República de Cuba

Universidad de las Ciencias Informáticas
Facultad 1

"Centro de Gestión de Emergencias y Seguridad Ciudadana (171)"
"Módulo de Administración"

Trabajo de Diploma
Presentado para optar por el título de
Ingeniero Informático

Autor: Yosvani Meneses Torres

Tutor: Ing. Yordanis Tornes Medina

"Año de la Revolución Energética en Cuba"
2006

Dedicatoria

…A mis padres

Agradecimientos

A papá y mamá, por el cariño y el amor que me han brindado toda la vida, por servirme de inspiración en todo plan que me he trazado, incluyendo este sueño realizado; a ellos les debo todo lo que soy y tengo, incluso mucho más.

A mi hermano, por su apoyo incondicional, por obligarme a servirle de ejemplo y por ser además, el mejor hermano del mundo.

A mami y papi, por su confianza, cariño y respeto. Por la obligación que tengo hacia ellos de convertirme en alguien en esta vida.

A abuelo y abuela, que siempre han estado conmigo en todo lo que hecho y dejado de hacer.

A Yaritza, por quererme y no dejarlo de hacer, sin importar la distancia.

A mi familia en general, por creer en mí.

A Yisel, que seriamos en el proyecto sin ella, le agradezco todo su apoyo y ayuda incondicional.

A mi tutor Tornes, por su consagración y apoyo para que este trabajo resultara lo mejor posible.

A Wilfredo, por ser mi cómplice en esta locura.

A mi mejor amigo Edgar, por su preocupación y confianza.

A Alexander, Bool, Molina y todos mis compañeros de cuarto, sobre todo por su honestidad y preocupación.

A mis compañeros de aula, por haber tenido la posibilidad de compartir estos años con ellos y haber conocido personas maravillosas.

A Daniel, por su gran ayuda y su alto grado de compromiso, sin él, este trabajo no habría sido posible.

A Albert, por su invaluable ayuda.

A todos los compañeros del proyecto, por su interés para que todo saliera bien y por darme su amistad.

A la Universidad de Oriente, por iniciarme en este mundo de ceros y unos.

A la UCI, por darme la posibilidad de cumplir este sueño.

A la Revolución, por permitir que todos, incluso hijos de obreros como yo tengan la posibilidad de graduarse en carreras universitarias.

A mi hijo, por darme fuerzas y servirme de inspiración, sobre todo a el, esa personita que existe, hace tan poco tiempo y tanta influencia ejerce en mi.

A Dios, por su amor infinito, por sus pruebas, por existir.

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo la Universidad de las Ciencias Informáticas para que haga el uso que estime pertinente con el mismo.

Para que así conste firmo la presente a los __ días del mes de _____ del año _____.

Yosvani Meneses Torres	**Ing. Yordanis Tornes Medina**
_____	_____
Firma del Autor	Firma del Tutor

OPINIÓN DEL TUTOR DEL TRABAJO DE DIPLOMA

Título: **Centro de Gestión de Emergencias y Seguridad Ciudadana (171). Módulo de Administración.**

Autor: **Yosvani Meneses Torres**

El tutor del presente Trabajo de Diploma considera que durante su ejecución el estudiante mostró las cualidades que a continuación se detallan.

<Aquí el tutor debe expresar cualitativamente su opinión y medir (usando la escala: muy alta, alta, adecuada) entre otras las cualidades siguientes:

- **Independencia**
- **Originalidad**
- **Creatividad**
- **Laboriosidad**
- **Responsabilidad>**

<Además, debe evaluar la calidad científico-técnica del trabajo realizado (resultados y documento) y expresar su opinión sobre el valor de los resultados obtenidos (aplicación y beneficios) >

Por todo lo anteriormente expresado considero que el estudiante está apto para ejercer como Ingeniero Informático; y propongo que se le otorgue al Trabajo de Diploma la calificación de **<nota>**. **<Además, si considera que los resultados poseen valor para ser publicados, debe expresarlo también>**

Ing. Yordanis Tornes Medina

_____ _____

Firma Fecha

Resumen

La seguridad ciudadana es uno de los principales deberes que todo gobierno debe garantizar; lograrlo depende en gran medida de la rapidez y efectividad con que sean atendidas las emergencias reportadas por la población.

Actualmente, en el Área Metropolitana de la República Bolivariana de Venezuela no existe un centro de gestión de emergencias y seguridad ciudadana automatizado, que integre a los órganos de seguridad y atienda de manera urgente y efectiva las emergencias solicitadas.

Para dar respuesta a esta problemática se promueve la realización del **Centro de Gestión de Emergencias y Seguridad Ciudadana (171)**, en su conjunto está compuesto por varios sistemas de tecnología actual y un sistema informático que gestionará el buen funcionamiento del mismo.

Este trabajo consiste en la creación del Módulo de Administración del **Sistema Informático del Centro 171**, surge a partir de la necesidad de una aplicación que gestione y controle todo lo referente a configuración y funcionalidad de los módulos que conforman el sistema, así como el nivel de acceso y visibilidad que poseen los usuarios a las acciones de cada aplicación.

Como objetivo se persigue realizar el análisis, diseño e implementación de una solución informática que realice todo el trabajo administrativo del centro de gestión de emergencias y seguridad ciudadana. El módulo de administración se encarga de realizar la configuración del sistema y de sus funcionalidades, permitiendo con ello un buen funcionamiento y la adaptabilidad ante situaciones excepcionales. Es de vital importancia para el sistema poder adaptarse en todo momento al contexto en que se desenvuelve, así como reasignar los recursos disponibles en el centro, esto se traduce en lograr mayor velocidad y efectividad del servicio de atención a emergencias. El módulo administrativo es el encargado de reconfigurar el sistema garantizando que se realicen los cambios a la configuración sin interrumpir el funcionamiento de las demás aplicaciones u ocasionarles demoras. Una pequeña demora puede conllevar a pérdidas tanto materiales como de vidas humanas, por lo que la rapidez es un requisito imprescindible. En el trabajo se realiza el análisis, diseño e implementación para un primer ciclo de desarrollo de la solución propuesta.

Índice

Introducción

"Todo individuo tiene derecho a la vida, a la libertad y a la seguridad de su persona" (Declaración Universal de los Derechos humanos. 1948), "derecho a la protección del estado, a través de los órganos de seguridad ciudadana, regulados por ley, frente a situaciones que constituyan amenazas, vulnerabilidad o riesgos para la integridad física de las personas, sus propiedades, y el disfrute de sus derechos" (Constitución de la Republica Bolivariana de Venezuela.).

Para una mejor protección de la población ante hechos delictivos, situaciones de emergencia y desastres naturales y para dar respuesta eficiente a cualquier irregularidad, el Ministerio del Interior y Justicia, atendiendo a su misión institucional de garantizar la seguridad ciudadana, promueve la formulación y puesta en marcha del **Centro de Gestión de Emergencias y Seguridad Ciudadana (171)**, partiendo de la premisa que la seguridad ciudadana es una condición necesaria para el desarrollo humano.

El Área Metropolitana de la República Bolivariana de Venezuela en la actualidad carece de un centro de gestión de emergencias y seguridad ciudadana automatizado, que integre a todos los órganos de seguridad ciudadana, que atienda con efectividad y rapidez las demandas de emergencias formuladas por la población.

Actualmente en Venezuela, existen varios números telefónicos para reportar emergencias, ofrecidos por empresas privadas de telecomunicaciones, gobernaciones y alcaldías. Algunos de estos centros funcionan las 24 horas de los 365 días del año, la mayoría sólo responde en horas de oficina, ninguno está interconectado entre sí y actúan como centros independientes, lo que demora la respuesta y repercute en el bienestar y seguridad de la población.

Aunque existen normativas de coordinación entre los órganos de seguridad ciudadana, han resultado insuficientes para la protección de la población ante hechos delictivos, situaciones de emergencia y desastres naturales.

El **Centro de Gestión de Emergencias y Seguridad Ciudadana (171),** de manera general pretende resolver los siguientes problemas y necesidades de la población, en el Área Metropolitana de la República Bolivariana de Venezuela:

- Fortalecer las acciones de coordinación entre los órganos de seguridad ciudadana y la unificación de criterios en la toma de decisiones.

- Disminuir el tiempo de respuesta a las demandas de emergencias formuladas por la población.
- Mantener un registro, en tiempo real e histórico, de información de hechos delictivos, emergencias y desastres.
- Medir la eficiencia y eficacia de los programas sociales en la disminución del delito dentro de las comunidades, y el uso de la información para el desarrollo de estrategias.

En su primera fase, el **Centro 171**, prioriza los siguientes servicios y funcionalidades:

- Capacidad de recepción de la información y denuncias de la población u otras fuentes.
- Despachar la atención de las emergencias hacia los centros de atención directa, y garantizar su seguimiento hasta lograr la asistencia solicitada.
- Mantener un registro de información que permita a los órganos de seguridad ciudadana, consultar y realizar análisis de la situación delictiva y de emergencias tanto en tiempo real como con carácter histórico.
- Posibilitar el análisis geográfico de la actividad delictiva y de emergencias, así como la localización en tiempo real de los recursos en servicio.
- Activar un proyecto piloto de cámaras de televisión digital operativa en áreas de interés ubicadas en el Distrito Metropolitano, para evaluar la factibilidad de implementación en etapas posteriores hacia el resto de los municipios.
- Acceso desde el centro a información de interés disponible en otros organismos.
- Disponer de un sistema de radiocomunicaciones TETRA que facilite la intercomunicación entre los diferentes cuerpos policiales de la región y el centro de control.
- Disposición de una sala situacional que permita la atención de emergencias en caso de catástrofes y desastres naturales.

Para lograr todo lo antes expuesto se hace necesario dotar al **Centro 171** de varios sistemas tecnológicos de punta y personal operativo, entre otros; para automatizar, agilizar y coordinar el gran proceso de atención de las emergencias. Integrados todos por un sistema informático que gestione toda la información generada por los mismos.

Por la diversidad de procesos implicados en la atención a las emergencias, existentes en el **Centro 171,** se realizó una división del sistema informático en varios módulos para así efectuar una rápida realización del mismo.

Los módulos que se implementarán son los siguientes:

- Módulo de Recepción de Llamadas (Operador).
- Módulo de Despacho de Emergencias (Despachador).
- Módulos/Sistemas de Supervisión (Supervisor).
- Módulo de Mapificación de la Información.
- Sistema de Notificación.
- Módulo de Grabación Digital.
- Módulo de Administración.
- Portal Web.
- Módulos de Acceso a Información de Otros Sistemas.

El módulo de administración es imprescindible para que el **Sistema Informático del Centro 171** brinde los servicios antes planteados. Este módulo es el encargado de realizar, tanto, la manipulación de la información de las entidades del sistema, como la configuración de las diferentes variables que determinan las funcionalidades del mismo.

El trabajo presenta como objetivo general analizar, diseñar e implementar el módulo administrativo del **Sistema Informático del Centro 171**, para lo cual se definen los siguientes objetivos específicos:

- Gestionar toda la información relativa a oficinas, usuarios, puntos y sitios de interés.
- Configurar las variables necesarias para el correcto funcionamiento del sistema.
- Generar reportes estadísticos del funcionamiento general del macrosistema.

La existencia de una aplicación administrativa, permite que pueda ser reconfigurado el sistema en cualquier momento, sin interrumpir ni retrasar el funcionamiento del mismo, además permite que los recursos del **Centro 171** sean redistribuidos y utilizados donde realmente sean necesarios. Garantiza la disponibilidad, efectividad y velocidad necesarias en un centro de gestión de emergencias y seguridad ciudadana. Una pérdida de tiempo, o interrupción del servicio, se traduce en pérdidas materiales o incluso de vidas humanas.

Durante la concepción del proyecto investigativo, y para la implementación satisfactoria del sistema, se plantea el siguiente grupo de tareas:

- Investigar sobre las características de instalación, configuración y explotación de las herramientas a utilizar.
- Evaluar las herramientas disponibles y las posibles alternativas según su funcionalidad y desempeño observado.
- Diseñar, implementar y evaluar una serie de pruebas que permitirán el conocimiento sobre la programación y la comunicación entre los distintos componentes a utilizar.
- Reunir conocimientos y experiencias acerca de la realización de las pautas y seleccionar un prototipo final.

El documento está estructurado en cuatro capítulos:

- Capítulo I denominado “Fundamentación del Tema”. Se incluyen todos los aspectos teóricos que soportan este proyecto; se analizan algunas de las herramientas y lenguajes de programación más utilizadas en el mundo para el desarrollo de las aplicaciones, y se plantea la metodología a seguir en la elaboración del mismo.
- Capítulo II denominado “Descripción del Sistema”. Se definen las reglas del negocio y los conceptos fundamentales del sistema, se determinan además los requerimientos funcionales y no funcionales del mismo, agrupándolos en casos de usos. Se da una idea general de la concepción del sistema.
- Capítulo III denominado “Construcción de la Solución Propuesta”. Se determinan las clases que se utilizarán en la implementación del sistema y la relación entre ellas. Además se muestra todo el proceso de obtención de la base de datos y se definen los principios de diseño de interfaz y codificación.
- Capítulo IV denominado “Implementación”. Se definen la topología de hardware sobre la que se sustenta la aplicación y sus componentes, se verifica su integridad y ajuste a los requerimientos planteados.

Capítulo I. Fundamentación del tema

Introducción

A lo largo del desarrollo de todo software es preciso realizar una serie de estudios que garanticen el avance del mismo, en búsqueda de cumplir los requerimientos planteados por el usuario. El presente capítulo muestra los aspectos teóricos que fue necesario investigar para desarrollar el Módulo de Administración del **Sistema Informático del Centro 171**. Se explica la necesidad de sistemas que garanticen la seguridad ciudadana, en la Venezuela actual, la importancia de los centros de gestión de emergencias y de un subsistema que lo administre. Se realiza un análisis de las tecnologías actuales de la informática y las comunicaciones consideradas a la hora de implementar el presente software.

Seguridad Ciudadana

Entiéndase por seguridad ciudadana el grado de respeto que se otorga al conjunto de derechos de los ciudadanos, no solo por parte del Estado, sino también por parte de las personas e instituciones públicas y privadas, que tienen que garantizar el bienestar de todos los componentes de las estructuras a las cuales están vinculadas, a las cuales prestan servicios, o de las que depende su seguridad.

Centros de Gestión de Emergencias

Los centros de gestión de emergencias son un servicio o conjunto de servicios que se brinda a la población, con el objetivo de ofrecer soluciones efectivas a las situaciones problemáticas de la sociedad y con esto otorgar mayor confianza en la seguridad ciudadana de cada individuo. Para lograr tales expectativas, estos centros deben ser un ente integrador de los organismos de seguridad y emergencias, encargado de recibir las llamadas de auxilio, prestando atención al ciudadano las 24 horas, los 365 días del año, manteniendo un servicio de comunicaciones que permite garantizar la adecuada supervisión y capacidad de respuesta de los organismos.

En la actualidad estos servicios se brindan con la ayuda de sistemas automatizados que se encargan de realizar la mayoría de las funciones de manera inmediata, estos se llaman Sistemas de Gestión de Emergencias.

Estos sistemas cuentan con diferentes subsistemas de cómputo, telefonía, radio y de información operativa. Ofreciendo a la población beneficios como, un número único al que reportar todo tipo de emergencias, disminución del tiempo de respuesta a las demandas formuladas, determinar la (s) institución (es) del estado más idóneas para atender el incidente. También visualiza toda la región que se atiende, los hechos que ocurren, cada llamada que se reciba, cuales carros están en servicio. Así se brinda un mejor servicio que contribuye de una forma u otra a que los ejecutivos tomen mejores decisiones para el bienestar de la seguridad del pueblo.

Centros de Gestión de Emergencias en Venezuela

Hoy en día, en la República Bolivariana de Venezuela existen varios estados que cuentan con centros automatizados para la atención de emergencias, algunos funcionan las 24 horas de los 365 días del año, la mayoría sólo responde en horas de oficina, no están interconectados entre sí, por lo que actúan como centros independientes. Tampoco presentan un sistema que visualice en un mapa digital de la ciudad o del estado una información detallada de los incidentes que ocurren a diario o con anterioridad, ni un sistema de administración centralizado que permita la configuración, del sistema en general, sin interrumpir el funcionamiento del mismo. Todo esto demora la respuesta por parte de los órganos de seguridad y repercute en el bienestar, la confianza y seguridad de la población.

A continuación se mencionan algunos de los centros que funcionan actualmente en Venezuela:

- Sistema Integral de Emergencia 171.
- Alcaldía Metropolitana 864-7191
- Alcaldía Libertador 545-4513 / 542-1711 / 409-8632
- Alcaldía Sucre 237-6343 / 271-0253 / 272-3360 / 0-800-76547
- Emergencias Bolívar 171 / (0286) 713-0279

Aunque existen normativas de coordinación entre estos centros que vinculan a los órganos de seguridad ciudadana, han resultado insuficientes para la protección de la población ante hechos delictivos, situaciones de emergencia y desastres naturales. Esta insuficiencia se ha traducido en la disminución de la capacidad de respuesta por parte del estado para enfrentar tales situaciones.

Tendencias y tecnologías actuales

Diseño Metodológico de la Investigación

La diversidad de procesos que se informatizan en la actualidad conlleva a la construcción de sistemas mucho más grandes y complejos; esto se debe principalmente al auge de las computadoras y al aumento del rigor del usuario. Estas tendencias actuales predisponen la existencia de un proceso bien definido y bien gestionado para poder lograr cumplir las exigencias requeridas en todo producto informático.

El **proceso de desarrollo de software** "*es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para su uso operativo". Concretamente "define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo.*"(IVAR JACOBSON 2000)

En concreto se traduce en el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software:

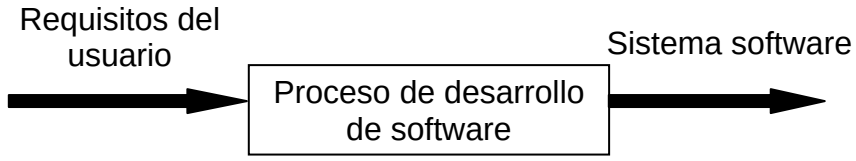


Figura 1. Proceso de desarrollo de software.

Existen varias metodologías para controlar el proceso para el desarrollo de software, una propuesta es el Proceso Unificado de Desarrollo de Software (Rational Unified Process, RUP), resultado de la evolución e integración de diferentes metodologías de desarrollo de software. RUP está basado en componentes, lo cuál quiere decir que el sistema en construcción está formado por componentes de software interconectados a través de interfaces bien definidas. Utiliza el *Lenguaje Unificado de Modelado* (Unified Modeling Languaje, UML) para preparar todos los esquemas, garantiza la elaboración de todas las fases de un producto de software orientado a objetos.

El UML es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema de software orientado a objetos. Entrega una forma de modelar conceptos como lo son procesos de

negocio y funciones de sistema, escribir clases en un lenguaje determinado, esquemas de bases de datos y componentes de software, entre otros.

Proceso Unificado de Desarrollo de Software (RUP)

Los principales rasgos que definen unívocamente al Proceso Unificado de Desarrollo de Software lo constituyen el ser dirigido por casos de uso, centrado en la arquitectura así como iterativo e incremental.

RUP está **dirigido por casos de uso,** el proceso de desarrollo sigue una trayectoria que avanza a través de los flujos de trabajo generados por los casos de uso. Los casos de uso se especifican y diseñan en el principio de cada iteración, y son la fuente a partir de la cual los ingenieros de prueba construyen sus casos de prueba. Los casos de uso describen la funcionalidad total del sistema, pensada en términos de la importancia de la misma para el usuario. Esto no significa que se desarrollen aisladamente respecto a la arquitectura, sino que se desarrollan a la vez, madurando ambos según avanza el ciclo de desarrollo. Los casos de uso guían a la arquitectura del sistema (como parte del proceso) y ésta influye en la selección de los casos de uso. La arquitectura involucra los elementos más significativos del sistema y está influenciada entre otros por las plataformas software, los sistemas operativos, los sistemas de gestión de bases de datos, además de otros como sistemas heredados y requerimientos no funcionales. Por esta razón RUP está **centrado en la arquitectura**, lo que invoca más la relación con los principios de la usabilidad. Desde que se planteó por primera vez el modelo **incremental** de desarrollo de software, y se establecieron sus ventajas con respecto al modelo de cascada, siempre se ha recomendado dividir los proyectos en pequeños ciclos o **iteraciones** a través de cada una de las fases por las que pase. RUP divide el proceso de desarrollo de software en cuatro fases, (inicio, elaboración, construcción y transición), dentro de las cuales se realizan varias iteraciones en número variable, según el proyecto, las cuales se definen de acuerdo al nivel de madurez que alcanza el producto que se van obteniendo con cada actividad ejecutada. La terminación de cada fase ocurre en el hito correspondiente a cada una, donde se evalúa que se hayan cumplido los objetivos de la fase en cuestión. Y con la terminación de la fase de inicio se puede ya determinar la factibilidad, tanto operativa como económica del proyecto, lo cuál nos lleva a tomar la decisión de continuarlo o no realizarlo.

El Lenguaje de Modelado Unificado (UML)

Es un lenguaje de propósito general para el modelado orientado a objetos. UML es también un lenguaje de modelación visual que permite una abstracción del sistema y sus componentes, se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Nos permite entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

UML no es un lenguaje de programación. Las herramientas de modelación pueden ofrecer generadores de código a partir de especificaciones UML para una gran variedad de lenguaje de programación, así como construir modelos por ingeniería inversa a partir de programas existentes.

Permite la modificación de todos sus miembros mediante estereotipos y restricciones. Un estereotipo nos permite indicar especificaciones del lenguaje al que se refiere el diagrama de UML. Una restricción identifica un comportamiento forzado de una clase o relación, es decir mediante la restricción estamos forzando el comportamiento que debe tener el objeto al que se le aplica.

UML es un estándar, su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que ha sido diseñado para modelar cualquier tipo de proyectos.

Sistemas Gestores de Bases de Datos (SGBD)

En 1964 se conciben los primeros SGBD, con estos se pretendió dar un viraje a los Sistemas de Archivos, los cuales se limitaban a la estructuración del almacenamiento físico de los datos. Con los SGBD se logró, por medio de actividades integradas, verlos físicamente en un solo almacenamiento, pero lógicamente se manipulan a través de esquemas compuestos por estructuras donde se establecen vínculos de integridad, métodos de acceso y organización física sobre los mismos, permitiendo así obtener valores agregados de utilización.

"*En 1970 el Dr. Edgar F. Codd propuso el Modelo de Datos Relacional, este modelo es el que ha marcado la línea de investigación por muchos años, representa al mundo real mediante tablas relacionadas entre sí por columnas comunes. Viendo la necesidad de mejorar este estándar se desarrollaron los Sistemas*

Gestores de Base de Datos Relacionales (SGBDR) cuyas características hacen al sistema mucho más eficiente que los sistemas de manejos de archivos(CORABEL 2004)" .

Un SGBDR es un conjunto de datos relacionados entre sí y un grupo de programas para tener acceso a esos datos, permitiendo concurrencia y recuperación. La persistencia de un SGBDR hace referencia a la conservación de los datos después de la finalización del proceso que los creó, y la concurrencia se refiere a la capacidad del sistema para gestionar a múltiples usuarios interactuando al mismo tiempo sobre el mismo. Entre los SGBDR más comunes se encuentran SQL Server y Oracle, a continuación se dará una breve explicación de los mismos.

Oracle

"*Oracle es un manejador de BD relacional que hace uso de los recursos del sistema informático en todas las arquitecturas de hardware, para garantizar su aprovechamiento al máximo en ambientes cargados de información. También proporciona la capacidad de almacenar y acude a los datos de forma consecuente con un modelo definido como relacional (…). Además es una suite de productos que ofrece una gran variedad de herramientas (CORABEL 2004)".*

"Es el mayor y más usado Sistema Manejador de Base de Dato Relacional (RDBMS) en el mundo e incluye cuatro generaciones de desarrollo de aplicación, herramientas de reportes y utilitarios"(CORABEL 2004).

Entre las características de Oracle se destaca su escalabilidad y alta disponibilidad, aportando un sistema de administración completo para gestionar todas las situaciones críticas de una Base de Datos, por ejemplo presenta: Sistema de seguridad basados en usuarios, grupos y roles, alertas, respaldo y restauración de datos. Oracle corre en computadoras personales, mainframes y computadoras con procesamiento paralelo masivo. Soporta unos 17 idiomas, corre automáticamente en más de 80 arquitectura de hardware y software distintos sin tener la necesidad de cambiar una sola línea de código. "*Esto es porque más del 80% de los códigos internos de Oracle son iguales a los establecidos en todas las plataformas de sistemas operativos(CORABEL 2004)"*.

Microsoft SQL Server

SQL Server es un sistema de gestión de bases de datos relacionales (SGBDR) basada en el lenguaje SQL, capaz de poner a disposición de muchos usuarios, grandes cantidades de datos de manera simultánea.

Entre sus características figuran:

- Soporte de transacciones.
- Gran estabilidad.
- Gran seguridad.
- Escalabilidad.
- Soporta procedimientos almacenados.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a esta información.
- Además permite administrar la información de otros servidores de datos.

Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle o Sybase.

Microsoft SQL Server presenta un entorno de desarrollo cómodo y de alto rendimiento a través de la implementación de aplicaciones de dos capas mediante el uso de formularios Windows.

Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para la mayoría de las plataformas de desarrollo, incluyendo .NET.

Microsoft SQL Server, al contrario de su más cercana competencia, no es multiplataforma, ya que sólo está disponible en Sistemas Operativos de Microsoft. Como política de desarrollo de Cuba y también de nuestro país hermano Venezuela, se encuentra la migración a software libre, por lo que utilizar un sistema gestor de bases de datos que dependa en su implantación de un sistema operativo propietario no se presenta como una opción factible.

Plataformas de desarrollo

Microsoft.NET

Microsoft.NET es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando durante los últimos años con el objetivo de obtener una plataforma sencilla y potente para distribuir el software, de manera que los lenguajes de programación y modelo de componentes con los que hayan sido desarrollados puedan comunicarse y combinarse sin importar la plataforma, a ella se le denominó **plataforma .NET**

Microsoft, para crear aplicaciones escalables y distribuidas ha publicado el Framework .NET SDK, que incluye las herramientas necesarias tanto para el desarrollo de aplicaciones como para su distribución y ejecución. Visual Studio.NET es la herramienta por excelencia que permite el desarrollo de aplicaciones desde una interfaz visual basada en ventanas.

Framework .NET

Se le llama Framework, ("entorno de trabajo"), a las Bibliotecas de Clase Base, (también llamadas BCL) y el Common Language Runtime, (CLR) que resulta ser el corazón de la plataforma .NET y ofrece una interoperatividad multi-lenguaje, o sea, la característica que códigos de un lenguaje, para una aplicación, pueda utilizar códigos de otro lenguaje sin inconvenientes.

Visual Studio .NET 2003.

Para .NET se distribuye el paquete Visual Studio .NET (VS.NET), con un sin fin de utilidades para el desarrollador.

Posee varios lenguajes, esto permite elegir el lenguaje más adecuado para la tarea a desarrollar. Proporciona eficaces herramientas para crear con rapidez las aplicaciones.

