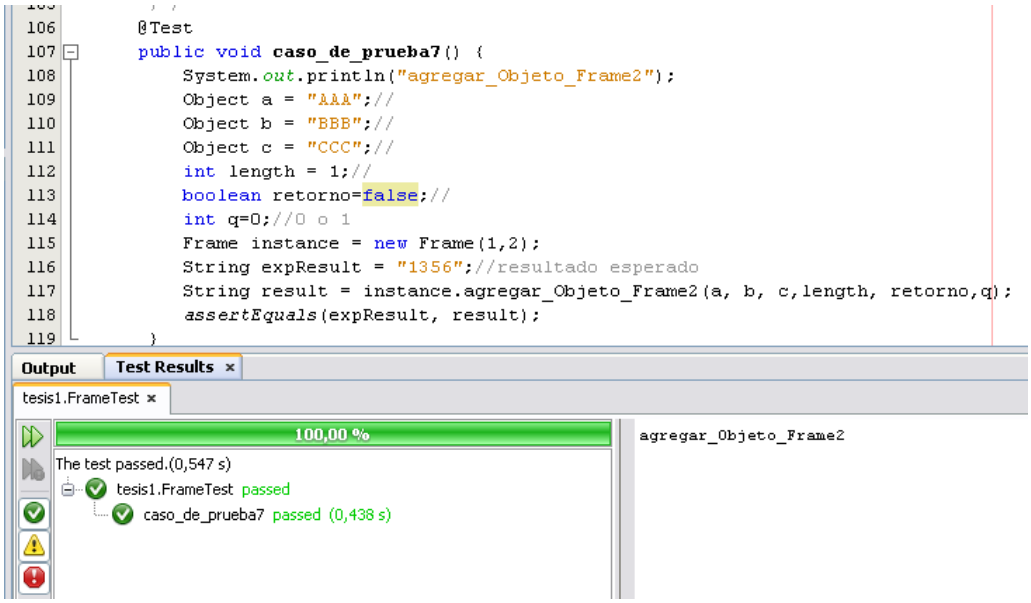

Figura 3.7: Implementación del caso de prueba 7.

- Método ***leer_fichero***

Este método pertenece a la clase **Conexión** que se encuentra en el paquete **Controlador**. Se encarga de obtener datos de un fichero de texto dado su dirección. El método se compone de una condición a evaluar en la sentencia *where* y dos sentencias que pueden elevar excepciones. El valor de retorno es un arreglo que contiene las líneas leídas del fichero. En la figura 3.8 se muestra el método con los nodos identificados.

```
141     public String[] leer_fichero(String direccion_fichero){
142      String linea;
143       String [] d= new String[5];
          File archivo = null;
145       // archivo = new File(archivo.getAbsolutePath() + ".txt");
146       archivo = new File(direccion_fichero);
147       try {
148           BufferedReader fr = new BufferedReader(new FileReader(archivo));
149           int i=0;
150           while ((linea = fr.readLine()) != null) {
151               System.out.println(linea);
152               d[i]=linea;
153               i++;
154               }
155           } catch (FileNotFoundException e) {
156           System.out.println("Fichero no encontrado");
157           JOptionPane.showMessageDialog(null,"Existe un error en la lectura
158       } catch (IOException e) {
159           System.out.println("No se pudo crear el fichero");
160       }
161       return d;
162   }
```

1 2 3 4 5 6

Figura 3.8: Método ***leer_fichero*** con los nodos identificados.

Para definir los casos de prueba se obtuvo el grafo de la figura 3.9 representando en color azul los nodos predicados.

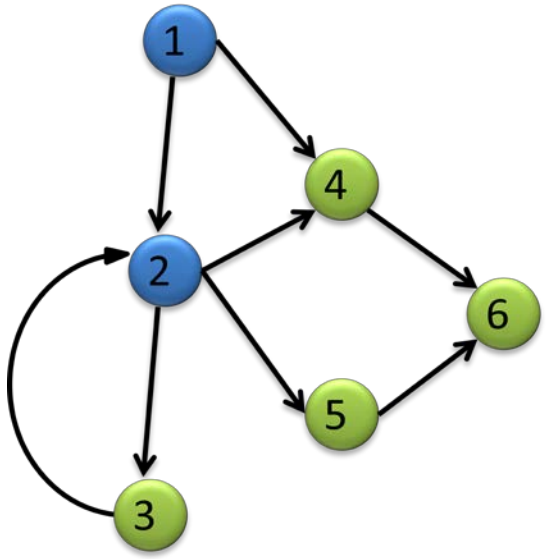

Figura 3.9: Grafo de flujo para el método ***leer_fichero***.

En la Tabla 3.2 se puede apreciar las descripciones de los casos de prueba. Los puntos suspensivos (…) en el cuarto caso de prueba indican que cualquier camino del resto de la estructura de control es aceptable. Las pruebas programadas y ejecutadas utilizando la herramienta JUnit se encuentran en el Anexo III.

Casos de prueba	Descripción	Valores de entrada	Resultado esperado
1. 1-4-6	El valor de entrada deberá ser un fichero que no exista para forzar el paso por el camino elevando la excepción FileNotFoundException.	direccion_fichero=”fichero1.txt”;	String[] expResult = null;
2. 1-2-4-6	Deberá haber un error en la E/S para elevar la excepción IOException.		
3. 1-2-5-6	El fichero de entrada debe estar vacío para que el algoritmo no pueda entrar al ciclo.	direccion_fichero=”fichero3.txt”;	String[] expResult = new String[5];
4. 1-2-3-2-…	El fichero de entrada tendrá una línea con la cadena “aaa” para que pueda entrar al ciclo.	direccion_fichero=”fichero4.txt”;	String[] expResult = "aaa",null,null,null,null;

Tabla 3.2: Casos de prueba para el método ***leer_fichero***.

3.2 Pruebas de integración

Las pruebas de integración se emplean para comprobar que las unidades de prueba, que han superado sus pruebas de unidad, funcionan correctamente cuando se integran. Durante la integración, las técnicas que más se utilizan son las de caja negra, aunque se pueden llevar a cabo algunas pruebas de caja blanca para asegurar que se cubren los principales flujos de comunicación entre las unidades (Pressman, 2010).

En la figura 3.10 se muestra el código fuente de una prueba que comprueba si un caso es introducido correctamente en la base de datos. Para ello se verifica que los datos de entrada coinciden con los datos de la fila insertada.

```
44      @Test
45      public void testAbrir_caso() throws SQLException   {
46          System.out.println("abrir_caso");
47          //definimos los parametros a pasar
48          String area="Yaguanabo";
            int numero=41;
50          String categoria="Actividad Enemiga";
51          String subcategoria="Infiltración";
52          String oficial="Felix Julio";
53          try {
54              c.abrir_conexion();
55          } catch (Exception ex) {
56              Logger.getLogger(CasoTest.class.getName()).log(Level.SEVERE, null, ex);
57          }
58          Caso instance = new Caso(area, numero, categoria, subcategoria, oficial,c);
59          try{
60          instance.abrir_caso();
61          }catch (SQLException e)
62          {
63              fail();
64          }
65          String[] arreglo=instance.getDatos(area, numero);
66          assertTrue(categoria.equals(arreglo[0]) && subcategoria.equals(arreglo[1]) &&  oficial.equals(arreglo[2]));
67        }
```

Figura 3.10: Prueba de integración para comprobar la inserción de un caso.

El código que se presenta en la figura 3.11 comprueba si se actualizan los datos en la tabla Caso al cerrar un caso. En este ejemplo se verifican los datos que se introdujeron con los pasados como parámetro al método que se está testeando.

```
@Test
 public void testCerrar_caso() {
     System.out.println("cerrar_caso");
     String fecha_cierre = "2017-12-23";
     String lug_ocu = "Cayo Carenas";
     String hora_ocu = "23:03";
     String fecha_ocu = "2017-12-20";
     String descripcion = "descripcion";
     String modus_operandi = "modus_operandi";
     String otros_datos = "otros_datos";
     String tiempo_perm = "tiempo_perm";
     String defic = "defic";
     String medidas_tomadas = "medidas tomadas";
     String frustrado = "No";
     String area = "Yaguanabo";
     int numero = 60;
     try {
         c.abrir_conexion();
     } catch (Exception ex) {
         Logger.getLogger(CasoTest.class.getName()).log(Level.SEVERE, null, ex);
     }
     Caso instance = new Caso(area, numero, c);
     try{
        instance.cerrar_caso(fecha_cierre, lug_ocu, hora_ocu, fecha_ocu, descripcion, modus_operandi, otros_datos, tiempo
     }catch (SQLException e)
     {
         Logger.getLogger(Caso.class.getName()).log(Level.SEVERE, null, e);
          fail();
     } System.out.println("aaaaaaaaaaaaaaaaaa");
     String[] arreglo=instance.getTodosDatos(area, numero);
     for(int i=0;i<11;i++)
     {
         System.out.println(arreglo[i]);
     }
     assertTrue(lug_ocu.equals(arreglo[0]) && fecha_ocu.equals(arreglo[1]) &&  descripcion.equals(arreglo[2]) &&  modus_
     }
```

Figura 3.11: Prueba de integración para comprobar la actualización de un caso.

