Universidad de las Ciencias Informáticas

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

Título

Desarrollo del módulo de Sistema de colectivos del Sistema Informativo de la Dirección de Establecimientos Penitenciarios (SIDEP).

Autor

Alexander Rodríguez Lozada.

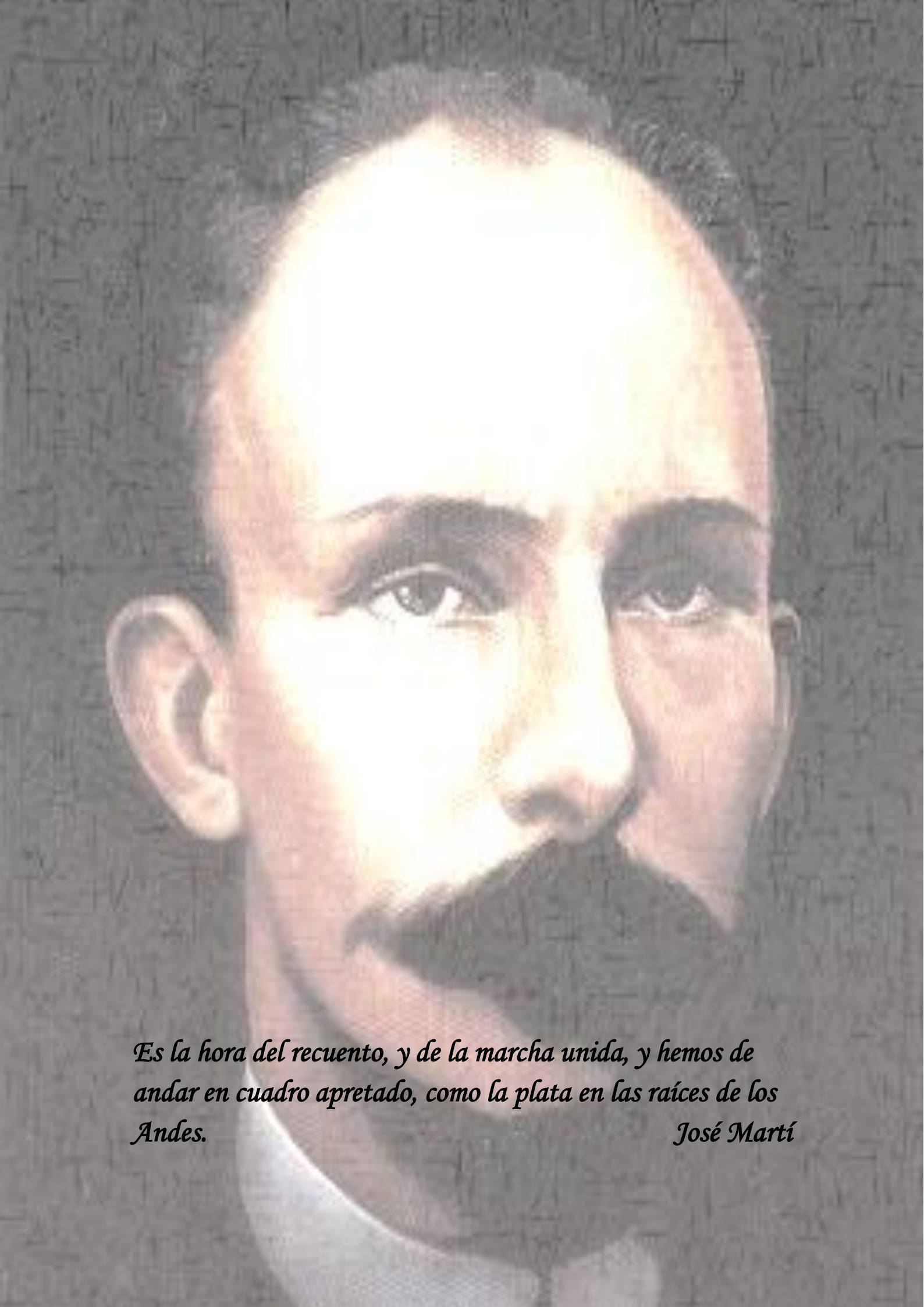
Tutor

MsC. Julio Omar Prieto Entenza.

Co-tutora

Ing. Liset Schery Sánchez.

"Año 54 de la Revolución"



Es la hora del recuento, y de la marcha unida, y hemos de andar en cuadro apretado, como la plata en las raíces de los Andes.

José Martí

DECLARACIÓN DE AUTORÍA

Declaro ser autor del presente trabajo de diploma y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo, con carácter exclusivo.

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

Alexander Rodríguez Losada

_____
Firma dela Autor

Julio Omar Prieto Entenza

_____
Firma del Tutor

Liset Schery Sánchez

_____
Firma dela Cotutora

DEDICATORIA

Dedicado**:**

A Estrella Lozada, porque hay un millón de buenos motivos por los cuales es la primera en esta lista.

A mis padres Nelson Rodríguez García y Leonardo Berrueta Acevedo.

A Sergio Pérez Espinoza.

A mis hermanos Arriannis, Daliana, Aylin y Leonel.

A la Universidad de las Ciencias Informáticas.

A todos los que han estado a mi lado y en mi mente durante todo este tiempo.

AGRADECIMIENTOS

A Estrella Lozada Martínez y a mi novia Katherine Zamora Mena.

A mis tutores, Julio Omar Prieto y Liset Schery.

A Roberto Granda.

A Yaidel Ferrales, Reinier Álamo y Leandro Roura.

RESUMEN

Con el desarrollo de las tecnologías de la informática y las comunicaciones en nuestro país, se ha llevado a cabo en el país un plan de informatización de las instituciones, ya sean empresariales o de otro tipo, proceso en el cual la Universidad de las Ciencias Informáticas (UCI) ha jugado un importante papel durante estos años con el desarrollo de proyectos productivos, dirigidos a prácticamente todas las esferas de la economía, dentro de los que se encuentra el Sistema Informativo de la Dirección de Establecimientos Penitenciarios (SIDEP), creado con el objetivo de mejorar el proceso de toma de decisiones y agregar nuevas funcionalidades con las que no cuenta el actual sistema de gestión que se utiliza en el país.

Este trabajo tiene como objetivo la documentación de todo el proceso de desarrollo del módulo Sistema de colectivos del proyecto SIDEP. La importancia de dicho módulo radica en la gestión del personal agrupado por colectivos de diferentes características, ya sean trabajadores, familiares o internos.

En este documento se recogen los aspectos fundamentales del proceso de desarrollo del módulo Sistema de colectivos, describiendo las herramientas y tecnologías, así como los flujos de información y relaciones establecidas entre los elementos que forman parte del proceso de gestión de la información correspondiente al Sistema de colectivos.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Luego del triunfo de la revolución el 1ro de enero de 1959 el gobierno cubano toma un conjunto de medidas con el objetivo de revertir muchas de las políticas y leyes que había dejado como legado la tiranía. Una de las medidas que tomó, en aras de hacer un ordenamiento jurídico y crear nuevas formas para enfrentar los delitos en correspondencia con los perjuicios que éstos producían, fue la de renovar el sistema penitenciario existente en aquel entonces. Muchas de las antiguas prisiones heredadas de la tiranía, donde el abuso, las torturas, los tratos inhumanos a los internos, así como los maltratos (los cuales constituían los métodos y procedimientos que caracterizaban la estancia de los detenidos en la prisión), fueron desactivadas y se construyó un sistema penitenciario totalmente humano, basado en el respeto y en una gestión de control inspirada en la regulación de normas y leyes internacionales en cuanto al tratamiento a los reclusos. Durante estos años de revolución el sistema penitenciario actual ha sufrido un sin número de transformaciones que no solo han posibilitado cumplir con los fines de la sanción penal, sino que también han ayudado a situar a nuestro sistema de justicia entre los de mayor carácter humanista del mundo. Entre sus fortalezas se encuentra el perfeccionamiento de la legislación penitenciaria y su base reglamentaria.

En el año 1989 comienza en Cuba el sistema de informatización de los centros penitenciarios, con la automatización de los datos principales del recluso y ciertos aspectos de control penal. Pero no es hasta algunos años más tarde que, a raíz del cumplimiento de la orden 43/99 del Vice Ministro Primero del Ministerio del Interior (MININT), se crea el Sistema Automatizado para el Control del Recluso (SACORE) el cual comenzó a dar respuesta en gran medida a muchos de los problemas descritos anteriormente.

Dicho sistema culminó su desarrollo a finales del 2002, poniéndose en marcha a principios del 2003 y cuenta en la actualidad con tres módulos principales: Control Penal, Reeducación Penal y el Orden Interior, los cuales aumentan las especificaciones de la automatización de los datos principales del recluso y los aspectos de control penal existentes, además de adicionar un mayor número de facilidades. En los nueve años de explotación del sistema, a pesar de sus facilidades y ayuda brindada a dichos centros, aún cuenta con requisitos incompletos o pendientes.

La Universidad de Ciencias Informáticas (UCI), como vanguardia en el desarrollo de las tecnologías de la información, ha asumido la modernización informática del sistema

penitenciario cubano a través del proyecto Sistema Informativo de la Dirección de Establecimientos Penitenciarios (SIDEP), siguiendo al pie de la letra todas las disposiciones legales del MININT, los Órganos de Justicia y del Estado. Uno de los procesos que aún no se han implementado es el del sistema de colectivos, que incluye al Educador Guía, el Consejo de Educadores, el Consejo de Internos y el Consejo de Familia, lo que constituye la estructura a través de la cual se organizan los internos con el objetivo de brindarles una mejor atención a su educación y garantizar su reinserción a la sociedad. Dichos consejos tienen como objetivo fundamental la discusión de problemas, elaboración de planes de actividades, toma de decisiones, entre otras, las cuales se registran en encuentros que se realizan periódicamente cada uno de estos consejos.

Actualmente el sistema de colectivos presenta determinados problemas que atentan en contra de su correcto funcionamiento:

- Lentitud en la entrega de actas, debido a que el proceso se realiza manualmente y se produce aglomeración de la documentación, por lo que el tiempo de entrega de estas en muchos casos se extiende más del tiempo requerido.
- Difícil acceso a la documentación de los encuentros que realizan los diferentes consejos, ya que la información no está centralizada en un mismo lugar y la consulta de actas se convierte en un proceso engorroso.

Derivado de estos problemas se hace necesaria la automatización del proceso de sistema de colectivos, para la obtención de una solución informática acorde a lo antes planteado.

Debido a lo anterior se plantea como **problema a resolver**:

- ¿Cómo gestionar la información referente a los elementos que integran la estructura organizativa de los internos en el sistema de colectivos del SIDEP?

Esto lleva a definir como **objeto de estudio**:

- Los procesos de tratamiento educativo en los sistemas penitenciarios.

Como **objetivo general** se define:

Desarrollar el módulo Sistema de colectivos del proyecto SIDEP a partir de las especificaciones de casos de usos definidas durante la fase de análisis y haciendo uso de la arquitectura, tecnologías y herramientas definidas por el proyecto.

Enmarcado en el **campo de acción**:

- La gestión de los procesos de sistema de colectivos en el sistema penitenciario cubano.

Para dar cumplimiento al objetivo general, se desglosó en los siguientes **objetivos específicos**:

- Realizar el estudio previo referente a los elementos utilizados para el desarrollo de la solución.
- Diseñar el módulo Sistema de colectivos.
- Implementar el módulo Sistema de colectivos.
- Realizar pruebas a la solución propuesta.

Para el correcto cumplimiento de los objetivos específicos es necesario definir una serie de **tareas de la investigación**:

- Estudio y análisis de las tendencias actuales de los sistemas penitenciarios en el mundo.
- Descripción de las tecnologías y herramientas a utilizar en el diseño e implementación del módulo Sistema de colectivos.
- Definición de patrones de diseño y arquitectura para la posible solución.
- Desarrollo del diagrama de clases del diseño correspondiente al módulo Sistema de colectivos.
- Realización del diseño del Modelo de Datos asociado a las funcionalidades del módulo Sistema de colectivos.
- Implementación del diseño del módulo Sistema de colectivos.
- Desarrollo del diagrama de componentes del módulo Sistema de colectivos.
- Diseño de los casos de prueba correspondientes al módulo Sistema de colectivos.
- Ejecutar las pruebas del módulo Sistema de colectivos.

Teniendo la siguiente **idea a defender**:

Con el desarrollo del módulo Sistema de colectivos del proyecto SIDEP se mejorará la forma de organización de los internos en el sistema penitenciario para su tratamiento educativo correspondiente.

La realización de las tareas estuvo sustentada por un conjunto de **métodos de investigación**:

Método Analítico - Sintético: en este caso se combinan dos métodos relacionados estrechamente entre sí. En primer lugar, el análisis es una operación intelectual que posibilita descomponer mentalmente un todo complejo en sus partes y cualidades; por otro lado el análisis permite la división mental del todo en sus múltiples relaciones y componentes. La síntesis es la operación inversa, que establece mentalmente la unión entre las partes, previamente analizadas, y posibilita descubrir relaciones y características generales entre los elementos de la realidad. Este método se ve reflejado durante el proceso de investigación para el desarrollo del módulo Sistema de colectivos, debido a que no existe un sistema penitenciario con características similares a este y se hace necesario estudiar los componentes y factores que intervienen dentro del proceso, haciendo una síntesis sobre las características de los mismos y por consiguiente, mediante el análisis determinar la influencia de las características de estos elementos, su repercusión y comportamiento dentro del proceso de gestión.

Método Inductivo – Deductivo: la inducción la podemos definir como una forma de razonamiento por medio de la cual se pasa del conocimiento de casos particulares a un conocimiento más general que refleja lo que hay de común en los fenómenos individuales. Por su parte, la deducción es una forma de razonamiento, mediante el cual se pasa de un conocimiento general a otro de menor nivel de generalidad. Estos métodos fueron utilizados conjuntamente ya que uno implica al otro. En el caso del módulo Sistema de colectivos no se conoce un sistema penitenciario con las características organizativas similares a este, sin embargo sí existen otros sistemas de organización sociales que se asemejan en cuanto al funcionamiento del sistema de colectivos

Método Modelación: el modelo científico es un instrumento de la investigación de carácter material o teórico, creado por los científicos para reproducir el fenómeno que se está estudiando. El modelo es una reproducción simplificada de la realidad, que cumple una función heurística, ya que permite descubrir y estudiar nuevas relaciones y cualidades del objeto de estudio. La modelación es justamente el proceso mediante el cual creamos modelos con vistas a investigar la realidad. Este método se ve reflejado en la modelación de los diferentes diagramas de los flujos que se desarrollan en este trabajo.

Como **posibles resultados** se obtendrá lo siguiente:

- Módulo Sistema de colectivos para la plataforma del proyecto SIDEP.
- Modelo de Diseño del módulo Sistema de colectivos.

- Modelo de Implementación del módulo Sistema de colectivos.

El presente trabajo de diploma está estructurado de la siguiente forma:

Capítulo 1: Marco teórico de la investigación.