Los equipos de desarrollo pueden mantener el control de versiones y compartir código fuente y documentación. Se puede analizar el rendimiento y la escalabilidad de las aplicaciones. .NET tiene una plataforma escalable y confiable para aplicaciones distribuidas por lo que los sistemas creados son seguros, confiables y de alto rendimiento.

Lenguajes

VS.NET, nos ofrece la posibilidad de programar en lenguajes como Visual C++ 7.0, Visual C#, Visual Basic .NET, ASP.NET y JSCRIPT, (además de incluir ADO.NET y muchas herramientas), todos con la posibilidad de utilizar la BCL. A continuación se hace una breve explicación de algunos de los lenguajes más utilizados.

- **Visual Basic .NET (VB)**

  De los lenguajes utilizados en versiones anteriores de Visual Studio, éste ha sido el más beneficiado y el que más ha cambiado. Gracias a la BCL, VB ha adquirido características que nunca antes había presentado, por ejemplo: la programación multi-hilo. Se han quitado instrucciones típicas de Basic reemplazándolas por unas más intuitivas y se han agregado y/o modificado otras. Ahora las aplicaciones de Visual Basic.NET son mucho más robustas.

- **C#, (o Visual C#)**

  Este es el nuevo lenguaje inspirado en C/C++, similar a Java. Este lenguaje tiene una sintaxis basada en C/C++ con una estructura similar a Java pero con características especiales que lo hacen muy estructurado, sencillo, poderoso y de alto nivel. Hace uso de las BCL lo cual sirve para cualquier otro lenguaje .NET.

  Ofrece una serie de ventajas basadas fundamentalmente en aspectos tales como:

  - Sencillez: Es un lenguaje simple y de fácil aprendizaje,
  - Soporta el trabajo con punteros manteniéndolos en espacio de código que el usuario declara como unsafe (inseguro).
  - Orientado a Objetos: Es un lenguaje diseñado para la implementación de software orientado a objetos. Implementa conceptos como la herencia, el tratamiento de estructuras, la abstracción, la herencia, el polimorfismo, la encapsulación, entre otros.
  - Modernidad: Es un lenguaje joven, surgido en el año 2000 y muy similar al java.
  - Distribuido: Está concebido para trabajar en un entorno conectado en red. Cuenta con una amplia biblioteca de clases para comunicarse mediante TCP/IP: HTTP, FTP, etc.
  - Seguridad de tipos.
  - Instrucciones seguras.
  - Sistema de tipos unificado.
  - Extensibilidad de tipos básicos.
  - Orientación a componentes.

- Extensibilidad de operadores.
- Extensibilidad de modificadores.
- Versionabilidad.
- Eficiencia.
- Compatibilidad.

Plataforma Java

La plataforma Java se ejecuta sobre otra plataforma hardware/software. Esta posee dos componentes fundamentales:

1. La Máquina Virtual Java (JVM). La JVM es el intérprete Java.
2. La Interfaz de Programación de Aplicaciones (API). El API Java es un conjunto de clases ya desarrolladas que ofrecen un gran abanico de posibilidades al programador.

Hoy en día esta plataforma ha evolucionado en concordancia con el avance tecnológico y se ha convertido en una de las plataformas de programación mas usadas por los desarrolladores. Su principal ventaja es que su entorno de desarrollo es independiente de la plataforma sobre la que se trabaje, es decir, sus aplicaciones son funcionales tanto en Linux como en Windows. Sin embargo Java no permite la interoperabilidad de múltiples lenguajes. Es posible compilar a bytecode desde múltiples lenguajes, pero no es posible alcanzar el nivel de interacción que se da en .NET. Además la intercomunicación entre aplicaciones es segura pero a su vez muy difícil, esto ocurre por el hecho que tiene que cumplir con los protocolos de seguridad de los SO sobre los que funciona. Actualmente existen distintas ediciones de la plataforma Java como por ejemplo: J2ME (Java2 Micro Edition), J2EE (Java2 Enterprise Edition) y J2SE (Java2 Standard Edition), esta última es la más usada por los desarrolladores.

Java:

Java es un lenguaje de programación surgido en 1991. Los creadores de Java se basaron en C++, pero eliminaron la mayoría de sus complejidades como por ejemplo: la herencia múltiple y la creación de punteros. Java presenta características que lo convierten en un lenguaje seguro, estándar y de alto nivel, algunas de las principales características se muestran a continuación:

- Orientado a Objetos.
- Distribuido.

- Interpretado.
- Robusto.
- Seguro.
- Portabilidad.
- Altas prestaciones.
- Multihilo.
- Dinámico.

Conclusiones

En este capítulo se analizaron las ventajas y la importancia de los Centros de Gestión Emergencias y Seguridad Ciudadana en la República Bolivariana de Venezuela, cómo se ha ido introduciendo poco a poco el uso de estos servicios en todos los estados, buscando de esta forma una mejor atención al ciudadano y que este se sienta identificado con su gobierno. Se realizó un análisis de la metodología a utilizar, de la herramienta para modelar el problema, los principales lenguajes de programación y plataformas de desarrollo.

Después de realizado todo el proceso de investigación se decidió que el Sistema Informático del Centro 171, en consecuencia su módulo de administración, se implementará utilizando la plataforma .NET, en el lenguaje C#, metodología RUP utilizando UML como lenguaje de modelación y como gestor de base de datos Oracle.

Capítulo II. Descripción del Sistema

Introducción

En este capítulo se realiza un análisis del proceso de administración del **Sistema Informático del Centro 171**, permitiendo que se logre una mayor comprensión de sus características y el marco en que se desenvuelve. El proceso de desarrollo del presente software se centró en la metodología RUP, haciendo uso del UML para la modelación de los artefactos. Como herramienta CASE para asistir el proceso de desarrollo fue utilizado el Rational Rose.

Se presenta el modelo de dominio, como alternativa al modelo de negocio, para entender el contexto en que se ubica el sistema, debido a la poca estructuración de los procesos que describen el negocio de los centros de gestión de emergencias en la República de Venezuela.

Se presentan los requerimientos funcionales y no funcionales especificados por el usuario. Se describen los casos de uso y la propuesta del sistema a desarrollar.

Modelo del dominio

Durante el análisis de los procesos surgieron varios conceptos relacionados entre si, estos definen el dominio del sistema, que permite a los usuarios, clientes, desarrolladores e interesados, utilizar un vocabulario común para poder entender el contexto en que se ubica el mismo, para capturar correctamente los requisitos y lograr una solución adecuada.

Conceptos del Modelo del Dominio

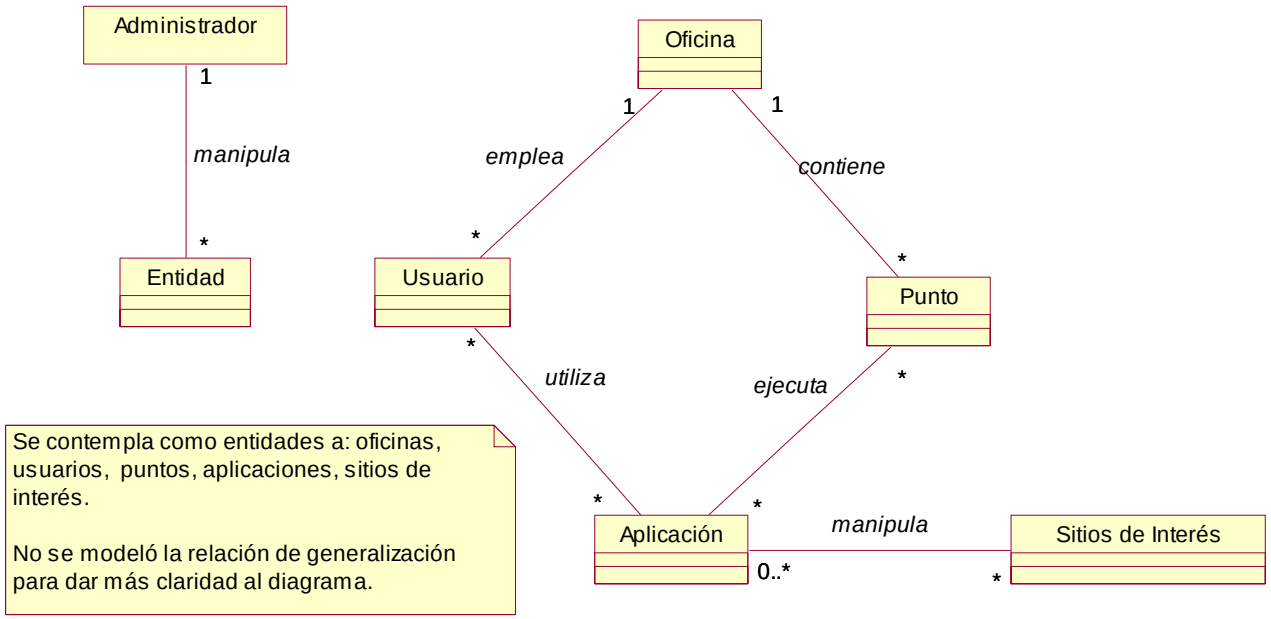
Administrador: Único actor y encargado de iniciar todas las acciones del módulo administrativo.

Entidad: Representa las entidades que intervienen en el sistema.

Oficina, Usuario, Aplicación, Punto, Sitios de interés: Instancias de Entidad.

A continuación se muestra como interactúan los conceptos manipulados en el sistema:

Diagrama 1. Modelo del dominio del módulo de administración.



Descripción del modelo del dominio

El administrador es el encargado de manipular toda la información de todas las entidades del sistema. Es utilizado el concepto entidad para definir a: Oficina, Usuario, Punto (estación de trabajo), Aplicación y Sitios de Interés. Geográficamente hablando el sistema contará con un grupo de oficinas, con usuarios y puntos propios. Las aplicaciones sólo pueden ser ejecutadas en aquellos puntos que han sido autorizadas. Un Usuario puede utilizar aquellas aplicaciones en las que tiene permiso para un punto donde esa aplicación pueda correr. Los sitios de interés o sitios que revisten gran importancia a la hora de garantizar la seguridad ciudadana, son manipulados por aplicaciones específicas encargadas de ello.

Especificación de los requerimientos

Requerimientos funcionales

Como punto de partida para determinar que debe lograr el sistema, se han identificado los siguientes requerimientos funcionales que representan las condiciones y capacidades que el software debe cumplir:

Iniciar Aplicación

1. **Iniciar Aplicación**
   1.1. Mostrar la aplicación al usuario con las opciones a las que tiene acceso, según los roles o permisos que tiene asignado.

Activar Aplicación

2. **Activar aplicación**
   2.1. Los datos necesarios son:
      2.1.1. Número de serie de la oficina: número único generado aleatoriamente por el sistema para identificar unívocamente a cada oficina.
      2.1.2. Nombre de usuario.
      2.1.3. Contraseña.
      2.1.4. Servidor: nombre del TNS utilizado para conectarse con el servidor de base de datos.
   2.2. La aplicación se activa en la base de datos y en el registro del sistema.

Autenticar usuario

3. **Autenticar Usuario**
   3.1. Introducir nombre de usuario y contraseña.
   3.2. Validar datos introducidos.
   3.3. Mostrar al administrador ya autenticado las opciones a las que tiene acceso.
   3.4. Obtener roles y permisos asociados al usuario.

Gestionar información de las entidades del sistema

4. **Gestionar oficina**
   4.1. Insertar datos de Oficina.
      4.1.1. Nombre de la Oficina.
      4.1.2. Dirección.

4.1.3. Parroquia.

4.1.4. Tipo de oficina.

4.1.5. Oficina a la que pertenece (En caso de ser una suboficina).

4.1.6. Estado de activación.

4.1.7. Teléfonos.

4.1.8. Números de Fax.

4.1.9. Serial de activación.

4.2. Modificar datos de Oficina.

4.2.1. Nombre de la Oficina.

4.2.2. Dirección.

4.2.3. Parroquia.

4.2.4. Tipo de oficina.

4.2.5. Oficina a la que pertenece (En caso de ser una suboficina).

4.2.6. Estado de activación.

4.2.7. Teléfonos.

4.2.8. Número de Fax.

5. Gestionar usuarios

5.1. Insertar datos del Usuario.

5.1.1. Nombre de Usuario.

5.1.2. Contraseña.

5.1.3. Oficina (Oficina donde pertenece el usuario).

5.1.4. Estado de activación.

5.1.5. Rol.

5.2. Modificar datos del Usuario.

5.2.1. Contraseña.

5.2.2. Oficina (Oficina donde pertenece el usuario).

5.2.3. Estado de activación.

5.2.4. Rol.

5.3. Establecer contraseña.

5.4. Eliminar un Usuario.

6. Gestionar aplicación

6.1. Insertar datos de una Aplicación.

6.1.1. Nombre de la aplicación.

6.2. Modificar datos de una Aplicación.

6.2.1. Nombre de la aplicación.

6.3. Eliminar una Aplicación.

7. Gestionar puntos de red

7.1. Insertar datos de un Punto.

7.1.1. Descripción.

7.1.2. Número IP.

7.1.3. Oficina (Oficina donde pertenece).

7.1.4. Estado de activación.

7.1.5. Aplicaciones (Aplicaciones autorizadas a ejecutarse en este punto).

7.2. Modificar datos de un Punto.

7.2.1. Descripción.

7.2.2. Número IP.

7.2.3. Oficina (Oficina donde pertenece).

7.2.4. Estado de activación.

7.2.5. Aplicaciones (Aplicaciones autorizadas a ejecutarse en este punto).

7.3. Eliminar un Punto.

8. Gestionar Sitios de interés

8.1. Insertar datos del sitio de interés.

8.1.1. Nombre.

8.1.2. Dirección.

8.1.3. Parroquia.

8.1.4. Números de Fax.

8.1.5. Teléfonos.

8.2. Modificar datos del sitio de interés.

8.2.1. Dirección.

8.2.2. Parroquia.

8.2.3. Números de Fax.

8.2.4. Teléfonos.

8.3. Eliminar sitio de interés.

Buscar Información de Entidades

9. Buscar Información de Entidades

9.1. Realizar búsquedas de información de las entidades del sistema:

9.1.1. Las entidades son: oficinas, puntos, usuarios, aplicaciones y lugares de interés.

9.1.2. El criterio de búsqueda puede ser simple (un solo dato de la entidad) o compuesto (varios datos de la entidad).

9.1.3. Algunos datos de las entidades no pueden ser utilizados como criterio de búsqueda. Para cada caso pueden utilizarse.

9.1.3.1. Oficina: nombre de la oficina, parroquia, tipo de oficina y estado de activación.

9.1.3.2. Usuarios: nombre de usuario, oficina a la que pertenece, rol y estado de activación.

9.1.3.3. Puntos: numero IP, oficina a la que pertenece y aplicaciones autorizadas a ejecutarse.

9.1.3.4. Aplicaciones: nombre de la aplicación.

9.1.3.5. Sitios de interés: por definir.

Configuración General del Sistema Informático del Centro 171

10. Configurar el Sistema Informático del Centro 171

10.1. Asignar usuario o grupo de usuarios a determinado punto.

10.2. Asignar usuario o grupo de usuarios a determinada aplicación.

Generación de Reportes Estadísticos

11. Generar Reportes Estadísticos

11.1. Generar Reporte de Excepciones.

11.1.1. Usuario.

11.1.2. Aplicación.

11.1.3. Fecha.

11.1.4. Tipo de Excepción.

11.1.5. Punto.

11.1.6. Oficina.

11.2. Generar Reporte de Actividad del Usuario.

11.2.1. Usuario.

11.2.2. Aplicación.

11.2.3. Acción.

11.2.4. Punto.

11.2.5. Oficina.

Requerimientos no funcionales

Lograr cumplir las expectativas del usuario, implica, además de lograr que el software funcione, que lo haga de la mejor manera posible en base a los requisitos no funcionales que especifique el mismo; el presente trabajo se centró en:

Apariencia o interfaz externa: El producto debe tener un diseño sencillo, permitiendo la utilización del sistema sin necesidad de mucho conocimiento informático.

- Las ventanas del sistema contendrán claro y bien estructurados los datos, al mismo tiempo permitirán la interpretación correcta e inequívoca de la información.
- La interfaz contará con teclas de función, teclas de atajo y menús desplegables que faciliten y aceleren su utilización.
- Todos los textos y mensajes en pantalla aparecerán en idioma español.
- Su funcionamiento será intuitivo y requerirá de información mínima.

Soporte: Garantía de instalación y prueba del sistema, además de un breve entrenamiento a los futuros usuarios.

Usabilidad: Para utilizar el sistema es necesario poseer conocimientos elementales de computación y conceptos propios de la información que se visualiza.

Seguridad y privacidad:

- Se mantendrá seguridad y control a nivel de usuarios y contraseñas, garantizando el acceso de los mismos sólo a los niveles establecidos de acuerdo a la función que realizan. Las contraseñas sólo podrán ser cambiadas por el administrador del sistema.

- Se mantendrá un segundo nivel de seguridad a nivel de estación de trabajo, garantizándose la ejecución sólo de las aplicaciones que hayan sido establecidas para la estación en cuestión.
- El sistema permitirá registrar todas las acciones que se realizan, llevando el registro de las actividades de cada usuario en todo momento.
- La transmisión de datos por la red es a través de un protocolo seguro, TCP/IP.
- Se utilizan transacciones para el trabajo con los datos, garantizan la integridad de los mismos y se verifican las acciones irreversibles como las eliminaciones.

Software: Para el adecuado funcionamiento del sistema, la computadora debe tener instalado alguno de los sistemas operativos y los programas que a continuación se muestran:

Tabla 2**:** Requisitos de software para el adecuado uso de la aplicación.

Software	**Requisitos**
Sistema Operativo	Windows XP Professional SP I Windows XP Professional SP II
Plataforma	Microsoft .NET Framework 1.1
Programas	Oracle 10g cliente

Hardware: Se requiere disponer, con vistas a que el sistema funcione como se tiene concebido, de una computadora con las características que se muestran en la tabla.

Tabla 3: Requisitos de hardware para el adecuado funcionamiento de la aplicación.

Componente de Hardware	**Requisitos**
Procesador	2.00 –GHz o superior Pentium IV
Memoria RAM	512 MB o más
Espacio libre en disco duro	100 MB o más
Tarjeta de Video	Cualquiera estándar que tenga como mínimo 8 MB, 32 MB simulados.
Monitor LCD	VGA, Súper VGA o LCD de 19” mínimo.

Descripción de la solución propuesta

La ausencia de procedimientos que garantizan atender con efectividad las demandas de emergencias formuladas por la población de la Gran Caracas, relacionadas con la policía, bomberos y salud pública; provocó que el Ministerio del Interior y Justicia planteara la necesidad de implantar un sistema que logre la coordinación de los diferentes órganos policiales e instituciones relacionadas con la seguridad ciudadana, para garantizar una mejor y eficaz respuesta a las situaciones de emergencia que puedan ocurrir en el área que abarca la Gran Caracas. A este sistema se le denominó **Centro de Gestión de Emergencias y Seguridad Ciudadana (171)**.

Este centro tiene como misión implementar un sistema con alto grado de automatización, eficiencia y profesionalidad, que brinde soluciones efectivas a problemáticas de la población e integre diferentes subsistemas de cómputo, telefonía, radio e información operativa; que en su conjunto, lo convierten en un sistema integral de emergencia y seguridad ciudadana.

El sistema informático, fue dividido en varios módulos por su complejidad y debido a la necesidad del mismo. Gestionar la información de las entidades manipuladas por el sistema, configurar las funcionalidades del mismo así como garantizar la adaptabilidad que tiene que poseer ante la ocurrencia de situaciones excepcionales, se presentan como tareas a resolver y surge para ello el módulo de administración.

Funcionamiento del módulo de administración

El Módulo de Administración del sistema informático del Centro de Gestión de Emergencias y Seguridad Ciudadana (171), es el encargado de gestionar la información de todas las entidades que se manipulan en el sistema. Entiéndase por gestionar, la inserción, modificación y eliminación de todos los datos de las entidades. Las entidades son:

- Oficinas (locales donde radican las diferentes dependencias del centro de emergencia).
- Usuarios (trabajadores de cada una de las oficinas: supervisores, despachadores y operadores).
- Aplicaciones (cada uno de los módulos que conforman el sistema informático del centro de emergencias).
- Puntos (cada una de las estaciones de trabajo ubicadas en las oficinas).

- Sitios de interés (lugares considerados estratégicos en los estados venezolanos y que pueden servir para atender una emergencia)

Este módulo es el encargado de configurar el sistema y sus funcionalidades, permitiendo con ello el buen funcionamiento del mismo y la adaptabilidad necesaria en el Centro 171 ante situaciones excepcionales. Poder adaptarse al entorno que se requiera, es de vital importancia; el sistema tiene que reconfigurarse, sin que deje de funcionar, de forma tal que se aprovechen todos los recursos disponibles del centro, con el objetivo de alcanzar la mayor rapidez de respuesta posible. Una pequeña demora en un centro de gestión de emergencias y seguridad ciudadana puede conllevar a pérdidas tanto materiales como de vidas humanas, por lo que la rapidez y vitalidad del sistema es un requisito imprescindible.

Modelo del Sistema

Modelo de Casos de Uso del Sistema

Los actores representan los usuarios del sistema y otras aplicaciones que interactúan con él, es decir, representan terceros fuera del sistema que guardan relación con el mismo. Estos suelen corresponderse con trabajadores o actores del negocio. Debido a las características particulares de los sistemas de administración generalmente ocurre que solamente tienen un actor, el administrador. Los casos de usos definidos para este modulo se relacionan exclusivamente con este actor.

Tabla 4: Actores en módulos de administración.

Actores	Justificación
Administrador	Es el encargado de gestionar toda la información relacionada con las oficinas, usuarios, aplicaciones, puntos y sitios de interés, así como configurar el sistema definiendo de esta manera el funcionamiento del mismo. Puede además buscar cualquier información sobre estas entidades por diferentes criterios de búsqueda.

Para cumplir con los requerimientos se definen los siguientes casos de uso:

Tabla 5: Casos de Uso.

Nombre CU	Propósito	RF
1. Iniciar Aplicación.	Iniciar el subsistema de administración y mostrar el formulario principal.	R1
2. Activar Aplicación.	Activar la aplicación en la base de datos y en el registro del sistema.	R2
3. Autenticar Usuario.	Permitir el acceso al sistema de administración, determinando del usuario autenticado el rol que le corresponde.	R3
4. Gestionar Oficinas.	Permite al administrador insertar o modificar información relacionada con una oficina.	R4
5. Gestionar Usuarios.	Permite al administrador insertar, modificar, establecer contraseña o eliminar información relacionada con un usuario.	R5
6. Gestionar Aplicación.	Permite al administrador insertar, modificar o eliminar información relacionada con una aplicación.	R6
7. Gestionar Puntos.	Permite al administrador insertar, modificar o eliminar información relacionada con un punto.	R7
8. Gestionar Sitios de Interés.	Permite al administrador insertar, modificar o eliminar información relacionada con un sitio de interés.	R8
9. Buscar Información Entidades.	Permite al administrador buscar información de interés asociada a una oficina, un usuario, una aplicación o un punto. En cada caso se pueden utilizar varios criterios de búsqueda.	R9
10. Configurar el Sistema Informático del Centro 171.	Realizar la configuración del sistema, definiendo la forma de operar en él.	R10

11. Generar Reportes Estadísticos.	Generar reportes de excepciones y actividades del usuario.	R11

Diagrama de Casos de Uso

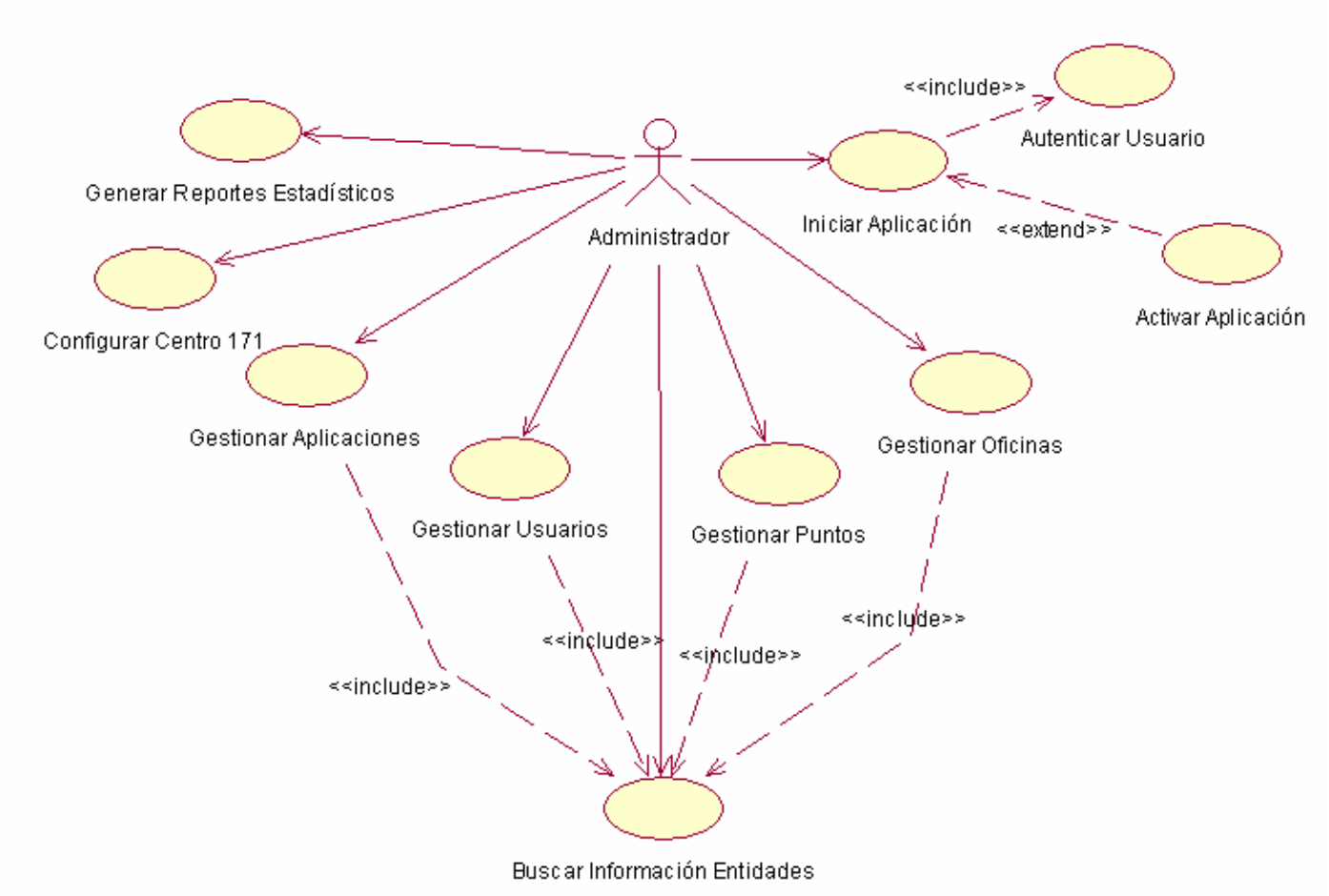


Diagrama 2: Modelo de Casos de Uso del Módulo de Administración

Basado en las características de RUP, se hizo una división de los casos de uso en 2 ciclos. A continuación se muestra una tabla con el nombre de cada caso de uso, el ciclo al que corresponde y la justificación de cada uno.