3.3 Descripción del sistema

El sistema cuenta con los dos actores siguientes: “administrador del sistema” y “usuario”. Cada uno de los cuales tienen diferentes funciones y privilegios condicionados por el uso de contraseñas.

3.3.1 Pasos iniciales

En la fase de despliegue del sistema, el actor “administrador del sistema” es el encargado de configurar los permisos de acceso a la base de datos y las copias de seguridad de los datos almacenados.

Los usuarios no deben tener todos los permisos sobre la base de datos por lo que se restringen sus accesos y privilegios. Para ello el actor “administrador del sistema” tiene que otorgar los permisos a los nuevos usuarios. Para cumplir con esto se crean los nuevos usuarios y se les otorga una contraseña para acceder a los datos, como se muestra en la figura 3.12.

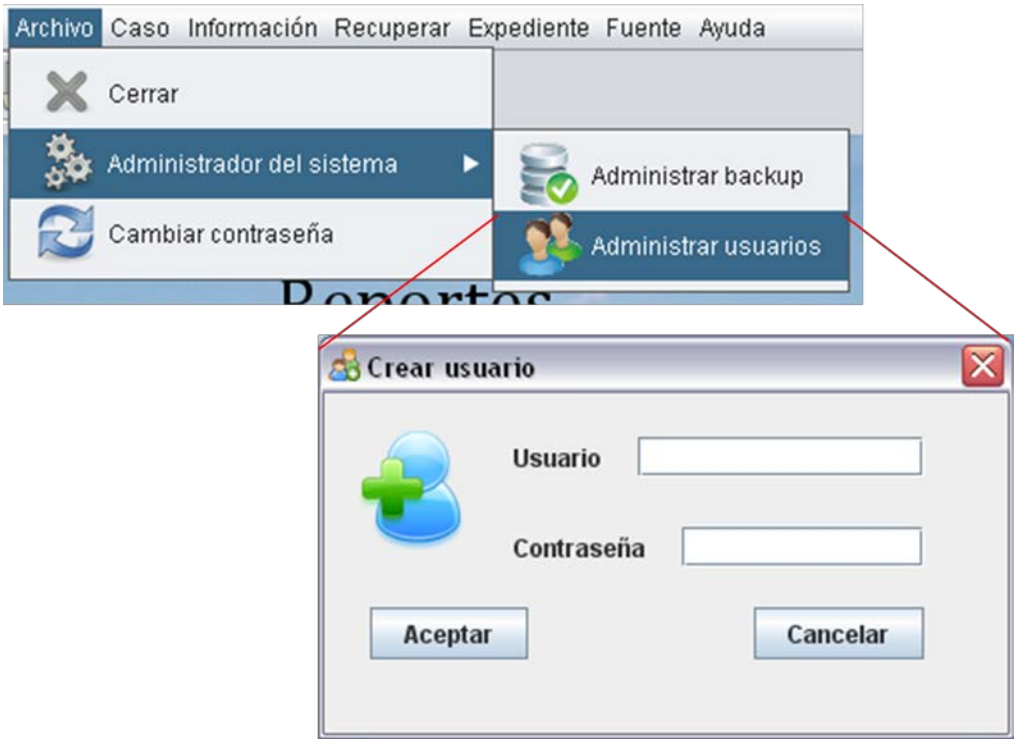

Figura 3.12: Creación de usuarios del sistema.

Se crea un usuario con permisos de acceder a los datos de las tablas de la base de datos, con el propósito de insertar, eliminar, actualizar o seleccionar, además de otorgarle permisos de ejecutar funciones y usar secuencias.

Para los datos, mantener una estricta seguridad sobre ellos es fundamental, aún más cuando puede haber fallas en el Sistema Operativo o un borrado indebido de los datos. Cuando existen estos problemas los datos almacenados en la partición donde se encuentra la instalación se pierden por lo que se hace necesario almacenar una copia de resguardo de la base de datos en otra unidad. Para resolver esto, el "administrador del sistema" configura los principales datos para realizar la copia como muestra la figura 3.13.

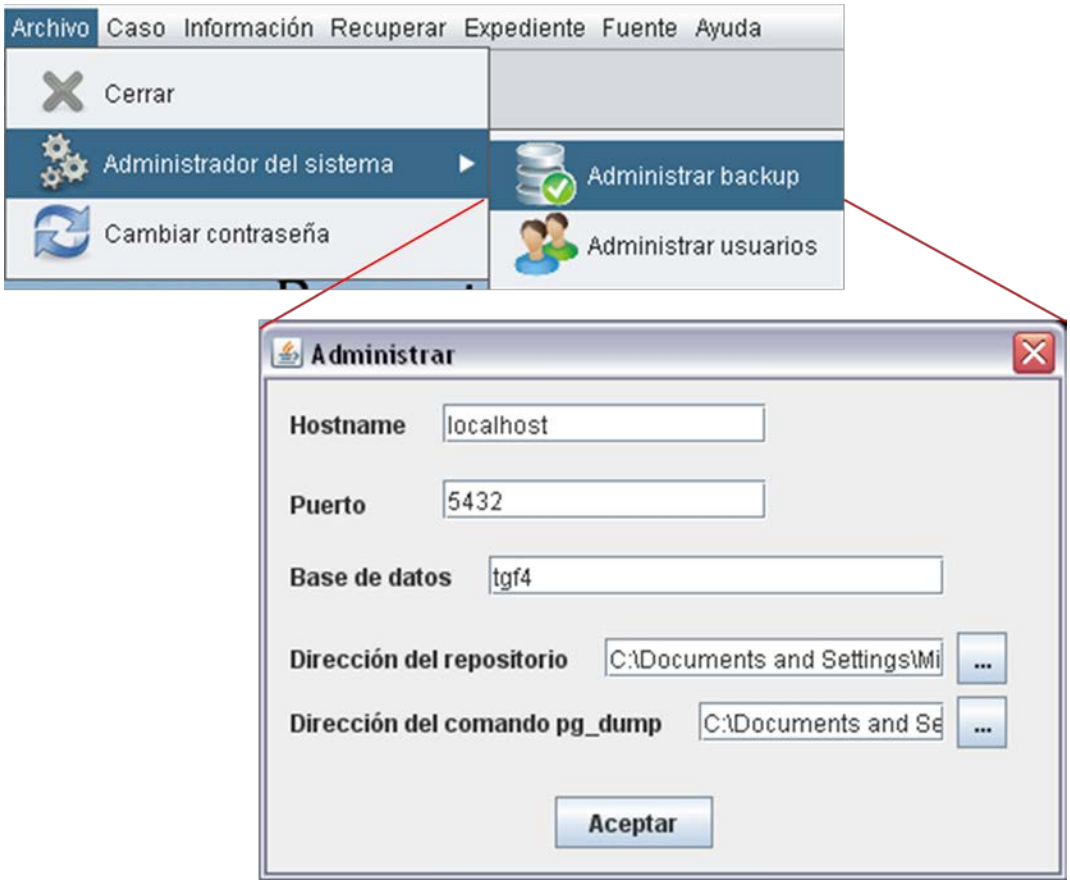

Figura 3.13: Administración de copias de la base de datos.

Del resto se encarga el sistema cada vez que se cierra, creando una copia nueva o actualizando una ya existente.

3.3.2 Accediendo a las funcionalidades del sistema

Una vez autenticado el usuario, este puede hacer uso de las funciones que el sistema ofrece y que se pueden acceder a través de la ventana principal como se observa en la Figura 3.14.

Figura 3.14: Ventana principal de la aplicación con las principales funcionalidades.