Se definen aspectos fundamentales del proceso de investigación, se especifica y describe la estrategia de investigación y los métodos a utilizar en la misma, así como las herramientas, tecnologías, lenguajes de programación y metodología de desarrollo a utilizar durante la confección del módulo Sistema de colectivos del proyecto SIDEP.

Capítulo 2: Diseño del sistema.

Aborda todo lo referente al diseño propuesto para la solución del software, así como la descripción de la arquitectura a utilizar. Además, se especifica el conjunto de funcionalidades a desarrollar dando un breve resumen de las mismas.

Capítulo 3: Implementación y pruebas.

Se tratan los aspectos referentes al desarrollo y validación del módulo Sistema de colectivos, se exponen características de los componentes que integran el módulo, así como las relaciones internas y externas con otros elementos del sistema. Además, se exponen los resultados relativos a las pruebas realizadas.

CAPÍTULO1. Marco teórico de la investigación

1.1. Introducción

En este capítulo se realizará una caracterización de algunos sistemas penitenciarios en uso actualmente, además de la descripción de las herramientas y tecnologías, así como los lenguajes de programación utilizados en el desarrollo del módulo Sistema de colectivos del proyecto SIDEP.

1.2. Sistemas penitenciarios

Manuel Osorio, creador del diccionario de Ciencias Jurídicas, Políticas y Sociales, asocia el término sistema penitenciario con régimen penitenciario, definiéndolo como: "Conjunto de normas legislativas o administrativas encaminadas a determinar los diferentes sistemas adoptados para que los penados cumplan sus condenas. Se encamina a obtener la mayor eficacia en la custodia o en la readaptación social de los delincuentes. Esos regímenes son múltiples, varían a través del tiempo y van desde el aislamiento absoluto y de tratamiento rígido, hasta el sistema de puerta abierta con libertad vigilada" (1). Se puede concluir que el sistema penitenciario es el conjunto de normas generales, establecidas y específicas referidas a las penas en sí, el modo de su cumplimiento y el tratamiento de los penados y procesados.

Cada gobierno define la estructura de su sistema penitenciario de acuerdo a la legislación y las condiciones reales que posee. En el caso de la República de Cuba, tal sistema está constituido por la legislación vigente, los métodos que se emplean para lograr su funcionamiento, las diferentes dependencias encargadas de su aplicación, los equipos de trabajo y la infraestructura carcelaria. El Reglamento del Sistema Penitenciario de Cuba establece que el sistema penitenciario cubano es el encargado de garantizar el proceso de ejecución de la sanción de privación de libertad, de la sanción de trabajo correccional con internamiento, la medida de seguridad reeducativa de internamiento y la medida cautelar de prisión provisional. Este sistema, dirigido por la Dirección de Establecimientos Penitenciarios (DEP) del MININT, se sustenta en la integración de principios, conceptos, procedimientos, fuerzas y medios que garantizan el funcionamiento de los centros destinados al internamiento y el tratamiento de los internos.

Una iniciativa en el sistema penitenciario cubano es la creación de un sistema de colectivos, como un componente importante en el proceso de readaptación y reinserción a la sociedad de cada uno de los internos en los penales. El objetivo de organizar el personal por colectivos es mantener una mejor organización y control

sobre las actividades que estos realicen. Un colectivo está formado por tres consejos, dependientes entre sí, uno es para los internos (Consejo de Internos), otro para los familiares de los internos que quieran formar parte de él (Consejo de Familia) y un consejo de educadores (Consejo de Educadores), formado por trabajadores del centro, específicamente del sistema educativo. Estos consejos realizan reuniones donde se hacen planteamientos; a través de las cuales los integrantes de los consejos plantean sus inquietudes y toman acuerdos a los que le son asignados responsables con el objetivo de garantizar su cumplimiento.

1.3. Sistemas informáticos penitenciarios

Como parte del desarrollo tecnológico, son varios los países que cuentan con sistemas informáticos capaces de apoyar la toma de decisiones y la gestión de los procesos que se realizan en los centros penitenciarios. Para el desarrollo del módulo Sistema de colectivos del SIDEP se tuvieron en cuenta varios sistemas informáticos. Algunos de ellos son:

- Sistema de Gestión Penitenciaria (SIGPEN).
- Sistema Penitenciario del Gobierno de Panamá.
- Establecimiento Penitenciario Virtual en Red (EPVNET).
- Sistema de Gestión Penitenciario (SIGEP).
- Sistema Automatizado para el Control del Recluso (SACORE).

Sistema de Gestión Penitenciaria (SIGPEN)

El SIGPEN, perteneciente a Ecuador, facilita el seguimiento y manejo de las actividades que realizan los Centros de Rehabilitación Social (CRS) en cada una de sus áreas, logrando de esta manera tener un control adecuado y oportuno de la información de las personas privadas de libertad.

El Ministerio de Justicia, Derechos Humanos y Cultos conjuntamente con la Dirección Nacional de Rehabilitación Social están en continuo monitoreo de las actividades realizadas en los CRS por medio del sistema SIGPEN, el cual permite obtener datos estadísticos reales de la situación de los sistemas penitenciarios, logrando de esta manera tomar decisiones eficaces y oportunas para mejorar el ambiente carcelario del país.

Está compuesto por diferentes secciones (Interno, Departamento Jurídico, Departamento Médico, entre otras) las cuales brindan toda la información referente al estado de los reclusos en la sección elegida. (28)

Sistema Penitenciario del Gobierno de Panamá.

Este sistema fue creado a principios de año 1997, como resultado de un proyecto financiado por las Naciones Unidas y el gobierno español en coordinación con la Dirección General de Sistemas Penitenciarios de Panamá (DGSP). La finalidad del software es mantener en una base de datos los registros de los internos que están detenidos en los centros penales a nivel nacional.

El proceso se inicia con la captura de la información por parte del personal ubicado en el centro penal. Esta información se refiere a los datos personales del interno, antecedentes médicos, datos socioeconómicos y jurídicos y algún otro dato de importancia.

La información capturada se registra automáticamente en una base de datos Oracle que radica en la sede central y la cual está disponible para los diferentes departamentos que tiene acceso al sistema. Esto garantiza que se refleje de forma inmediata los cambios en el expediente del interno tales como traspasos de autoridad, traslados de un centro a otro, libertades, diligencias médicas, jurídicas y la realización del cómputo automático de la sentencia.

Las limitaciones se evidencian en los centros que no poseen enlace aún con la dirección central lo que debe llevar a una transformación de la red externa, reestructuración de la red interna de la sede, la dotación de internet a la institución, la actualización de toda la información de los centros penitenciarios y actualización de los equipos.(29)

Establecimiento Penitenciario Virtual en Red (EPVNET).

EPVNET es un proyecto de software libre para la gestión informatizada de centros penitenciarios. Tiene como precedente en España algunas aplicaciones basadas en la arquitectura Cliente-Servidor y bajo entorno Windows que cubren algunas áreas o nóminas de internos, empleados, la productividad de los talleres, tramitación de los expedientes penales y penitenciarios de los internos; pero con la limitante de no contemplar áreas de gestión como la encargada de la supervisión de los individuos para su reinserción en la sociedad.

En el año 2004 se convierte en un proyecto de software libre con licencia GNU/GPL, para que pudiera ser utilizado por otras personas. Para la realización del mismo se utilizó como lenguaje de programación PHP 4 y como gestor de base de datos PostgreSQL. Puede ser utilizado en sistemas operativos como Linux y Windows.

El EPVNET se encuentra basado en la legislación española, garantizando el control de los movimientos, de comunicaciones y de informes que se realizan en los centros penitenciarios. Tiene automatizado los procesos de: Ingresos y Salidas, Centro de Control, Comunicaciones, Seguridad y Accesos, Situaciones Regimentales, Incidentes Regimentales, Registros de Correspondencia Oficial de Entrada y Salida y Requisas, entre otras; según el propio creador.(30)

Sistema de Gestión Penitenciario (SIGEP).

SIGEP es un software diseñado e implementado para el sistema penitenciario de Venezuela. Contiene un módulo para la Instrucción Escolar, el cual permite registrar los distintos tipos de actividades educativas que se desarrollan en los Establecimientos Penitenciarios. El módulo gestiona la matrícula de las diferentes actividades educativas, permite mantener un registro de la asistencia de los reclusos a las actividades y registrar las evaluaciones finales. En este caso los reclusos no son organizados por colectivos y consejos, sino que se registran las actividades que realizan en conjunto.

Sistema Automatizado para el Control del Recluso (SACORE).

El Ministerio del Interior (MININT) no está ajeno a las tecnologías informáticas, por lo que se ha trazado como objetivo reorganizar y facilitar el trabajo dentro de las instituciones penales del país, garantizando la gestión y control de los datos automatizando el sistema penitenciario.

Con la realización de este sistema, el MININT abre paso al desarrollo tecnológico e informático utilizando nuevas herramientas automatizadas que complementan la información del SACORE. Aún con el uso de dicho sistema muchos datos son manejados y conservados en formato duro, o por herramientas informáticas que no ofrecen grandes facilidades para su gestión pero de manera independiente, por lo que no se garantiza rapidez en dicho proceso.

Análisis general

Muchos de estos sistemas no se encuentran actualizados ya que existen muchos centros que por falta de infraestructura no están conectados a la base de datos central. También existen sistemas cuyos centros penitenciarios no poseen interconexión, imposibilitando la integración y la centralización de las informaciones que se generan. Además, casi todos reflejan la situación legislativa y judicial

perteneciente a un país específico, lo cual no permite que estos sean adaptados para su uso en otros sistemas penitenciarios.

Ninguno de estos sistemas contemplan el trabajo en colectivos (tanto de los reclusos, como de los familiares y los supervisores) enfocados como un proceso integrado. Solo se contemplan procesos elementales de los centros penitenciarios y las actividades referentes a los colectivos se encuentran dispersas en varias funcionalidades.

Con la gestión de todo el trabajo relacionado con los colectivos, reuniones y otras actividades de los internos, se evalúa sistemáticamente la conducta del interno en el medio en que se desenvuelve. Siendo esto un punto fundamental para la rebaja adicional de la sanción por excepcional conducta y resultados relevantes, permitiendo un control de su comportamiento de manera más eficiente y con mínimas posibilidades de error.

1.4. Tecnologías y herramientas utilizadas para el desarrollo de la aplicación

1.4.1. Metodología de Desarrollo

Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información. Está formada por fases, cada una de las cuales se puede dividir en sub-fases, que guiarán a los desarrolladores de sistemas a elegir las técnicas más apropiadas en cada momento del proyecto y también a planificarlo, gestionarlo, controlarlo y evaluarlo.

Rational Unified Process (RUP)

En la arquitectura del proyecto se definió como metodología de desarrollo RUP. Esta constituye una metodología estándar utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en un número variable según el proyecto y en las que se hace un mayor o menor hincapié en las distintas actividades. Las fases que RUP propone son: Inicio, Elaboración, Construcción y Transición. Esta metodología define los roles y sus respectivas actividades, los productos que se deben generar y los flujos de trabajo (2).

1.4.2. Lenguaje de Modelado

Unified Modeling Language (UML)

UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

Es importante resaltar que UML es un lenguaje de modelado para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir (2).

1.4.3. Herramienta CASE

Visual Paradigm Suite 3.4

Para llevar a cabo una de las fases fundamentales en el desarrollo de software, que es la fase de modelado y diseño, se hace uso de la herramienta Visual Paradigm Suite 3.4, por las funcionalidades que brinda para el desarrollo en lo que se refiere a ingeniería de software.

Esta herramienta está desarrollada por Visual Paradigm Internacional, una de las compañías más prestigiosas en el desarrollo de herramientas CASE (por sus siglas en inglés Ingeniería de Software Asistida por Computadora). Su mayor éxito consiste en la capacidad de ejecutarse sobre diferentes sistemas operativos, lo que le confiere la característica de ser multiplataforma. Visual Paradigm utiliza UML como lenguaje de modelado ofreciendo soluciones de software que permiten a las organizaciones desarrollar las aplicaciones de calidad más rápido y más barato. Es muy fácil de usar y presenta un ambiente gráfico agradable para el usuario. Su notación es muy parecida a la estándar, permite configurar las líneas de redacción, el modelado de base de datos, el modelado de requerimientos, el modelado del proceso de negocio, la interoperabilidad, la generación de documentación y la generación de código base para diferentes lenguajes de programación como Java, C# y PHP, además de permitir la integración con herramientas de desarrollo (3).

1.4.4. Plataforma de desarrollo

Java 2 Enterprise Edition (J2EE)

Las empresas productoras de software y servicios en la actualidad se ven obligadas a incluir nuevas funcionalidades con el fin de satisfacer las necesidades del usuario, estos servicios deben cumplir con ciertos requisitos:

- Alta disponibilidad, de forma tal que el servicio pueda ser usado sin inconvenientes la mayoría del tiempo.
- Seguridad, para asegurar la privacidad de los usuarios y la integridad y confidencialidad de las transacciones y la información procesada.
- Escalabilidad, que garantice que los servicios seguirán operativos aunque el número de usuarios, de transacciones o el volumen de información sufran aumentos importantes.

El proceso de adaptación de estos requisitos y las funciones en un servicio nuevo consume tiempo, factor en el cual es favorable el uso de esta plataforma por las facilidades que brinda dada la rapidez con que es posible el desarrollo sobre la misma.

Es una plataforma de programación, parte de la Plataforma Java, para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java con arquitectura de N capas distribuidas, y que se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. Habilita una plataforma que reduce de manera significativa los costos y la complejidad de desarrollo de soluciones multicapas, resultando en servicios que pueden ser desarrollados rápidamente y ampliados fácilmente. J2EE añade el soporte completo para componentes Enterprise Java Beans, el API Java Servlets y la tecnología Java Server Pages. El estándar J2EE incluye todas las especificaciones y pruebas de conformidad que permiten la portabilidad de las aplicaciones a través de la amplia gama de sistemas empresariales compatibles con J2EE (4).

1.4.5. Framework

Dojo Toolkit

Dojo es un framework que contiene *Apis* y *widgets* (*controles*) para facilitar el desarrollo de aplicaciones *Web* que utilicen tecnología AJAX. Contiene un sistema de empaquetado inteligente, los efectos de interfaz de usuario, *drag and drop Apis*, *widget Apis*, abstracción de eventos, almacenamiento de *Apis* en el cliente, e interacción de *Apis* con AJAX. Resuelve asuntos de usabilidad comunes como pueden ser la navegación y detección del navegador, soportar cambios de URL en la barra de URLs para luego regresar a ellas, y la habilidad de degradar cuando AJAX/JavaScript no es completamente soportado en el cliente. Es conocido como "la navaja suiza del ejército de las bibliotecas Javascript". Proporciona una gama más amplia de opciones en una sola biblioteca JavaScript y es compatible con navegadores antiguos (5).

Grails

Es un framework creado principalmente para el desarrollo web, se ejecuta sobre plataforma Java, lo que aumenta su uso como como software multiplataforma e interpreta varios lenguajes, es posible escribir en lenguaje Java, Groovy o un híbrido de estos dos (6).