Tabla 6: Casos de Uso por ciclo.

Nombre de Caso de Uso	**Ciclo**	**Justificación de la selección**
Iniciar Aplicación.	1	Estos casos de uso se encuentran implicados en el primer ciclo de desarrollo porque en su conjunto permiten realizar la configuración básica del sistema, imprescindible para el funcionamiento del mismo.
Autenticar Usuario.		
Activar Aplicación.		
Gestionar Oficinas.		
Gestionar Usuarios.		
Gestionar Puntos.		
Gestionar Aplicaciones.		
Buscar Información de Entidades.		
Configurar.	2	En este ciclo es dedicado a la configuración del sistema en general, a la gestión de los datos que son funcionalmente de gran importancia dentro de un centro de gestión de emergencias y seguridad ciudadana, pero que no le son
Gestionar Sitios de Interés.		

Generar Reportes Estadísticos.		imprescindibles a la hora de iniciar su funcionamiento, así como a la generación de reportes estadísticos.

Casos de Uso expandidos

Mediante los casos de uso expandidos se describe paso a paso la secuencia de eventos que los actores utilizan para completar un proceso a través del sistema. A continuación se describen en formato expandido los casos de uso antes definidos:

Caso de uso Iniciar Aplicación

Tabla 7: Caso de uso, Iniciar Aplicación.

Caso de uso		
CU -1 Iniciar Aplicación		
Propósito	Iniciar el subsistema de administración y mostrar el formulario principal al administrador.	
Actores: Administrador (inicia)		
Resumen: El CU se inicia cuando el administrador va a utilizar el subsistema de administración, para ello se necesita tener la aplicación instalada y autenticarse. El sistema determina si realmente es un administrador; en caso que lo sea, se ejecuta la aplicación de administración mostrándole todas las funcionalidades del mismo.		
Referencias:	**R1**	
Acción del actor:		**Respuesta del sistema:**
1 - El usuario accede al sistema.		1.1 - El sistema determina que la aplicación está activada en ese punto.
		1.2 – El sistema realiza el caso de uso

	Autenticar.
Flujo alternativo	
Acción 1.1	El sistema determina que la aplicación no está activada en ese punto. El sistema realiza el caso de uso, Activar Aplicación. Ir a la acción 1.2.

Caso de uso Activar Aplicación

Tabla 8: Caso de uso, Activar Aplicación.

Caso de uso	
CU -2 Activar Aplicación	
Propósito	Activar la aplicación en la base de datos y en el registro del sistema.
Actores: Usuario	
Resumen: El caso de uso se inicia cuando se va a utilizar la aplicación por primera vez y esta no ha sido registrada. Para activar la aplicación es necesario ser un usuario con privilegios de administración en el sistema.	
Referencias:	**R2**
Acción del actor:	**Respuesta del sistema:**
	1. – El sistema muestra el formulario Activar Aplicación solicitando al usuario la información necesaria: - Número de Serie de la Oficina. - Usuario. - Contraseña. - Servidor.

2 – El usuario proporciona la información solicitada y presiona el botón Validar.	2.1 – El sistema verifica que los datos sean correctos.
	2.2 – El sistema muestra el nombre de la oficina asociada al número de serie proporcionado, la descripción del punto.
3 – El usuario consulta la información proporcionada y presiona el botón Aceptar.	3.1 – El sistema activa la aplicación en la base de datos y en el registro del sistema. Se cierra el formulario Activar Aplicación.
Flujo alternativo	
Acción 2.1	El sistema verifica que los datos no son correctos, muestra un mensaje de error.
Acción 3	El usuario consulta la información proporcionada y presiona el botón Cancelar. El sistema no realiza ninguna acción y cierra el formulario Activar Aplicación.

Caso de uso Autenticar Usuario

Tabla 9: Caso de uso, Autenticar Usuario.

Caso de uso	
CU -3 Autenticar Usuario	
Propósito	Permitir el acceso al módulo de administración, otorgándole privilegios de acceso a sus funcionalidades sólo a los administradores del sistema.
Actores: Usuario	
Resumen: El caso de uso se inicia cuando un posible administrador introduce en el sistema su usuario y contraseña, el sistema verifica la validez de estos datos para determinar si realmente es un administrador; en caso que lo sea, se ejecuta la aplicación de administración	

mostrándole todas las funcionalidades del mismo.	
Referencias:	**R3**
Acción del actor:	**Respuesta del sistema:**
	1 - El sistema muestra el formulario Autenticar Usuario solicitando nombre de usuario y contraseña.
2 - El administrador introduce los datos solicitados.	
3 - El administrador presiona el botón Aceptar.	3.1 - El sistema verifica que los datos sean correctos determinando si realmente presenta rol de administrador.
Flujo alternativo	
Acción 3	El administrador presiona el botón Cancelar.
Acción 3.1	El sistema verifica que los datos son incorrectos, muestra un mensaje de error y cuenta el intento de autenticación. Al llegar a tres se cancela la acción y se envía una notificación al módulo de administración indicando que se ha intentado violar la seguridad. . Pasa directamente a la acción 2.

Caso de uso Gestionar Oficinas

Tabla 10: Caso de uso, Gestionar Oficinas.

Caso de uso	
CU -4 Gestionar Oficinas.	
Propósito	Brindar al administrador la posibilidad de insertar o modificar los datos referentes a una oficina.

<table>
<tr><td colspan="2">Actores: Administrador (inicia)</td></tr>
<tr><td colspan="2">Resumen: El caso de uso inicia cuando un administrador necesita adicionar datos de una nueva oficina o modificar los de una oficina ya existente. Si selecciona adicionar una oficina el sistema solicita todos los datos, verifica que tengan el formato correcto y los almacena. Si selecciona modificar una oficina el sistema solicita un criterio para buscar la oficina que se desea; solicita los datos a modificar, los verifica y almacena si todo es correcto.</td></tr>
<tr><td>Referencias:</td><td>R4</td></tr>
<tr><td>Acción del actor:</td><td>Respuesta del sistema:</td></tr>
<tr><td>1. El administrador selecciona el grupo de acciones Oficina en el menú, indicando que quiere operar sobre ellas.</td><td>1.1 El sistema despliega en el menú las opciones referentes a las operaciones que pueden realizarse sobre las oficinas del sistema.
a) Adicionar oficina (Ir a sección Adicionar Oficina).
b) Modificar oficina (Ir a sección Modificar Oficina).</td></tr>
<tr><td colspan="2">Sección Adicionar Oficina.</td></tr>
<tr><td>2. El administrador selecciona la opción Adicionar Oficina.</td><td>2.1 El sistema muestra un formulario solicitando la información necesaria para insertar la oficina:
a) Nombre de la Oficina.
b) Dirección.
c) Parroquia.
d) Oficina a la que pertenece.
e) Estado de activación.
f) Números de Fax.
g) Teléfonos.</td></tr>
</table>

3. El administrador introduce los datos solicitados.	
4. El administrador presiona el botón Adicionar.	4.1 El sistema verifica que los datos proporcionados sean correctos.
	4.2 El sistema almacena la información en la base de datos.
Flujo alternativo.	
Acción 4.1	El sistema verifica que los datos no son correctos, algunos no cumplen con el formato adecuado. Se indica donde está el error para que sea corregido. Retorna a 3
Acción 4.2	El sistema no pudo almacenar los nuevos datos. Se muestra una notificación indicando el error. Retorna a 4
Sección Modificar Oficina.	
2. El administrador selecciona la opción Modificar Oficina.	2.1 El sistema realiza el caso de uso **Buscar Información Entidades** (incluido).
3. El administrador localiza la oficina cuyos datos desea cambiar.	3.1 El sistema muestra toda la información correspondiente a la oficina seleccionada, permitiéndole al administrador corroborar que realmente es la oficina deseada (Aquí no es posible cambiar ninguno de los valores, sólo consultarlos.
4. El administrador presiona el botón Modificar.	4.1 El sistema muestra, de forma editable, todos los campos de la oficina que pueden ser modificados. (En este caso el único dato que no se puede modificar es el serial de activación.)
5. El administrador introduce la nueva	

información.	
6. El administrador presiona el botón Aceptar.	6.1 El sistema verifica que los datos proporcionados sean correctos.
	6.2 El sistema almacena la nueva información, de la oficina seleccionada, en la base de datos.
Flujo alternativo.	
Acción 6	El administrador presiona el botón Cancelar.
	El sistema muestra los resultados de la búsqueda. Retorna a 3
Acción 6.1	El sistema verifica que los datos no son correctos, algunos no cumplen con el formato adecuado. Se indica donde está el error para que sea corregido. Retorna a 5
Acción 6.2	El sistema no pudo almacenar los nuevos datos. Se muestra una notificación indicando el error. Retornar a 6

Caso de uso Gestionar Usuarios

Tabla 11: Caso de uso, Gestionar Usuarios.

Caso de uso	
CU -5 Gestionar Usuarios.	
Propósito	Brindar al administrador la posibilidad de insertar, modificar, establecer

	contraseña o eliminar los datos referentes a un usuario.
Actores: Administrador (inicia)	
Resumen: El caso de uso inicia cuando un administrador necesita insertar datos de un nuevo usuario, modificar datos, establecer contraseña o eliminar usuarios existentes. Si selecciona adicionar usuario, el sistema solicita todos los datos, verifica que tengan el formato correcto, los almacena y actualiza los usuarios en el servidor de bases de datos. Si selecciona modificar un usuario, el sistema solicita un criterio para buscar el usuario que se desea modificar; solicita los datos a cambiar, los verifica y almacena si todo es correcto. Si se selecciona establecer contraseña, el sistema solicita un criterio para buscar el usuario al que se le quiere establecer la nueva contraseña, solicita los datos para establecerla, verifica que estén correctos y si lo están actualiza la contraseña del usuario en el servidor de bases de datos. Si se selecciona eliminar, el sistema solicita un criterio para buscar el usuario, se selecciona el que se desea eliminar, se confirma y elimina de la base de datos y del servidor de bases de datos.	
Referencias:	**R5**

Acción del actor:	**Respuesta del sistema:**
1. El administrador selecciona el grupo de acciones Usuario en el menú.	1.2 El sistema despliega en el menú las opciones referentes a las operaciones sobre los usuarios del sistema. a) Adicionar usuario (Ir a sección Adicionar Usuario). b) Modificar usuario (Ir a sección Modificar Usuario). c) Establecer contraseña (Ir a sección Establecer Contraseña). d) Eliminar usuario (Ir a sección Eliminar Usuario).
Sección Adicionar Usuario.	

2. El administrador selecciona la opción Adicionar usuario del menú.	2.1 El sistema muestra un formulario solicitando la información necesaria para insertar el usuario: a) Nombre de Usuario. b) Contraseña. c) Confirmación de contraseña. d) Oficina a la que pertenece. e) Roles. f) Estado de activación.
3. El administrador introduce los datos solicitados.	
4. El administrador presiona el botón Adicionar.	4.1 El sistema verifica que los datos proporcionados sean correctos.
	4.2 En caso de estar correctos el sistema almacena la información en la base de datos y actualiza los usuarios del servidor de bases de datos.
Flujo alternativo.	
Acción 4.1	El sistema verifica que los datos no son correctos, algunos no cumplen con el formato adecuado. Se indica donde está el error para que sea corregido. Retorna a 3
Acción 4.2	El sistema no pudo almacenar los datos del usuario. Se muestra una notificación indicando el error. Retorna a 4
Sección Modificar Usuario.	

2. El administrador selecciona la opción Modificar usuario del menú.	2.1. El sistema realiza el caso de uso **Buscar Información Entidades** (incluido).
3. El administrador localiza el usuario cuyos datos desea cambiar.	3.1. El sistema muestra toda la información correspondiente al usuario seleccionado, permitiéndole al administrador corroborar que realmente es el usuario deseado (Aquí no es posible cambiar ninguno de los valores, sólo consultarlos).
4. El administrador presiona el botón Modificar.	4.1. El sistema muestra de forma editable todos los campos del usuario que pueden ser modificados. (En este caso los únicos datos que no pueden ser modificados son el nombre de usuario y la contraseña, esta última se modifica a través de la seccion Establecer Contraseña).
5. El administrador introduce la nueva información.	
6. El administrador presiona el botón Aceptar.	6.1. El sistema verifica que los datos proporcionados sean correctos.
	6.2. En caso de estar correctos el sistema almacena la nueva información en la base de datos.
Flujo alternativo.	
Acción 5	El administrador presiona el botón Cancelar.
	El sistema muestra los resultados de la búsqueda. Retorna a 3

Acción 6.1	El sistema verifica que los datos no son correctos, algunos no cumplen con el formato adecuado. Se indica donde está el error para que sea corregido. Retorna a 5
Acción 6.2	El sistema no pudo almacenar los nuevos datos. Se muestra una notificación indicando el error. Retornar a 6
Sección Establecer Contraseña.	
2. El administrador selecciona la opción Establecer contraseña.	2.1 El sistema realiza el caso de uso **Buscar Información Entidades** (incluido).
3. El administrador selecciona el usuario cuya contraseña desea cambiar.	3.1 El sistema muestra toda la información correspondiente al usuario seleccionado, permitiéndole al administrador corroborar que realmente es el usuario deseado (Aquí no es posible cambiar ninguno de los valores, sólo consultarlos).
	3.2 El sistema habilita dos campos editables uno para la nueva contraseña y otro para confirmarla.
4. El administrador introduce el nuevo valor de la contraseña en ambos campos.	
5. El administrador presiona el botón Aceptar.	5.1 El sistema verifica si no están en blanco y si no representan la misma cadena.
	5.2 Si esto ocurre se muestra un mensaje solicitando la confirmación para cambiarle la

	contraseña al usuario.
6. El administrador confirma que desea reemplazar la contraseña del usuario.	6.1 El sistema actualiza la contraseña del usuario en el servidor de bases de datos.
Flujo alternativo.	
Acción 5.1	El sistema verifica que la contraseña está en blanco, si esto ocurre se muestra un mensaje de error aclarando que debe teclear una contraseña. Retorna a 4
Acción 5.1	El sistema verifica que la contraseña y la confirmación no contengan la misma cadena si esto ocurre se muestra un mensaje de error aclarando que no pueden ser diferentes. Retorna a 4
Acción 6	El administrador confirma que no desea cambiar la contraseña del usuario. Retorna a 3
Acción 6.1	El sistema no pudo almacenar los nuevos datos. Se muestra una notificación indicando el error. Retornar a 5
Sección Eliminar Usuario	
2. El administrador selecciona la opción Eliminar usuario del menú.	2.1 El sistema realiza el caso de uso **Buscar Información Entidades** (incluido).
3. El administrador localiza el usuario que desea eliminar como usuario del sistema.	3.1 El sistema muestra toda la información correspondiente al usuario seleccionado, permitiéndole al administrador corroborar que

	realmente es el usuario deseado (Aquí no es posible cambiar ninguno de los valores, sólo consultarlos).
4. El administrador presiona el botón Eliminar.	4.1 Se muestra un mensaje solicitando confirmación para la eliminación.
5. El administrador confirma la eliminación.	5.1 El sistema elimina los datos del usuario seleccionado, en el servidor de bases de datos, se marca como eliminado en la base de datos y se almacena un historial del mismo.
	5.2 Se muestra el resultado de la búsqueda sin los datos de este usuario.
Flujo alternativo.	
Acción 5.1	El sistema no pudo eliminar los datos del usuario.
	El sistema muestra una notificación indicando el error. Retorna a 4

Caso de uso Gestionar Aplicaciones

Tabla 12: Caso de uso, Gestionar Aplicaciones.

Caso de uso	
CU -6 Gestionar Aplicaciones.	
Propósito	Brindar al administrador la posibilidad de insertar, modificar o eliminar los datos referentes a una aplicación.
Actores: Administrador (inicia)	
Resumen: El caso de uso inicia cuando un administrador necesita insertar datos de una	

<table>
<tr><td colspan="2">nueva aplicación, modificar datos de una ya existente o eliminarla. Si selecciona adicionar aplicación, el sistema solicita todos los datos, verifica que tengan el formato correcto y los almacena. Si selecciona modificar una aplicación, el sistema solicita un criterio para buscar la aplicación que se quiere modificar; solicitan los datos que se desean cambiar, los verifica y almacena si todo es correcto. Si se selecciona eliminar, el sistema solicita un criterio para buscar la aplicación, el administrador selecciona la que desea eliminar, se pide confirmación y elimina la aplicación.</td></tr>
<tr><td>Referencias:</td><td>R6</td></tr>
<tr><td>Acción del actor:</td><td>Respuesta del sistema:</td></tr>
<tr><td>1 El administrador selecciona el grupo de acciones, Aplicación.</td><td>1.1 El sistema despliega en el menú las opciones referentes a las operaciones sobre las aplicaciones del sistema.
a) Adicionar aplicación (Ir a sección Adicionar Aplicación).
b) Modificar Aplicación (Ir a sección Modificar Aplicación).
c) Eliminar Aplicación (Ir a sección Eliminar Aplicación).</td></tr>
<tr><td colspan="2">Sección Adicionar Aplicación.</td></tr>
<tr><td>2 El administrador selecciona la opción Adicionar aplicación del menú.</td><td>2.1 El sistema muestra un formulario solicitando la información necesaria para insertar la aplicación:
a) Nombre de aplicación.</td></tr>
<tr><td>3 El administrador introduce los datos solicitados.</td><td></td></tr>
<tr><td>4 El administrador presiona el botón</td><td>4.1 El sistema verifica que los datos proporcionados sean correctos.</td></tr>
</table>

Adicionar.	4.2 En caso de estar correctos el sistema almacena la información en la base de datos.
Flujo alternativo.	
Acción 4.1	El sistema verifica que los datos no son correctos, algunos no cumplen con el formato adecuado. Se indica donde está el error para que sea corregido. Retorna a 3
Acción 4.2	El sistema no pudo almacenar los nuevos datos. Se muestra una notificación indicando el error. Retorna a 4
Sección Modificar Aplicación.	
2. El administrador selecciona la opción Modificar aplicación del menú.	2.1. El sistema realiza el caso de uso **Buscar Información Entidades** (incluido).
3. El administrador selecciona la aplicación cuyos datos desea cambiar.	3.1. El sistema muestra toda la información correspondiente a la aplicación seleccionada, permitiéndole al administrador corroborar que realmente es la aplicación deseada (Aquí no es posible cambiar ninguno de los valores, sólo consultarlos).
4. El administrador presiona el botón Modificar.	4.1. El sistema muestra de forma editable todos los campos de la aplicación que pueden ser modificados.
5. El administrador introduce la nueva información.	
6. El administrador presiona el botón Aceptar.	6.1. El sistema verifica que los datos proporcionados sean correctos.
	6.2. En caso de estar correctos el sistema almacena la nueva información en la

	base de datos.
Flujo alternativo.	
Acción 6	El administrador presiona el botón Cancelar.
	El sistema muestra los resultados de la búsqueda. Retorna a 3
Acción 6.1	El sistema verifica que los datos no son correctos, algunos no cumplen con el formato adecuado. Se indica donde está el error para que sea corregido. Retorna a 5
Acción 6.2	El sistema no pudo almacenar los nuevos datos. Se muestra una notificación indicando el error. Retornar a 6
Sección Eliminar Aplicación.	
2. El administrador selecciona la opción Eliminar aplicación.	2.1. El sistema realiza el caso de uso **Buscar Información Entidades** (incluido).
3. El administrador selecciona la aplicación que desea eliminar del sistema.	3.1. El sistema muestra toda la información correspondiente a la aplicación seleccionada, permitiéndole al administrador corroborar que realmente es la aplicación deseada (Aquí no es posible cambiar ninguno de los valores, sólo consultarlos).
4. El administrador presiona el botón Eliminar.	4.1. Se muestra un mensaje solicitando confirmación para la eliminación.
5. El administrador confirma la	5.1. El sistema elimina los datos de la aplicación

eliminación.	seleccionada, en la base de datos.
	5.2. Se muestra el resultado de la búsqueda sin los datos de esta aplicación.
Flujo alternativo.	
Acción 5.1	El sistema no pudo eliminar los datos de la aplicación.
	El sistema muestra una notificación indicando el error. Retorna a 3

Caso de uso Gestionar Puntos

Tabla 13: Caso de uso, Gestionar Puntos.

Caso de uso	
CU -7 Gestionar Puntos.	
Propósito	Brindar al administrador la posibilidad de insertar, modificar o eliminar los datos referentes a un punto de red o estaciones de trabajo.
Actores: Administrador (inicia)	
Resumen: El caso de uso inicia cuando un administrador necesita insertar datos de un nuevo punto, modificar datos de uno existente o eliminarlo. Si selecciona adicionar punto, el sistema solicita todos los datos, verifica que tengan el formato correcto y los almacena. Si selecciona modificar un punto, el sistema solicita un criterio para buscar el punto que se quiere modificar; solicita los datos que se quieren cambiar, los verifica y almacena si todo es correcto. Si se selecciona eliminar, el sistema solicita un criterio para buscar el punto, se selecciona el que se desea eliminar, se confirma la eliminación y se elimina de la base de datos.	
Referencias:	**R7**
Acción del actor:	**Respuesta del sistema:**

1. El administrador selecciona el grupo de acciones Punto.	1.2 El sistema despliega en el menú las opciones referentes a las operaciones sobre los puntos del sistema. a) Adicionar Punto (Ir a sección Adicionar Punto). b) Modificar Punto (Ir a sección Modificar Punto). c) Eliminar Punto (Ir a sección Eliminar Punto)
Sección Adicionar Punto.	
2. El administrador selecciona la opción Adicionar punto del menú.	2.1 El sistema muestra un formulario solicitando la información necesaria para insertar el punto: a) Descripción. b) Número IP. c) Aplicaciones (Aplicaciones que se permiten ejecutar en el punto). d) Oficina (Oficina a la que pertenece el punto). e) Estado Activación.
3. El administrador introduce los datos solicitados.	
4. El administrador presiona el botón Adicionar.	4.1 El sistema verifica que los datos proporcionados sean correctos.
	4.2 En caso de estar correctos el sistema almacena la información en la base de datos.
Flujo alternativo.	

Acción 4.1	El sistema verifica que los datos no son correctos, algunos no cumplen con el formato adecuado. Se indica donde está el error para que sea corregido. Retorna a 3
Acción 4.2	El sistema no pudo almacenar los nuevos datos. Se muestra una notificación indicando el error. Retorna a 4
Sección Modificar Punto.	
2. El administrador selecciona la opción Modificar punto.	2.1. El sistema realiza el caso de uso **Buscar Información Entidades** (incluido).
3. El administrador selecciona el punto cuyos datos desea cambiar.	3.1. El sistema muestra toda la información correspondiente al punto seleccionado, permitiéndole al administrador corroborar que realmente es el punto deseado (Aquí no es posible cambiar ninguno de los valores, sólo consultarlos).
4. El administrador presiona el botón Modificar.	4.1. El sistema muestra de forma editable todos los campos del punto que pueden ser modificados.
5. El administrador introduce la nueva información.	
6. El administrador presiona el botón Aceptar.	6.1. El sistema verifica que los datos proporcionados sean correctos.
	6.2. En caso de estar correctos el sistema almacena la nueva información en la base de datos.
Flujo alternativo.	
Acción 6	El administrador presiona el botón Cancelar.

	El sistema muestra los resultados de la búsqueda. Retorna a 3
Acción 6.1	El sistema verifica que los datos no son correctos, algunos no cumplen con el formato adecuado. Se indica donde está el error para que sea corregido. Retorna a 5
Acción 6.2	El sistema no pudo almacenar los nuevos datos. Se muestra una notificación indicando el error. Retornar a 6
Sección Eliminar Punto.	
2. El administrador selecciona la opción Eliminar punto.	2.1. El sistema realiza el caso de uso **Buscar Información Entidades** (incluido).
3. El administrador selecciona el punto que desea eliminar.	3.1. El sistema muestra toda la información correspondiente al punto seleccionado, permitiéndole al administrador corroborar que realmente es el punto deseado (Aquí no es posible cambiar ninguno de los valores, sólo consultarlos
4. El administrador presiona el botón Eliminar.	4.1. Se muestra un mensaje solicitando confirmación para la eliminación.
5. El administrador confirma la eliminación.	5.1. El sistema elimina los datos del punto seleccionado, en la base de datos.
	4. Se muestra el resultado de la búsqueda sin los datos de este punto.

Flujo alternativo.	
Acción 4.1	El sistema no pudo eliminar los datos del punto.
	El sistema muestra una notificación indicando el error. Retorna a 3

Caso de uso Buscar Información de Entidades

Tabla 14: Caso de uso, Buscar Información de Entidades.

Caso de uso	
CU -8 Buscar Información de Entidades.	
Propósito	Mostrarle al administrador la información de las entidades del sistema filtradas por los diferentes parámetros de búsqueda.
Actores: Administrador (inicia)	
Resumen: El caso de uso comienza cuando el administrador necesita obtener la información de alguna de las entidades del sistema, Oficina, Usuario, Aplicación y Punto, ya sea seleccionando la opción buscar en el menú principal o desde una acción que necesite la búsqueda de entidades para su ejecución. La búsqueda se puede realizar teniendo en cuenta uno de los datos de la entidad o mediante una combinación de varios de ellos. El sistema realiza la búsqueda y muestra el resultado.	
Referencias:	**R9**
Acción del actor:	**Respuesta del sistema:**
1. El administrador necesita obtener información relacionada a alguna de las entidades. La entidad es seleccionada por el administrador desde el menú principal o determinada por una acción de	

modificación o eliminación que se quiere realizar sobre la misma. La búsqueda puede realizarse a: a) Oficinas (Ir a Sección Buscar Oficinas). b) Usuarios (Ir a Sección Buscar Usuarios). c) Puntos (Ir a Sección Buscar Puntos). d) Aplicaciones (Ir a Sección Buscar Aplicaciones).	
Sección Buscar Oficina.	
2. El administrador selecciona la opción Buscar oficina.	2.1 Se muestra un formulario solicitando un criterio de búsqueda, este puede ser una o varias características de la oficina: a) Nombre de la oficina. b) Parroquia. c) Estado de activación.
3. El administrador introduce los datos solicitados o sólo llena los campos que son de su interés para la búsqueda.	
4. Presiona el botón Buscar.	4.1 Se validan los datos proporcionados verificando que todo es correcto.
	4.2 Se busca la información solicitada y se elabora una lista de todas las oficinas que cumplen con el criterio de búsqueda. De cada oficina se muestran sólo los datos necesarios para identificarla inequívocamente: a) Nombre de la oficina.

	b) Dirección. c) Teléfono. d) Número de Fax.
Sección Buscar Usuarios.	
1. El administrador selecciona la opción Buscar usuario.	1.1 Se muestra un formulario solicitando un criterio de búsqueda, este puede ser una o varias características del usuario: a) Nombre de usuario. b) Oficina. c) Estado de activación.
2. El administrador introduce los datos solicitados o sólo llena los campos que son de su interés para la búsqueda.	
3. Presiona el botón Buscar.	3.1 Se validan los datos proporcionados verificando que todo es correcto.
	3.2 Se busca la información solicitada y se elabora una lista de los usuarios que cumplen con el criterio de búsqueda. De cada uno se muestran sólo los datos necesarios para identificarlo inequívocamente. a) Nombre de usuario.
Sección Buscar Puntos.	
2. El administrador selecciona la opción Buscar Punto.	2.1 Se muestra un formulario solicitando un criterio de búsqueda, este puede ser una o varias características del punto: a) Número IP. b) Oficina. c) Estado de activación.
3. El administrador introduce los datos	

solicitados o sólo llena los campos que son de su interés para la búsqueda.	
4. Presiona el botón Buscar.	4.1 Se validan los datos proporcionados verificando que todo es correcto.
	4.2 Se busca la información solicitada y se elabora una lista de los puntos que cumplen con el criterio de búsqueda. De cada uno se muestran sólo los datos necesarios para identificarlo inequívocamente. a) Número IP. b) Descripción.
Sección Buscar Aplicaciones.	
2. El administrador selecciona la opción Buscar Aplicaciones.	2.1 Se muestra un formulario solicitando un criterio de búsqueda, este puede ser una o varias características de la aplicación: a) Nombre de la aplicación.
3. El administrador introduce los datos solicitados.	
4. Presiona el botón Buscar.	4.1 Se validan los datos proporcionados verificando que todo es correcto.
	4.2 Se busca la información solicitada y se elabora una lista de todas las aplicaciones que cumplen con el criterio de búsqueda. De cada una se muestran sólo los datos necesarios para identificarla inequívocamente: a) Nombre de la aplicación.