El usuario puede acceder a la gestión de los casos, expedientes e informaciones, recuperar datos almacenados, acopiar información en forma de reportes, además de administrar su contraseña. Aunque se le restringe la opción de configurar las copias de la base de datos y crear usuarios.

3.3.3 Gestionando los casos

El caso de uso más importante es el de gestionar los casos porque a partir de este se utilizan los demás. En las siguientes figuras se visualizan las interfaces gráficas para dar cumplimiento a este caso de uso.

En la figura 3.15 se muestra el diseño de la interfaz gráfica para introducir un caso en la base de datos.

Figura 3.15: Interfaz gráfica para la inserción de un caso.

La figura 3.16 muestra el diseño de la ventana para cerrar un caso. En ella es necesario seleccionar el área y número del caso, además de la fecha de cierre.

Figura 3.16: Interfaz gráfica para cerrar un caso.

3.3.4 Gestionando las informaciones

Las informaciones proporcionadas por las fuentes contienen un gran número de datos. Inicialmente cuando se crea una información en el sistema, este debe permitir la introducción de esta gran cantidad de datos. En la figura 3.17 se contempla el diseño que permite lo anteriormente planteado.

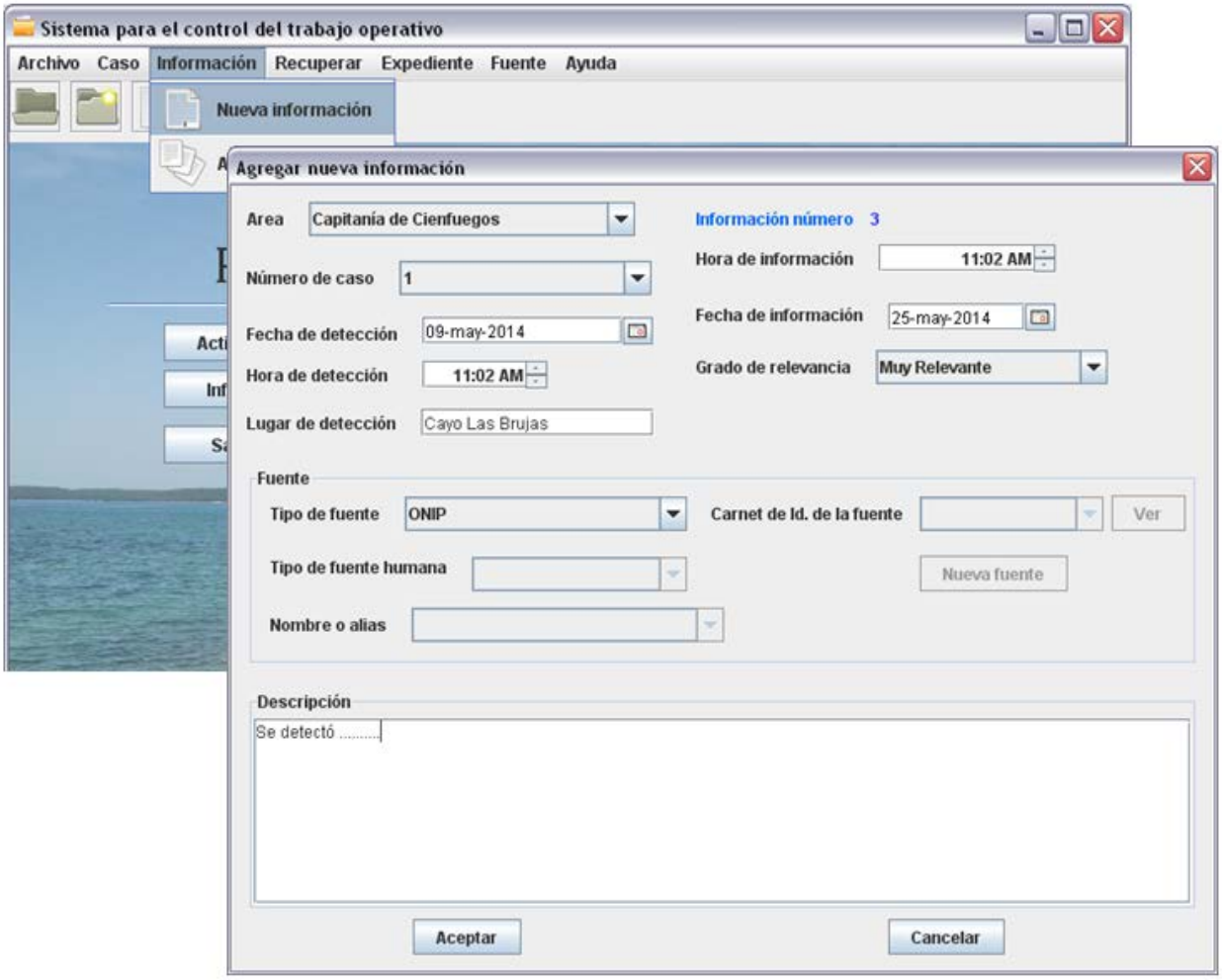

Figura 3.17: Interfaz gráfica para agregar una nueva información a un caso.

Para granular aún más la información, en la figura 3.18 se muestra la interfaz que permite introducir los datos de objetos, embarcaciones, personas involucradas, armamentos y drogas que se relacionan con la información, junto a la documentación gráfica vista como fotos.

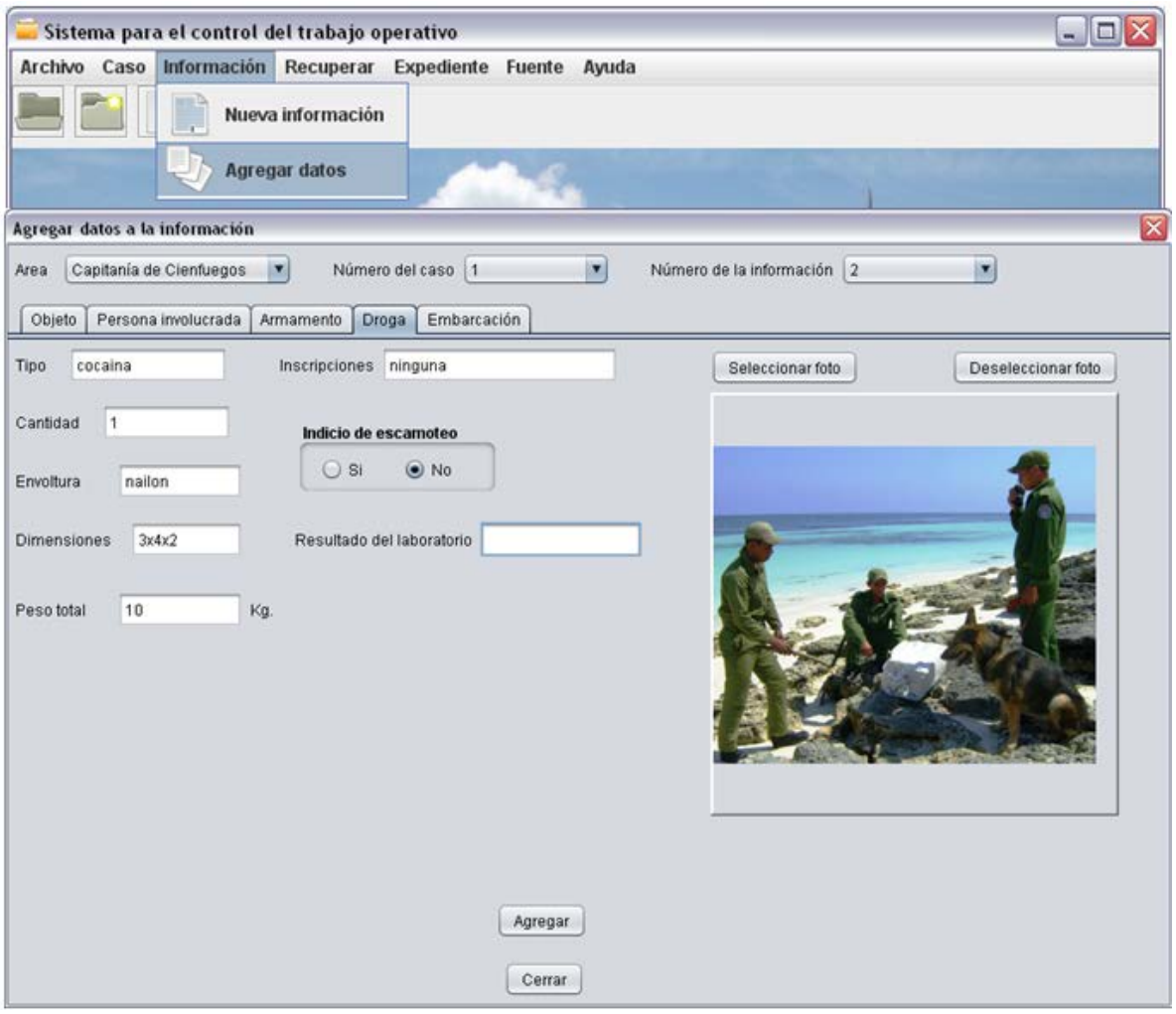

Figura 3.18: Ventana para introducir los datos de las informaciones.