Está construido sobre cinco tecnologías fundamentales del desarrollo web (7):

- Groovy: para la creación de propiedades y métodos dinámicos en los objetos de la aplicación (8).
- Spring: para los flujos de trabajo e inyección de dependencias
- GORM: para la persistencia del dominio.
- SiteMesh: para la composición de la vista.
- Ant: para la gestión de procesos de desarrollo.

Sus principales características son:

- Alta productividad: la inexistencia de configuración XML, un entorno de desarrollo preparado para funcionar desde el primer momento y el empleo de métodos dinámicos, basándose en Groovy, hacen que presente un productividad comparada a los framework más tradicionales en Java.
- Integración con la plataforma Java: al estar construido sobre la plataforma Java, con lo que es muy fácil integrarlo con librerías Java, Framework y código existente. La mejor característica que Grails ofrece en este ámbito es una integración transparente con clases mapeada mediante el Framework Hibernate ORM. Esto significa que aplicaciones existentes que utilicen Hibernate pueden utilizar Grails sin recompilar el código o reconfigurar las clases Hibernate, aprovechando los métodos de persistencia que se mencionan anteriormente. Una consecuencia es que se puede utilizar scaffolding con las clases Java mapeadas con Hibernate. Otra consecuencia es que las capacidades de Grails están totalmente disponibles para estas clases y las aplicaciones que las usan.
- Persistencia: el modelo de datos en Grails, denominado GORM, se basa en una integración de Hibernate y los métodos dinámicos de Groovy. Esta librería permite una fácil manipulación de las clases de dominio y de acceso a datos.

Con la visión de convertirse en un marco de trabajo altamente productivo no puede estar ajeno al uso de patrones por lo que utiliza como principales paradigmas en este sector dos patrones fundamentales: Convención sobre Configuración o COC (por sus

siglas en inglés, *Convention Over Configuration*) y DRY (por sus siglas en inglés, *Don't Repeat Yourself*).

COC es un paradigma de programación de software que busca disminuir el número de decisiones que un desarrollador necesita hacer, ganando así en simplicidad pero no perdiendo flexibilidad por ello. DRY es una filosofía de definición de procesos que promueve la reducción de la duplicación. Ambos patrones en general proporcionan un entorno de desarrollo estandarizado y ocultan, en gran parte, detalles de configuración (7).

Con la utilización del framework Spring, como componente de Grails, se utilizan diferentes patrones de dicho framework que aportan un mejor diseño e implementación al sistema. Un ejemplo de esto es la utilización del patrón Modelo Vista Controlador (MVC). Spring-MVC es uno de los módulos del framework de Spring, y como su nombre indica, implementa una arquitectura MVC que se utilizó como base para desarrollar la capa de presentación de la aplicación (10).

La inversión de control (IoC) es otro patrón utilizado en Grails, según el cual las dependencias de un componente no deben gestionarse desde el propio componente para que este solo contenga la lógica necesaria para hacer su trabajo (6).

1.4.6. Arquitectura en Capas según Grails

La arquitectura de Grails está conformada por 3 capas lógicas principales: Capa Web, Capa de Servicios y Capa de Datos, donde cada capa está separada de la siguiente e interactúan mediante interfaces que definen funcionalidades que la misma debe brindar o también llamadas fachadas, las cuales aseguran que el acoplamiento sea el más bajo posible y la abstracción del funcionamiento de la capa inferior, sea casi total. Cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. Esta arquitectura en capas por su diseño proporciona la facilidad de modificar cada capa todo lo posible sin infligir daños o alteraciones a la inmediata (6). A continuación se brinda una descripción de las mismas:

Capa Web: en esta capa se encuentran las vistas y la lógica de presentación, las cuales, según la arquitectura que propone Grails, estarán enmarcadas sus clases en los paquetes Controladores, el cual contendrá las clases controladoras, y Vista y Lógica de Presentación, donde se encontrarán las *Groovy Server Pages* o más conocidas como GSP. En la lógica de presentación se manejará todo el flujo web utilizando la implementación del patrón MVC que brinda Grails mediante Spring MVC.

Esto permite cambiar fácilmente el aspecto de la aplicación, sin modificar su comportamiento. La capa de presentación se compone principalmente de: controladores, modelo y vistas.

Controlador: un controlador de Grails es una clase responsable por el manejo de los pedidos provenientes de la aplicación. El controlador recibe la petición, realiza algún trabajo potencial con la misma y finalmente decide que sucederá a continuación, lo cual puede incluir algunos de los siguientes flujos:

- Ejecutar otra función de controlador.
- Mostrar una vista.
- Mostrar información directamente con la respuesta de la petición.

Un controlador es prototipado, lo cual significa que una nueva instancia es creada por cada petición, por tanto los desarrolladores no necesitan manejarlos en modo “singleton”. Proveen la entrada principal para cualquier aplicación de Grails, coordinando los pedidos entrantes, delegando hacia los servicios o clases de dominio para la lógica de negocios y renderizando las vistas.

Modelo: Una de las actividades fundamentales llevadas a cabo por los controladores es obtener los datos que serán mostrados en la vista. El controlador puede recoger esta información directamente, delegarla a algún servicio u otro componente comprendido en la capa de acceso a datos; esta información es pasada a la vista en forma de un mapa u objeto de información. Dicho objeto representa el modelo.

Vista: Grails utiliza para la interacción con el usuario la tecnología JSP, pero basada en una implementación mediante GSP, que es una extensión de JSP y puede incluir Groovy. Grails permite a los desarrolladores mezclar etiquetas de lenguajes de marcas tradicionales como HTML con código Java para producir vistas dinámicas. Las Vistas son los recursos que junto al modelo generado por los controladores le permiten al cliente visualizar la información, estos pueden ser páginas HTML, documentos en formato PDF, hojas de cálculo, entre otras. Las mismas están representadas en el paquete de clase Vista y Lógica de Presentación.

Capa de servicios: En esta capa se encapsula toda la lógica de la aplicación en fachadas de negocio que son utilizadas por los controladores en la capa de presentación y se exponen algunos procesos de negocio a través de interfaces de servicios. Sus clases radicarán según la arquitectura propuesta por Grails en el paquete Servicios. A estas fachadas de negocio se le aplican la seguridad a nivel de

métodos y de objetos de negocio, auditorías, cache, política de transacciones, entre otros.

Capa de datos: Maneja los objetos de acceso a datos abstrayéndolos del mecanismo de persistencia usado, a través de interfaces que exponen las operaciones de persistencia. Grails, para evitar trabajar directamente con un gestor de base de datos y sus tablas y permitir trabajar con objetos en su lugar, utiliza Hibernate 3 como una herramienta ORM, pero esta vez, dada la naturaleza dinámica de Grails y la adopción del convenio sobre la configuración, crea sobre una versión superior de una nueva implementación de Hibernate llamado Grails Objeto Mapeo Relacional (GORM) que simplifica el trabajo con Hibernate y elimina cualquier configuración externa (6).

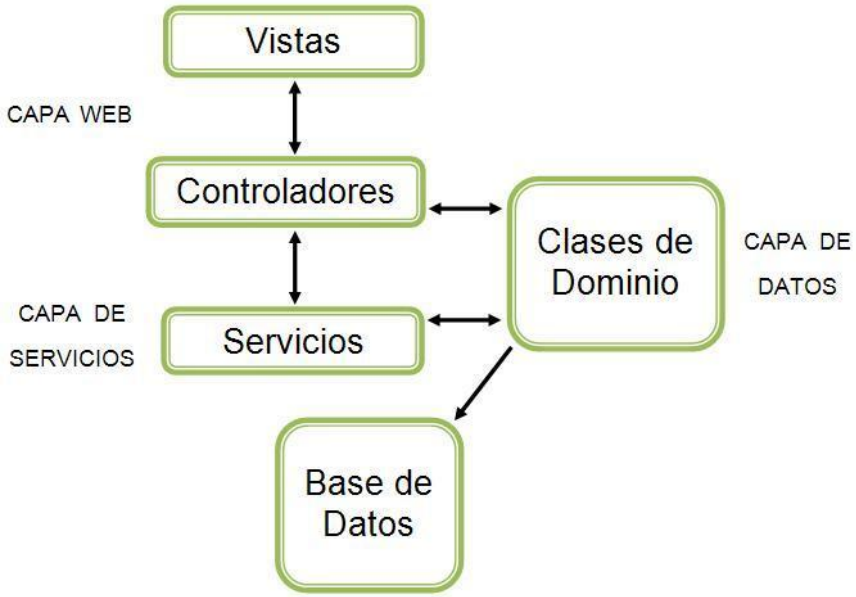


Figura 1 Arquitectura en capas según Grails

Otro elemento importante dentro del entorno Grails es el empleo de un marco de trabajo para el uso de módulos como plugins. Estos pueden ser librerías externas basadas en Java o módulos que separan tanto la lógica como, estructuralmente, diversas partes de una aplicación. Estas nuevas funcionalidades pueden ser desde lógica de negocio, nuevas etiquetas para las páginas GSP o una determinada interfaz de usuario. Una de las características de Grails es que se implementa como plugins, un ejemplo clásico es el módulo de persistencia GORM (11). Un esquema de dicha arquitectura puede ser visto en la Figura 2.

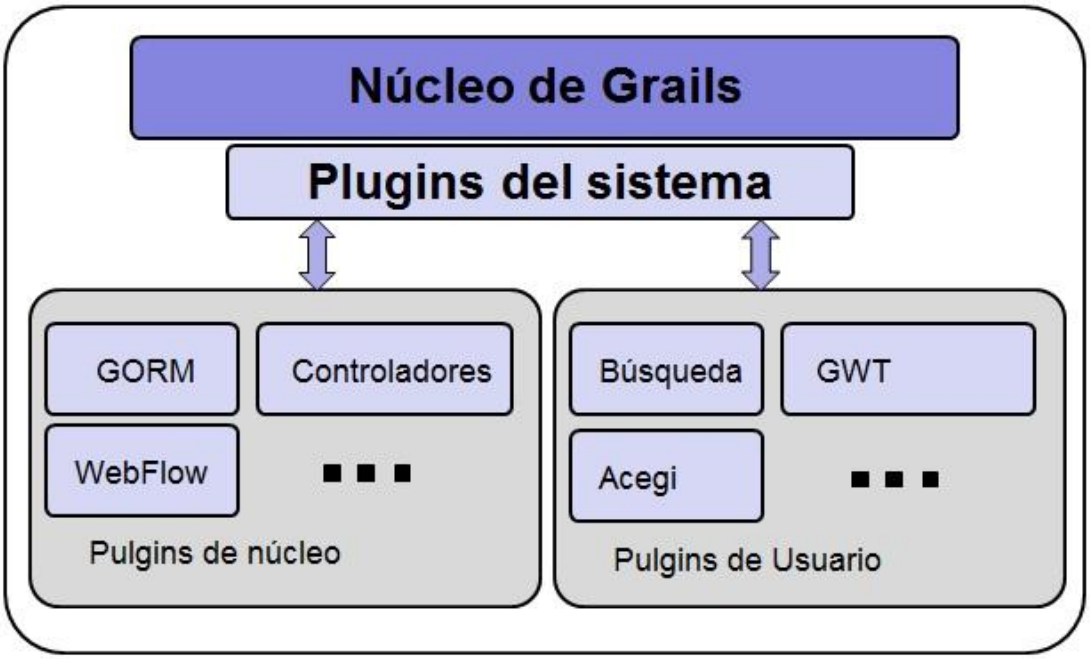


Figura 2 Arquitectura de plugins

1.4.7. Entorno de Desarrollo

NetBeans IDE 6.9

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de Java escritas para interactuar con las APIs de NetBeans y un archivo especial que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software (12).

1.4.8. Sistemas Gestor de Base de Datos

Oracle 11g.

Oracle es un Sistema de Gestión de Base de Datos objeto-relacional, por el acrónimo en inglés de *Object-Relational Data Base Management System* (ORDBMS), desarrollado por Oracle Corporation.

Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacándose por presentar las siguientes características: soporte de transacciones, estabilidad, escalabilidad y soporte multiplataforma. Se seleccionó a Oracle 11g como sistema gestor para la base de datos porque es robusto y garantiza la seguridad e integridad de los datos (13) (14).

1.4.9. Herramienta de diseño de Base de Datos

ER/Studio

ER/Studio está equipado para crear y manejar diseños de bases de datos funcionales y confiables. Ofrece fuertes capacidades de diseño lógico, sincronización bidireccional de los diseños físicos y lógicos, construcción automática de bases de datos, documentación y fácil creación de reportes. Su ambiente es de gran alcance, de varios niveles del diseño.

Ofrece las siguientes funcionalidades:

- Capacidad fuerte en el diseño lógico.
- Sincronización bidireccional de los diseños lógico y físico.
- Construcción automática de Base de Datos.
- Reingeniería inversa de Base de Datos.
- Documentación basada en HTML.
- Un Repositorio para el Modelado.

1.4.10. Contenedor Web

Apache Tomcat (6.0.25)

Es un servidor de páginas WEB que permite acceder a páginas alojadas en un ordenador. Es un software de código abierto multiplataforma. Ha evolucionado hasta convertirse en uno de los mejores servidores en términos de eficiencia, funcionalidad y velocidad. Tomcat es el servidor web más utilizado a la hora de trabajar con Java en entornos web. Incluye el compilador Jasper, que compila *Java Server Pages* (JSPs) convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor web Apache. Cada versión del Apache Tomcat incluye nuevas características, así como nuevas versiones de los componentes que lo conforman, en este caso la versiones Tomcat 6.x (15).

Para la determinación del uso de estas herramientas, fueron tomadas en cuenta las características definidas en la línea base del proyecto, cabe destacar las sugerencias

del cliente en relación al uso de algunas de estas tecnologías debido a que en su infraestructura tecnológica ya emplean algunas.

1.5. Pruebas de software

La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación.

Se puede definir la prueba como una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones específicas, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente.

Las pruebas de software implican ejecutar una implementación del software con datos de prueba. Se examinan las salidas del software y su entorno operacional para comprobar que funcionan tal y como se requiere. Las pruebas son una técnica dinámica de verificación y validación. (16)

Tienen dos objetivos fundamentales:

- Demostrar al desarrollador y al cliente que el software satisface sus requerimientos.
- Descubrir defectos en el software en que el comportamiento de este es incorrecto, no deseable o no cumple su especificación.

Los casos de prueba son especificaciones de las entradas para la prueba y la salida esperada del sistema, más una afirmación de lo que se está probando. Los datos de prueba son las entradas que han sido ideadas para probar el sistema y veces pueden generarse automáticamente. La generación automática de casos de prueba es imposible. La salida de las pruebas sólo puede predecirse por personas que comprenden lo que debería hacer el sistema.

La prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo.

Según Pressman se distinguen los siguientes niveles de pruebas (17):

- Prueba de Unidad
- Prueba de Integración
- Prueba de Validación
- Prueba de Sistema

Para realizar las pruebas en el módulo Sistema de colectivos se desarrollaron pruebas de unidad.