Conclusiones

El capítulo muestra las peculiaridades y características que tiene que presentar el software para cumplir los requerimientos especificados por el cliente. Para describir el contexto en que se desenvuelve el módulo administrativo se realiza el modelo del dominio y una descripción de la solución propuesta. Se definieron los requisitos funcionales y no funcionales que debe cumplir, se elaboró el diagrama de casos de uso del sistema, con su descripción y ciclo al que pertenece y se realizó la descripción expandida de cada uno.

Capítulo III: Construcción de la solución propuesta

Introducción

Este capítulo muestra cómo se elaboró el módulo de administración en base a las condiciones específicas que debe cumplir el **Sistema Informático del Centro 171**, se describe la arquitectura de la aplicación, el diagrama de clases y el diseño de la base de datos utilizados.

También se definen los principios de diseño de la aplicación, las pautas a seguir para el diseño de la interfaz y la programación a realizar y el tratamiento a seguir para cada posible error; contribuyendo de esta forma a lograr un sistema más robusto, legible y de fácil comprensión.

Arquitectura propuesta para la implementación del Sistema Informático del Centro 171

La arquitectura del sistema informático del Centro 171 esta basada en dos características fundamentales. La primera es que presenta un modelo de desarrollo basado en casos de uso, y la segunda se refiere a que todo el ciclo de elaboración del sistema se ajustará a un modelo iterativo e incremental.

El sistema informático del Centro 171 está dividido en varios módulos: Módulo de Administración, Módulo de Mapificación de la Información, Módulo de Despacho, entre otros. Cada uno responde a un conjunto de funcionalidades específicas del cliente. Todos estos módulos interactúan entre sí y comparten datos de interés en dependencia de la funcionalidad de cada uno, siguen un estricto régimen de seguridad de la información y se desarrollan de forma paralela.

La arquitectura esta organizada desde dos enfoques, uno horizontal y otro vertical.

Enfoque horizontal

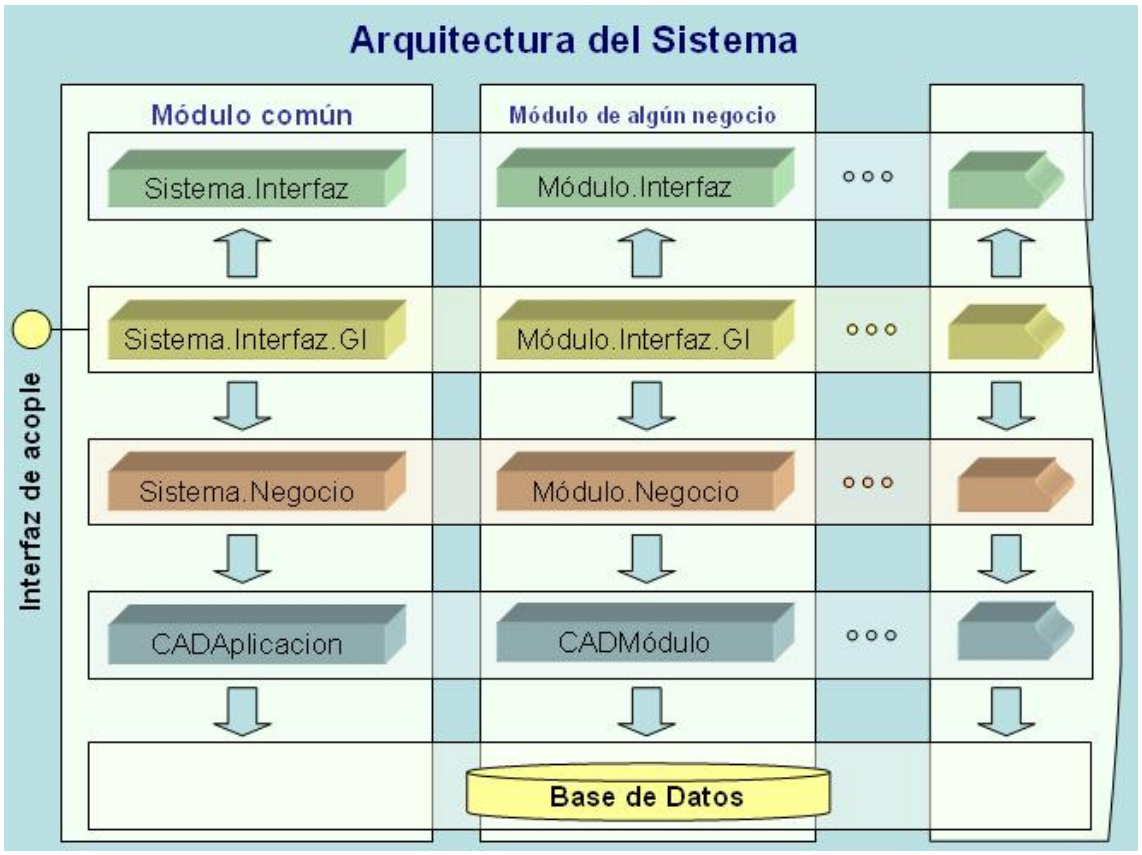
El desarrollo de cada módulo responde a un modelo multicapas (Véase Figura 2).

1. Capa interfaz.
2. Capa mediadora interfaz basada en acciones.

3. Capa lógica del negocio.
4. Capa acceso a datos.
5. Capa de datos.

A continuación se ilustra la arquitectura del sistema.

Figura 2: Arquitectura del Sistema.



Capa interfaz.

La capa de interfaz responde a un patrón estándar, todos los módulos tienen prácticamente el mismo diseño con vistas a facilitar el trabajo con ellos y la estandarización del sistema completo.

Capa mediadora interfaz basada en acciones.

Una acción es una clase que representa precisamente la ejecución de alguna tarea concreta. Estas tareas no deben ser excesivamente complejas y la clase debe:

- Heredar de la clase *Accion* o *AccionSegura.*
- Tener opcionalmente un formulario asociado.
- Atribuir propiedades en función del objetivo específico de la misma.
- Reescribir el método CrearForma con el objetivo de mostrar las características que se deseen en el formulario.
- En caso de que la acción tenga asignado un formulario, se deben programar dentro de la misma todos los eventos que puedan ocurrir a partir de la interacción con el usuario.

Capa lógica del negocio.

El diseño de esta capa depende directamente del negocio específico que se éste automatizando.

Es preciso se mantenga un estricto seguimiento de los documentos y artefactos de software definidos para la documentación del proceso, los cuales se explican más adelante.

Capa acceso a datos.

Esta capa esta compuesta por dos subcapas. La primera se corresponde con una capa compuesta por clases que constituyen factorías de las entidades del negocio que deben persistir. La segunda subcapa debe ser generada con la herramienta TierDeveloper versión 4.0 y debe garantizar todo el manejo de la información que requiera el negocio del problema en cuestión y la conexión con la base de datos.

Se debe garantizar desde el TierDeveloper que se genere toda la documentación apropiadamente, que se expliquen todos los métodos y las clases que se generan automáticamente.

Capa de datos.

Esta capa corresponde a los almacenes de datos. A ella pertenecen las bases de datos en Oracle. Esta capa es única y común a todos los módulos del sistema general. Se debe caracterizar porque el modelo de datos debe estar en al menos la tercera forma normal.

Enfoque Vertical

Cada modulo implementa sus funcionalidades y en caso que sea necesario se redefine aquella capa de la arquitectura que no se corresponda con su lógica de funcionamiento.

Diagrama de clases del diseño

Diagrama de clases de Paquete Servicios

Diagrama 3: Clases del Paquete Servicio.

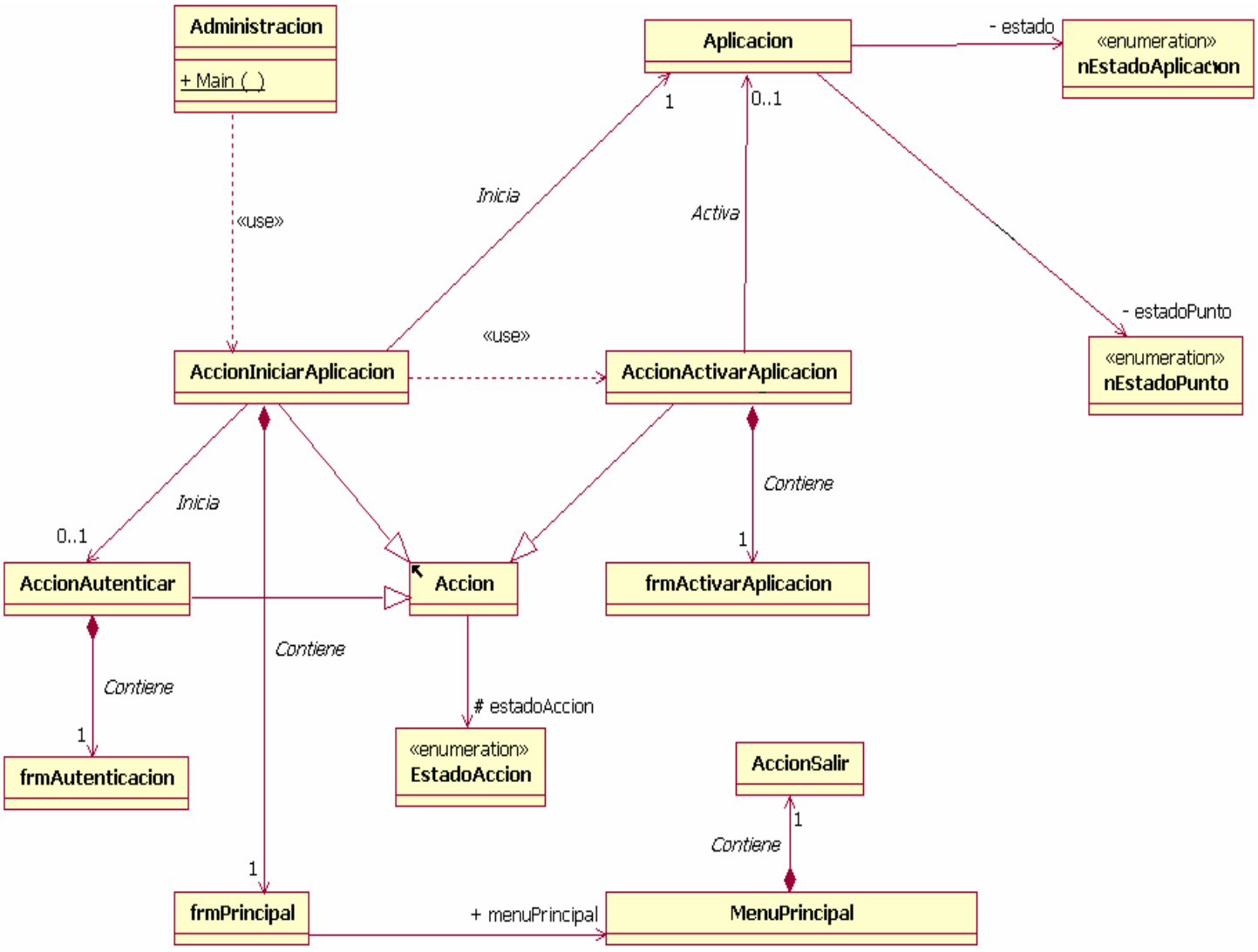


Diagrama de clases Paquete Control de Oficina

Diagrama 4: Clases del Paquete Control Oficina.

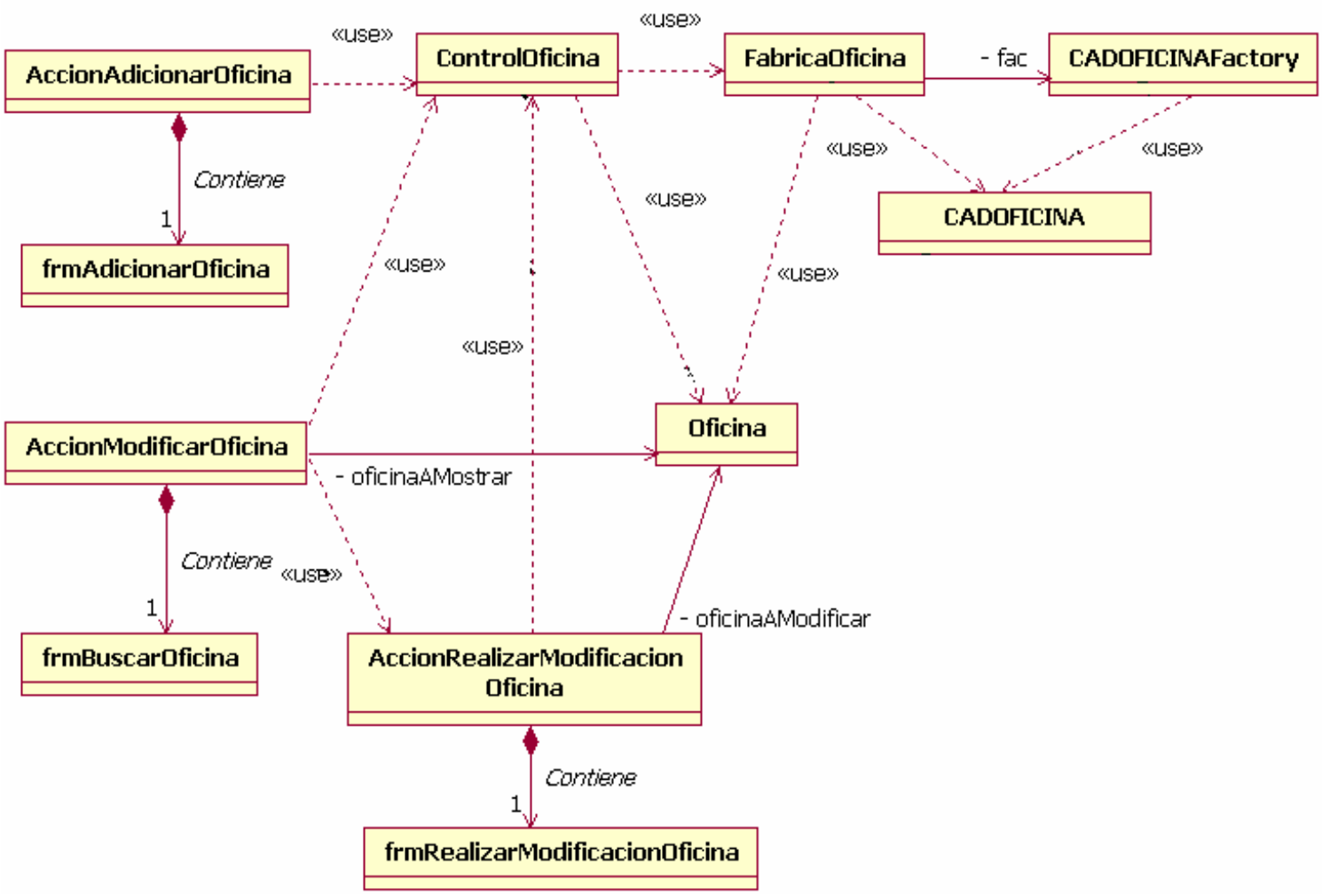


Diagrama de clases Paquete Control de Usuario

Diagrama 5: Clases del Paquete Control Usuario.

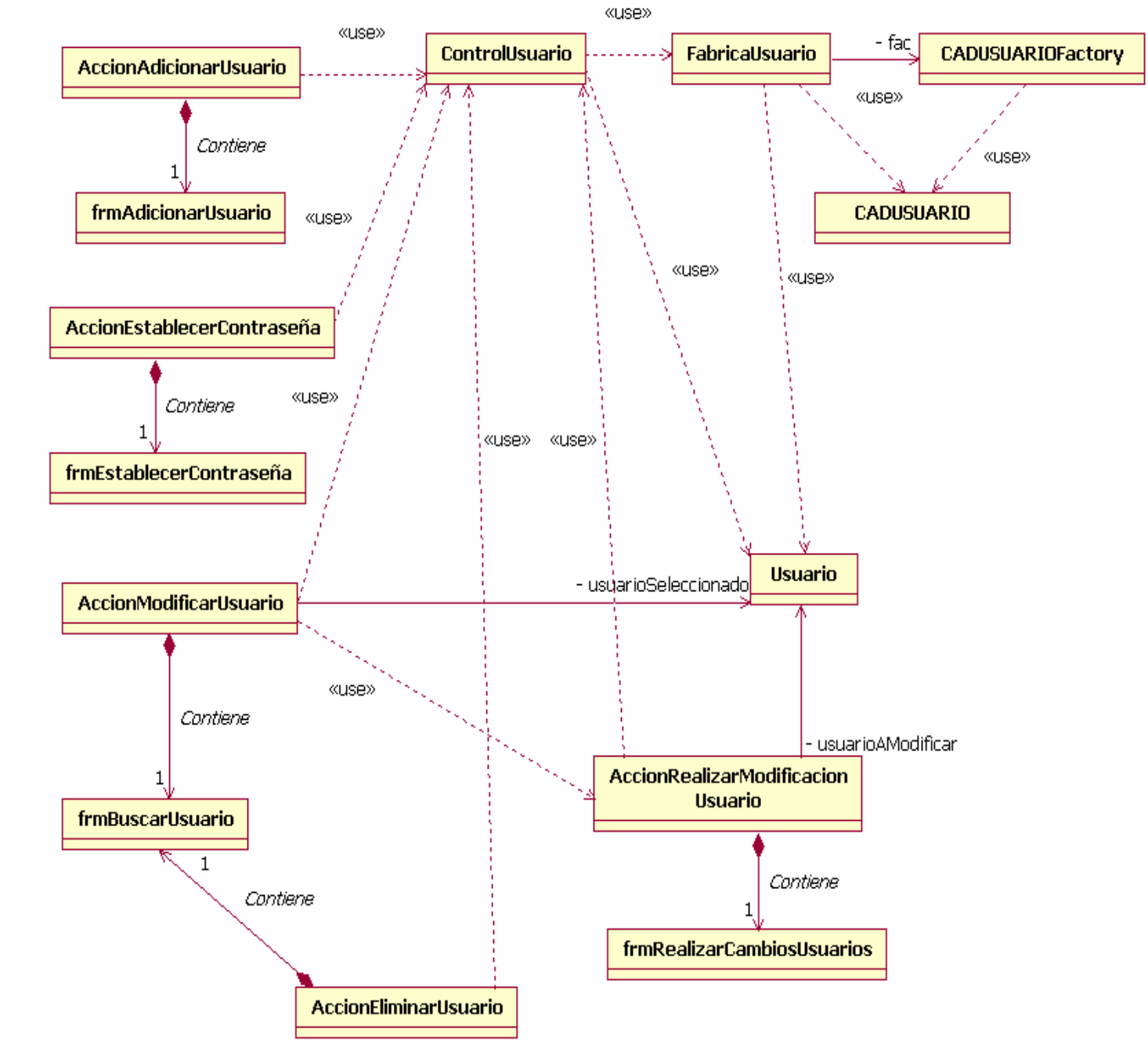


Diagrama de clases Paquete Control de Punto

Diagrama 6: Clases del Paquete Control Punto.

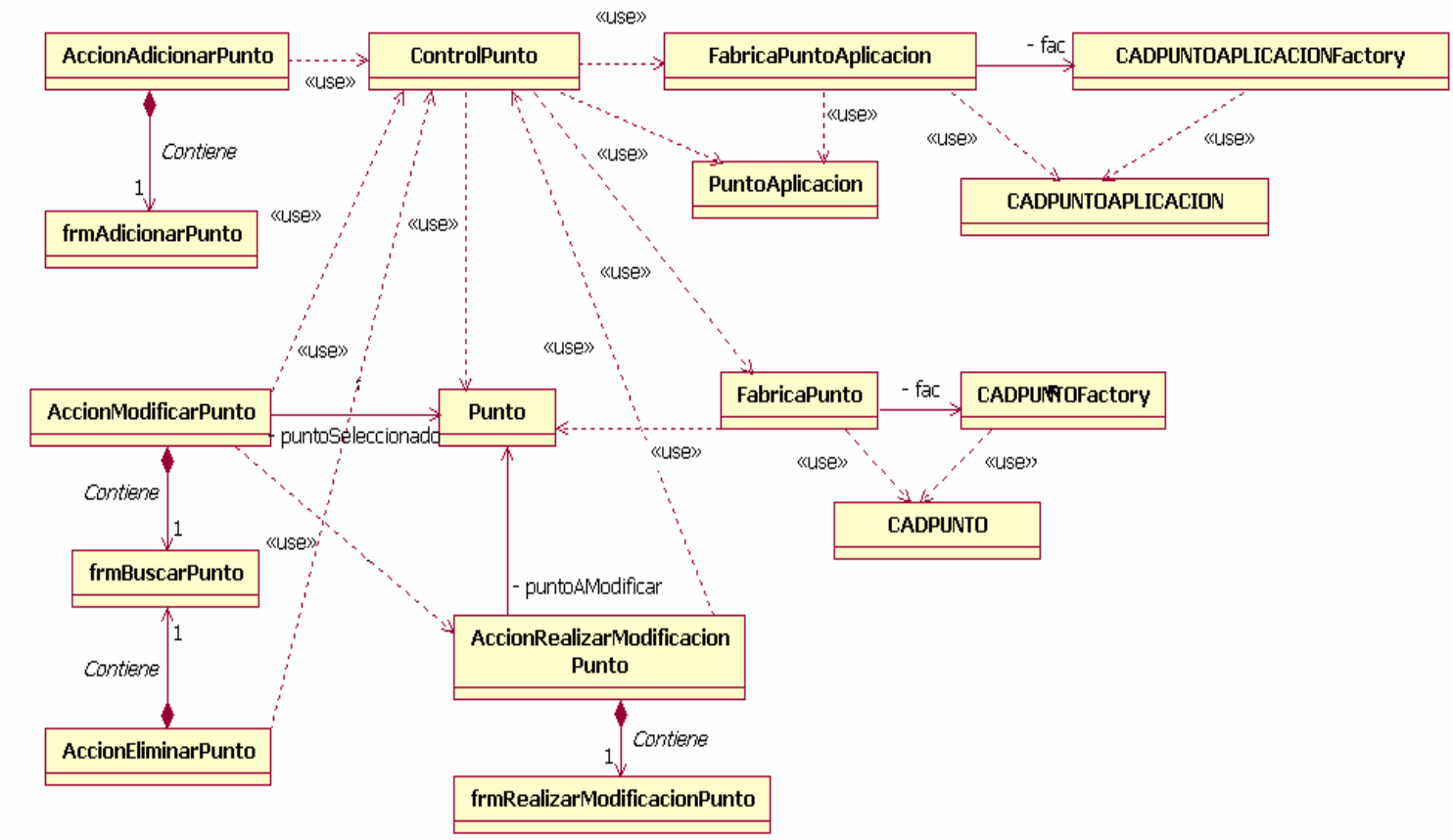
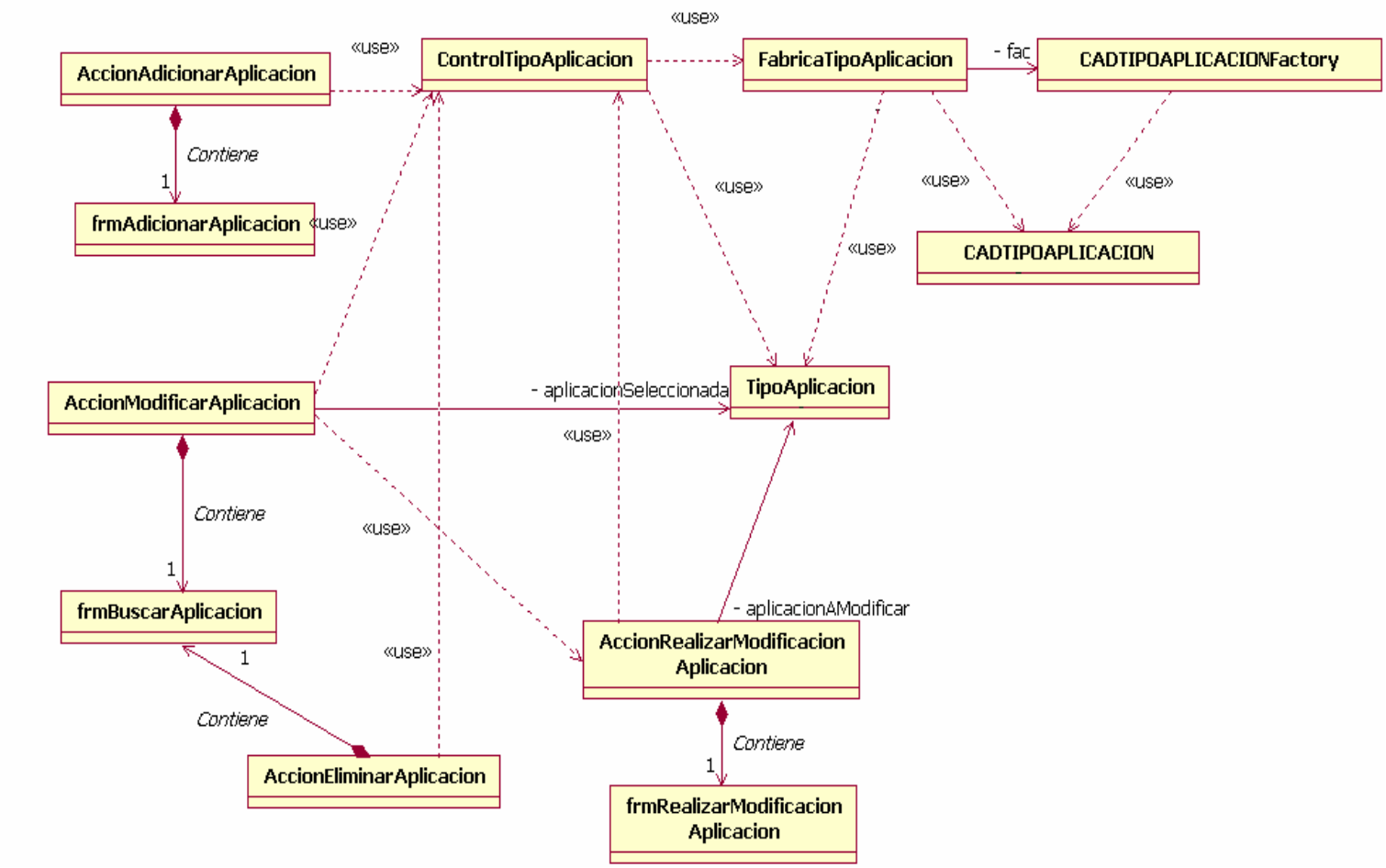


Diagrama de clases Paquete Control de Aplicación

Diagrama 7: Clases del Paquete Control Aplicación.