3.4 Diseño del mercado de datos

Siguiendo la metodología para la modelación dimensional planteada por (Kimball and Ross, 2011) se muestran las siguientes cuatro fases, las cuales se utilizan como guía para diseñar una adecuada modelación del mercado de datos.

Las cuatro fases son:

- Elegir el proceso de negocio.
- Establecer el nivel de granularidad.
- Elegir las dimensiones.
- Identificar tablas de hechos y medidas.

Otros pasos a realizar son:

- Identificar los atributos de las tablas de dimensiones y hechos.
- Implementar el modelo dimensional detallado.

Elegir el proceso de negocio

Se identificaron los siguientes tres subprocesos que generan un conjunto de hechos y dimensiones descriptivas. A continuación se detallan cada uno de estos subprocesos.

- *Subproceso drogas*. En el subproceso drogas interesa conocer la cantidad de drogas incautadas y el peso total, ya sea por recalo o por ocupación.
- *Subproceso fuentes de información.* En el subproceso fuentes de información es de interés conocer las informaciones proporcionadas por las fuentes en un caso. Además se controla la cantidad de informaciones y el grado de relevancia de cada información.
- *Subproceso salidas ilegales.* En este subproceso se necesita conocer las personas involucradas en casos migratorios, como entradas y salidas del territorio nacional. Es de interés medir la cantidad de veces que una persona realiza alguno de estos tipos de violaciones.

Establecer el nivel de granularidad.

- En el caso de las drogas: interesa conocer la cantidad de drogas incautadas y el peso total, por caso, fecha y droga.
- En el caso fuentes de información: es de interés conocer la cantidad de informaciones por caso, tipo de fuente, fechas de detección e información y por la fuente que la proporciona.
- En el caso de las salidas del país: se necesita medir la cantidad de salidas que una persona ha realizado agrupado por fecha, caso e involucrado.

Elegir las dimensiones.

Tablas de dimensiones identificadas:

1. *dimension_fecha*

Es la tabla de dimensión más común en el diseño de almacenes de datos debido a que define una línea de tiempo para enmarcar la información almacenada, contiene las dimensiones fecha, día de la semana, día, mes, semestre y año.

2. *dimensión_caso*

   Esta tabla de dimensión describe las propiedades del universo de valores de los casos, contiene las dimensiones número, subcategoría, categoría y área.

3. *dimensión_droga*

   Esta tabla de dimensión describe las propiedades del universo de valores de drogas existentes, contiene las dimensiones tipo y las dimensiones o sea el tamaño de la droga.

4. *dimensión_fuente*

   Esta tabla de dimensión describe las propiedades del universo de valores de las fuentes que proporcionan las informaciones, contiene las dimensiones nombre o alias, carnet de identidad y tipo de fuente.

5. *dimensión_información*

   Esta tabla de dimensión describe las propiedades del universo de valores de las informaciones, contiene las dimensiones código de la información, descripción y relevancia.

6. *dimensión_involucrado*

   Esta tabla de dimensión describe las propiedades del universo de valores de las personas involucradas, contiene las dimensiones carnet de identidad o pasaporte, nombre y apellidos y nacionalidad.

Identificar tablas de hechos y medidas.

Tablas de hechos identificadas:

1. *fact_droga*

   Esta tabla de hechos recoge toda la información detallada de las drogas que se han incautado. Se relaciona con las dimensiones: dimensión_droga, dimension_fecha y con dimension_caso. Las medidas que almacena son peso_tot y cantidad_tot.

2. *fact_fuentes*

Esta tabla de hechos recoge todos los datos detallados de las fuentes que proporcionan informaciones. Se relaciona con las dimensiones dimensión_fecha, dimensión_fuente, dimensión_información y dimensión_caso. Cuenta con las medidas *cant* y *relevancia*.

3. *fact_salidas*

   Esta tabla de hechos recoge todas las informaciones detalladas de los intentos de salidas ilegales del territorio nacional o entradas ilegales. Se relaciona con las dimensiones dimensión_fecha, dimensión_involucrado y dimensión_pertenecer. Para contar la cantidad de violaciones de una persona se cuenta con la medida cantidad que toma valor 1.

Una vez identificadas las dimensiones y tablas de hechos se propone el modelo lógico dimensional, en el cual se utiliza una estructura de constelación de hechos, pues intervienen varias tablas de hechos, quienes a su vez comparten las siguientes dimensiones conformadas: *dimension_fecha* y *dimension_caso*. Además la tabla de dimensión fecha juega rol en la tabla de hechos fuentes de información, una vez como fecha detección de la información (*id_fecha_det*) y otra como fecha en la que se informa el hecho ocurrido (*id_fecha_info*), véase figura 3.19.

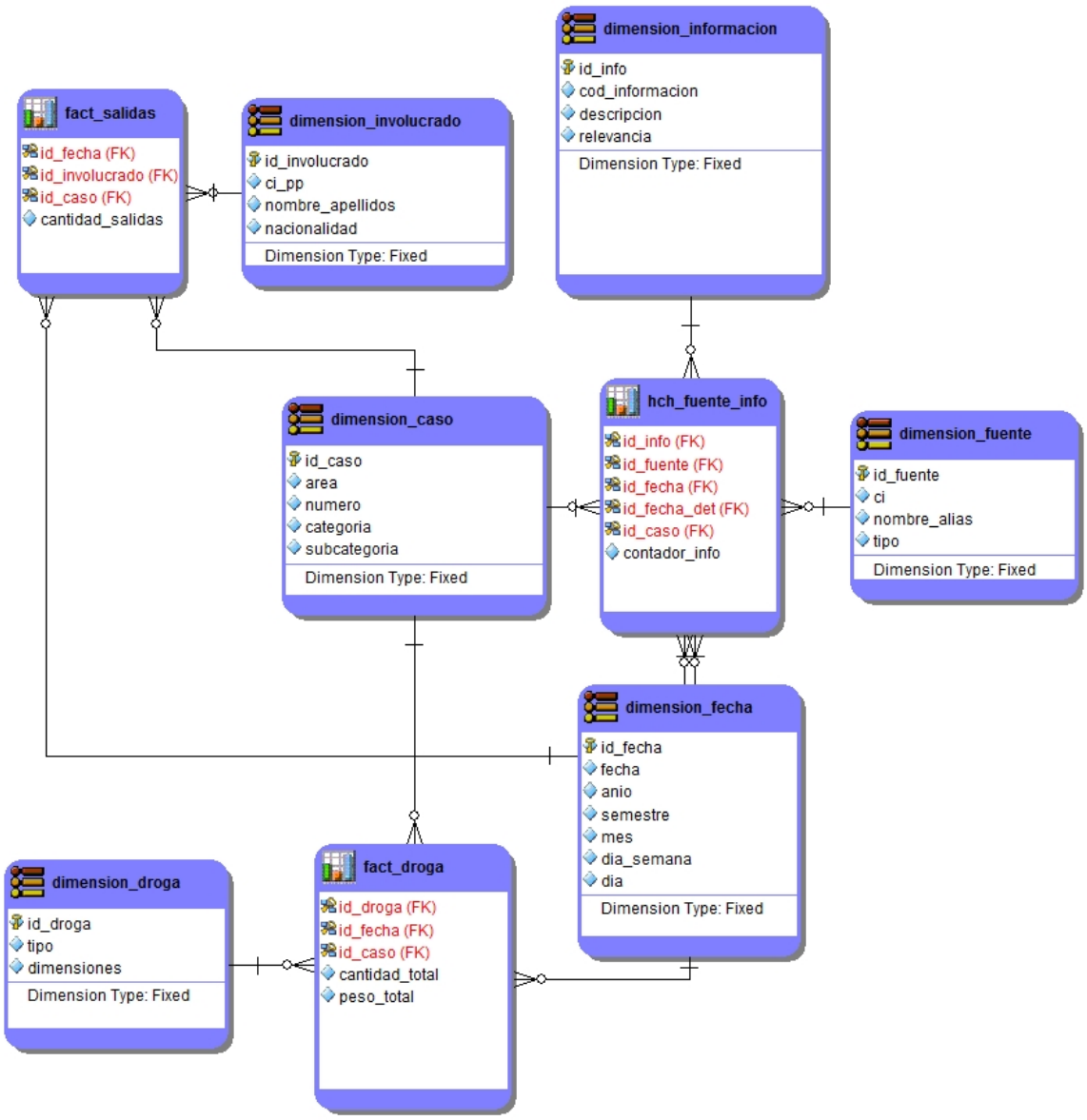

Figura 3.19: Modelo dimensional del mercado de datos propuesto.