La prueba de unidad centra el proceso de verificación en la menor unidad del diseño del software. Es aplicable a componentes representados en el modelo de implementación para verificar que los flujos de control y de datos están cubiertos, y que ellos funcionen según lo esperado.

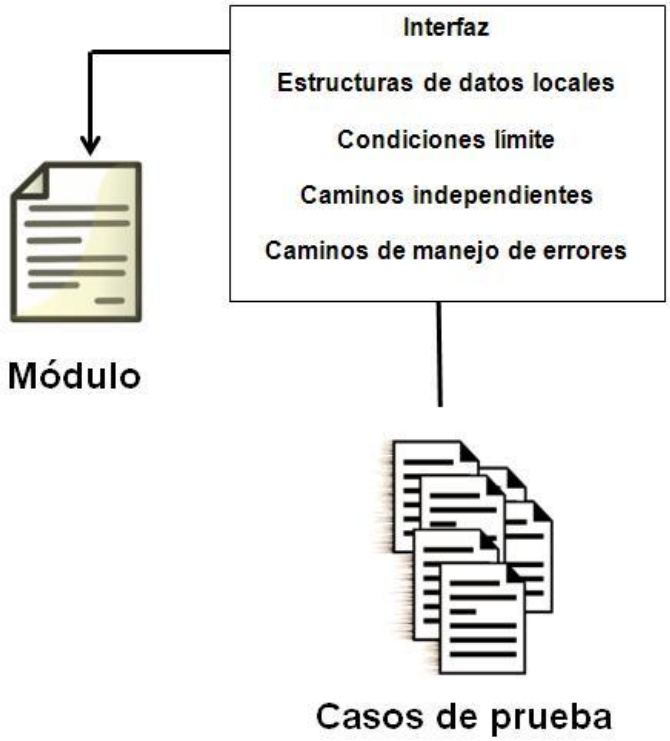


Figura 3 Prueba de unidad.

Se prueba la interfaz del módulo para asegurar que la información fluye de forma adecuada hacia y desde la unidad de software que está siendo probada. Se examinan las estructuras de datos para asegurar que los datos conservan la integridad durante la ejecución de los diferentes procesos. Se prueban las condiciones límites para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones de procesamiento. Se ejecutan todos los caminos independientes (caminos básicos) de la estructura de control con el fin de asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez. Finalmente se prueban los caminos de manejo de errores.

Según lo expuesto anteriormente se realizan pruebas de funcionalidad al módulo Sistema de colectivos. El objetivo fundamental es asegurar el cumplimiento de los requisitos funcionales principalmente al nivel de interfaz de usuario, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados.

El método de prueba empleado es el de Caja Negra. Este se refiere a las pruebas que se llevan a cabo sobre la interfaz del software, por lo que los Casos de Prueba (CP) pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta, así como que la integridad de la información externa se mantiene. Esta prueba examina algunos aspectos del modelo fundamentalmente del sistema sin tener mucho en cuenta la estructura interna del software (17).

Se ejecuta cada caso de uso, flujo de caso de uso, o función, usando datos válidos e inválidos, para verificar lo siguiente:

- Que se aplique apropiadamente cada regla de negocio.
- Que los resultados esperados ocurran cuando se usen datos válidos.
- Que sean desplegados los mensajes apropiados de error y precaución cuando se usan datos inválidos.

1.6. Conclusiones parciales

En este capítulo se realizó un estudio de todo lo concerniente con el sistema penitenciario nacional, lo que permitió comprobar y analizar el estado actual del mismo en todo lo referente al proceso de sistema de colectivos. Se estudiaron varios sistemas informáticos penitenciarios demostrando las insuficiencias y diferencias que estos presentan, determinándose la necesidad de implementar en el SIDEP el módulo Sistema de colectivos. Se abordó además la metodología, herramientas y tecnologías propuestas por el proyecto SIDEP, además de los elementos fundamentales definidos en la estrategia de prueba.

CAPÍTULO 2. Diseño del sistema

2.1. Introducción

En el desarrollo de este capítulo se dará una descripción más detallada del sistema, así como las funcionalidades y características específicas del módulo Sistema de colectivos. Dentro de las características a explicar se encuentran aspectos ligados a los elementos arquitectónicos del sistema, para ello se mostrarán los diagramas de interacción, diagramas de clases y el diagrama de base de datos, además de la descripción de los patrones de diseño e implementación que se utilizaron.

2.2. Requerimientos

La IEEE (*Institute of Electrical and Electronics Engineers*) define un requerimiento como una condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo. Un requerimiento no es más que esa condición que debe ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Por lo tanto, los requerimientos de software le brindan al usuario la posibilidad de que el sistema cumpla las condiciones que le interesen, siendo estos una manera de verificar la calidad del producto. Por esto, el levantamiento de requerimientos se hace fundamental a la hora de desarrollar cualquier sistema.

Requerimientos funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Son considerados características requeridas del sistema que expresan una capacidad de acción del mismo, es decir, una funcionalidad; generalmente expresada en una declaración en forma verbal.

Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer (14).

Nº	Funcionalidad	Descripción
RF1	Crear Colectivo	Permite crear un nuevo colectivo
RF2	Modificar Colectivo	Permite modificar los datos de un colectivo

RF3	Crear Consejo de Internos	Permite crear un consejo de internos
RF 4	Modificar Consejo de Internos	Permite modificar la información del consejo de internos
RF6	Crear Consejo de Familia	Permite crear un consejo de familia
RF7	Modificar consejo de Familia	Permite modificar la información del consejo de familia
RF8	Crear Consejo de Educadores	Permite crear un Consejo de Educadores
RF9	Modificar Consejo de Educadores	Permite modificar un Consejo de Educadores
RF10	Consultar Documentos Consejo	Permite buscar los diferentes documentos generados en las reuniones de los consejos
RF11	Registrar Reunión	Permite registrar los datos de una reunión
RF12	Generar Modelo	Permite Generar los modelos u otros documentos que se requieran imprimir.
RF13	Consultar Colectivo	Permite al educador guía consultar toda la información sobre el consejo del cual es responsable

Tabla 1 Requisitos funcionales del módulo Sistema de colectivos

2.3. Diagramas de caso de uso

Un Diagrama de Casos de Uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa (15).

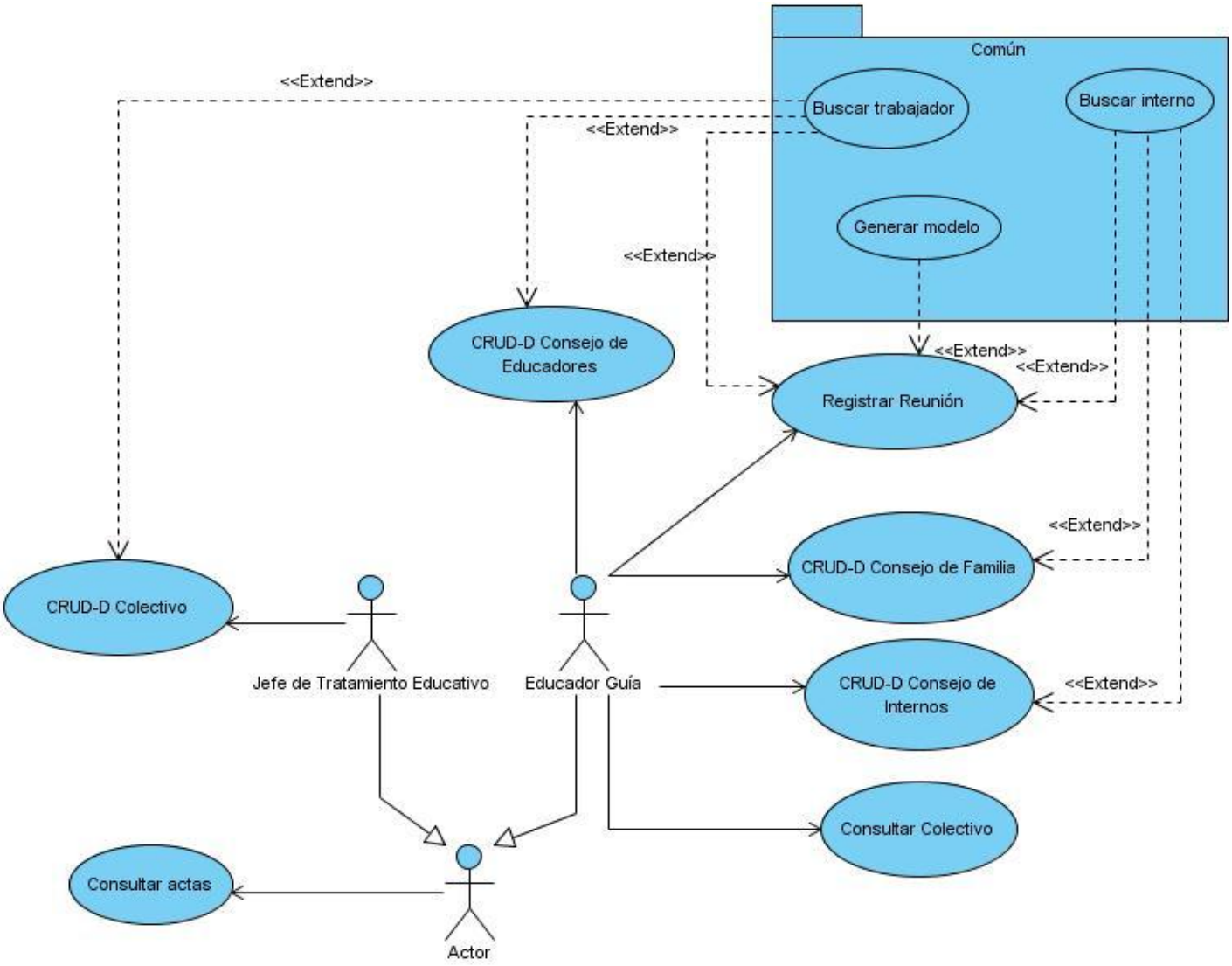


Figura 4 Diagrama de CU Sistema de colectivos

El Educador Guía es el responsable fundamental de la gestión del Consejo de Educadores, Consejo de Familia y Consejo de Internos. Además es el encargado de monitorear las reuniones que realizan estos consejos.

El Jefe de Tratamiento Educativo es el encargado de gestionar los diferentes colectivos que estarán presentes en el centro penal. Por último un Usuario, que representa a un trabajador del centro penitenciario, puede consultar las actas emitidas en las reuniones de los distintos consejos.

A continuación se muestran las principales funcionalidades del módulo Sistema de colectivos.

Nombre del caso de uso	Descripción
CRUD-D Colectivo	El sistema permite crear y actualizar los datos de un colectivo. Lo que incluye asignarle un Educador Guía, quien será el encargado de la gestión del colectivo.

Registrar Reunión	El sistema muestra un listado de los consejos existentes y permite seleccionar uno de ellos para registrar una reunión. Además permite registrar invitados a la misma, tomar la asistencia y asignar responsables a los acuerdos tomados en la misma.
CRUD-D Consejo de Educadores	El sistema muestra un listado con los colectivos existentes en el centro penitenciario, permite seleccionar uno de ellos y agregarle o actualizar los datos de un Consejo de Educadores. Se le asigna a consejo el mismo número del colectivo al que pertenece.
CRUD-D Consejo de Familia	El sistema muestra un listado con los colectivos existentes en el centro penitenciario permite seleccionar uno de ellos y registrar o actualizar los datos del Consejo de Familia, además de registrar los internos con los cuales estos familiares se relacionan.
CRUD-D Consejo de Internos	El sistema muestra un listado con los colectivos existentes en el centro penitenciario, permite seleccionar uno de ellos y registrar o actualizarlos datos del Consejo de Internos.
Consultar actas	El sistema permite consultar las actas de las reuniones de los diferentes consejos.

Tabla 2 Principales funcionalidades del módulo Sistema de colectivos.

Para los casos de uso extendidos Buscar trabajador, Buscar interno y Generar modelo los mismos son representados dentro del paquete común debido a que son funcionalidades que ofrecen otros módulos y son usadas en Sistema de colectivos como complemento de algunas de sus funcionalidades. De esta forma se evita la redundancia y se fomenta la reutilización de módulos comunes que pueden ser usados por otros subsistemas.

Para un mejor entendimiento del caso de uso CRUD-D Colectivo correspondiente al módulo Sistema de colectivos se muestra una descripción más detallada.

Objetivo	Crear y actualizar los datos de un colectivo.
Actores	Jefe de Tratamiento Educativo (JTE)
Resumen	El Caso de Uso (CU) se inicia cuando el JTE selecciona la opción de “Sistema de colectivos”. El sistema permite actualizar o crear un nuevo colectivo. El sistema registra la información y termina el CU.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	El JTE del sistema debe estar autenticado y con los permisos asignados para gestionar un colectivo.
Postcondiciones	Se crea o se actualiza un colectivo.

Flujo de eventos

Flujo básico Gestionar Colectivo

	Actor	**Sistema**
1.	Selecciona la opción “Sistema de colectivos”.	
2.		Muestra un listado de los colectivos existentes en el centro penitenciario. El listado contiene: • Número del colectivo • Educador guía Del educador guía muestra los siguientes datos: • Primer nombre • Segundo nombre • Primer apellido • Segundo apellido Y muestra las opciones de registrar o actualizar un colectivo.

3.	Si selecciona: La opción “Nuevo”, ver Sección 1: Registrar Colectivo. De un colectivo creado la opción “Actualizar”, ver Sección 2: Modificar Colectivo.	
Flujos alternos		
*** Cancelar Gestionar Colectivo**		
	Actor	**Sistema**
1.	Oprime el botón “Cancelar”.	
2.		Regresa al paso 2 del flujo básico “Gestionar Colectivo”
Sección 1: “Registrar Colectivo”		
Flujo básico Registrar Colectivo		
	Actor	**Sistema**
1.		Muestra la estructura del centro penitenciario, para que el JTE seleccione la composición del colectivo. La estructura del centro penitenciario contiene: • Locales del centro penitenciario De cada uno de los locales muestra: • Número de local • Tipo de local
2.	Selecciona las celdas, cubículos o pabellones que integrarán el colectivo y	

	oprime el botón “Siguiente”.	
3.		Verifica que los locales seleccionados no pertenecen ya a otro colectivo. Si existe un colectivo con al menos uno de los locales seleccionados, ver flujo alterno 3a. “Local asignado a otro colectivo”.
4.		Muestra los siguientes campos para registrar el colectivo: • Designación • Cantidad máxima de internos • Régimen Y le asigna un número consecutivo al colectivo teniendo en cuenta que no se le haya asignado a otro colectivo registrado anteriormente.
5.	Introduce los datos y oprime el botón “Siguiente”.	
6.		Valida los datos introducidos. Si los datos son incorrectos, ver flujo alterno 6a. “Los datos son incorrectos”. Si hay campos vacíos, ver flujo alterno 6b. “Campos vacíos”.
7.		Se ejecuta el CU extendido “Buscar trabajador”.
8.	Selecciona un educador guía y oprime el botón “Aceptar”	
9.		Valida la asignación del educador guía.