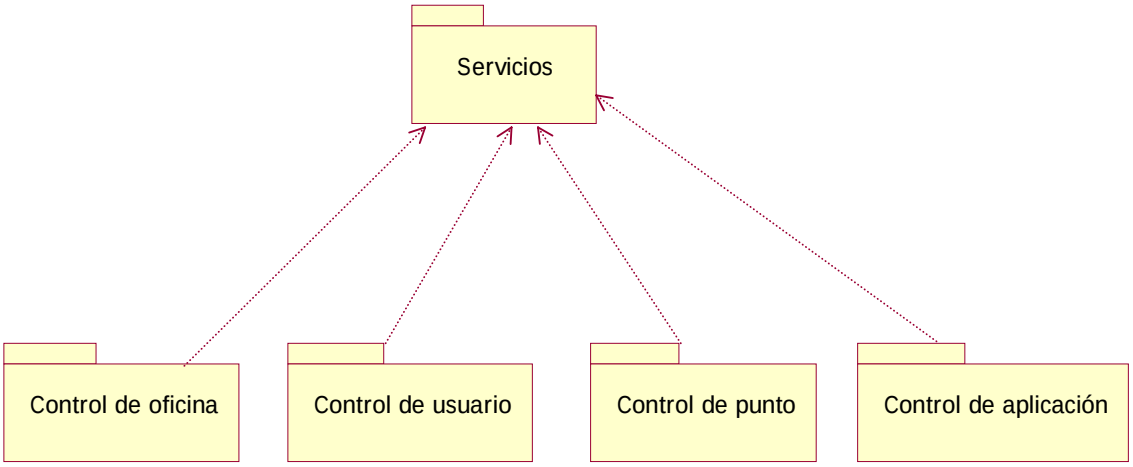


Todas las clases que su nombre comiencen con Accion heredan de AccionSegura excepto AccionAutenticar, AccionActivarAplicacion y AccionIniciarAplicacion que hereda de Accion, estas relaciones no se muestran para dar claridad al diagrama. AccionSegura hereda de Accion.

Todas las clases que su nombre comiencen con frm heredan de frmAreaTrabajo excepto frmPrincipal de la cual hereda la frm AreaTrabajo, estas relaciones no se muestran para dar claridad al diagrama.

Diagrama de clases interacción entre paquetes

Diagrama 8: Interacción entre paquetes.

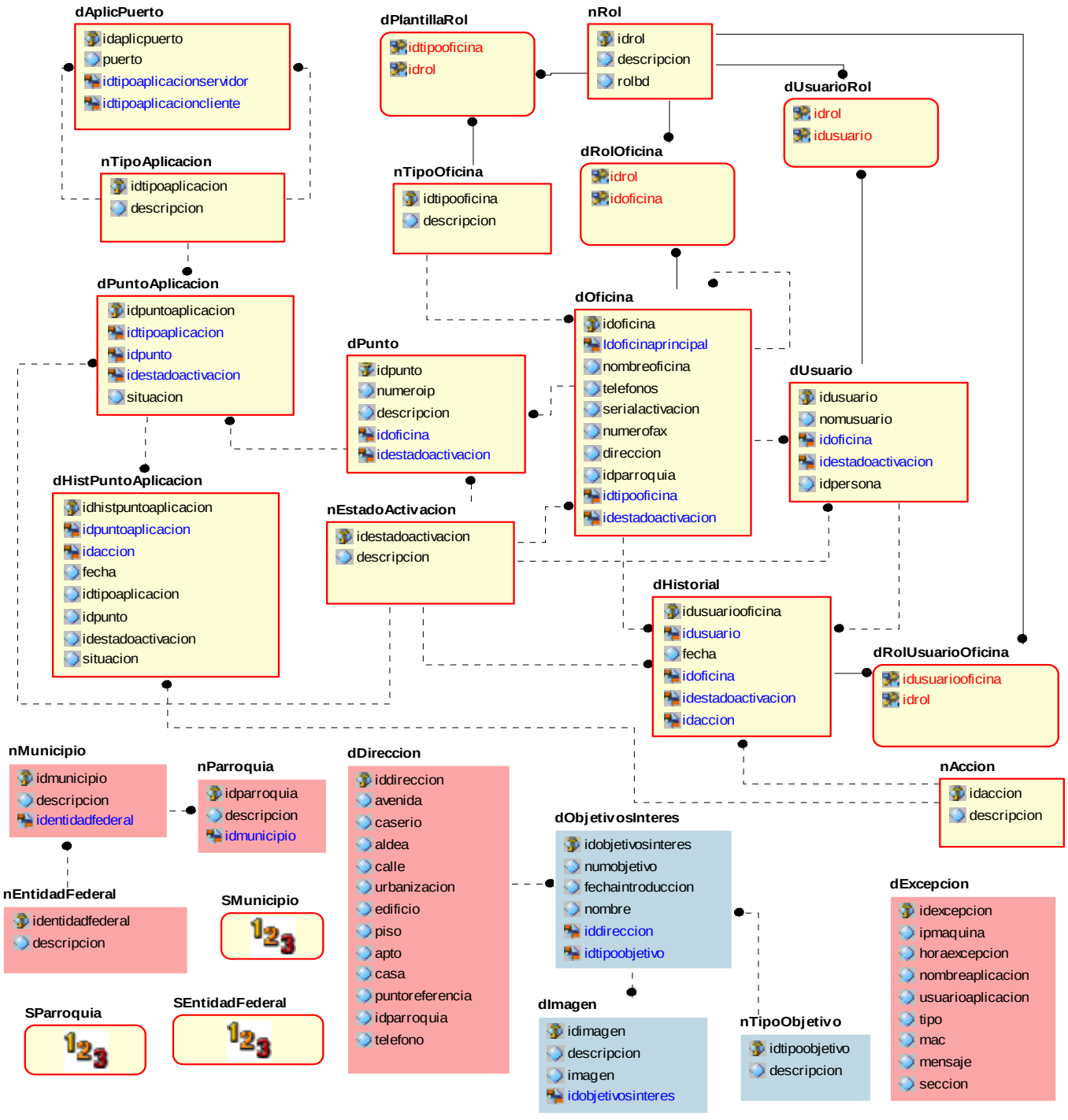


El menú principal contiene las siguientes Acciones:

- AccionAdicionarOficina
- AccionModificarOficina
- AccionAdicionarUsuario
- AccionEstablecerContraseña
- AccionModificarUsuario
- AccionEliminarUsuario
- AccionAdicionarPunto
- AccionModificarPunto
- AccionEliminarPunto
- AccionAdicionarAplicacion
- AccionModificarAplicacion
- AccionEliminarAplicacion
- AccionSalir

De ahí la relación entre los paquetes, estas no se muestran para darle claridad al diagrama.

Diseño de la base de datos



Descripción de los campos presentes en las tablas de la base de datos

Nombre: dOficina		
Descripción: Oficinas pertenecientes al sistema		
Atributo	**Tipo**	**Descripción**
idoficina	byte[]	Identificador de la oficina.
idoficinaprincipal	byte[]	Identificador de oficina principal. Un departamento es una suboficina con todas las relaciones y características que estas tiene. Se tratan a todas estas entidades como nodos donde pueden o no ser raíz de un árbol. La oficina es principal si su campo idoficinaprincipal es nulo.
nombreoficina	string	Nombre de la oficina que representa la tupla.
telefonos	string	Relación de los teléfonos que tiene la oficina.
serialactivacion	string	Serial de activación de la oficina, cadena que identifica inequívocamente una oficina desde su creación, utilizada a la hora de activar la misma ratificando que realmente se creó por los mecanismos existentes en el sistema.
numerofax	string	Relación de los números de Fax que tiene la oficina.
direccion	string	Dirección de residencia del local de la oficina.
idparroquia	decimal	Identificador de la parroquia a la que pertenece.
idtipooficina	decimal	Identificador del tipo de oficina presente.
idestadoactivacion	decimal	Identificador del estado de activación presente.

Nombre: dUsuario		
Descripción: Usuarios que se encuentran registrados en el sistema.		
Atributo	**Tipo**	**Descripción**
idusuario	byte[]	Identificador del usuario.
nomusuario	string	Nombre del usuario, cadena que identifica inequívocamente a los usuarios, no puede existir más de un usuario con el mismo.
idoficina	byte[]	Identificador de la oficina de la cual es usuario.
idestadoactivacion	decimal	Identificador del estado de activación presente.
idpersona	byte[]	Identificador del usuario en la base de datos persona.

Nombre: dPunto		
Descripción: Punto o estación de trabajo del sistema.		
Atributo	**Tipo**	**Descripción**
idpunto	byte[]	Identificador del punto o estación de trabajo.
numeroip	string	Dirección IP del punto.
descripcion	string	Nombre asociado a este puesto de trabajo.
idoficina	byte[]	Identificador de la oficina a la que pertenece.
idestadoactivacion	decimal	Identificador del estado de activación presente.

Nombre: nTipoAplicacion		
Descripción: Grupo de todos los tipos de aplicaciones que formarán el software.		
Atributo	**Tipo**	**Descripción**
idtipoaplicacion	decimal	Identificador del tipo de aplicación.
descripcion	string	Nombre completo de la aplicación.

Nombre: dPuntoAplicacion		
Descripción: Relación entre los puntos y las aplicaciones que pueden ejecutar ellos.		
Atributo	**Tipo**	**Descripción**
idpuntoaplicacion	byte[]	Identificador de la relación entre los puntos y las aplicaciones que pueden ejecutar.
idtipoaplicacion	decimal	Identificador de la aplicación que el punto puede ejecutar.
idpunto	byte[]	Identificador del punto autorizado a ejecutar la aplicación.
idestadoactivacion	decimal	Identificador del estado de activación que tiene esta relación.
situación	bool	Situación bajo la cual se le permitió a este punto ejecutar este tipo de aplicación.

Nombre: nEstadoActivacion		
Descripción: Estado de activación en el que pueden estar los objetos del sistema.		
Atributo	**Tipo**	**Descripción**
idestadoactivacion	decimal	Identificador del estado de activación.
descripcion	string	Nombre del estado de activación.

Nombre: dHistorial		
Descripción: Historial de las acciones que se realizan sobre los usuarios.		
Atributo	**Tipo**	**Descripción**
idusuariooficina	byte[]	Identificador del usuario relacionado con una oficina.
idusuario	byte[]	Identificador del usuario en la tabla dUsuario.
fecha	DateTime	Fecha de la ocurrencia de la acción.
idoficina	byte[]	Identificador de la oficina de la tabla dOficina.

idestadoactivacion	Decimal	Identificador del estado de activación en la tabla nEstadoActivacion.
idaccion	byte[]	Identificador de la acción que se realizó.

Nombre: dUsuarioRol		
Descripción: Rol asignado a cada usuario en el sistema.		
Atributo	**Tipo**	**Descripción**
idrol. idusuario	Decimal byte[]	Identificador del rol de la tabla nRol. Identificador del usuario en la tabla dUsuario.

Nombre: nRol		
Descripción: Rol con el que pueden actuar los objetos en el sistema.		
Atributo	**Tipo**	**Descripción**
idrol. descripcion	Decimal String	Identificador del rol. Nombre del rol.
rolbd	String	Rol que tiene este en el servidor de base de datos.

Nombre: dRolUsuarioOficina		
Descripción: Rol que tiene un usuario en la oficina a la que pertenece en el sistema.		
Atributo	**Tipo**	**Descripción**
idusuariooficina idrol.	byte[] Decimal	Identificador del usuario relacionado con una oficina. Identificador del rol de la tabla nRol.

Nombre: nAccion		
Descripción: Acciones que se realizan sobre determinados objetos.		
Atributo	**Tipo**	**Descripción**
idaccion descripcion	byte[] string	Identificador de la acción. Nombre de la acción.
Nombre: dHistPuntoAplicacion		
Descripción: Historial de las acciones que se realizan sobre un punto relacionado con una aplicación.		
Atributo	**Tipo**	**Descripción**
idhistpuntoaplicacion idpuntoaplicacion	byte[] byte[]	Identificador de la acción producida en una fecha. Identificador de la relación entre los puntos y las aplicaciones que pueden ejecutar.
idaccion	byte[]	Identificador de la acción que se realizó.
fecha	DateTime	Fecha de la ocurrencia de la acción.
idtipoaplicacion	decimal	Identificador de la aplicación que el punto puede ejecutar.
idpunto	byta[]	Identificador del punto.
idestadoactivacion	decimal	Identificador del estado de activación que tiene esta relación.
situación	bool	Situación bajo la cual se le permitió a este

		punto ejecutar este tipo de aplicación.

Nombre: dAplicPuerto		
Descripción: Controla la que aplicaciones se comunican con otras y a través de que puerto lo hacen.		
Atributo	**Tipo**	**Descripción**
idaplicpuerto	byte[]	Identificador de la relación entre aplicaciones, por un puerto determinado. Campo llave.
puerto	string	Puerto por el cual se comunicarán las aplicaciones.
idtipoaplicacionservidor	decimal	Identificador de la aplicación servidor.
idtipoaplicacioncliente	decimal	Identificador del tipo de aplicación cliente.

Nombre: nTipoOficina		
Descripción: Tipo de oficina del sistema.		
Atributo	**Tipo**	**Descripción**
idtipooficina	decimal	Identificador del tipo de oficina.
descripcion	string	Nombre del tipo de oficina.

Nombre: dRolOficina		
Descripción: Rol que posee la oficina.		
Atributo	**Tipo**	**Descripción**
idrol.	decimal	Identificador del rol de la tabla nRol.
idoficina	byte[]	Identificador de la oficina de la tabla dOficina.

Nombre: dPlantillaRol		
Descripción: Control del rol que puede tener una oficina.		
Atributo	**Tipo**	**Descripción**
idtipooficina	decimal	Identificador del tipo de oficina de la tabla nTipoOficina.
idrol	decimal	Identificador del rol de la tabla nRol.

Principios de Diseño de la Aplicación

En todo proceso de desarrollo de software se hace necesario analizar los principios de diseño que determinarán las características del producto de software final. Las facilidades que brinda un software con un diseño coherente y consistente, son elementos determinantes para una buena aceptación por parte de los usuarios.

Para el diseño del Módulo de Administración se definen ciertas técnicas, principios y estándares con suficientes detalles como para permitir su realización física, a continuación se explica cuáles fueron.

Interfaz

Las interfaces del **Sistema Informático del Centro 171** se implementan utilizando formularios de escritorio de Windows, de esta forma se permite procesar y dar formato a los datos de los usuarios, así como adquirir y validar los datos procedentes de éstos. Las funcionalidades de cada aplicación serán variables dependiendo del rol que desempeñe el usuario autenticado; solo se visualizan en el menú las acciones a las que tiene acceso, se evita con esto, recorridos de ida y vuelta innecesarios a los componentes del lado del servidor, al tratar de acceder a datos que no tiene permiso. En el módulo administrativo solo acceden a sus funcionalidades, los administradores del sistema.

Estándares en la interfaz de la aplicación

Con vistas a garantizar la homogeneidad del código en el desarrollo del diseño del **Sistema Informático del Centro 171**, se establecen los siguientes prefijos:

Tabla 15: Prefijos sugeridos para los controles de interfaz.

Tipo de control	**Prefijo**	**Ejemplo**
Panel	pnl	pnlDatos
CheckBox	chk	chkSoloLectura
Diálogo común (OpenFileDialog, SaveFileDialog, etc.)	dlg	dlgArchivoAbrir
Formulario	frm	frmEntrada
DataGrid	grd	grdPrecios
HScrollBar	hsb	hsbVolumen
VScrollBar	vsb	vsdColor
PictureBox	img	imgIcono
Label	lbl	lblNombre
MainMenu	mnu	mnuArchivoAbrir
ContextMenu	cmnu	cmnuEliminar
TextBox	txt	txtApellido
Timer	tmr	tmrAlarma
ImageList	ilst	ilsIconosGrandes
TreeView	trv	trvOrganizacion
ToollBar	tlb	tlbAcciones
TabControl	tab	tabOpciones
StatusBar	stb	stbFechaHora
ProgressBar	pgb	pgbCargarArchivo
RichTextBox	rtxt	rtfInforme
Nomenclador	nmc	nmcEstados
ValidadorTextbox	vldtxt	vldtxtInsertarOficina
ValidadorCombobox	vldcbx	vldcbxInsertarUsuario
Direccion	dir	dirOficina
GroupBox	gbx	gbxSexo

RadioButton	rb	rbEstado
ListBox	lstbx	lstbxGrupos
Listar	lst	lstOficina
ComboBox	cbx	cbxMunicipio
NumericUpDown	nud	nudCantidad

Estándares de codificación

Con vistas a garantizar la homogeneidad del código dentro del grupo de desarrollo, se establece el estilo de código descrito a continuación:

Organización de los ficheros

Ficheros de código

Clases/ficheros cortos, que no excedan las 2000 líneas de código.

Hacer las estructuras/clases claras.

Cada clase en un fichero independiente, con nombre igual que la clase.

Directorio

Crear un directorio para cada espacio de nombre. Por ejemplo, para **Administracion.Interfaz.GI** usar **Administracion/interfaz/GI**, no usar puntos en el nombre de los espacios de nombre.

Líneas largas

Cuando una expresión ocupa más de una línea, debemos dividir la misma atendiendo a los siguientes aspectos:

- Después de una coma.
- Después de un operador.
- Alinear la nueva línea al inicio de la expresión.

Ejemplos de expresiones aritméticas:

```
longName1 = longName2 * (longName3 + longName4 - longName5)
                + 4 * longname6;
var = a * b / (c - g + f) +
```

```
    4 * z;
```

Mal estilo (EVITAR):

```
var = a * b / (c - g +
    f) + 4 * z;
```

Declaración de clases e interfaces

Se deben seguir las siguientes reglas:

- No dejar espacios en blanco entre el nombre del método y el paréntesis ‘(‘ que comienza la lista de parámetros.
- La llave ‘{‘ que comienza el bloque de código del método no debe aparecer a continuación de la declaración.
- La llave que cierra el bloque de código del método debe estar sola en una línea. Indentada en correspondencia a la llave ‘{‘ que abre el bloque. Por ejemplo:

```
class Ejemplo : OtraClase, IUnaInterface
{
    int entero;

    public Ejemplo(int entero)
    {
        this.entero = entero;
    }

    void Inc()
    {
        ++entero;
    }
    void MetodoVacio()
    {
    }
```

```
}
```

Convenciones de declaración

Estilos de capitalización

Estilo de Pascal

Capitaliza la primera letra de cada palabra, ejemplo: UnContador

Estilo camello

Capitaliza la primera letra de cada palabra, excepto para la primera palabra, ejemplo unContador.

Mayúsculas

Usar este convencionalismo sólo para el caso de identificadores que consisten en abreviaturas de uno o dos caracteres de largo, por ejemplo:

```
public class Math
{
        public const PI = ...
        public const E = ...
        public const NumeroBaum = ...
}
```

Directivas de declaración

- El uso de undescore es considerado una mala práctica. Fue utilizado un tiempo atrás para describir el tipo de una variable, pero ya debe considerarse obsoleto.
- Exceptuando los nombres de los controles de interfaz, un buen nombre de variable describe la semántica, no el tipo.
- Utilizar enumerados para los nomencladores.
- Todas las declaraciones a partir del espacio de nombre Sistema.
- Ejemplo: Sistema.Comunes.Excepciones, Sistema.Administracion, entre otros.

Clases

- Los nombres de las clases deben ser sustantivos o frases en sustantivo.
- Usar el estilo de Pascal.

- No usar ningún prefijo excepto para las clases que representen acciones que se debe utilizar el prefijo Accion.

Interfaces

- Utilizar sustantivos, frases en sustantivo o adjetivos que describan comportamiento.
- Usar el estilo de Pascal.
- Usar I como prefijo, seguido por una letra mayúscula (primera letra del nombre de la interfaz).

Enumerados

- Usar el estilo de Pascal.
- No poner prefijos (sufijos) al identificador ni a los valores.
- Identificadores en singular.

Campos de sólo lectura y constantes

- Utilizar sustantivos o frases en sustantivo.
- Usar el estilo de Pascal.
- Nombre completo en mayúsculas para las constantes.

Parámetros/campos no constantes

- Usar nombres descriptivos, deben ser suficientes para determinar el significado de la variable y su tipo, preferiblemente su significado.
- Usar el estilo camello.

Variables

- Utilizar i, j, k, l, m, n para contadores en ciclos triviales, en caso contrario utilizar nombres de variables más descriptivos.
- Utilizar el estilo camello.
- Utilizar el prefijo acc para las variables que representen acciones.

Métodos

- Nombrar los métodos con verbos o frases verbales.
- Utilizar el estilo de Pascal.

Propiedades

- Nombrar las propiedades utilizando sustantivos o frases en sustantivo.
- Utilizar el estilo de Pascal.

- Considerar nombrar las propiedades con el mismo nombre de su tipo.

Eventos

- Nombrar los manejadores de eventos con el sufijo Evento.
- Usar el estilo de Pascal.
- Nombrar los eventos que utilizan el concepto de presente y pasado utilizando el tiempo en que ocurren.

Sumario de capitalización

Tabla 16: Estilos de codificación.

Tipo	**Estilo**	**Notas**
Clase / Estructura	Pascal	
Interfase	Pascal	Comenzando con I
Valores de un enumerado	Pascal	
Nombre de un enumerado	Pascal	
Eventos	Pascal	
Clases de excepciones	Pascal	Comenzando con Excepcion
Métodos	Pascal	
Espacios de nombres	Pascal	
Propiedades	Pascal	
Campos protegidos / privados	Camello	
Campos públicos	Camello	
Parámetros	Camello	
Variables locales	Camello	

Idioma

Todas las declaraciones deben ser en idioma español.

Tratamiento de errores

El tratamiento de errores posibilita el buen funcionamiento de una aplicación, dándole una mejor apariencia ante los clientes. En el sistema se controla la entrada de datos del usuario a nivel de interfaz, se controlan todas las posibilidades en las que pueda haber un error, cuando éste se produce ya sea por la entrada incorrecta de un valor o por algún motivo desconocido para el usuario, se brinda información sobre lo ocurrido en la pantalla y donde se encuentra para que sea rectificado. Esto se hace de dos formas principales: utilizando el control Validador para controles de texto, y la librería de Excepciones implementada, cuya funcionalidad se explica posteriormente.

Validador

El Control Validador es un control no visual que extiende propiedades a los controles TextBox y ComboBox contenidos en un formulario, y de esta forma permiten validar los datos de entrada tanto por teclado como por asignación. En caso que haya datos incorrectos, permite configurar la manera de señalar el control donde ocurrió el error. Esto puede hacerse de las siguientes formas: modificando el color de fondo, utilizar un control ErrorProvider para señalarlo o mostrar un ToolTip con el mensaje de error o cualquier combinación de ellas.

Excepciones

La librería Excepciones permite conocer las excepciones que se generan en una aplicación (Ver Figura 3), así como todos sus detalles (Ver Figura 4), lo que permite conocer el origen del error.

Los datos de la excepción son almacenadas en un log de la base de datos y en caso que no pudiera insertarse en la base de datos, se crearía un log en el Registro de Eventos del Sistema Operativo con esta misma información, además, la librería Excepciones permite propagar la excepción hacia capas superiores para un mejor tratamiento.

Figura 3: Mensaje de error emitido por el sistema.

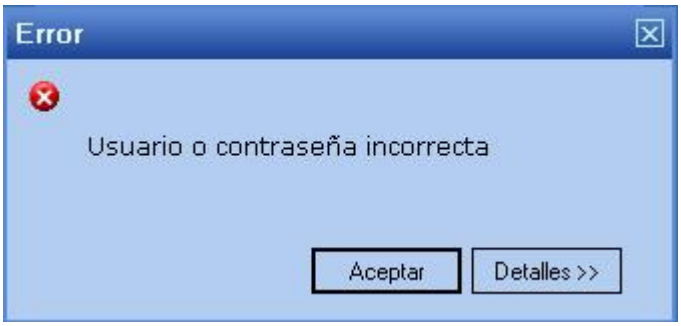
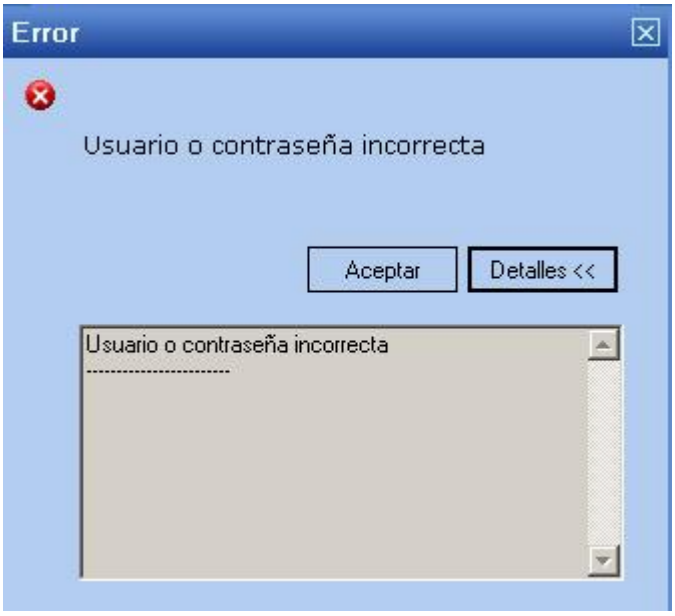


Figura 4: Mensaje de error detallado emitido por el sistema.



Conclusiones

En el capitulo se realizó un análisis de la arquitectura de la aplicación; los diagramas se presentaron divididos por paquetes para una mayor comprensión de los mismos. Se obtuvo el modelo físico de la Base de Datos con la que interactúa el módulo de administración, con la descripción de cada tabla. Se realizó un análisis de los principios de diseño de interfaz de la aplicación, los estándares de codificación y el tratamiento de errores utilizado.