3.5 Conclusiones parciales

En este capítulo se realizaron pruebas unitarias a tres métodos implementados en la aplicación, mediante el uso del método del camino básico para identificar los casos de prueba y pruebas de integración entre la aplicación principal y la base de datos. Se describió el sistema como producto final, destacándose las principales funcionalidades, la manera de manejar la autenticidad de los usuarios y la interfaz gráfica de usuario. Se propuso un diseño dimensional con estructura constelación de hechos, para una futura

implementación de un mercado de datos, identificándose y describiendo seis dimensiones y tres tablas de hechos que lo componen, basados en la metodología de Kimball.

Conclusiones

Al finalizar el trabajo de diploma “Sistema de Información para el control del trabajo operativo en Tropas Guardafronteras de Cienfuegos”:

1. Se analizaron las principales características de los SI, haciendo énfasis en los operacionales y los de apoyo a la toma de decisiones.
2. Se diseñó una base de datos relacional orientada a los requerimientos de usuario detectados.
3. Se implementó una aplicación con una interfaz gráfica de fácil uso para los usuarios.
4. Se confeccionaron reportes con vista a la toma de decisiones por parte de la jefatura.
5. Se propuso un modelo lógico dimensional para una futura implementación de un mercado de datos.
6. Se diseñaron e implementaron pruebas unitarias y de integración que permitieron validar parcialmente el sistema de información propuesto.

Recomendaciones

1. Implantar el sistema en las tropas Guardafronteras de Cienfuegos y continuar las validaciones del mismo.
2. Permitir el acceso a los datos por parte del sistema propuesto a las bases de datos pertenecientes al MININT tales como: “Registro de embarcaciones”, “Registro de personas” y al “Catálogo de drogas”.
3. Implementar los procesos de extracción, transformación y carga hacia el mercado de datos, para poblarlo y mantenerlo actualizado.
4. Crear un cubo de datos para navegar por él mediante los operadores OLAP.
5. Confeccionar reportes estáticos y dinámicos que visualicen la información en apoyo a la toma de decisiones.
6. Crear cuadros de mando integral que permitan obtener una visión global del cumplimento de los objetivos de la empresa.

Referencias bibliográficas

AYALA, A. P. 2006. Ingeniería de Software: una guía para crear Sistemas de Información. *México: sn,* 6**,** 2306-2495.

B., E. M. Y. H. 2008. *PostgreSQL Administration*, Oreilly.

BURCH, J. G. & GRUDNITSKI, G. 1992. *Diseño de sistemas de información: teoría y práctica*, Grupo Noriega.

BUSCHMANN, F., HENNEY, K. & SCHMIDT, D. C. 2007. *Pattern-Oriented SoftwareArchitecture*, John Wiley & Sons Inc.

CATALANI, E. 2007. *ARQUITECTURA Modelo/Vista/Controlador* [Online]. Available: http://exequielc.wordpress.com/2007/08/20/arquitectura-modelovistacontrolador/ [Accessed 27/01/2014.

CHEN, P. 1976. *The entity-relationship model: Toward a unified view of data. ACM Transactions on Database Systems*.

GARCÍA, C. E. 2009. *Ayuda a la modelación conceptual y lógica de bases de datos relacionales.* Marta Abreu de las Villas.

GÓMEZ, D., JÚSTIZ, D. & DELGADO, M. 2013. Unit Tests of Software in a University Environment. *Computación y Sistemas,* 17**,** 69-77

GRADY BOOCH, J. R., IVAR JACOBSON 2009. *El Lenguaje Unificado de Modelado*.

INMON, W. H. 2005. *Building the data warehouse*, John wiley & sons.

KAREN, C. & LARES, E. D. A. 2000. Sistemas de Información para los Negocios. Un enfoque de toma de decisiones. McGraw-Hill. México.

KIMBALL, R. & ROSS, M. 2011. *The data warehouse toolkit: the complete guide to dimensional modeling*, John Wiley & Sons.

LAUDON, C. 2008. Kenneth y P. LAUDON, Jane. Sistemas de información gerencial: Administración de la empresa digital. México: Pearson Educación de México, SA de CV.

LAUDON, K. C., LAUDON, J. P. & RODRÍGUEZ, J. R. 2008. *Administración de los Sistemas de Información: Organización y tecnología*, Prentice Hall Hispanoamericana.

LAUDON, K. L. Y. J. P. 1996. *Administración de los Sistemas de Información.*, México., Editorial Prentice Hall.

MONTILVA, J. 1999. *Desarrollo de sistemas de información*, Universidad de Los Andes.

MYATT, A. 2008. *Pro NetBeans IDE 6 Rich Client Platform Edition*, Apress.

MYERS, G. J., SANDLER, C. & BADGETT, T. 2012. *THE ART OF SOFTWARE TESTING*.

ORACLE. 2011. *The Java Tutorial* [Online]. Available: http://download.oracle.com/javase/tutorial [Accessed 5/5/2014 2014].

POSTGRESQL, C. 2012. *PostgreSQL 9.2.2 Documentation* [Online]. Available: http://www.postgresql.org/docs/9.2/static/release-9-2-2.html 2014].

PRESSMAN, R. S. 2010. *Software Engineering. A Practitioner's Approach*.

PRIETO, A. & MARTÍNEZ, M. 2004. Sistemas de información en las organizaciones: una alternativa para mejorar la productividad gerencial en las pequeñas y medianas empresas. *Revista de Ciencias Sociales (Ve)*, 10**,** 322-337.

REENSKAUG, T. & COPLIEN, J. 2009. The DCI Architecture: A New Vision of Object-Oriented Programming.

RODRÍGUEZ, D. R. D. J. M. 2006. Sistemas de Información. Introducción.

SILBERSCHATZ, A., SUDARSHAN, S. & KORTH, H. F. 2011. *Database System Concepts*, McGraw-Hill.

SOMMERVILLE, I. 2005. *Ingeniería del software*, Pearson Educación.

TAHCHIEV, P., LEME, F., MASSOL, V. & GREGORY, G. 2010. *JUnit in action*, Manning Publications Co.

YENISLEIDY F. ROMERO, Y. D. G. 2012. Patrón Modelo-Vista-Controlador. *Telemática*.

Anexos

Anexo I Diagramas de caso de uso

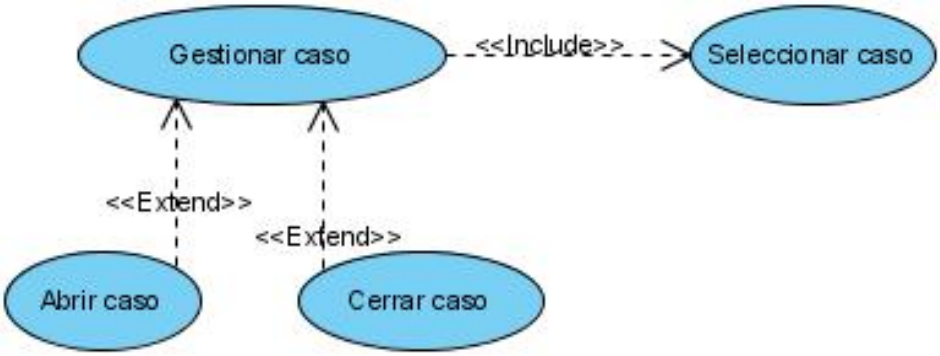

Figura A. 1: Diagrama del caso de uso “Gestionar caso”.