		Si el educador guía ya está asignado a otro colectivo, ver el flujo alterno 9a. “Educador guía asignado a otro colectivo”.
10.		Registra los datos del colectivo.
11.		Termina el caso de uso.
Flujos alternos		
3a. Local asignado a otro colectivo		
	Actor	**Sistema**
1.		Muestra el mensaje: “El local seleccionado ya ha sido asignado a otro colectivo” y señala el local.
2.		Regresa al paso 1 del flujo básico “Registrar Colectivo”.
6a. Los datos son incorrectos		
	Actor	**Sistema**
1.		Muestra el mensaje de error “Corrija los datos erróneos introducidos” y señala los campos erróneos introducidos.
2.	Regresa al paso 5 del flujo básico “Registrar Colectivo”.	
6b. Campos vacíos		
	Actor	**Sistema**
1.		Muestra un mensaje de error “Los campos en rojo son requeridos” y señala los campos obligatorios que no fueron introducidos.

2.	Regresa al paso 5 del flujo básico “Registrar Colectivo”.	
9a. Educador guía asignado a otro colectivo		
	Actor	**Sistema**
1.		Muestra el mensaje: “El educador guía seleccionado ya ha sido asignado a otro colectivo”.
2.	Regresa al paso 8 del flujo básico “Registrar Colectivo”.	
Sección 2: “Actualizar Colectivo”		
Flujo básico Actualizar Colectivo		
	Actor	**Sistema**
1.		Muestra los datos del colectivo y la estructura del centro penitenciario: Del colectivo muestra los datos: • Locales del colectivo • Número del colectivo • Educador guía De cada local: • Número de local • Tipo de local Del educador guía: • Primer nombre • Segundo nombre • Primer apellido • Segundo apellido La estructura del centro penitenciario contiene: • Locales del centro

		penitenciario Y brinda la opción de asignar nuevos locales al colectivo, además se puede quitar un local de los asignados al colectivo o cambiar el educador guía que atiende al colectivo.
2.	Selecciona los locales de la estructura del centro penitenciario que desea asignarle al colectivo y oprime el botón “Adicionar”. Si selecciona un local del colectivo y oprime el botón “Eliminar”, ver el flujo alterno 2a. “Quitar local asignado”. Si selecciona la opción “Cambiar educador guía”, ver el flujo alterno 2b. “Cambiar educador guía”	
3.		Verifica que los locales seleccionados no pertenecen ya a otro colectivo. Si existe un colectivo con al menos uno de los locales seleccionados, ver flujo alterno 3a. “Local asignado a otro colectivo”.
4.		Actualiza los datos.
5.		Termina el caso de uso.
Flujos alternos		
2a. Quitar local asignado		
	Actor	**Sistema**
1.		Muestra un mensaje de confirmación “¿Desea realmente eliminar el local seleccionado?”

2.	Oprime el botón “Aceptar”. Si oprime el botón “Cancelar”, ver flujo alterno 2a.3a. “No Confirma”.	
3.		Regresa al paso 4 del flujo básico “Actualizar Colectivo”.
2a.3a.No confirma		
	Actor	**Sistema**
1.		No elimina el local del colectivo.
2.		Regresa al paso 1 del flujo básico “Actualizar Colectivo”.
2b. Cambiar educador guía		
	Actor	**Sistema**
1.		Se ejecuta el CU extendido “Buscar trabajador”.
2.	Selecciona un educador guía y oprime el botón “Aceptar”.	
3.		Valida el educador guía seleccionado. Si el educador guía ya está asignado a otro colectivo, ver el flujo alterno 2b.3a. “Educador guía asignado ya”.
4.		Regresa al paso 4 del flujo básico “Actualizar colectivo”.
2b.3a. Educador guía asignado ya		
	Actor	**Sistema**
1.		Muestra el mensaje: “El educador guía seleccionado ya ha sido

		asignado a otro colectivo".
2.		Regresa al paso 1 del flujo alterno "Cambiar educador guía".
3a. Local asignado a otro colectivo		
	Actor	**Sistema**
1.		Muestra el mensaje: "El local seleccionado ya ha sido asignado a otro colectivo" y señala el local.
2.		Regresa al paso 1 del flujo básico "Actualizar Colectivo".

Relaciones	**CU Incluidos**	No aplicable
	CU Extendidos	Buscar trabajador (Ver Especificación de Casos de uso Comunes)
Requisitos no funcionales	No aplicable	
Asuntos pendientes	No aplicable	

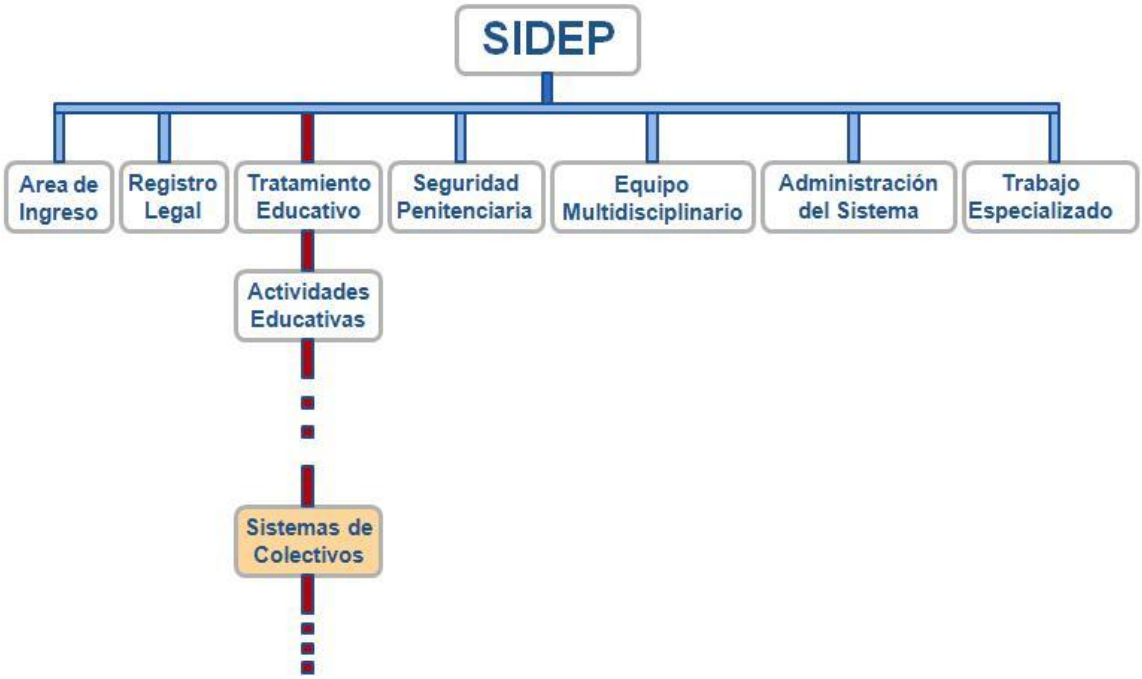
Tabla 3 Descripción del caso de uso CRUD-D Colectivo.

2.4. Arquitectura del sistema.

La arquitectura de software se define como "la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos el ambiente y los principios que orientan su diseño y evolución". O como la organización o estructura (en un punto dado en el tiempo) de los componentes importantes que interactúan a través de interfaces, los componentes que se componen de sucesivos elementos más pequeños y las interfaces. Una arquitectura puede ser descompuesta recursivamente en partes que interactúan a través de interfaces, las relaciones que conectan a las partes, y las limitaciones para el montaje de piezas. Partes que interactúan a través de interfaces, incluyen clases, componentes y subsistemas (2).

Una arquitectura de software se selecciona y diseña con base en objetivos y restricciones. Los objetivos son aquellos prefijados para el sistema de información, pero no solamente los de tipo funcional, también otros objetivos como la adaptabilidad, flexibilidad e interacción con otros sistemas de información. Las restricciones son aquellas limitaciones derivadas de las tecnologías disponibles para implementar sistemas de información.

SIDEP se basa en una arquitectura modular donde a cada módulo se le asigna el mismo nivel de importancia y seguimiento (**Figura 5**). El sistema cuenta con un módulo base, SIDEP, y a él se integran todos los demás componentes del sistema. Por ahora los subsistemas definidos son Área de ingreso, Registro legal, Seguridad penitenciaria, Equipo multidisciplinario, Administración del sistema, Trabajo especializado y Tratamiento educativo. Este último es el encargado de dar seguimiento y tomar decisiones sobre la reinserción de los privados de libertad. Entre sus misiones principales está la de ejecutar el proceso de estudio, observación, evaluación y diagnóstico del interno. Se encarga además de dirigir el tratamiento educativo que se les dispensa a los sancionados y lograr una educación en la formación de valores, que



contribuyan a reforzar las cualidades positivas, logrando normas de convivencia social y una cultura general integral en el proceso de resocialización.

Figura 5 Vista de los módulos de SIDEP

Como se observa en la Figura 5, el módulo de Sistema de colectivos es uno de los componentes que conforman el subsistema Tratamiento educativo. La arquitectura definida para la construcción de este sistema se basará a su vez en el estilo arquitectónico Cliente–Servidor y la propuesta de Arquitectura en Capas que propone Grails, ver Figura 6.

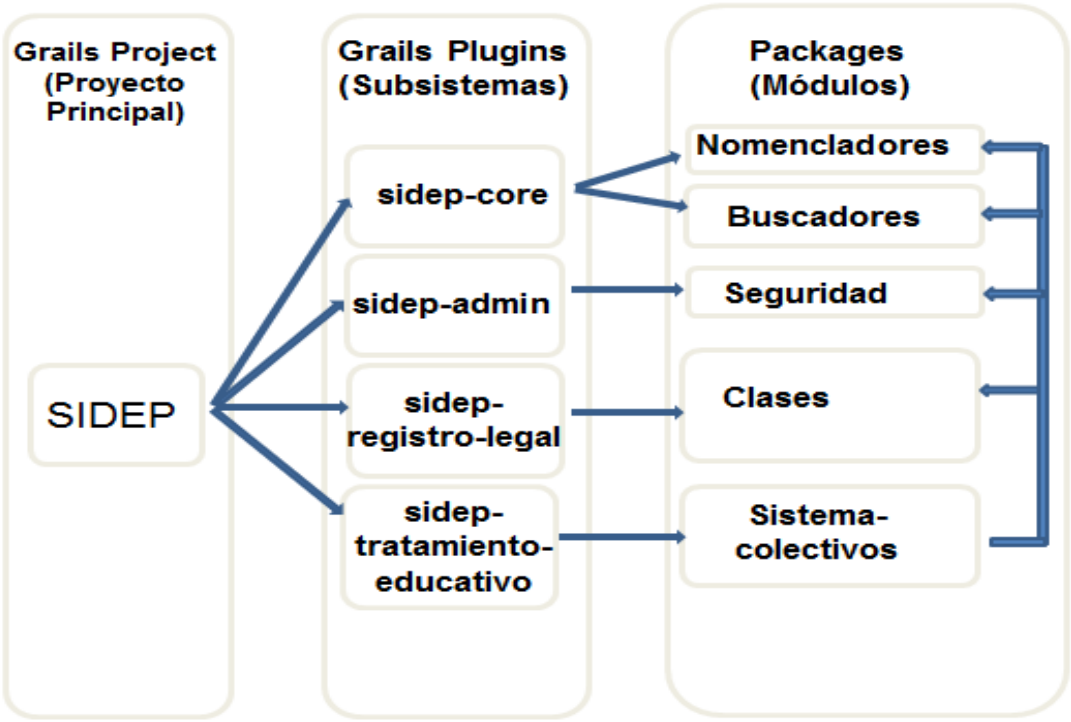


Figura 6 Arquitectura del sistema

Capa de presentación

En esta capa se encuentran las Vistas y la Lógica de Presentación, las cuales según la arquitectura que propone Grails, ver epígrafe 1.4.6, sus clases estarán enmarcadas en los paquetes GSP, JavaScript y Controladores, el cual contendrá las clases controladoras y las vistas. En este caso está compuesto por todas las páginas GSP, agrupadas en el componente (Sistema de colectivos (GSP)), los ficheros javascript (Sistema de colectivos (javascript)) y las clases controladoras dentro del componente (Sistema de colectivos (controladores)).

Capa de servicios

En esta capa se encapsula toda la lógica de la aplicación en fachadas de negocio que son utilizadas por los controladores en la capa de presentación y se exponen algunos procesos de negocio a través de interfaces de servicios. Sus clases radican según la arquitectura propuesta en el paquete Servicios.

Capa de datos

Esta capa maneja los objetos de acceso a datos, abstrayéndose del mecanismo de persistencia usado, a través de interfaces que exponen las operaciones de persistencia. Las clases relacionadas con esta capa radican en el paquete Clases de Dominio.

Es necesario mencionar que, debido a la arquitectura del sistema, este módulo interactúa directamente con los módulos base Sidep-Core, Sidep-Admin y Registro-Legal. Estos presentan algunas interfaces, componentes y servicios necesarios para el correcto funcionamiento e integración del módulo Sistema de colectivos.

2.5. Patrones de diseño

Un patrón de diseño es una solución a un problema de diseño, un estándar para resolver situaciones comunes. Para que una solución sea considerada un patrón debe haberse probado su efectividad en situaciones anteriores similares, además de ser un componente reutilizable que permita adaptarlo a otros problemas de la misma índole en circunstancias diferentes. Su utilización facilita el proceso de desarrollo de software en gran medida, pues en ellos se definen procedimientos específicos en dependencia del tipo de problema a resolver (2). En el caso de Grails, el uso de estos es intensivo, llegando incluso a hacer que el desarrollador los implemente obligatoriamente para el cumplimiento de determinadas funcionalidades.

De los patrones que propone el grupo GOF, también conocido como *Gang of Four*, se utiliza de forma general el Singleton (Solitario), todos los servicios lo implementan, y quedando como propuesta a su vez el uso de patrón Facade (Fachada). Este último está comprendido entre los patrones de creación y se emplea para separar las clases relacionadas con la lógica de negocio con las clases de la lógica de presentación, evitando así las constantes peticiones al servidor, y permitiendo a su vez estructurar las llamadas o peticiones que se realizarán desde el cliente. En el caso de que se decida cambiar la implementación de las clases no es necesario que la capa superior sufra afectaciones, sencillamente la fachada no permitiría que esto ocurra, al no existir una dependencia directa entre las capas.

De los patrones GRASP (Patrones Generales de Asignación de Responsabilidad de Software, por sus siglas en inglés *General Responsability Assignment Software Patterns*) se utilizarán de forma general los patrones Alta Cohesión, Bajo Acoplamiento, *Controller* (Controlador) y *Front-Controller*.

El patrón *Controller* propone asignar la responsabilidad de controlar el flujo de eventos de un sistema, a clases específicas llamadas controladores. Los controladores no ejecutan las tareas sino que las delegan en otras clases, con las que mantiene un modelo de alta cohesión. Cada clase de Grails promueve el uso de este patrón como parte de los convenios del framework. A su vez, el patrón *Front-Controller* propone utilizar un controlador como el punto inicial de contacto para manejar las peticiones del usuario en una aplicación.