Capítulo IV Implementación

Introducción

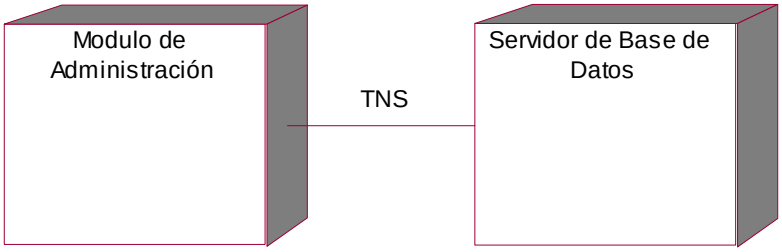
En el presente capítulo se realiza un análisis de la organización física de la implementación, tanto desde el punto de vista de los dispositivos necesarios para lograr implantar el módulo, como de la organización que presenta a nivel de ficheros. Se mencionan los protocolos de comunicación entre los nodos que intervienen en la aplicación administrativa, y la forma en que interactúan los componentes de la misma.

Modelo de despliegue

El diagrama de despliegue contiene los nodos (procesador o dispositivo) que forman la topología de hardware sobre la que se ejecuta el sistema.

El sistema se diseña sobre la base del siguiente diagrama de despliegue, contiene 2 nodos que representan la ubicación física del módulo de administración. En un nodo, el modulo cliente de administración donde descansa toda la lógica del sistema y con la que interactúa directamente el administrador. Un segundo nodo, contiene el servidor de bases de datos con la información relacionada al modulo cliente de administración. Como protocolo de comunicación entre los nodos se utiliza el TNS, implementado por el servidor de bases de datos Oracle para una mayor seguridad e integridad de la información.

Diagrama 9: Modelo de despliegue.



Modelo de Implementación

El modelo de implementación describe la implementación, en términos de componentes, o sea ficheros de código fuente, ficheros de código binarios, scripts, ejecutables y similares. Mediante el mismo se muestra como se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación, así como las dependencias que puedan existir entre ellos.
Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño. A continuación se muestra el Modelo de Implementación del Módulo de Administración.

Diagrama 10: Modelo de implementación.

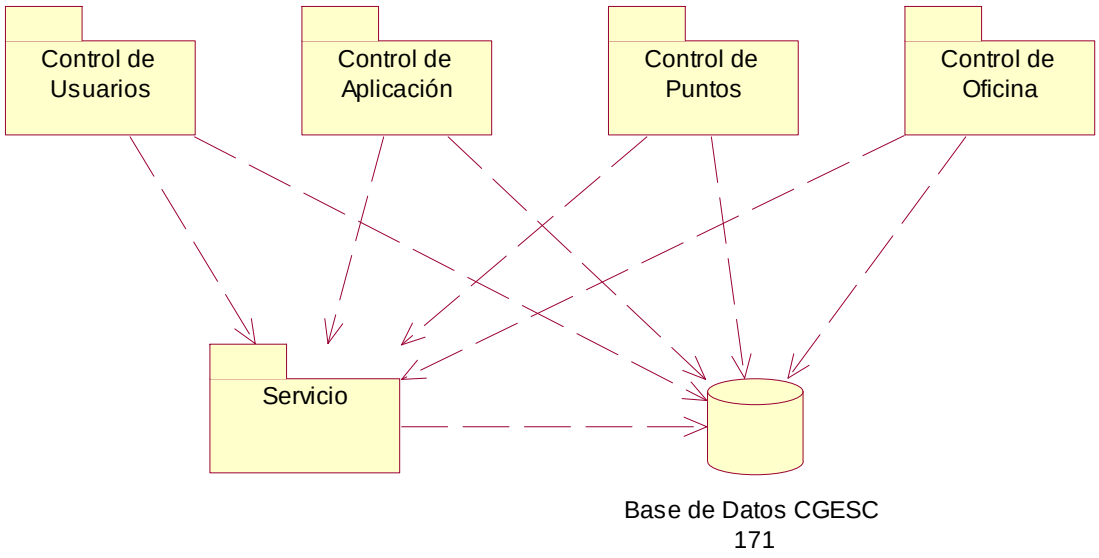


Diagrama de componentes del paquete Servicio

En el paquete Servicio se agrupan un conjunto de componentes que definen las operaciones y acciones que permiten controlar el acceso de los usuarios a las aplicaciones y brindar un entorno de interfaz común para todos los módulos, estandarizándose el comportamiento visual y funcional del Sistema Informático del Centro 171. De esta forma se logra uniformidad, garantizando calidad y profesionalidad mínimas requeridas para un producto de software.

Diagrama 11: Componentes del paquete servicios.

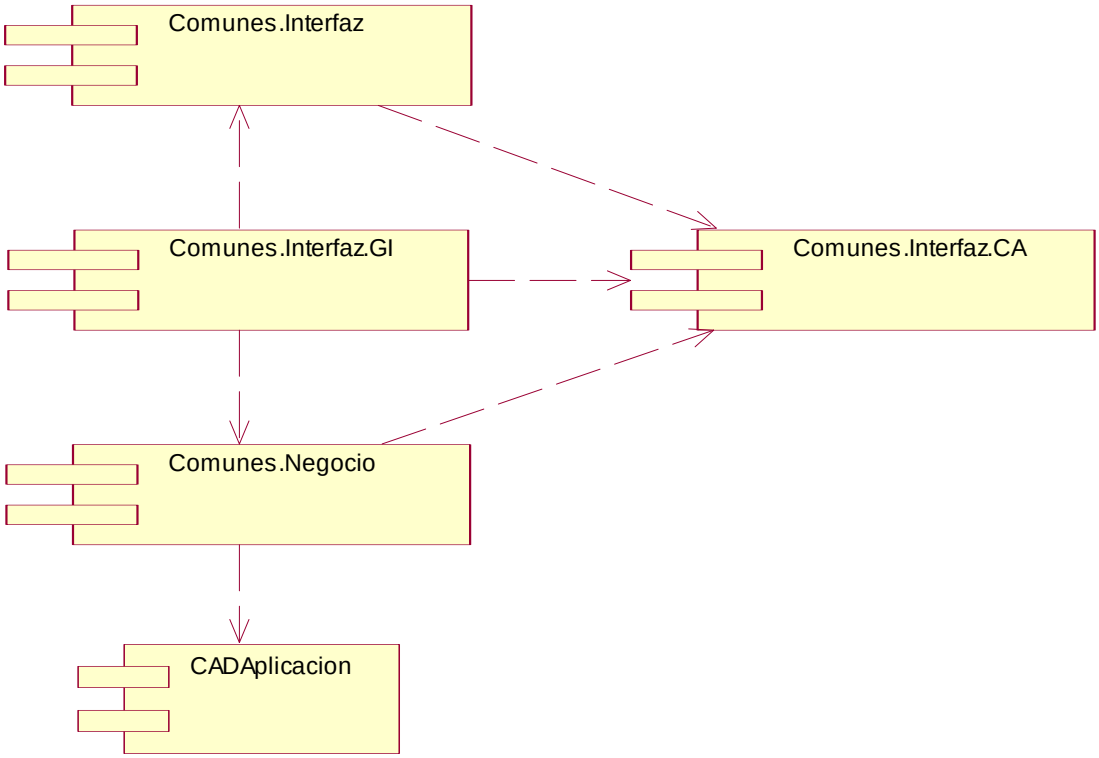


Diagrama de componentes del paquete Control de Oficina

En el paquete Control de Oficina se agrupan los componentes que definen las operaciones insertar y modificar toda la información referente a la entidad oficina.

Diagrama 12: Componentes del paquete control de oficinas.

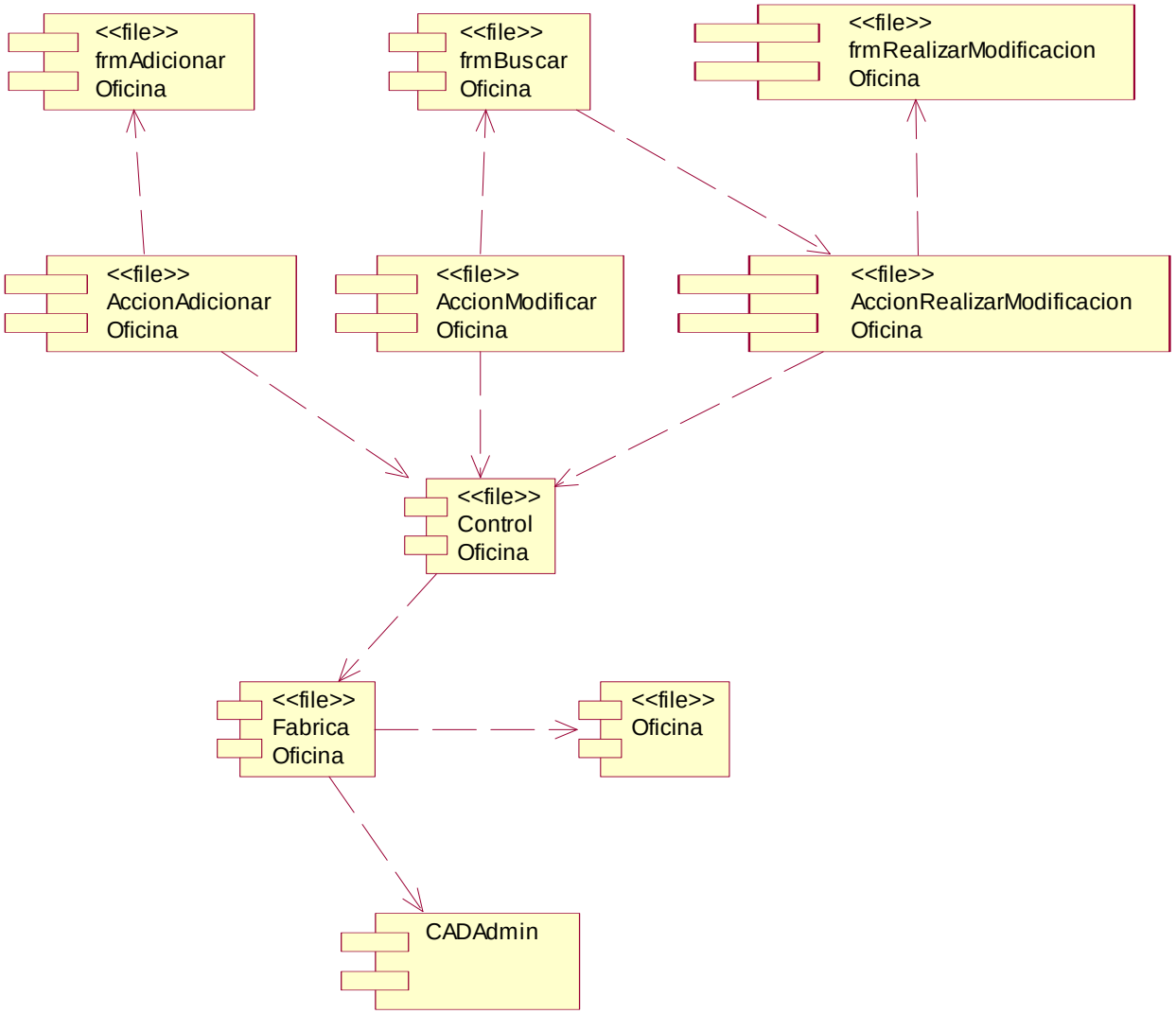


Diagrama de componentes del paquete Control de Usuario

En el paquete Control de Usuario se agrupan los componentes que definen las operaciones insertar, modificar y eliminar toda la información referente a la entidad usuario.

Diagrama 13: Componentes del paquete control de usuario.

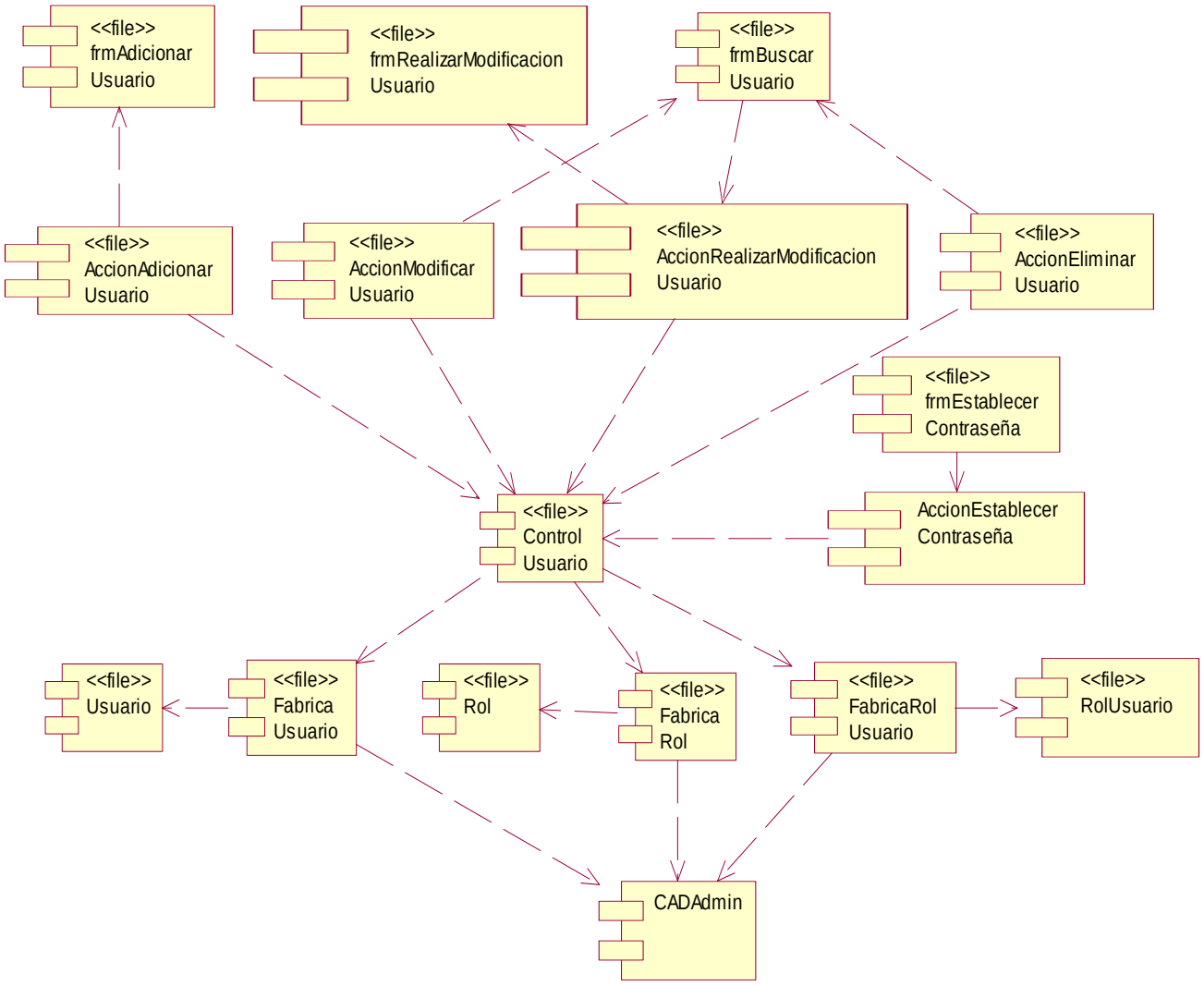


Diagrama de componentes del paquete Control de Punto

En el paquete Control de Punto se agrupan los componentes que definen las operaciones insertar, modificar y eliminar toda la información referente a la entidad punto.

Diagrama 14: Componentes del paquete control de punto.

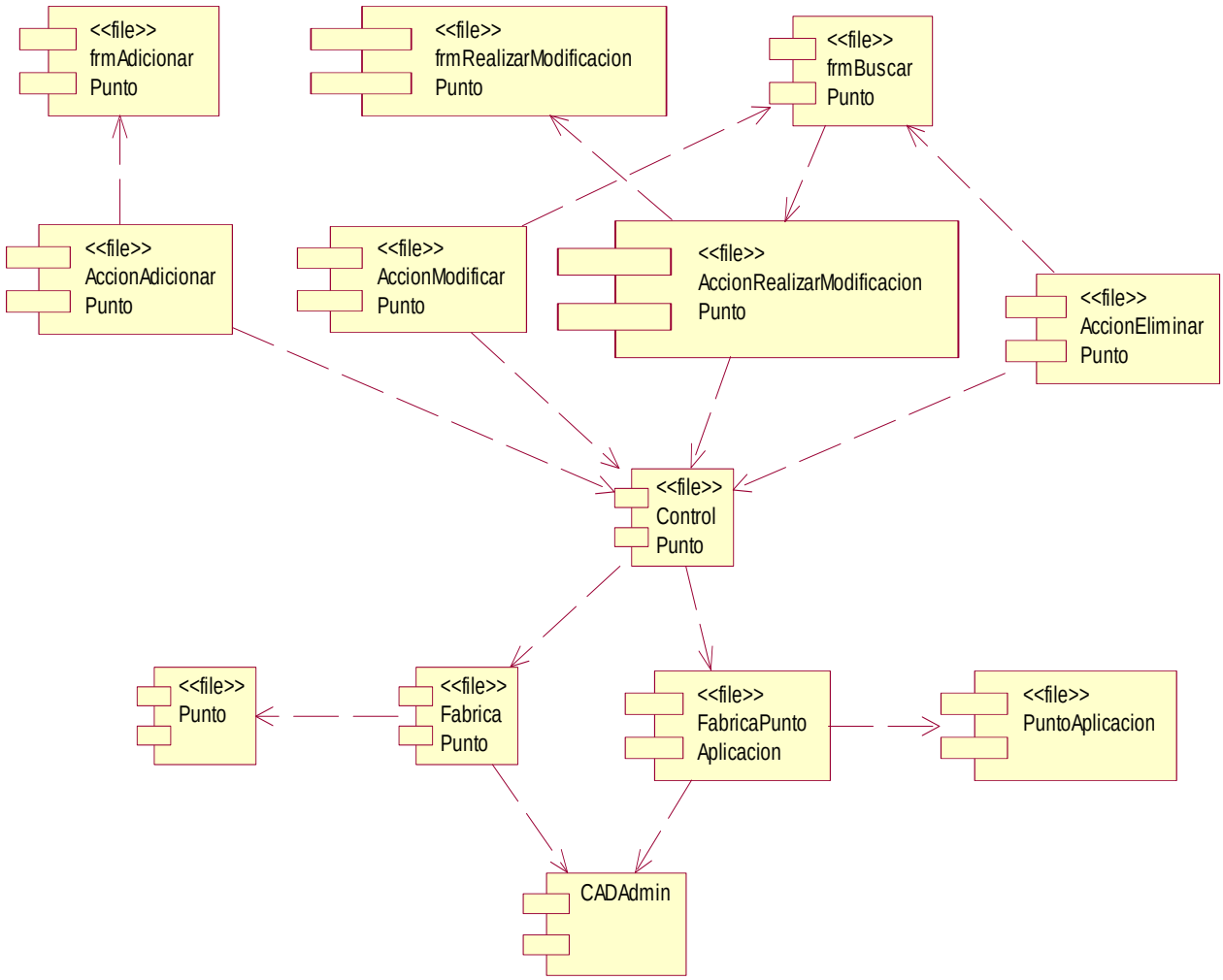
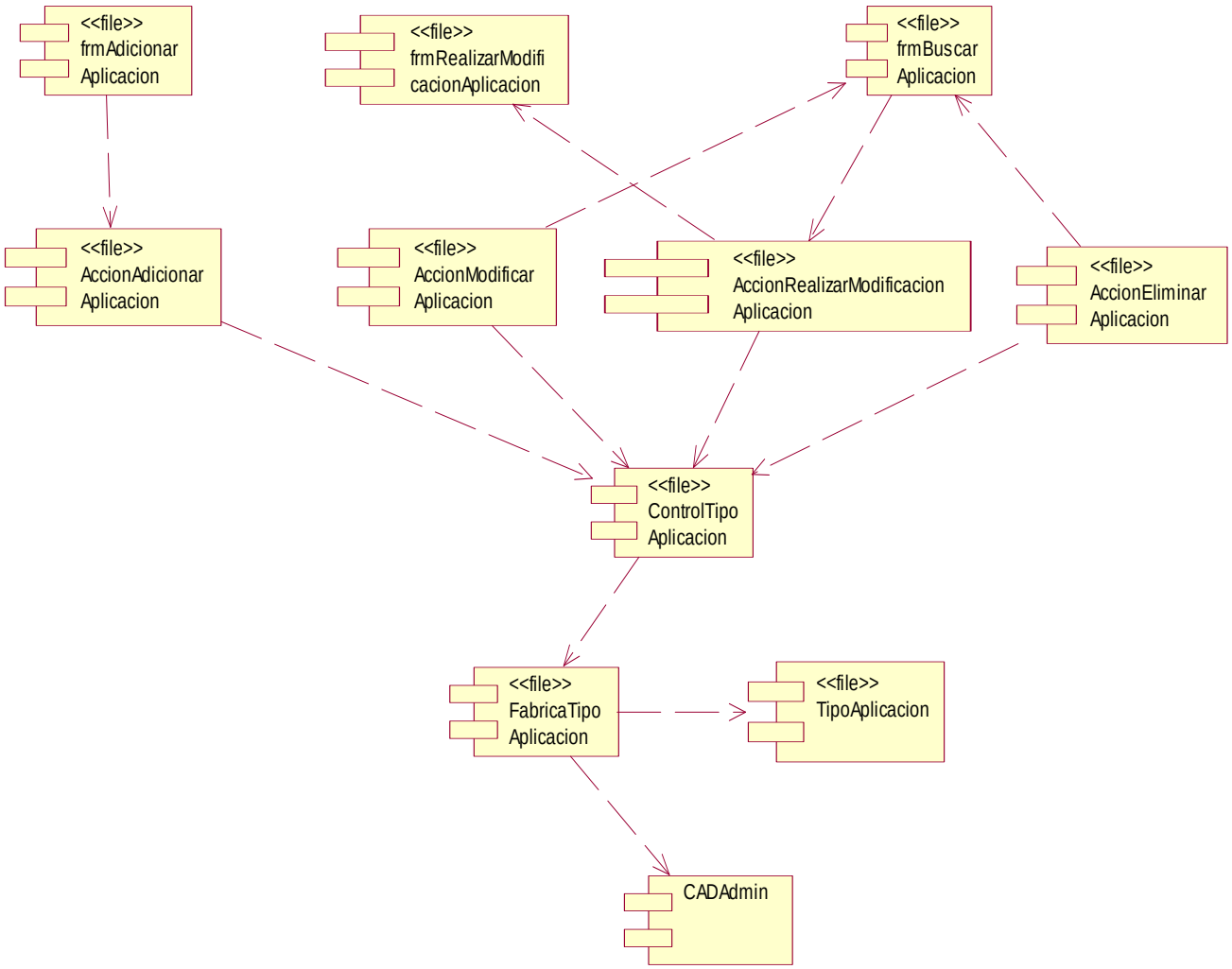


Diagrama de componentes del paquete Control de Aplicación

En el paquete Control de Aplicación se agrupan los componentes que definen las operaciones insertar, modificar y eliminar toda la información referente a la entidad aplicación.

Diagrama 15: Componentes del paquete control de aplicación.



Conclusiones

El capítulo que concluye nos permite tener una idea de cómo se organiza físicamente el Módulo de Administración del **Sistema Informático del Centro 171,** mediante la elaboración del diagrama de despliegue, donde se muestran los procesadores y dispositivos necesarios para ejecutar la aplicación, así como los protocolos usados para la comunicación entre ellos. Se describió mediante diagramas de componentes la forma que interactúan los diferentes paquetes, como se relacionan los componentes dentro de los mismos logrando con esto una mayor claridad a la hora de entender como se distribuyó el software implementado.

Conclusiones generales

A partir de una de las primeras iteraciones en el desarrollo del Módulo de Administración del **Centro de Gestión de Emergencias y Seguridad Ciudadana 171**, se concluye que se ha dado cumplimiento, parcialmente, al objetivo general que se había trazado para una primera etapa; el análisis, diseño e implementación de las diversas funcionalidades especificadas por los objetivos específicos:

- Se logró gestionar toda la información relativa a oficinas, aplicaciones, usuarios y puntos.
- Se abordó, hasta una primera etapa, el análisis de la generación de reportes estadísticos. Quedaron para posteriores iteraciones del desarrollo general del sistema, la conclusión de este objetivo.

Para dar cumplimiento a los objetivos específicos se definieron los requerimientos del sistema, tanto funcionales como no funcionales, orientados principalmente al trabajo de gestión de la información de las entidades que se manipulan, así como para generar reportes estadísticos. A partir de estos requisitos se estructuró y describió el modelo de casos de uso del sistema. Se estructuró el modelo de datos hasta una tercera forma normal, para un óptimo manejo del elevado volumen de información con que trabaja el Módulo; este modelo es una representación física de una parte de la base de datos de todo el **Sistema Informático del Centro 171**.

Se elaboró el modelo de despliegue, el modelo de componentes y de implementación, para una detallada descripción de la ubicación física del módulo y un análisis de la interacción entre sus partes y relación con otros módulos del sistema.

Se plantearon los principios a seguir en el diseño de la interfaz de usuario y algunas convenciones a respetar durante la escritura del código fuente.

Es resultado de este trabajo, un software de alta fiabilidad que cumple los requisitos del usuario para una primera etapa, acompañado de la descripción de sus partes y de los procesos involucrados en su ambiente.

Recomendaciones

Continuar con el análisis, diseño e implementación de las funcionalidades del segundo ciclo de desarrollo del Módulo de Administración del sistema informático del **Sistema Informático del Centro 171**, pues estas son las que permitirán configurar todas las variables necesarias a la hora de prestar los servicios esperados.

En vista a la política de migración hacia software libre llevada a cabo por nuestro país y el país hermano de Venezuela, se recomienda hacer un profundo estudio sobre la migración paulatina del módulo a software libre.