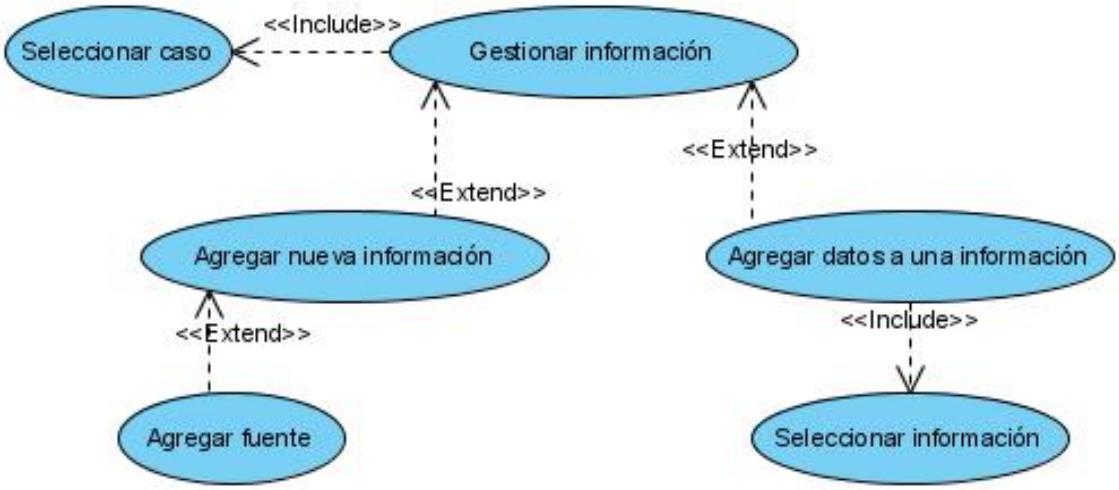

Figura A. 2: Diagrama del caso de uso “Gestionar información”.

Figura A. 3: Diagrama del caso de uso “Gestionar expediente”.

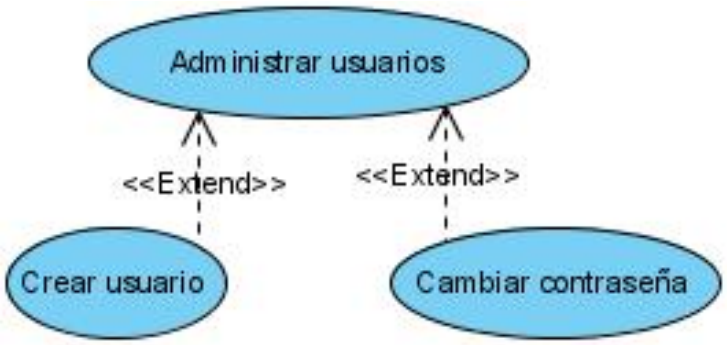

Figura A. 4: Diagrama del caso de uso “Administrar usuarios”.

Anexo II Diagramas de actividad

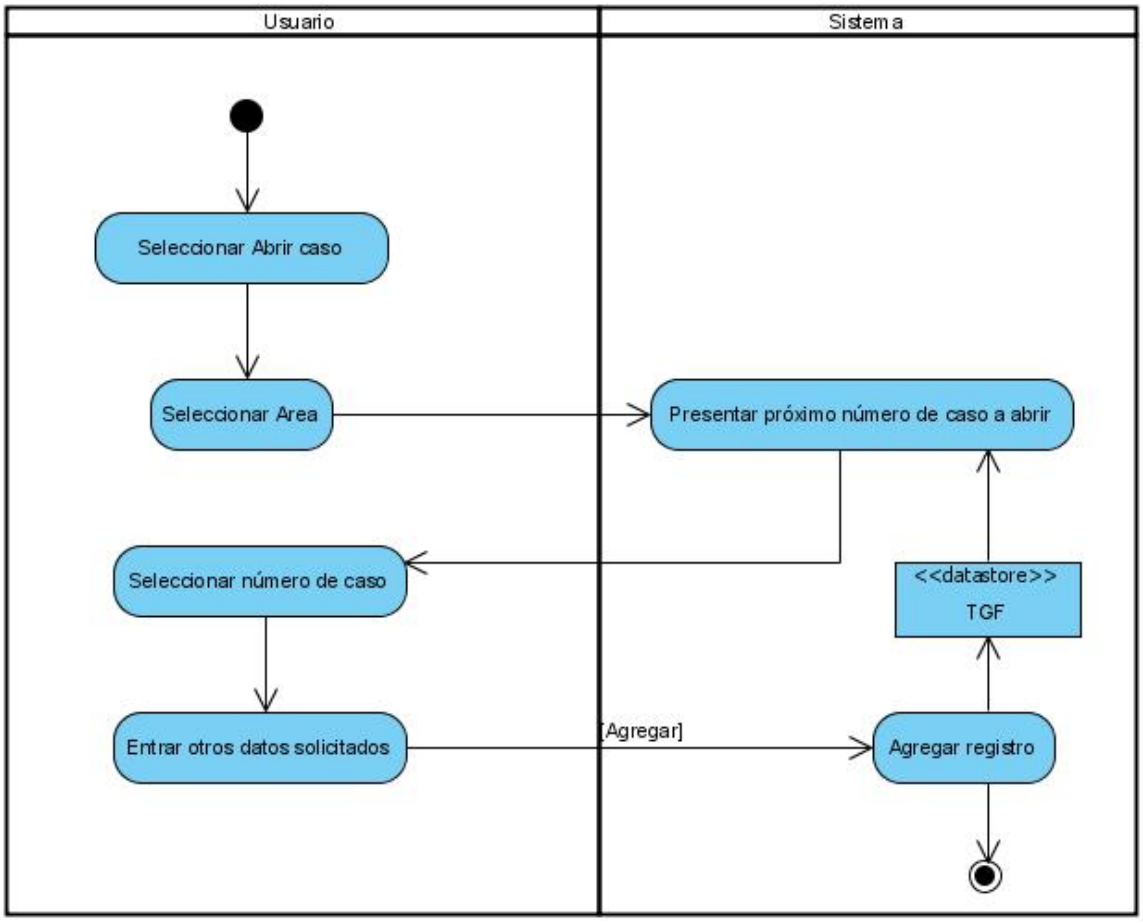

Figura A. 5: Diagrama de actividad para Abrir caso (extensión de “Gestionar caso”).

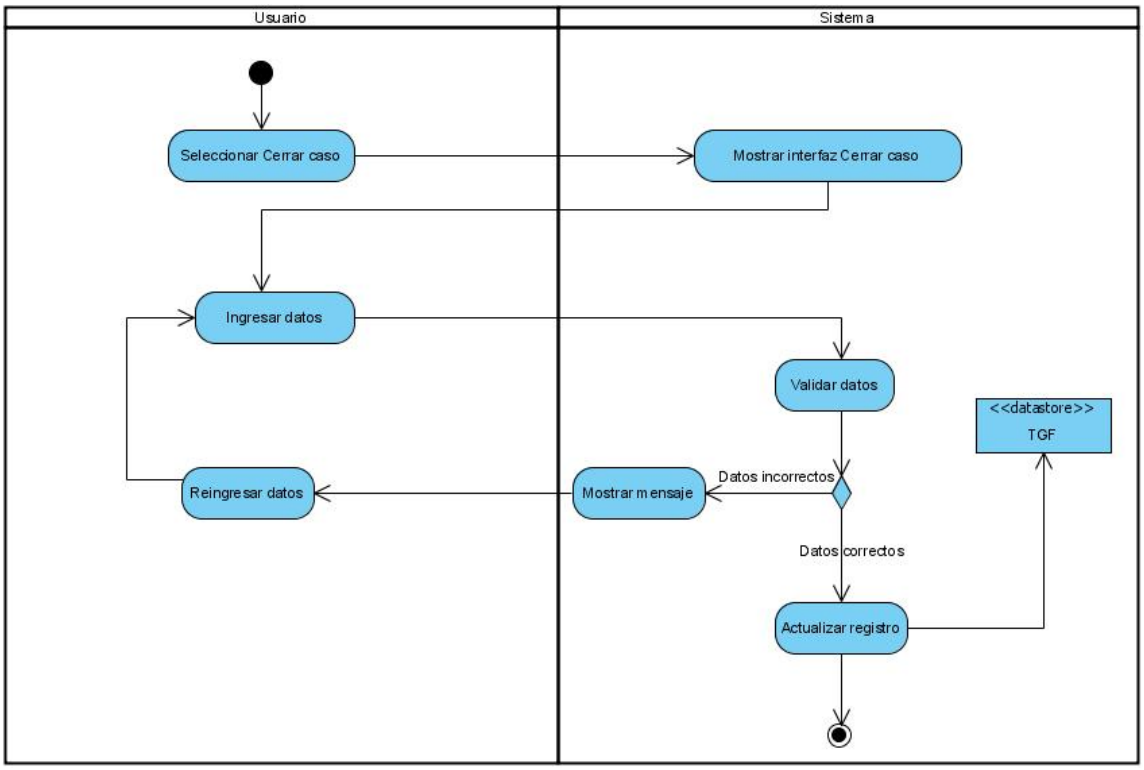

Figura A. 6: Diagrama de actividad para Cerrar caso (extensión de "Gestionar caso").