2.6. Diagramas de clases

2.6.1. Diagramas de clases de diseño

Los diagramas de clases de diseño constituyen un artefacto generado a partir del proceso de análisis en el ciclo de desarrollo de un software. A través de estos se logra tener una visión específica de todas aquellas clases que intervienen, así como su estructura y composición. Esto se traduce en la representación de clases, asociaciones y atributos, interfaces, formularios, métodos, navegabilidad y dependencias.

A continuación se muestra el diagrama de clases de diseño para el caso de uso CRUD-D Colectivo. Para los demás diagramas de clases del diseño ver el anexo 1.

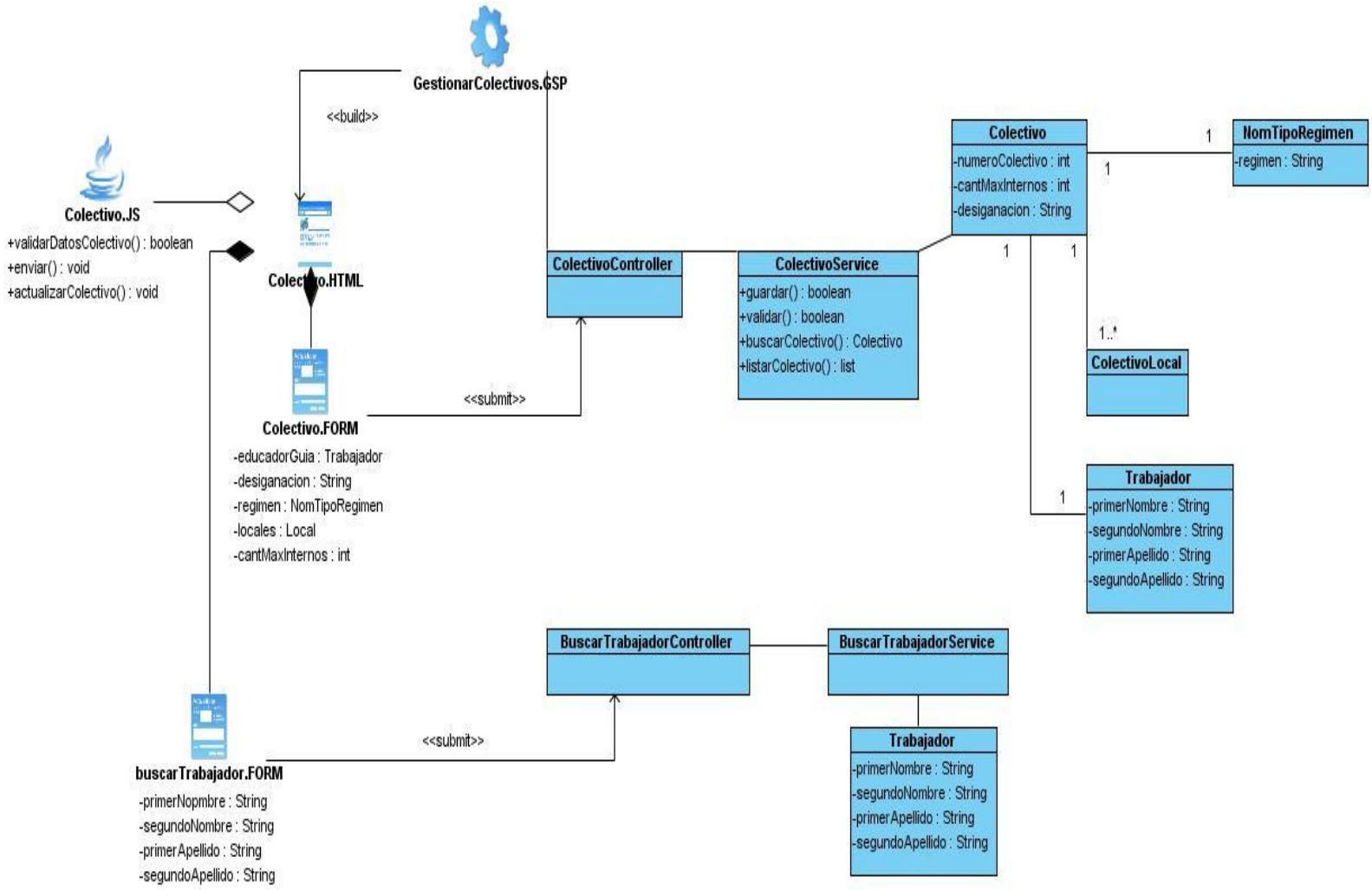


Figura 7 Diagrama de clases de diseño CRUD-D Colectivos

2.6.2. Descripción de las clases más significativas

A continuación se describen las clases más significativas correspondientes al módulo Sistema de colectivos.

Nombre de la clase: GestionarColectivo.gsp		
Tipo de clase: Interfaz		
Descripción: Esta clase es la encargada de gestionar los datos referentes a un colectivo (insertar un colectivo y actualizar). Cuenta con un formulario donde el usuario introduce los datos necesarios para crear o actualizar un colectivo, a su vez este está conectado a un fichero javascript, el cual se encarga de la validación previa y el envío de los datos al controlador.		
Atributos	**Tipo**	**Descripción**
numeroColectivo	int	Registra en número de colectivo asignado por la aplicación de manera consecutiva.
régimen	NomTipoRegimen	Registra el tipo de régimen del colectivo.

designación	String	Registra la designación del colectivo.
cantMaxInternos	int	Registra la cantidad máxima de internos con que contará el colectivo.
educadorGuia	Trabajador	Registra el Educador Guía encargado del colectivo.
locales	Lista	Registra el conjunto de locales que conformarán el colectivo.

Tabla 4 Clase estática GestionarColectivo.gsp del módulo Sistema de colectivos.

Nombre de la clase: Colectivo.js	
Tipo de clase: Interfaz	
Descripción: esta clase es la encargada de validar y enviar los datos del colectivo.	
Funciones	**Descripción**
submit	Se encarga de enviar los datos del formulario hacia el controlador.
validar	Se encarga de validar los campos del formulario.
actualizarTabla	Se encarga de actualizar los valores de la tabla Colectivos.

Tabla 5 Archivo javascript Colectivo.js del módulo Sistema de colectivos.

Nombre de la clase: ColectivoController		
Tipo de clase: Controlador		
Descripción: Esta clase es la encargada de gestionar la información de los colectivos.		
Atributos	**Tipo**	**Descripción**
colectivoService	ColectivoSevice	Ejecuta las operaciones asignadas por el controlador (guardar, buscar, listar).

Tabla 6 Clase controladora Colectivo Controller del módulo Sistema de colectivos.

Nombre de la clase: ColectivoService		
Tipo de clase: Servicio		
Descripción: Esta clase es la encargada de ejecutar las operaciones sobre las entidades.		
Atributos	**Tipo**	**Descripción**

localSevice	LocalService	Mediante él se acede a las funcionalidades definidas para la clase Local del subsistema Registro Legal.
colectivo	Colectivo	Mediante este atributo se accede a la clase colectivo del subsistema Tratamiento Educativo.
trabajador	Trabajador	Mediante este atributo se accede a la clase Trabajador.
Funciones		Descripción
guardarColectivo		Guarda un colectivo en la base de datos.
obtenerNumColectivo		Obtiene el número que le será asignado a un colectivo cuando sea creado.

Tabla 7 Clase servicio ColectivoService del módulo Sistema de colectivos.

Nombre de la clase: Colectivo	
Tipo de clase: Entidad	
Atributos	**Tipo**
numeroColectivo	int
cantMaxInternos	int
régimen	NomTipoRegimen
designación	String
educadorGuia	Trabajador
locales	Lista<Local>
Consejos	Lista<Consejo>

Tabla 8 Entidad Colectivo del módulo Sistema de colectivos.

A continuación se muestra la entidad Local que es externa del módulo Sistema de colectivos pero que es utilizada en la gestión de datos con dicho módulo.

Nombre de la clase: Local	
Tipo clase: Entidad	
Atributos	**Tipo**

numeroLocal	String
cantidadSalElecTechoT	int
cantidadSalElecTechoR	Int
cantidadLamparasT	Int
cantidadLamparasR	Int
tipoLocal	NomTipoLocal
area	Area

Tabla 9 Entidad Local del módulo Registro Legal.

2.7. Diagrama de interacción

Un diagrama de interacción explica gráficamente las interacciones existentes entre las instancias (y las clases) del modelo de estas. El punto de partida de las interacciones es el cumplimiento de las postcondiciones de los contratos de operación (14).

UML define dos tipos de estos diagramas; ambos sirven para expresar interacciones semejantes o idénticas de mensaje:

- Diagramas de colaboración
- Diagramas de secuencia

Los diagramas de colaboración dan una descripción gráfica de las interacciones del actor y de las operaciones a que da origen. Describen en el curso particular de los eventos de un caso de uso, los actores externos que interactúan directamente con el sistema (como caja negra) y con los eventos del sistema generados por los actores.

A continuación se muestran los principales diagramas de colaboración del sistema.