Referencias bibliográficas

1. Constitución de la Republica Bolivariana de Venezuela., Asamblea nacional constituyente.

2. CORABEL, S. Manejadores de Bases de Datos - SQL, ORACLE, INFORMIX, 2004. [2005]. Disponible en:
3. http://www.ilustrados.com/publicaciones/EpZVVlyFyAbRDtMKhI.php

4. Declaración Universal de los Derechos humanos., Resolución de la Asamblea General 217 A (iii). 1948.

5. IVAR JACOBSON, G. B., JAMES RUMBAUGH. "El Proceso Unificado de Desarrollo de Software". 2000. 464 p.

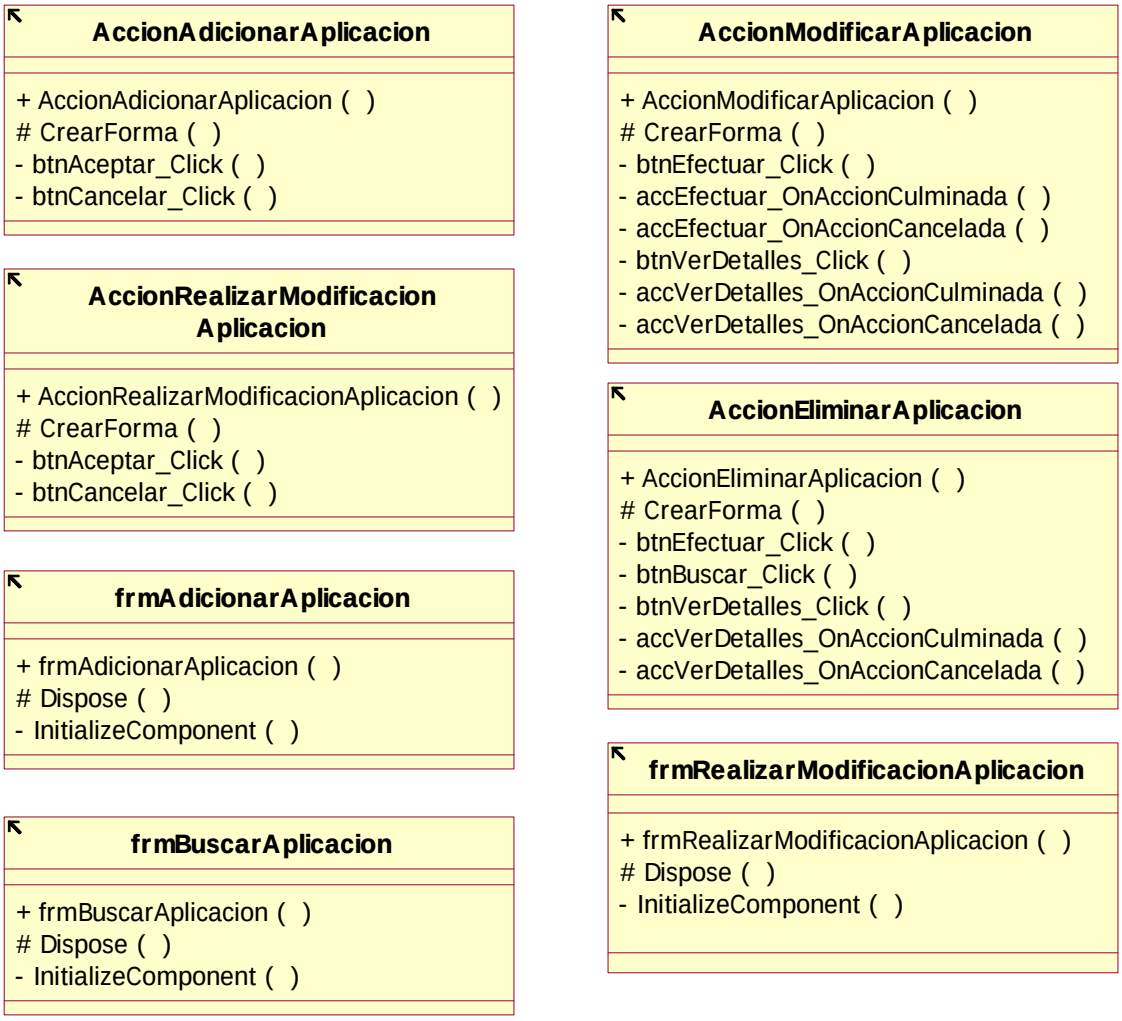
Bibliografía

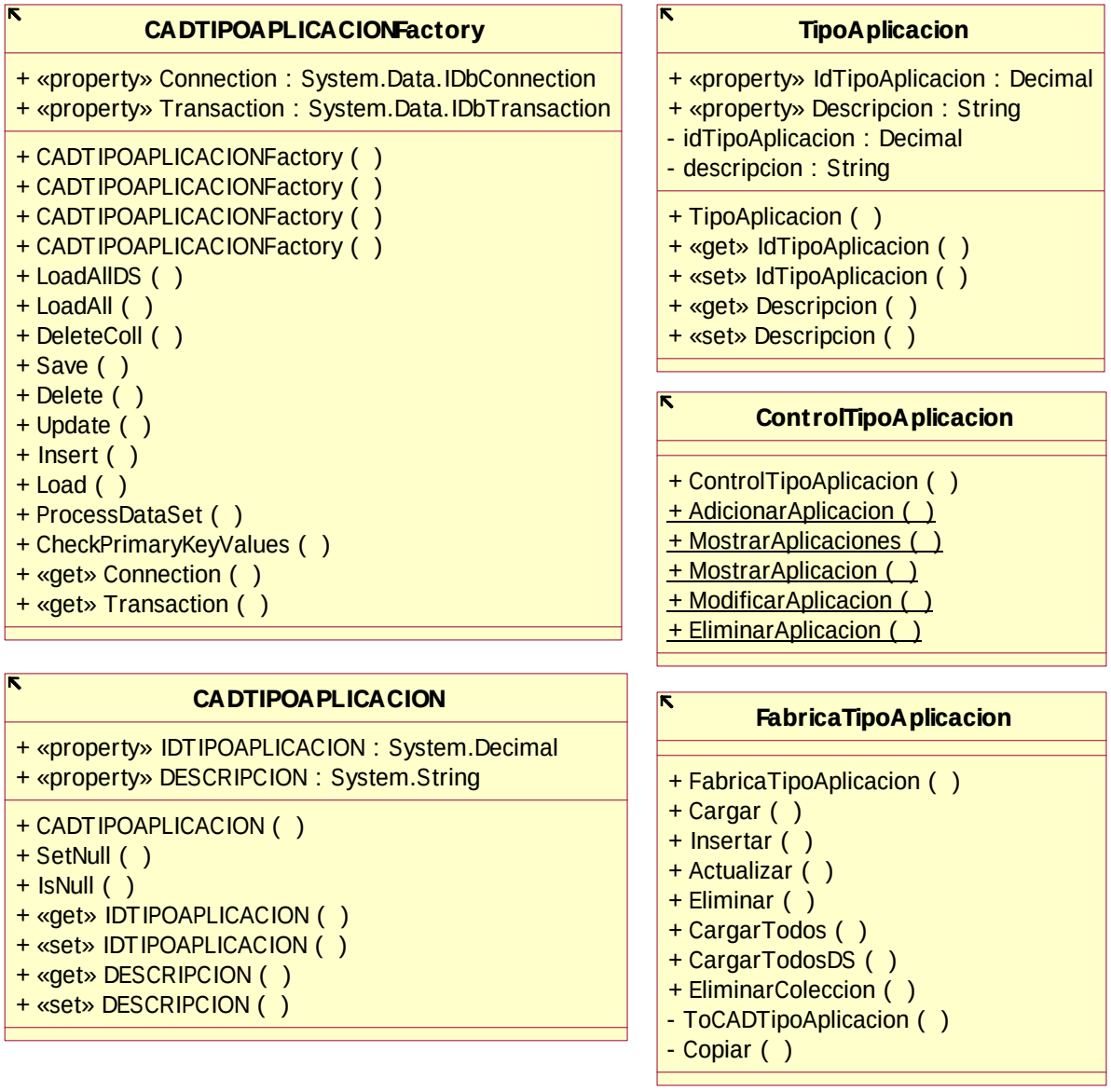
1. BECERRIL, F. *Java a su alcance*, 1999. [Disponible en: http://bibliodoc.uci.cu/pdf/reg01327.pdf
2. CLIKEAR. *Arquitectura .NET y Metodologías de Programación*, 2005. [Disponible en: http://www.clikear.com/Arquitectura/
3. CLIKEAR. *SQL SERVER*, 2005. [Disponible en: http://www.clikear.com/sqlserver/
4. *Constitución de la Republica Bolivariana de Venezuela.*, Asamblea nacional constituyente.
5. CORABEL, S. *Manejadores de Bases de Datos - SQL, ORACLE, INFORMIX*, 2004. [2005]. Disponible en: http://www.ilustrados.com/publicaciones/EpZVVlyFyAbRDtMKhI.php
6. *Declaración Universal de los Derechos humanos.*, Resolución de la Asamblea General 217 A (iii). 1948.
7. IVAR JACOBSON, G. B., JAMES RUMBAUGH. *"El Proceso Unificado de Desarrollo de Software"*. 2000. 464 p.
8. JAMES RUMBAUGH, I. J., GRADY BOOCH. *El lenguaje unificado de modelado. Manual de Referencia*, 2001. [Disponible en: http://bibliodoc.uci.cu/pdf/reg03050.pdf
9. JEFF FERGUSON, B. P., JASON BERES. *La Biblia de C#*, 2002. [Disponible en: http://bibliodoc.uci.cu/pdf/8441514844.pdf
10. MICHAEL ABBEY, M. C., IAN ABRAMSON. *Oracle 9i. Guía de aprendizaje*, 2000. [Disponible en: http://bibliodoc.uci.cu/pdf/reg02472.pdf
11. MICROSOFT, C. *SQL Server 2000*, 2000. [Disponible en: http://www.microsoft.com/sql/default.mspx
12. MICROSOFT, C. *Versioning, Compatibility, and Side-by-Side Execution in the .NET Framework*, 2004. [Disponible en: http://msdn.microsoft.com/netframework/technologyinfo/versioncomparison/default.aspx
13. MICROSOFT, C. *Interoperation Using .NET*, 2005. [Disponible en: http://msdn.microsoft.com/netframework/technologyinfo/infrastructure/interop/default.aspx
14. ---. *What is .NET?*, 2005. [Disponible en: http://www.microsoft.com/net/basics.mspx
15. ORACLE. *Oracle Ibérica*, 2005. [Disponible en: http://www.oracle.com/global/es/index.html
16. RALMALHO, J. A. *SQL Server. Iniciación y referencia*, 2000. [Disponible en: http://bibliodoc.uci.cu/pdf/reg01719.pdf

17. ROGER S, P. *Ingeniería del Software*, 2001. [1]. Disponible en: http://bibliodoc.uci.cu/pdf/reg02689.pdf
18. SCHMULLER, J. *Aprendiendo UML en 24 horas*, 2000. [1]. Disponible en: http://bibliodoc.uci.cu/pdf/reg00004.pdf
19. URMAN, S. *Oracle 8i. Programación avanzada con PL-SQL*, 2001. [Disponible en: http://bibliodoc.uci.cu/pdf/reg01431.pdf
20. NAVARRO, J. M. *Iniciación a Oracle*, 2005. [Disponible en: http://www.wikilearning.com/introduccion_a_este_manual-wkccp-3861-1.htm

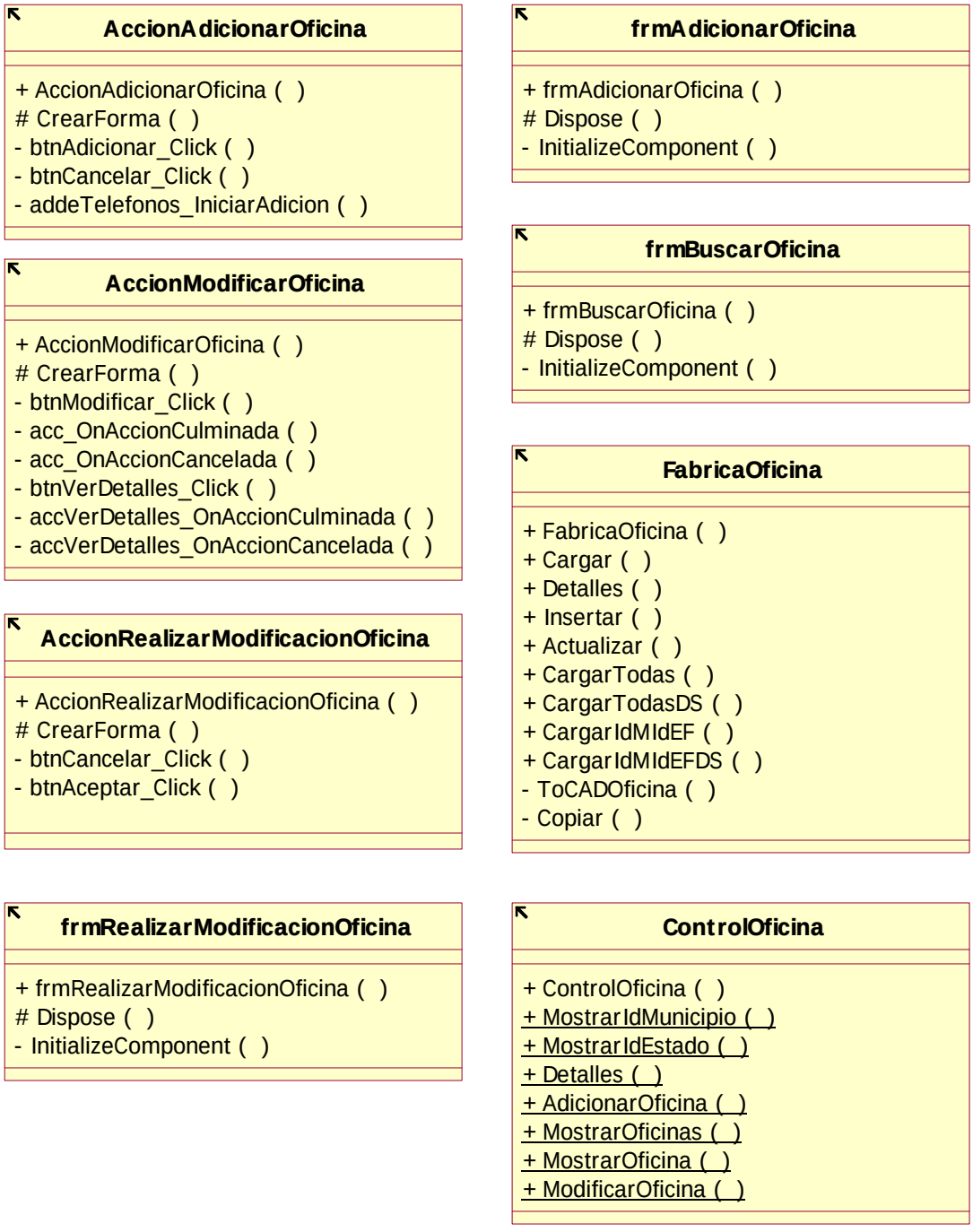
Anexos

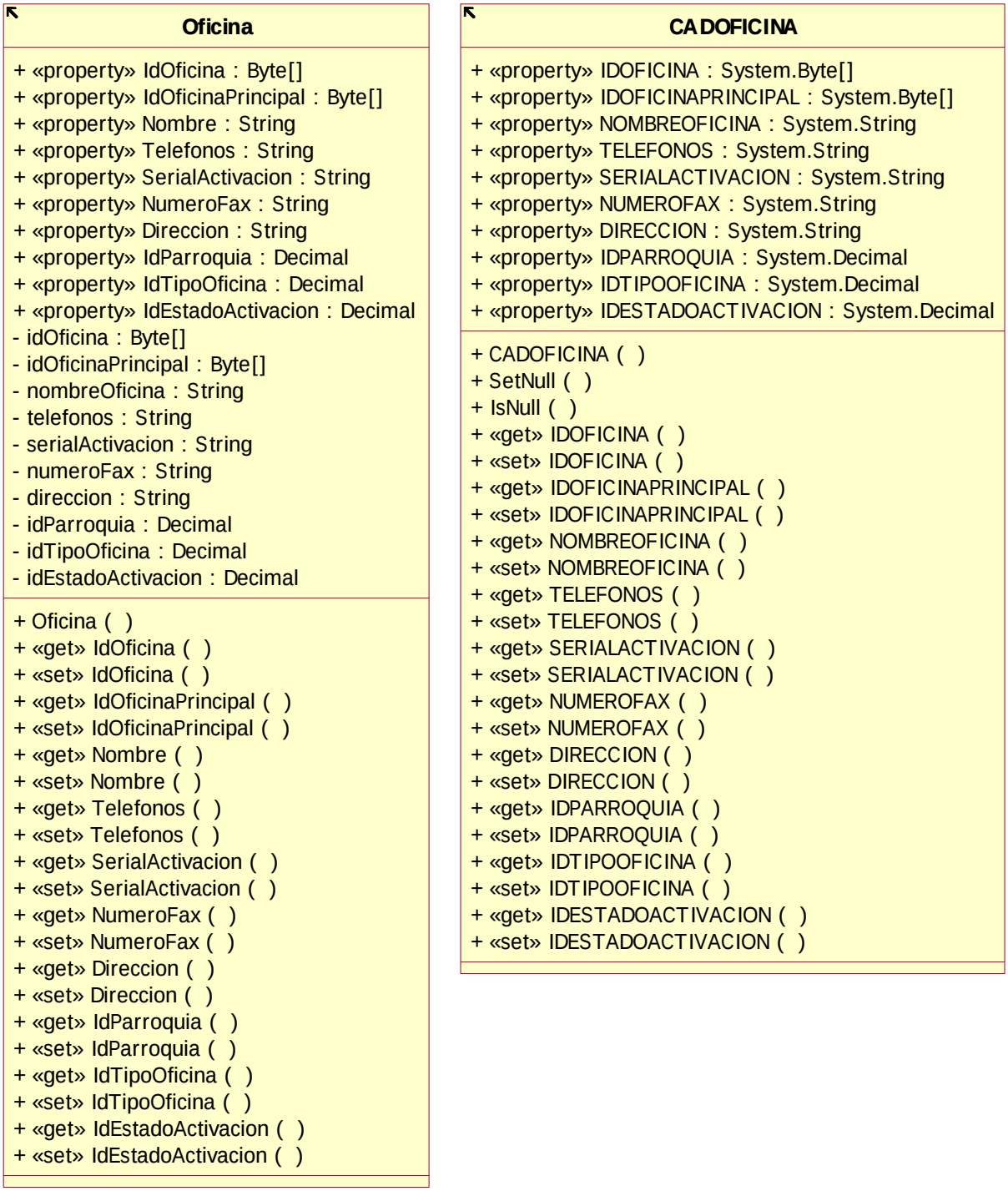
Clases del paquete Control de Aplicación

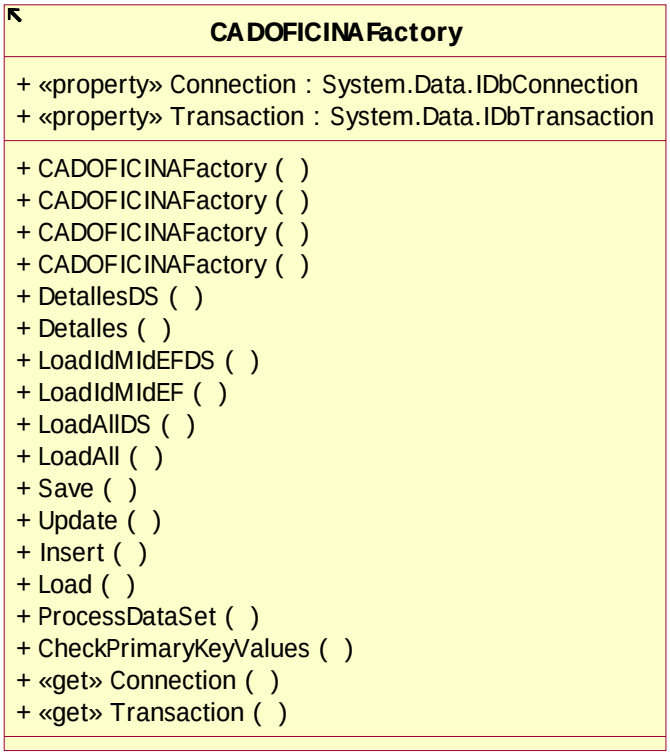



CADTIPOAPLICACIONFactory
+ «property» Connection : System.Data.IDbConnection
+ «property» Transaction : System.Data.IDbTransaction
+ CADTIPOAPLICACIONFactory ( )
+ CADTIPOAPLICACIONFactory ( )
+ CADTIPOAPLICACIONFactory ( )
+ CADTIPOAPLICACIONFactory ( )
+ LoadAllDS ( )
+ LoadAll ( )
+ DeleteColl ( )
+ Save ( )
+ Delete ( )
+ Update ( )
+ Insert ( )
+ Load ( )
+ ProcessDataSet ( )
+ CheckPrimaryKeyValues ( )
+ «get» Connection ( )
+ «get» Transaction ( )
CADTIPOAPLICACION
+ «property» IDTIPOAPLICACION : System.Decimal
+ «property» DESCRIPCION : System.String
+ CADTIPOAPLICACION ( )
+ SetNull ( )
+ IsNull ( )
+ «get» IDTIPOAPLICACION ( )
+ «set» IDTIPOAPLICACION ( )
+ «get» DESCRIPCION ( )
+ «set» DESCRIPCION ( )
TipoAplicacion
+ «property» IdTipoAplicacion : Decimal
+ «property» Descripcion : String
- idTipoAplicacion : Decimal
- descripcion : String
+ TipoAplicacion ( )
+ «get» IdTipoAplicacion ( )
+ «set» IdTipoAplicacion ( )
+ «get» Descripcion ( )
+ «set» Descripcion ( )
ControlTipoAplicacion
+ ControlTipoAplicacion ( )
+ AdicionarAplicacion ( )
+ MostrarAplicaciones ( )
+ MostrarAplicacion ( )
+ ModificarAplicacion ( )
+ EliminarAplicacion ( )
FabricaTipoAplicacion
+ FabricaTipoAplicacion ( )
+ Cargar ( )
+ Insertar ( )
+ Actualizar ( )
+ Eliminar ( )
+ CargarTodos ( )
+ CargarTodosDS ( )
+ EliminarColeccion ( )
- ToCADTipoAplicacion ( )
- Copiar ( )

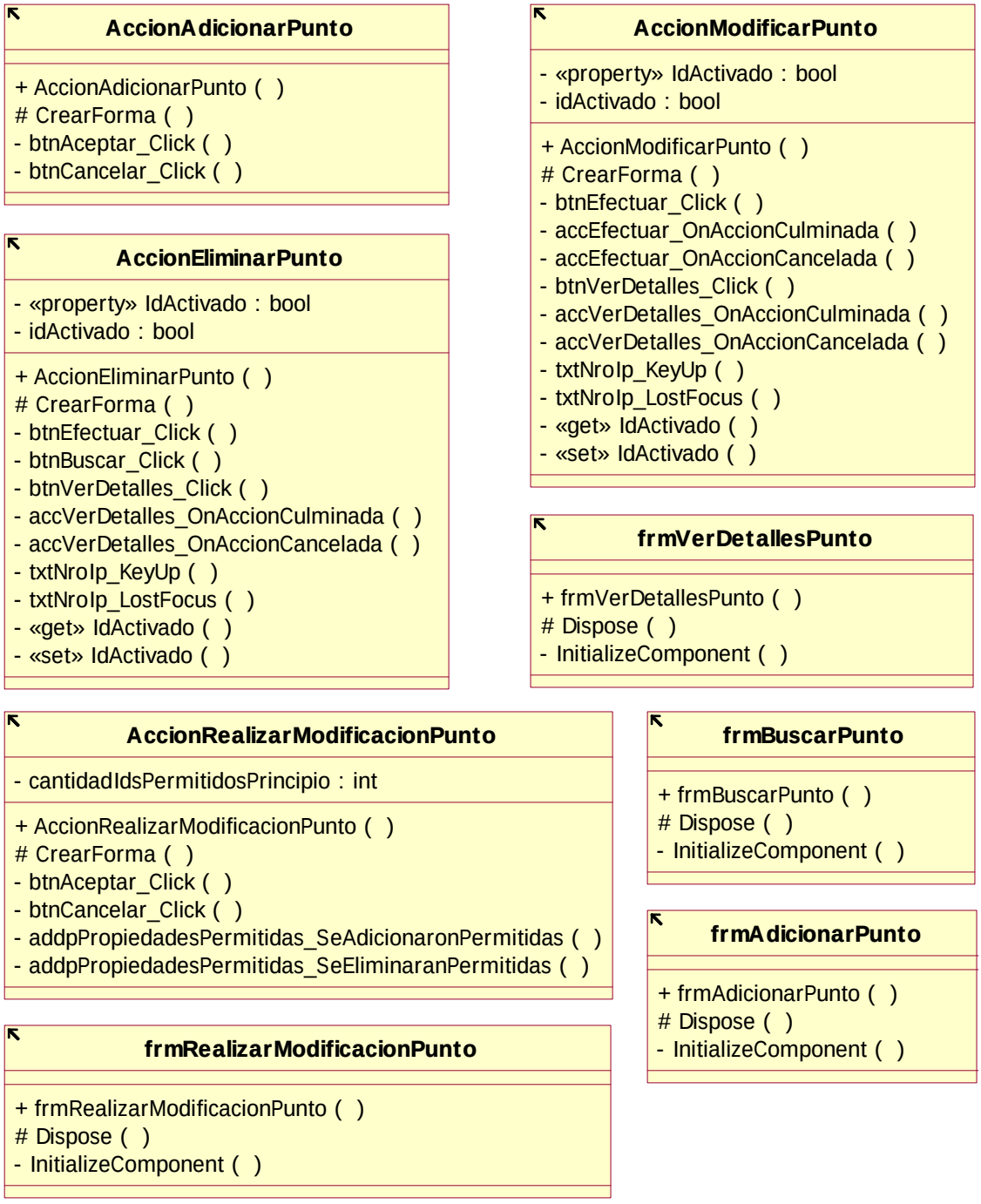
Clases del paquete Control de Oficina




Oficina
+ «property» IdOficina : Byte[]
+ «property» IdOficinaPrincipal : Byte[]
+ «property» Nombre : String
+ «property» Telefonos : String
+ «property» SerialActivacion : String
+ «property» NumeroFax : String
+ «property» Direccion : String
+ «property» IdParroquia : Decimal
+ «property» IdTipoOficina : Decimal
+ «property» IdEstadoActivacion : Decimal
- idOficina : Byte[]
- idOficinaPrincipal : Byte[]
- nombreOficina : String
- telefonos : String
- serialActivacion : String
- numeroFax : String
- direccion : String
- idParroquia : Decimal
- idTipoOficina : Decimal
- idEstadoActivacion : Decimal
+ Oficina ( )
+ «get» IdOficina ( )
+ «set» IdOficina ( )
+ «get» IdOficinaPrincipal ( )
+ «set» IdOficinaPrincipal ( )
+ «get» Nombre ( )
+ «set» Nombre ( )
+ «get» Telefonos ( )
+ «set» Telefonos ( )
+ «get» SerialActivacion ( )
+ «set» SerialActivacion ( )
+ «get» NumeroFax ( )
+ «set» NumeroFax ( )
+ «get» Direccion ( )
+ «set» Direccion ( )
+ «get» IdParroquia ( )
+ «set» IdParroquia ( )
+ «get» IdTipoOficina ( )
+ «set» IdTipoOficina ( )
+ «get» IdEstadoActivacion ( )
+ «set» IdEstadoActivacion ( )
CADOFICINA
+ «property» IDOFICINA : System.Byte[]
+ «property» IDOFICINAPRINCIPAL : System.Byte[]
+ «property» NOMBREOFICINA : System.String
+ «property» TELEFONOS : System.String
+ «property» SERIALACTIVACION : System.String
+ «property» NUMEROFAX : System.String
+ «property» DIRECCION : System.String
+ «property» IDPARROQUIA : System.Decimal
+ «property» IDTIPOOFICINA : System.Decimal
+ «property» IDESTADOACTIVACION : System.Decimal
+ CADOFICINA ( )
+ SetNull ( )
+ IsNull ( )
+ «get» IDOFICINA ( )
+ «set» IDOFICINA ( )
+ «get» IDOFICINAPRINCIPAL ( )
+ «set» IDOFICINAPRINCIPAL ( )
+ «get» NOMBREOFICINA ( )
+ «set» NOMBREOFICINA ( )
+ «get» TELEFONOS ( )
+ «set» TELEFONOS ( )
+ «get» SERIALACTIVACION ( )
+ «set» SERIALACTIVACION ( )
+ «get» NUMEROFAX ( )
+ «set» NUMEROFAX ( )
+ «get» DIRECCION ( )
+ «set» DIRECCION ( )
+ «get» IDPARROQUIA ( )
+ «set» IDPARROQUIA ( )
+ «get» IDTIPOOFICINA ( )
+ «set» IDTIPOOFICINA ( )
+ «get» IDESTADOACTIVACION ( )
+ «set» IDESTADOACTIVACION ( )


CADOFICINAFactory
+ «property» Connection : System.Data.IDbConnection
+ «property» Transaction : System.Data.IDbTransaction
+ CADOFICINAFactory ( )
+ CADOFICINAFactory ( )
+ CADOFICINAFactory ( )
+ CADOFICINAFactory ( )
+ DetallesDS ( )
+ Detalles ( )
+ LoadIdMIdEFDS ( )
+ LoadIdMIdEF ( )
+ LoadAllDS ( )
+ LoadAll ( )
+ Save ( )
+ Update ( )
+ Insert ( )
+ Load ( )
+ ProcessDataSet ( )
+ CheckPrimaryKeyValues ( )
+ «get» Connection ( )
+ «get» Transaction ( )

Clases del paquete Control de Punto



ControlPunto

+ ControlPunto ( )
+ AdicionarPunto ( )
+ MostrarPuntos ( )
+ MostrarPunto ( )
+ Detalles ( )
+ MostrarAplicacionesPermitidasEnElPunto ( )
+ MostrarAplicacionesNoPermitidasEnElPunto ( )
+ ModificarPunto ( )
...