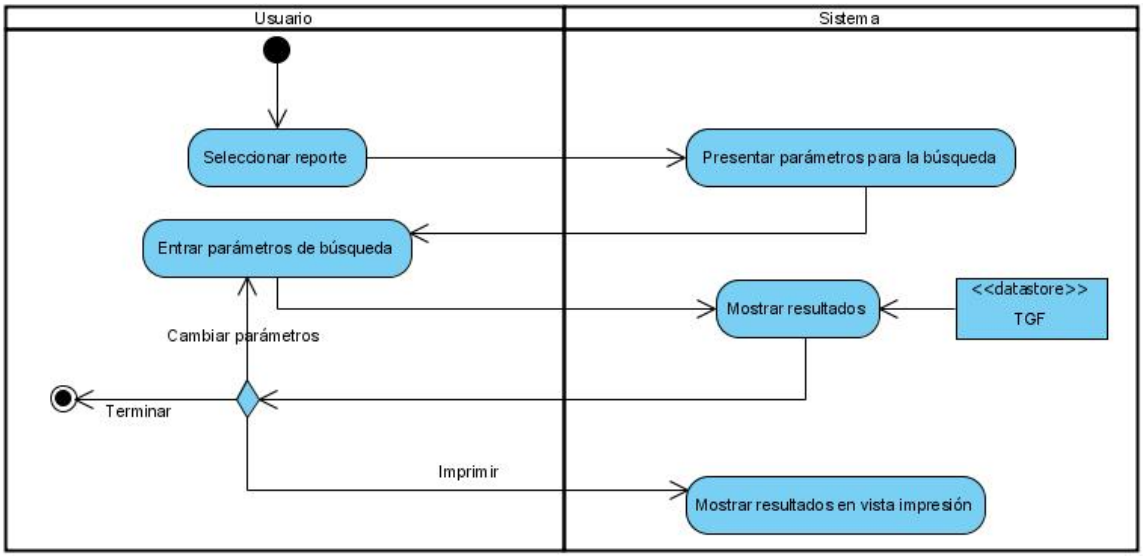

Figura A. 7: Diagrama de actividad para "Mostrar reportes".

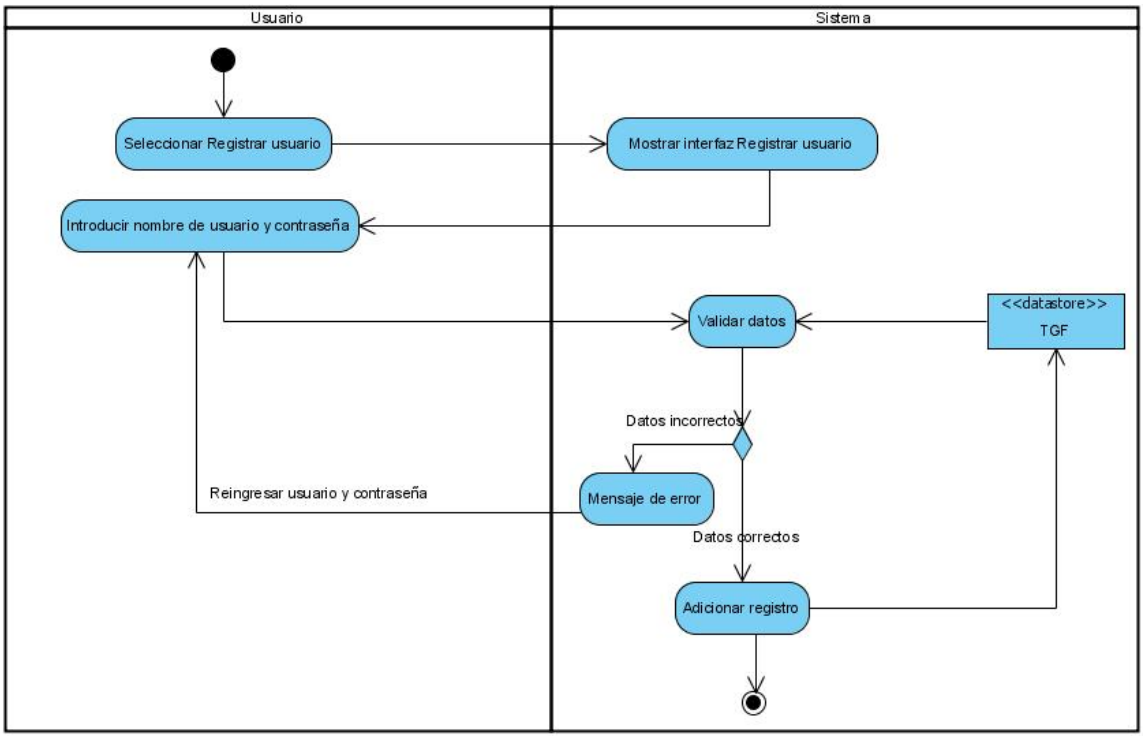

Figura A. 8: Diagrama de actividad para Crear usuario (extensión del caso de uso "Gestionar usuario").

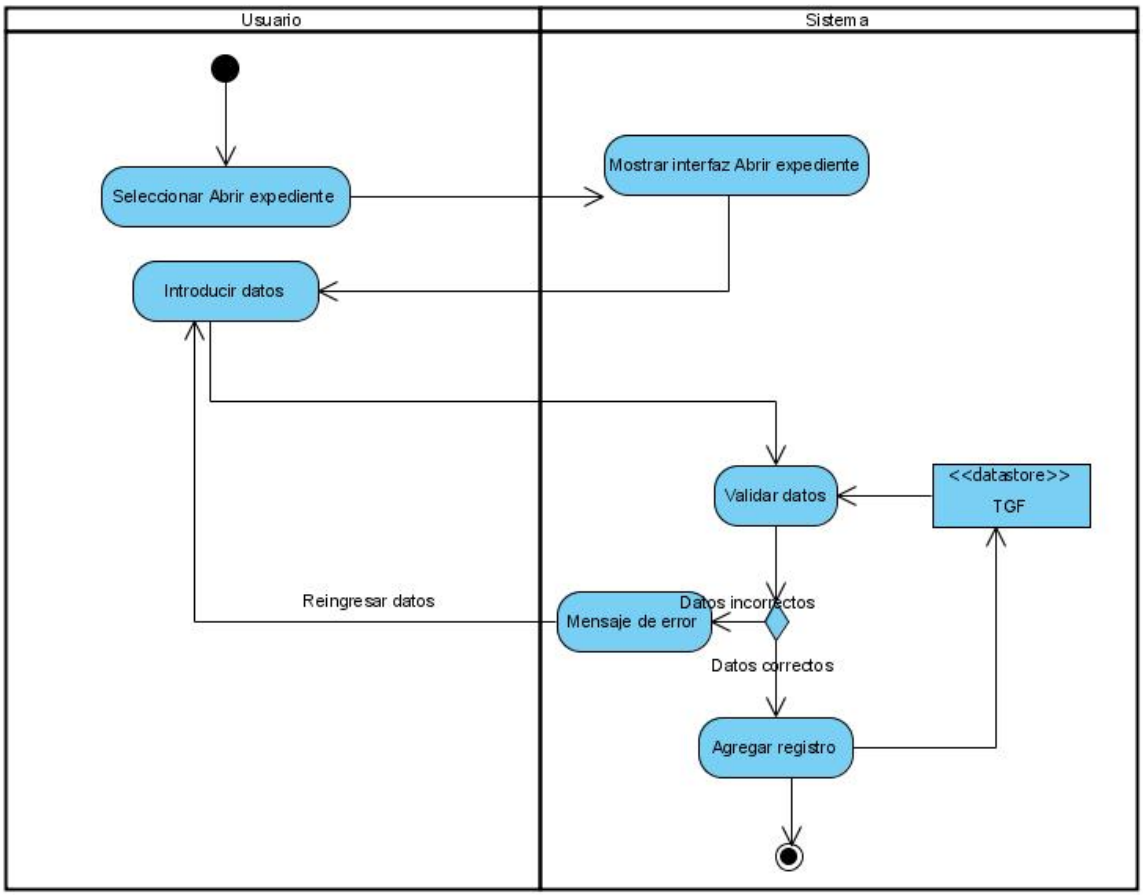

Figura A. 9: Diagrama de actividad para Abrir expediente (extensión de "Gestionar expedientes").