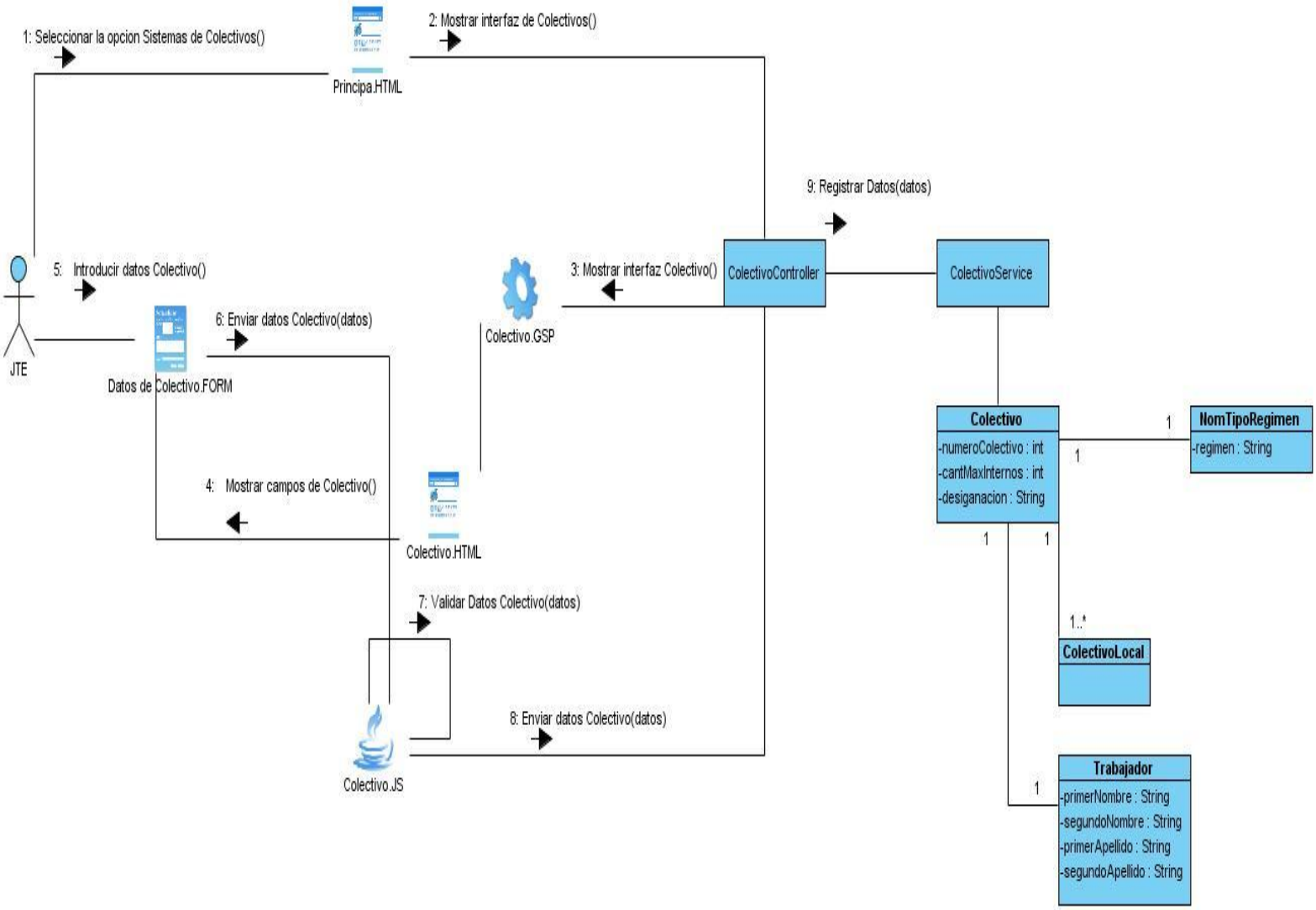


Figura 8 CRUD-D Colectivo. Sección 1 Registrar Colectivo

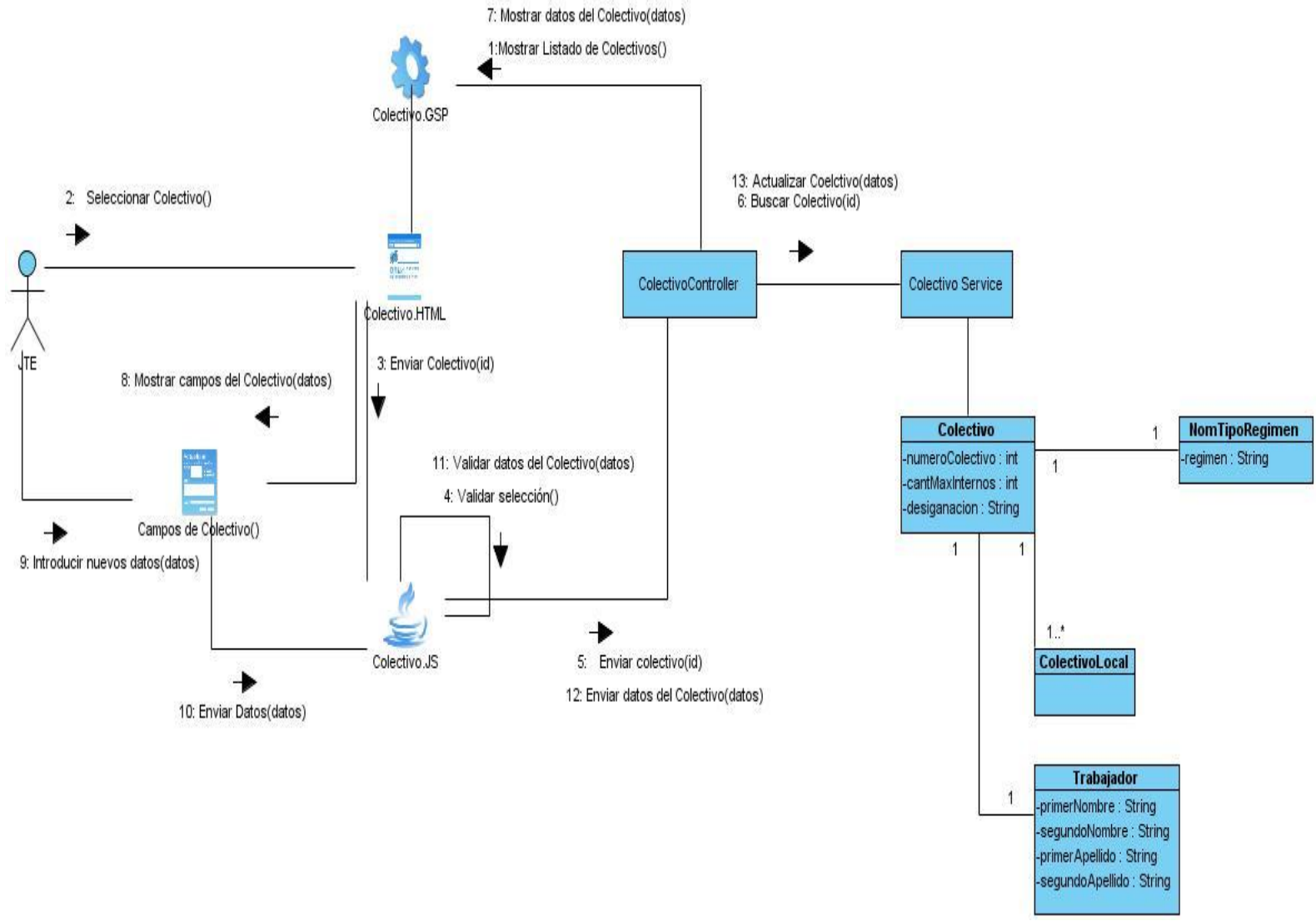


Figura 9 CRUD-D Colectivo. Sección 2 Actualizar Colectivo

2.8. Diseño de base de datos

El diseño de la base de datos debe almacenar toda la información referente a los procesos descritos anteriormente, y permitir a los funcionarios de control penal recuperarla y actualizarla en base a sus peticiones. La información de estos objetos de la base de datos es analizada detenidamente y se decide siempre que implique mayor rendimiento en el acceso a los datos y un acoplamiento mínimo al gestor de base de datos. Grails, en lo relacionado al acceso a datos, propone tres funcionalidades fundamentales para el trabajo con la base de datos (*create, update* y *drop*). Estos métodos permiten un mejor manejo de los datos, además de garantizar que el almacenamiento y recuperación de la información ocurra de manera adecuada y que se cumplan las restricciones identificadas, a través de la integridad referencial. A continuación se muestra el diagrama de base de datos correspondiente al módulo Sistema de colectivos.

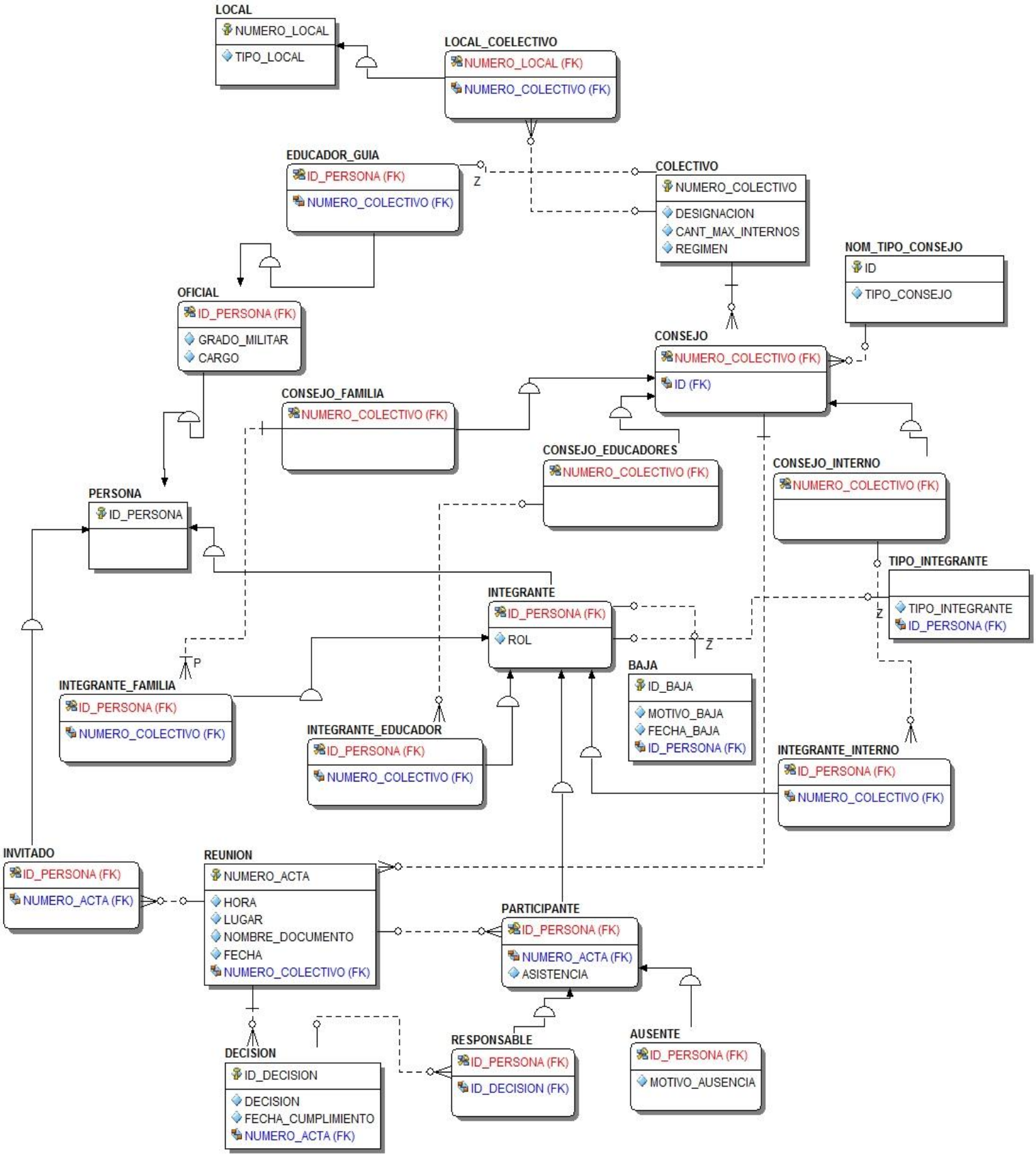


Figura 10 Diseño de base de datos del módulo Sistema de colectivos.

2.8.1. Descripción de las tablas de base de datos más significativas

A través de la descripción de las tablas de la base de datos se tiene una idea más clara de las características de las mismas, facilitando su comprensión. A continuación se describen algunas de las tablas más significativas del módulo Sistema de colectivos contenidas en la base de datos.

Nombre: Colectivo		
Descripción: Datos de un Colectivo		
Atributo	**Tipo**	**Descripción**
NUMERO_COLECTIVO	CHAR(10)	Enumerador para un colectivo.
DESIGNACION	VARCHAR	Designación de un colectivo.
CANT_MAX_INTERNOS	CHAR	Cantidad máxima de internos de un colectivo.
REGIMEN	VARCHAR	Régimen para un colectivo.

Tabla 10 Descripción de la tabla Colectivo del módulo Sistema de colectivos.

Nombre: Reunión		
Descripción: Datos de las reuniones.		
Atributo	**Tipo**	**Descripción**
NUMERO_ACTA	CHAR	Número del acta de una reunión.
HORA	VARCHAR	Hora en que se efectuará la reunión.
LUGAR	VARCHAR	Lugar donde se efectuará la reunión.
NOMBRE_DOCUMENTO	VARCHAR	Nombre del documento.
FECHA	DATE	Fecha de la reunión.
ID_COLECTIVO	CHAR	Identificador del colectivo al que pertenece el consejo que realiza esta reunión.

Tabla 11 Descripción de la tabla Reunión del módulo Sistema de colectivos.

Nombre: Consejo
Descripción: Datos de los consejos.

Atributo	Tipo	Descripción
NUMERO_COLECTIVO	CHAR	Número del colectivo al que pertenece el consejo
ID_TIPO_CONSEJO	CHAR	Identificador del colectivo al que pertenece el consejo.

Tabla 12 Descripción de la tabla Consejo del módulo Sistema de colectivos.

2.9. Conclusiones parciales

Con el desarrollo de este capítulo fueron descritos los procesos fundamentales de la fase de diseño, de esta manera se logra una correcta interpretación del funcionamiento del módulo Sistema de colectivos. Haciendo uso de las herramientas de modelado y de diseño fueron generados los diagramas correspondientes, donde quedaron expuestos aspectos esenciales de la lógica del negocio, estructura y arquitectura de la aplicación. Además, se utilizó patrones de diseño para promover la flexibilidad y facilitar la capacidad de extensión de las implementaciones.

CAPÍTULO 3. Implementación y pruebas

3.1. Introducción

El objetivo principal del desarrollo de este capítulo consiste en obtener una visualización más sólida y cercana al producto final, que en este caso se resume al correcto funcionamiento del módulo Sistema de colectivos, dividiendo esta fase en dos partes fundamentales; implementación y prueba; durante la implementación serán expuestos todos los factores que intervienen en la misma con el fin de determinar de una forma más clara la interacción entre los componentes del proyecto, y por último, la fase de prueba, esta se realizará con el fin de garantizar que la solución cumpla con las especificaciones definidas principalmente a nivel de interfaz de usuario.

3.2. Modelo de implementación

Un modelo de Implementación incluye suficiente información para construir el sistema. Debe incluir, no solamente la semántica lógica del sistema y los algoritmos, las estructuras de datos y los mecanismos que aseguran su funcionamiento apropiado, sino también las decisiones de organización sobre los artefactos del sistema que son necesarios, permitiendo así el trabajo cooperativo de las personas y el procesamiento por parte de las herramientas.

3.3. Diagramas de componentes

Los diagramas de componentes muestran la estructura de los componentes, incluyendo los clasificadores que los especifican y los artefactos que los implementan. También se pueden utilizar para mostrar la estructura de alto nivel del modelo de implementación en términos de subsistemas de implementación, y las relaciones entre los elementos de implementación (2).

En el caso específico de Sistema de Colectivos una representación general de las relaciones de este módulo con otros que integran el proyecto SIDEP puede verse en la figura 11.

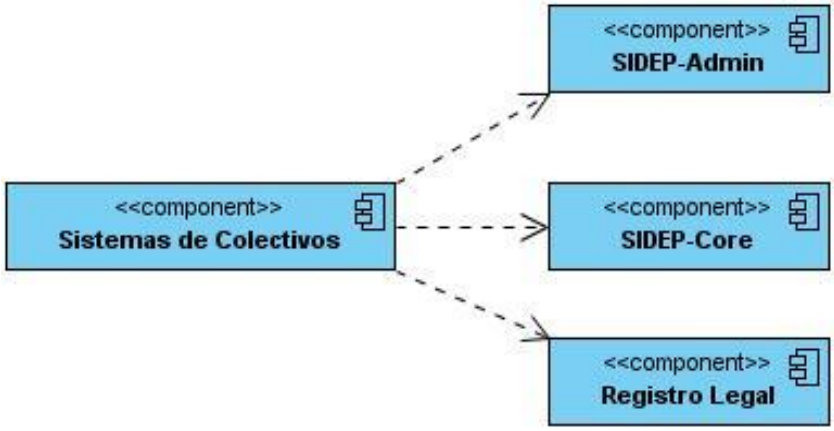


Figura 11 Representación de la dependencia entre el módulo Sistema de colectivos y otros componentes de SIDEP.

Los componentes SIDEP-Admin, SIDEP-Core y Registro Legal contienen elementos que son necesarios para el funcionamiento del módulo según la arquitectura del sistema vista en la figura 6.

Una representación más detallada del módulo Sistema de colectivos puede verse en la Figura 12. Como se especificó en la descripción de la arquitectura, este subsistema implementa el patrón MVC con una capa de Servicio según las especificaciones de Grails.

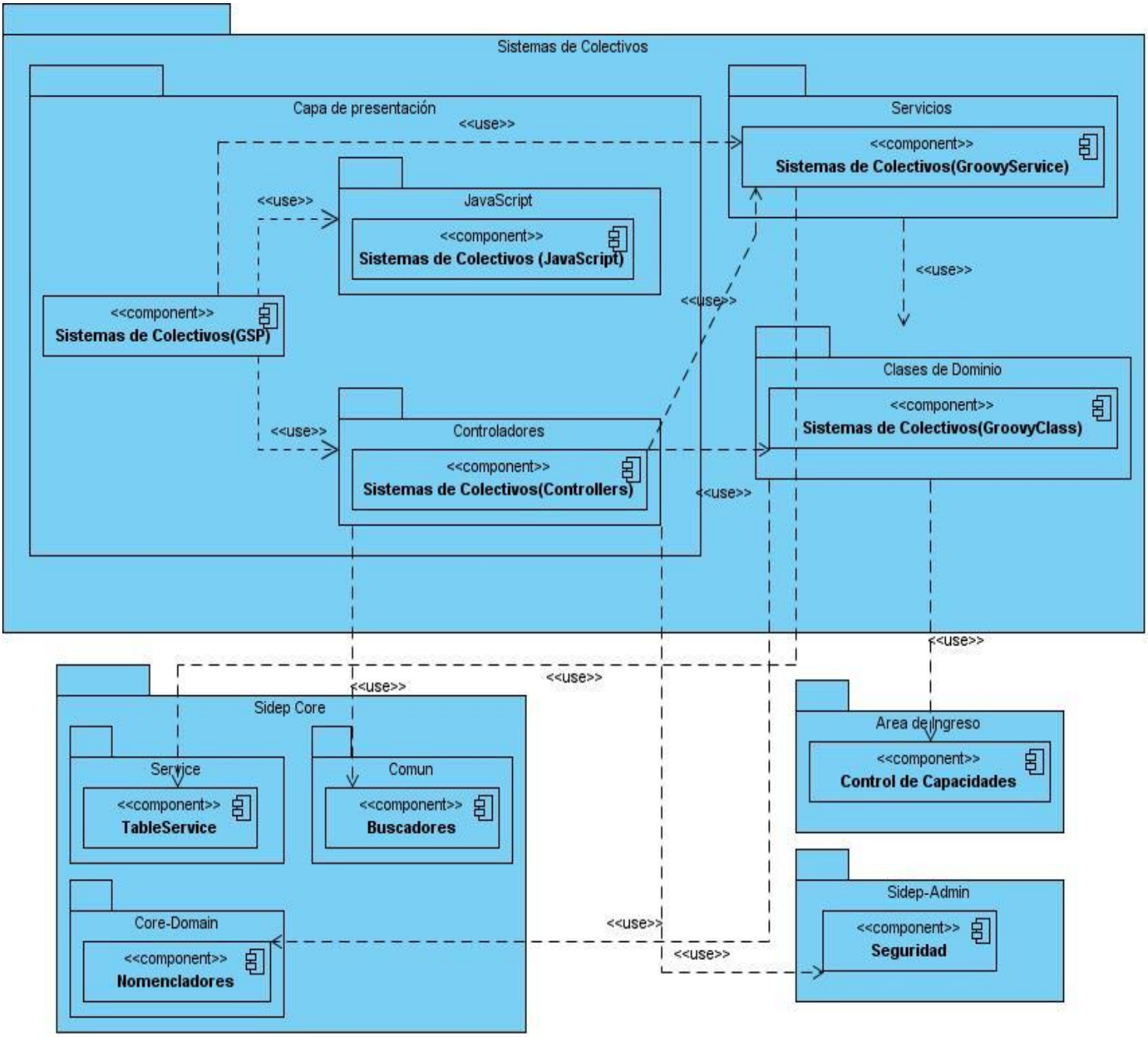


Figura 12 Representación de la dependencia entre los componentes que integran el módulo Sistema de colectivos.