FabricaPunto

+ FabricaPunto ( )
+ Cargar ( )
+ Detalles ( )
+ Insertar ( )
+ Actualizar ( )
+ Eliminar ( )
+ EliminarColeccion ( )
+ CargarTodos ( )
+ CargarTodosDS ( )
+ CargarAplicacionesPermitidasEnElPunto ( )
+ CargarAplicacionesNoPermitidasEnElPunto ( )
- ToCADPunto ( )
- Copiar ( )

FabricaPuntoAplicacion

+ FabricaPuntoAplicacion ( )
+ Cargar ( )
+ Insertar ( )
+ Actualizar ( )
+ Eliminar ( )
+ CargarTodos ( )
+ CargarTodosDS ( )
+ EliminarColeccion ( )
- ToCADPuntoAplicacion ( )
- Copiar ( )

Punto

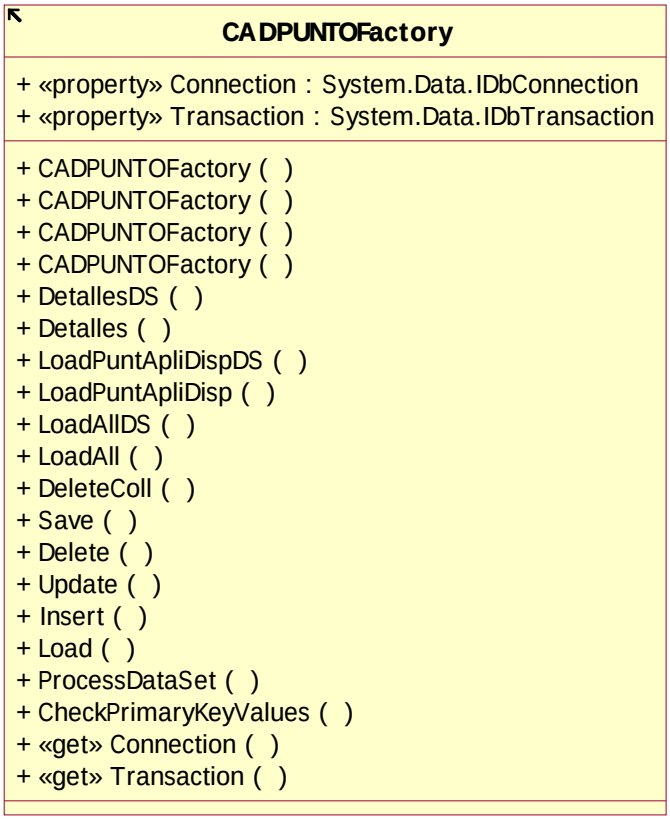
+ «property» IdPunto : Byte[]
+ «property» NumeroIp : String
+ «property» Descripcion : String
+ «property» IdOficina : Byte[]
+ «property» IdEstadoActivacion : Decimal
- idPunto : Byte[]
- numeroIp : String
- descripcion : String
- idOficina : Byte[]
- idEstadoActivacion : Decimal

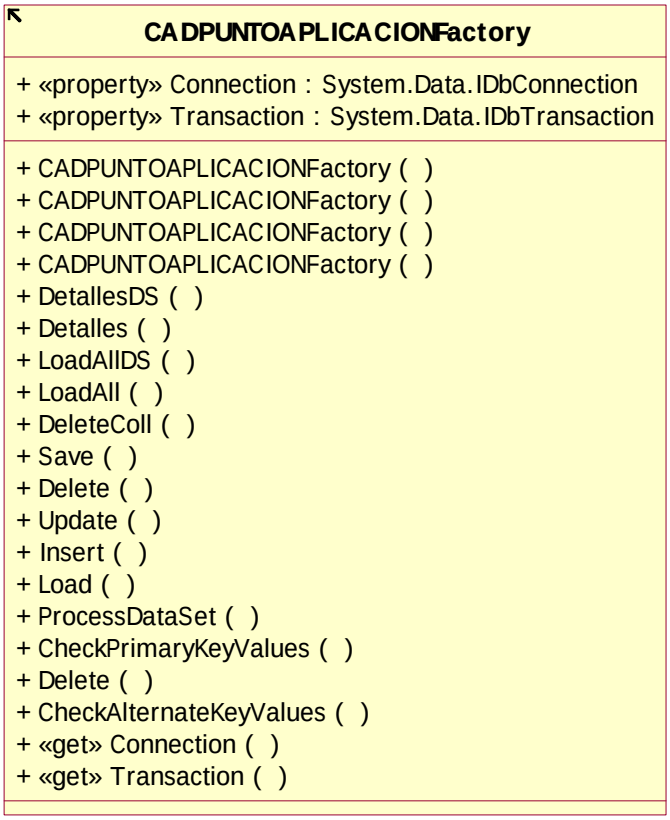
+ Punto ( )
+ «get» IdPunto ( )
+ «set» IdPunto ( )
+ «get» NumeroIp ( )
+ «set» NumeroIp ( )
+ «get» Descripcion ( )
+ «set» Descripcion ( )
+ «get» IdOficina ( )
+ «set» IdOficina ( )
+ «get» IdEstadoActivacion ( )
+ «set» IdEstadoActivacion ( )

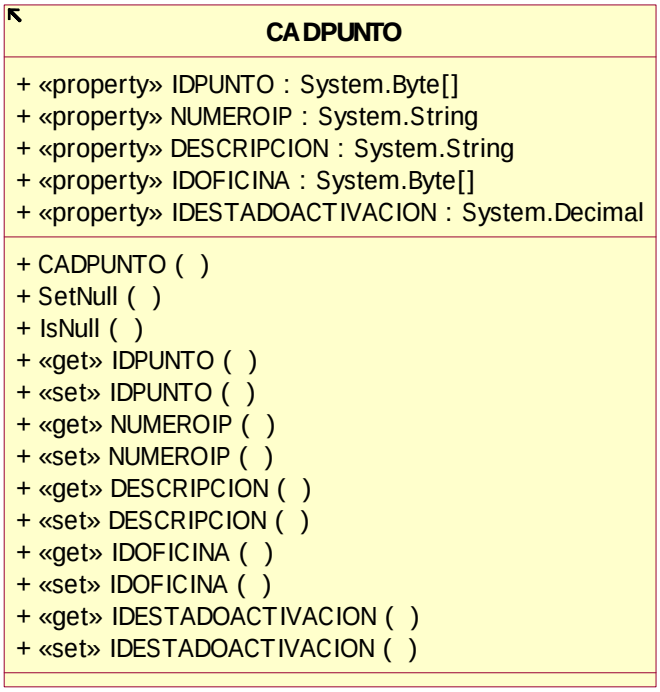
PuntoAplicacion

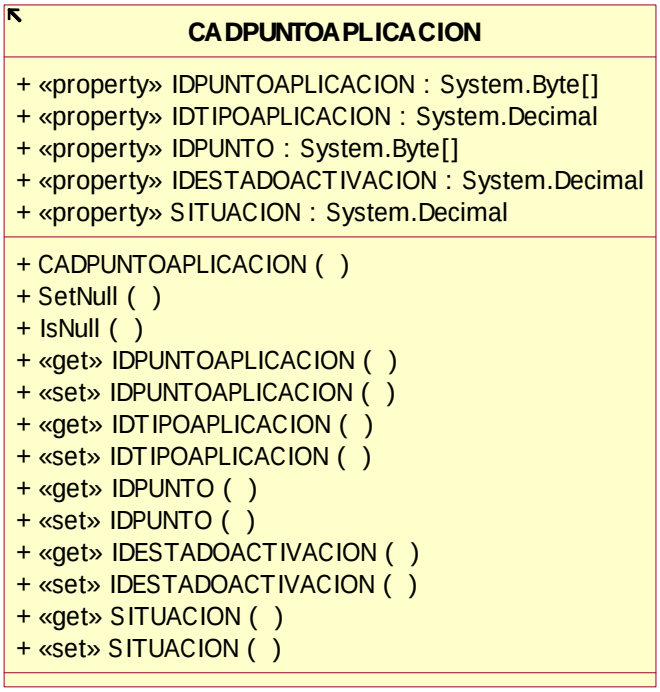
+ «property» IdPuntoAplicacion : Byte[]
+ «property» IdTipoAplicacion : Decimal
+ «property» IdPunto : Byte[]
+ «property» IdEstadoActivacion : Decimal
+ «property» Situacion : Decimal
- idPuntoAplicacion : Byte[]
- idTipoAplicacion : Decimal
- idPunto : Byte[]
- idEstadoActivacion : Decimal
- situacion : Decimal

+ PuntoAplicacion ( )
+ «get» IdPuntoAplicacion ( )
+ «set» IdPuntoAplicacion ( )
+ «get» IdTipoAplicacion ( )
+ «set» IdTipoAplicacion ( )
+ «get» IdPunto ( )
+ «set» IdPunto ( )
+ «get» IdEstadoActivacion ( )
+ «set» IdEstadoActivacion ( )
+ «get» Situacion ( )
+ «set» Situacion ( )

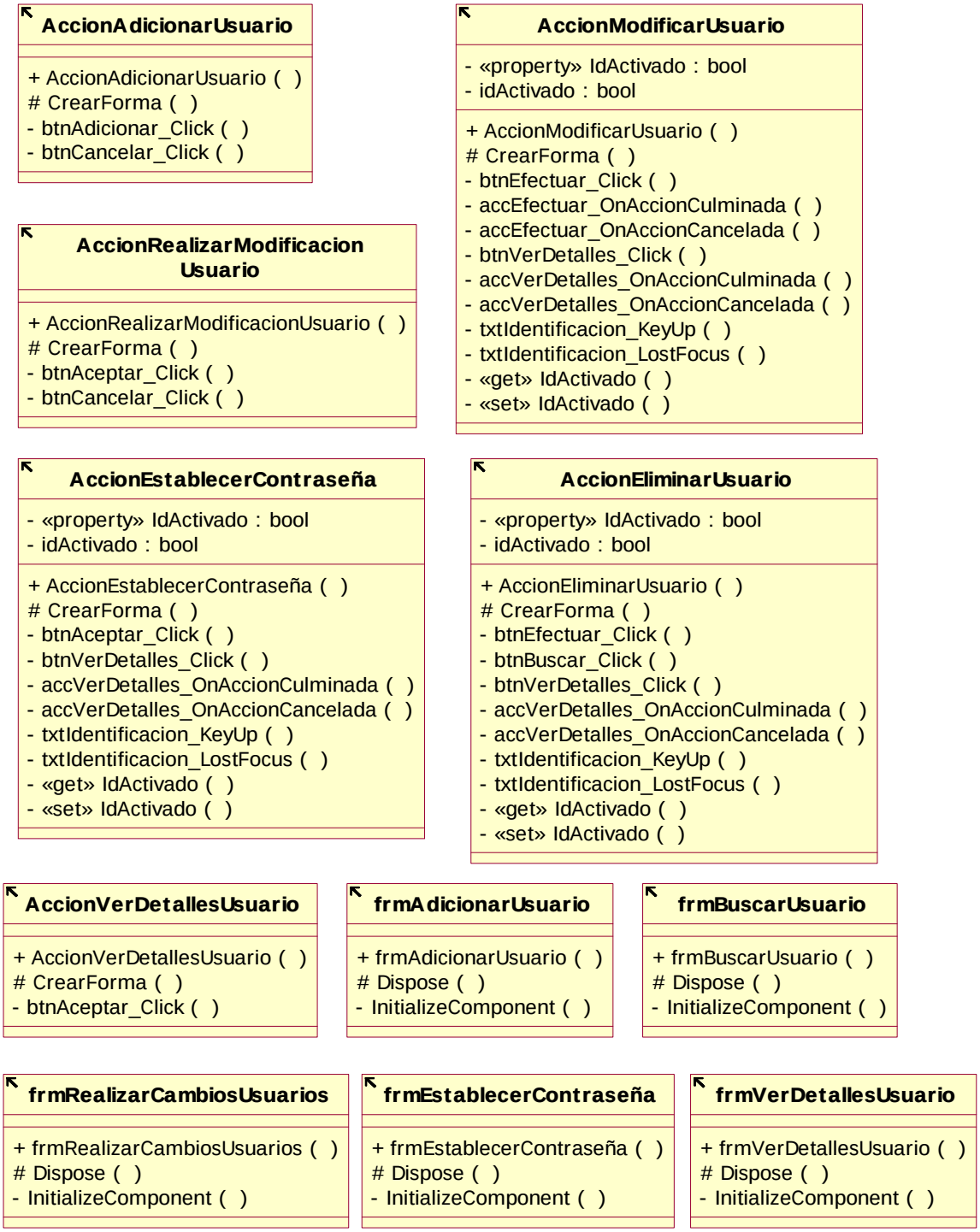

CADPUNTOFactory
+ «property» Connection : System.Data.IDbConnection
+ «property» Transaction : System.Data.IDbTransaction
+ CADPUNTOFactory ( )
+ CADPUNTOFactory ( )
+ CADPUNTOFactory ( )
+ CADPUNTOFactory ( )
+ DetallesDS ( )
+ Detalles ( )
+ LoadPuntApliDispDS ( )
+ LoadPuntApliDisp ( )
+ LoadAllDS ( )
+ LoadAll ( )
+ DeleteColl ( )
+ Save ( )
+ Delete ( )
+ Update ( )
+ Insert ( )
+ Load ( )
+ ProcessDataSet ( )
+ CheckPrimaryKeyValues ( )
+ «get» Connection ( )
+ «get» Transaction ( )

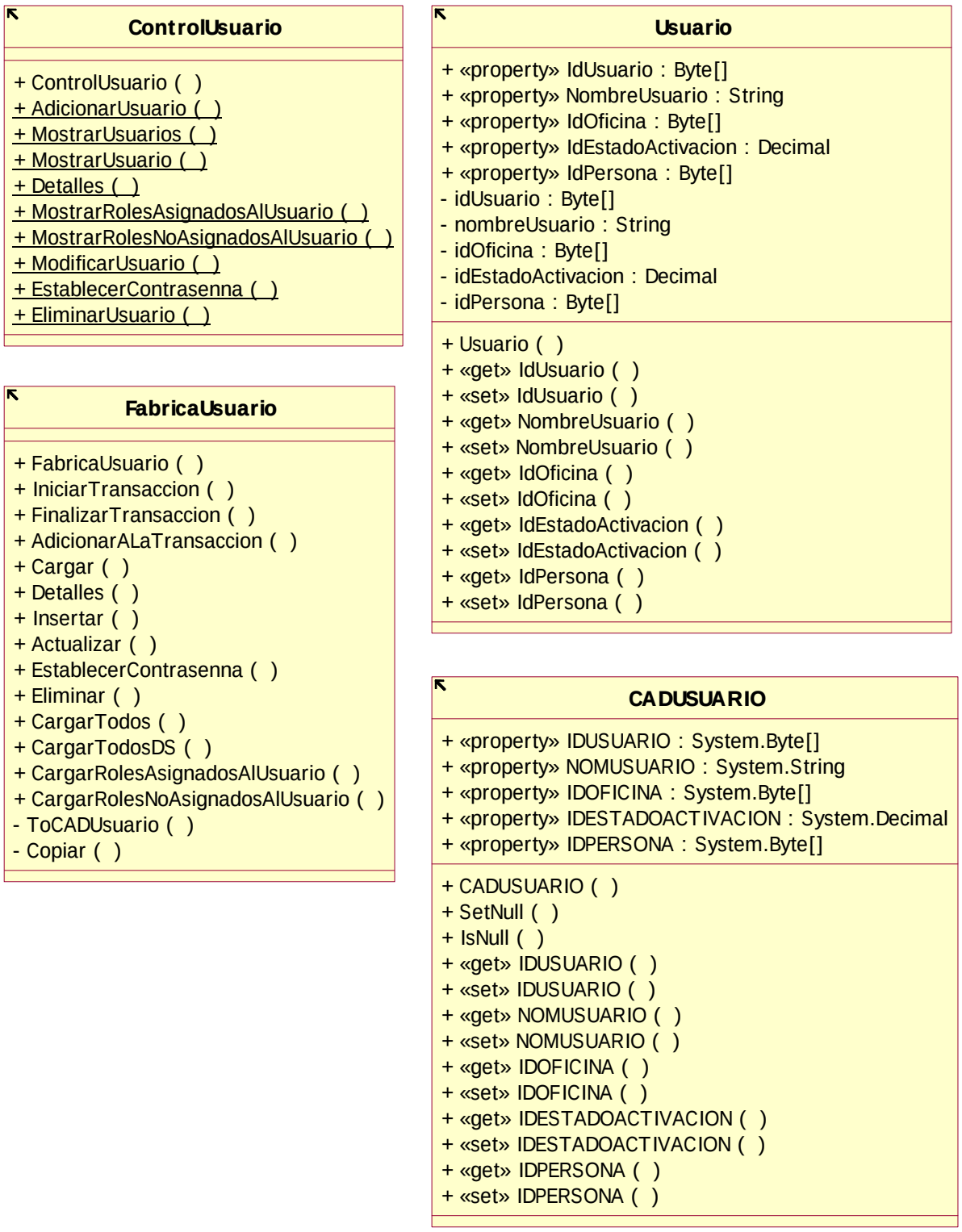

CADPUNTOAPLICACIONFactory
+ «property» Connection : System.Data.IDbConnection
+ «property» Transaction : System.Data.IDbTransaction
+ CADPUNTOAPLICACIONFactory ( )
+ CADPUNTOAPLICACIONFactory ( )
+ CADPUNTOAPLICACIONFactory ( )
+ CADPUNTOAPLICACIONFactory ( )
+ DetallesDS ( )
+ Detalles ( )
+ LoadAllDS ( )
+ LoadAll ( )
+ DeleteColl ( )
+ Save ( )
+ Delete ( )
+ Update ( )
+ Insert ( )
+ Load ( )
+ ProcessDataSet ( )
+ CheckPrimaryKeyValues ( )
+ Delete ( )
+ CheckAlternateKeyValues ( )
+ «get» Connection ( )
+ «get» Transaction ( )

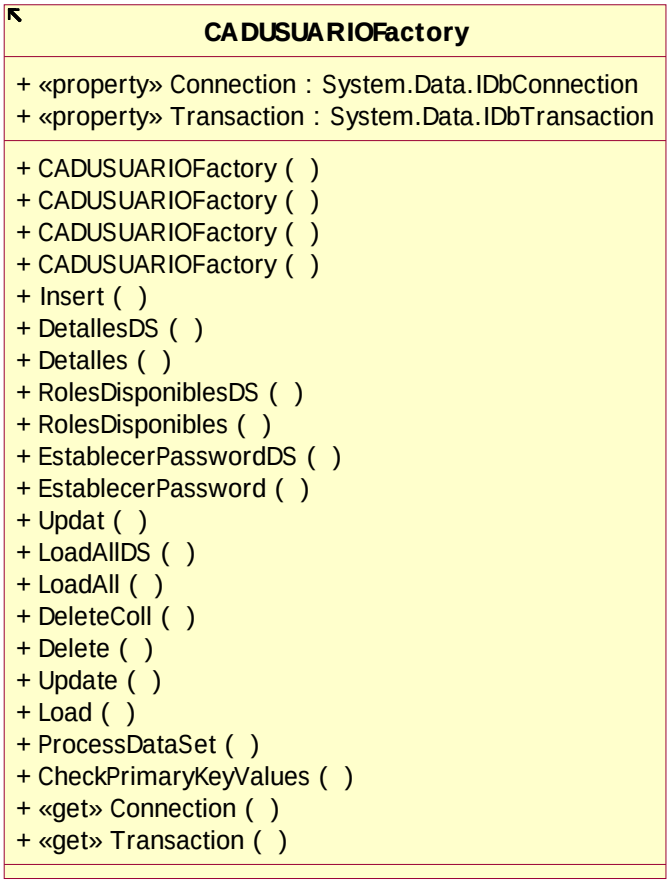

CADPUNTO
+ «property» IDPUNTO : System.Byte[]
+ «property» NUMEROIP : System.String
+ «property» DESCRIPCION : System.String
+ «property» IDOFICINA : System.Byte[]
+ «property» IDESTADOACTIVACION : System.Decimal
+ CADPUNTO ( )
+ SetNull ( )
+ IsNull ( )
+ «get» IDPUNTO ( )
+ «set» IDPUNTO ( )
+ «get» NUMEROIP ( )
+ «set» NUMEROIP ( )
+ «get» DESCRIPCION ( )
+ «set» DESCRIPCION ( )
+ «get» IDOFICINA ( )
+ «set» IDOFICINA ( )
+ «get» IDESTADOACTIVACION ( )
+ «set» IDESTADOACTIVACION ( )


CADPUNTOAPLICACION
+ «property» IDPUNTOAPLICACION : System.Byte[]
+ «property» IDTIPOAPLICACION : System.Decimal
+ «property» IDPUNTO : System.Byte[]
+ «property» IDESTADOACTIVACION : System.Decimal
+ «property» SITUACION : System.Decimal
+ CADPUNTOAPLICACION ( )
+ SetNull ( )
+ IsNull ( )
+ «get» IDPUNTOAPLICACION ( )
+ «set» IDPUNTOAPLICACION ( )
+ «get» IDTIPOAPLICACION ( )
+ «set» IDTIPOAPLICACION ( )
+ «get» IDPUNTO ( )
+ «set» IDPUNTO ( )
+ «get» IDESTADOACTIVACION ( )
+ «set» IDESTADOACTIVACION ( )
+ «get» SITUACION ( )
+ «set» SITUACION ( )

Clases del paquete Control de Usuario




ControlUsuario
+ ControlUsuario ( )
+ AdicionarUsuario ( )
+ MostrarUsuarios ( )
+ MostrarUsuario ( )
+ Detalles ( )
+ MostrarRolesAsignadosAlUsuario ( )
+ MostrarRolesNoAsignadosAlUsuario ( )
+ ModificarUsuario ( )
+ EstablecerContrasenna ( )
+ EliminarUsuario ( )
FabricaUsuario
+ FabricaUsuario ( )
+ IniciarTransaccion ( )
+ FinalizarTransaccion ( )
+ AdicionarALaTransaccion ( )
+ Cargar ( )
+ Detalles ( )
+ Insertar ( )
+ Actualizar ( )
+ EstablecerContrasenna ( )
+ Eliminar ( )
+ CargarTodos ( )
+ CargarTodosDS ( )
+ CargarRolesAsignadosAlUsuario ( )
+ CargarRolesNoAsignadosAlUsuario ( )
- ToCADUsuario ( )
- Copiar ( )
Usuario
+ «property» IdUsuario : Byte[]
+ «property» NombreUsuario : String
+ «property» IdOficina : Byte[]
+ «property» IdEstadoActivacion : Decimal
+ «property» IdPersona : Byte[]
- idUsuario : Byte[]
- nombreUsuario : String
- idOficina : Byte[]
- idEstadoActivacion : Decimal
- idPersona : Byte[]
+ Usuario ( )
+ «get» IdUsuario ( )
+ «set» IdUsuario ( )
+ «get» NombreUsuario ( )
+ «set» NombreUsuario ( )
+ «get» IdOficina ( )
+ «set» IdOficina ( )
+ «get» IdEstadoActivacion ( )
+ «set» IdEstadoActivacion ( )
+ «get» IdPersona ( )
+ «set» IdPersona ( )
CADUSUARIO
+ «property» IDUSUARIO : System.Byte[]
+ «property» NOMUSUARIO : System.String
+ «property» IDOFICINA : System.Byte[]
+ «property» IDESTADOACTIVACION : System.Decimal
+ «property» IDPERSONA : System.Byte[]
+ CADUSUARIO ( )
+ SetNull ( )
+ IsNull ( )
+ «get» IDUSUARIO ( )
+ «set» IDUSUARIO ( )
+ «get» NOMUSUARIO ( )
+ «set» NOMUSUARIO ( )
+ «get» IDOFICINA ( )
+ «set» IDOFICINA ( )
+ «get» IDESTADOACTIVACION ( )
+ «set» IDESTADOACTIVACION ( )
+ «get» IDPERSONA ( )
+ «set» IDPERSONA ( )


CADUSUARIOFactory
+ «property» Connection : System.Data.IDbConnection
+ «property» Transaction : System.Data.IDbTransaction
+ CADUSUARIOFactory ( )
+ CADUSUARIOFactory ( )
+ CADUSUARIOFactory ( )
+ CADUSUARIOFactory ( )
+ Insert ( )
+ DetallesDS ( )
+ Detalles ( )
+ RolesDisponiblesDS ( )
+ RolesDisponibles ( )
+ EstablecerPasswordDS ( )
+ EstablecerPassword ( )
+ Updat ( )
+ LoadAllDS ( )
+ LoadAll ( )
+ DeleteColl ( )
+ Delete ( )
+ Update ( )
+ Load ( )
+ ProcessDataSet ( )
+ CheckPrimaryKeyValues ( )
+ «get» Connection ( )
+ «get» Transaction ( )

Glosario de términos

Órganos de seguridad ciudadana Venezolanos: Entidades del estado que velan por el bienestar social de cada ciudadano:

- La Policía Nacional.
- Las Policías de cada Estado.
- Las Policías de cada Municipio, y los servicios mancomunados de policías prestados a través de la Policía Metropolitana.
- El cuerpo de investigaciones científicas, penales y criminalísticas.
- El cuerpo de bomberos y administración de emergencias de carácter civil.
- La organización de protección civil y administración de desastre.

Centro 171: Centro de Gestión de Emergencias y Seguridad Ciudadana (171).

CU: Casos de Uso.

TNS: (Transparent Network Substrate, Sustrato de red transparente) Permite la comunicación entre los clientes y los servidores de bases de datos Oracle, con independencia del protocolo de comunicaciones que se utilice.

Entidad: Oficina, usuario, estación de trabajo, aplicación y sitios de interés pertenecientes al sistema.