Anexo III Pruebas unitarias

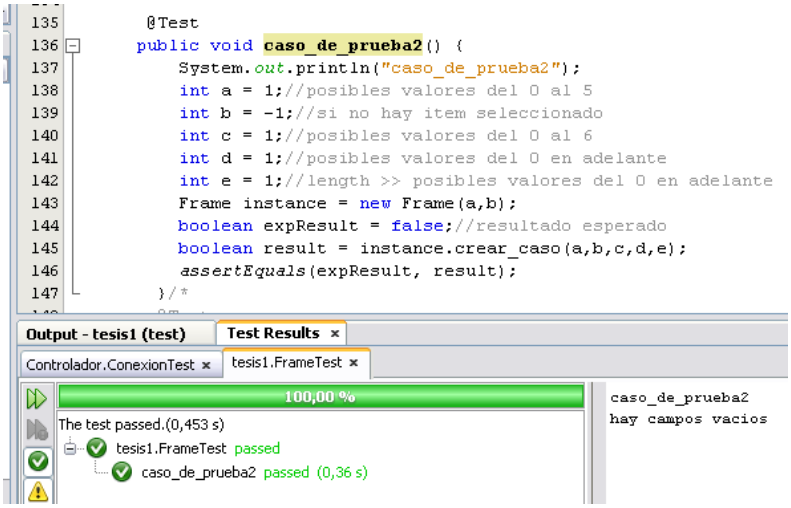

Figura A. 10: Caso de prueba 2 para el método *crear_caso*.

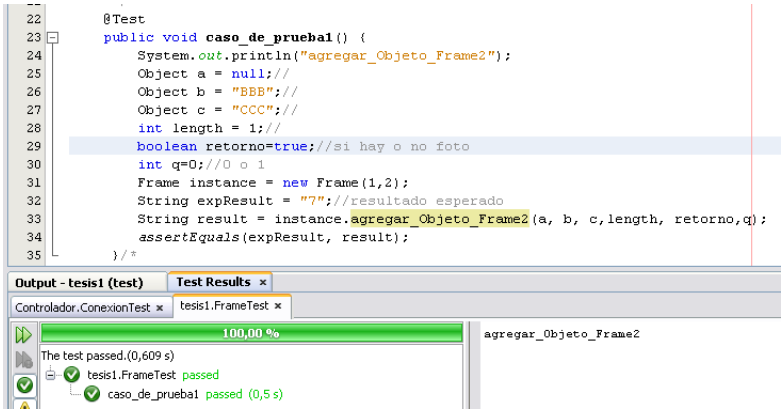

Figura A. 11: Caso de prueba 1 para el método *agregar_Objeto_Frame*.

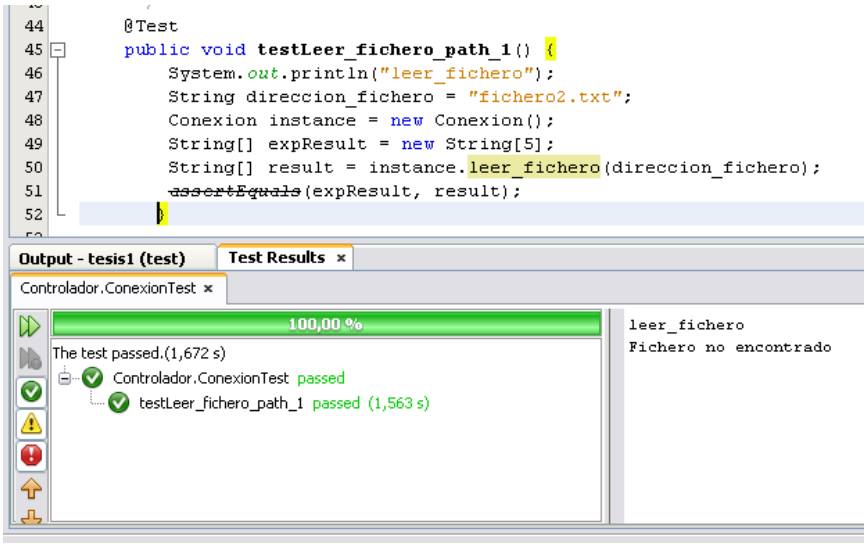

Figura A. 12: Caso de prueba 1 para el método *leer_fichero*.

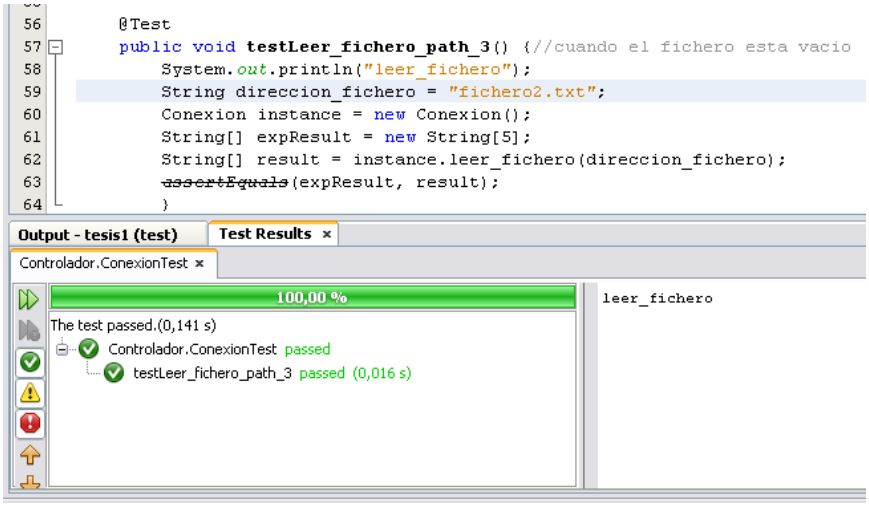

Figura A. 13: Caso de prueba 3 para el método *leer_fichero*.

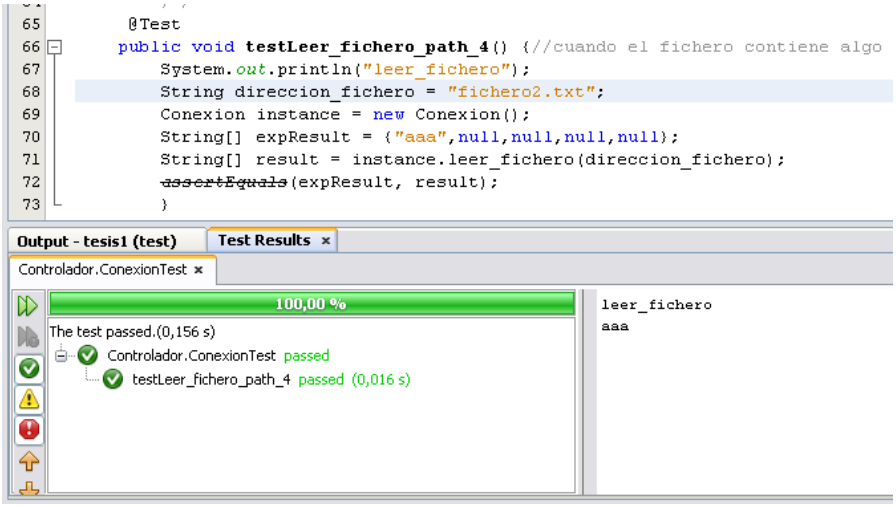

Figura A. 14: Caso de prueba 4 para el método *leer_fichero*.