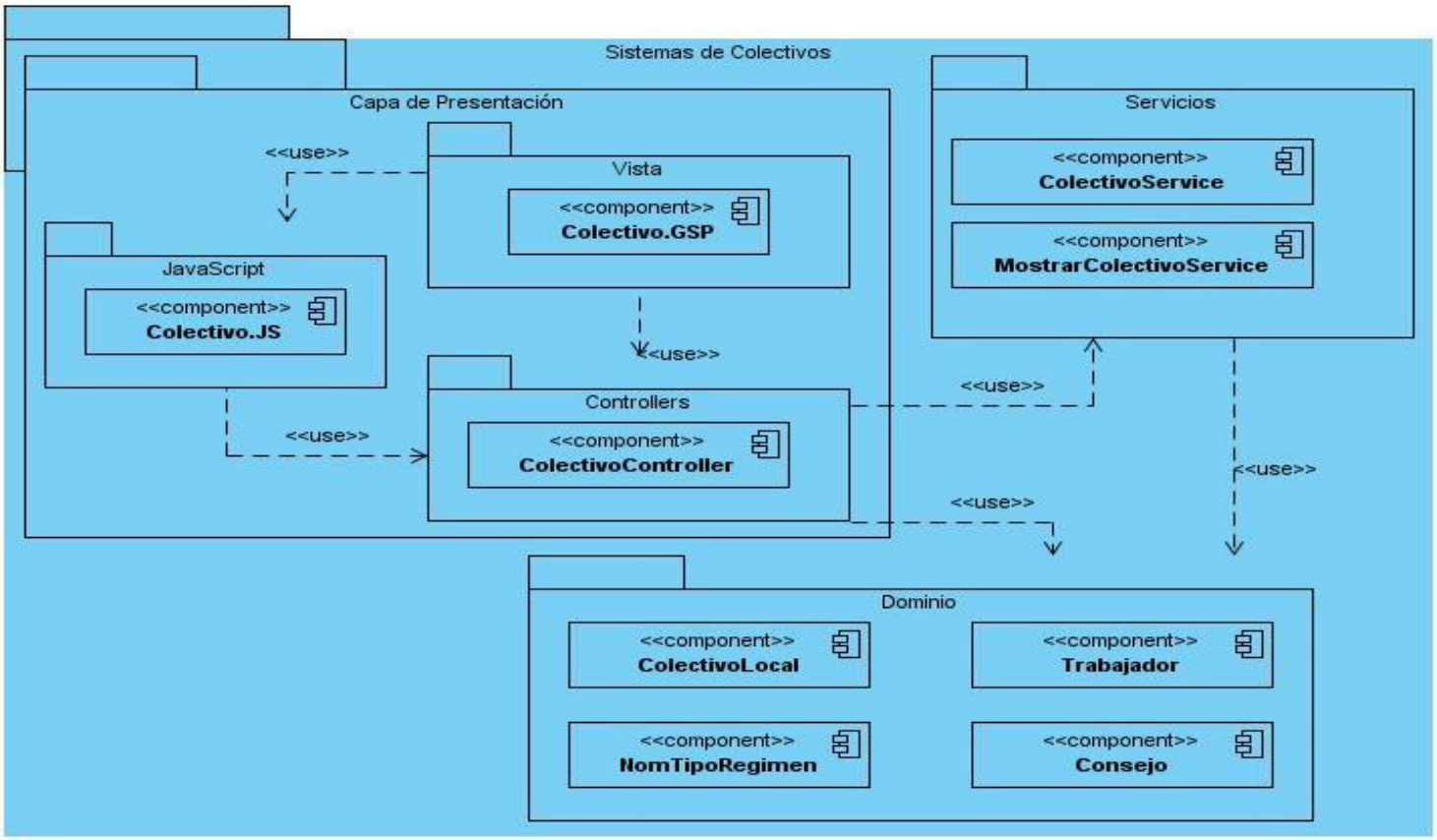


Figura 13 Representación del diagrama de componentes del CU Colectivo

3.4. Diagrama de despliegue

El Diagrama de Despliegue muestra las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. Además, muestra la configuración en funcionamiento del sistema, incluyendo su software y su hardware. Para cada componente de un diagrama es necesario documentar las características técnicas requeridas, el tráfico de red, el tiempo de respuesta, etc.

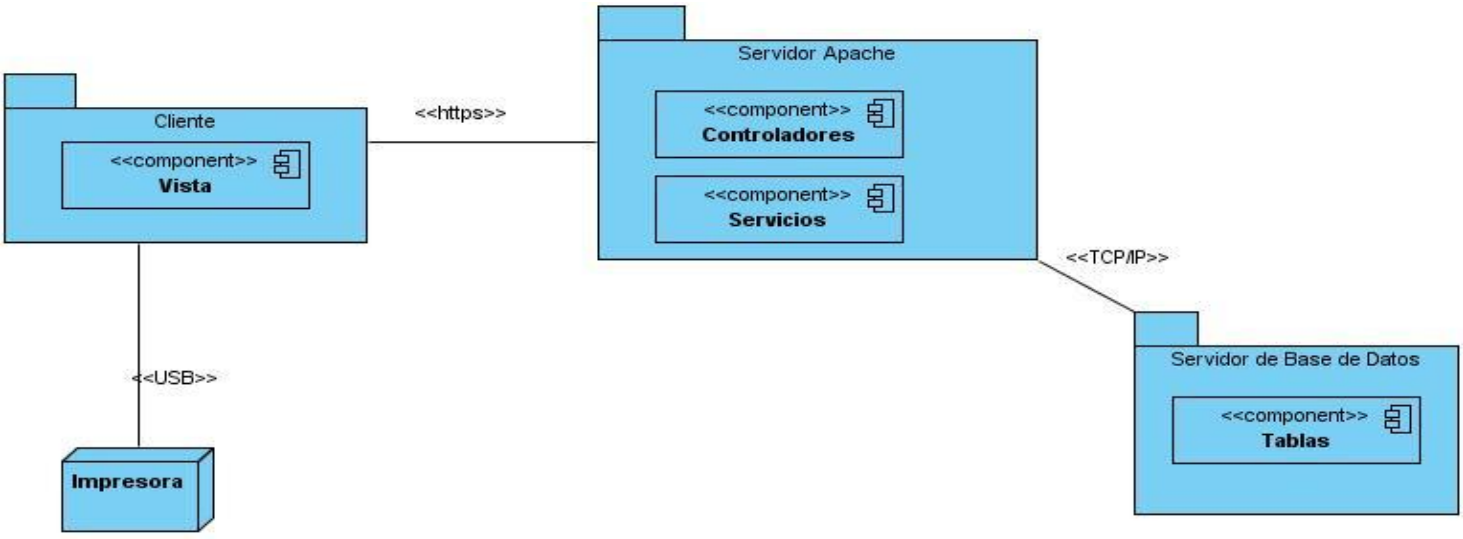


Figura 14 Diagrama de despliegue.

3.5. Pruebas

En este acápite se describen pruebas realizadas sobre la interfaz de la aplicación desarrollada. Estas se llevaron a cabo mediante casos de pruebas (CP) que tenían como objetivo demostrar que sus funcionalidades son operativas, que los datos de entrada se aceptan de forma adecuada y que se produce una salida correcta garantizando la integridad de la información que se almacena y procesa. Se examinan fundamentalmente algunos aspectos del modelo del sistema sin profundizar mucho en la estructura interna del software.

Los CP son un conjunto de entradas con datos de prueba, condiciones de ejecución y resultados esperados, cuyo propósito es identificar y comunicar las condiciones que se llevarán a cabo en la prueba. Los CP son necesarios para verificar que la solución sea aceptable acorde con los requisitos planteados para el producto (casos de uso).

Los CP del módulo Sistema de colectivos se encuentran en el Expediente de proyecto del SIGEP. Implementación y Pruebas.

La plantilla de no conformidades recoge los errores que son detectados durante la revisión de la documentación del sistema. Se elabora un documento por cada revisión que se haga y se controlan a través de versiones según se vayan eliminando los errores, hasta que finalmente se hayan erradicado todos los defectos que posea el elemento que se prueba. Además de estar inmerso en la planilla de diseño de CP, estas no conformidades se van registrando en un documento aparte, para luego enviarlo al equipo de desarrolladores.

En la siguiente tabla se expone un resumen de los resultados de las no conformidades detectadas en el módulo Sistema de colectivos, distinguiendo de ellas cuáles no proceden y las que ya fueron resueltas.

No	No conformidad	Etapa de detección	Clasificación	Ubicación	Estado de la NC
1	Cubículo C1 esta repetido	Iteración I	Baja	1. Tratamiento educativo. 2. Sistema de colectivo. 3. Gestionar colectivo	Resuelta 24/05/2012
2	Sale un alerta cuando se desea actualizar un colectivo.	Iteración I	Media	1. Tratamiento educativo. 2. Sistema de colectivo.	Resuelta 24/05/2012

				3. Gestionar colectivo	
3	Falta designación y cantidad máxima de internos.	Iteración I	Media	1. Tratamiento educativo. 2. Sistema de colectivo. 3. Consultar colectivos.	Resuelta 24/05/2012
4	Aparecen educadores repetidos dentro de los colectivos.	Iteración I	Media	1. Tratamiento educativo. 2. Sistema de colectivo. 3. Consultar colectivos.	Resuelta 24/05/2012
5	Los botones aceptar, cerrar y cancelar no realizan acción alguna.	Iteración I	Media	1. Tratamiento educativo. 2. Sistema de colectivo. 3. Consultar colectivos.	Resuelta 24/05/2012
6	Se sugiere poner un cerrar o cancelar que lleve a la lista de colectivos.	Iteración I	Media	1. Tratamiento educativo. 2. Sistema de colectivo. 3. Gestionar Consejo de Educadores	Resuelta 24/05/2012
7	Cuando se va a eliminar un integrante falta el motivo de la baja y fecha de la baja.	Iteración I	Media	1. Tratamiento educativo. 2. Sistema de colectivo. 3. Gestionar colectivo	Resuelta 24/05/2012

Tabla 13 Resultados de las pruebas (No conformidades)

3.6. Conclusiones parciales

En este capítulo se describieron las características principales de las fases de implementación y prueba relacionadas con el desarrollo del módulo Sistema de colectivos. En la fase de implementación quedaron establecidas las relaciones funcionales entre los componentes que interactúan durante los procesos gestión y se logró dar solución al problema propuesto haciendo uso de las herramientas, las tecnologías y los estándares de desarrollo definidos por el proyecto.

En la fase de pruebas, indispensable en la conclusión del desarrollo de software, mediante las pruebas de software se verifico que la solución cumpliera con los requerimientos planteados por el cliente. Además, se documentaron los resultados arrojados por las mismas, teniendo en cuenta que constituye un artefacto importante para mitigar las deficiencias encontradas en el producto y garantizar el correcto funcionamiento del software.

CONCLUSIONES GENERALES

Al finalizar de la investigación del presente trabajo de diploma se concluye lo siguiente:

- Se detallaron los elementos necesarios para la búsqueda de la solución al problema propuesto, donde el estudio y análisis de la metodología, herramientas, tecnologías y lenguajes permitió demostrar la elección de estos para el desarrollo del módulo Sistema de colectivos.
- Con el modelo de diseño obtenido se logró sentar las bases para la futura implementación de la solución acorde a las necesidades del módulo Sistema de colectivos obtenidas a partir de los requisitos planteados por el cliente.
- Se desarrolló el módulo Sistema de colectivos del sistema SIDEP a partir de las especificaciones de caso de uso definidas por el equipo de análisis y con la utilización de herramientas y tecnologías definidas en el proyecto.
- Se verificaron las funcionalidades de la solución propuesta con la realización de pruebas, demostrándose que cada una de estas responde apropiadamente a los requisitos funcionales definidos, incluyendo la navegación, la entrada de datos, el procesamiento y la obtención de resultados. De manera general los resultados obtenidos en estas pruebas fueron satisfactorios.

RECOMENDACIONES

Comprobar el rendimiento del módulo frente a bases de datos de grandes volúmenes de información para validar que el la solución propuesta satisface tales condiciones, puesto que actualmente el sistema está probado para una cantidad limitada de datos que no ponen al límite la solución propuesta.

BILIOGRAFÍA

1. **calahorro, Brito Nacho.** *Manual_de_desarrollo_web_con_Grails.*

2. www.grails.org. [En línea]

3. **López, Javier.** *Fundamentos para el desarrollo de aplicasiones web.*

4. **Abbey, Michael.** *Oracle guia de aprendizaje.*

5. **Flanagan, David.** *Java en pocas palabras.*

6. **Smith, Glen.** *Grails in Action, Peter Ledbrook.*

7. http://netbeans.org. [En línea]

8. **Javier J. Gutiérrez, María J. Escalona, Manuel Mejías y Antonia M. Reina.** *MODELOS DE PRUEBAS PARA PRUEBAS DEL SISTEMA.*

9. **MEJÍA, ELÍAS MEJÍA.** *METODOLOGÍA DE LA INVESTIGACIÓN CIENTÍFICA.*

10. **Gonzalez, Alberto Ramirez.** *Metodología de la investigación científica.*

11. **Brown, Greame Rocher & Jeff.** *The definitive gudie to Grails.*

12. **Martinto, MSc. Pedro Carlos Pérez.** *El diseño metodológico de la investigación científica.*

13. **Brown, Jeff and Graeme, Rocher.** *The Definitive Guide to Grails, Second Edition.* s.l. : Apress, 2009.

14. **Riecke, Craig, Gill, Rawld y Russell, Alex.** *Mastering Dojo. JavaScript and Ajax Tools for Great Web Experiences.* s.l. : Jacquelyn Carter, 2008.

15. **Larman, Craig.** *UML y patrones.* s.l. : Prentice Hall.

16. **Software, Rational.** *RUP. Rational Unified Process.* 2003.

17. **Jacobson, Ivar, Booch, Grady y Rumbauch, James.** *El Proceso Unificado de desarrollo del software.* s.l. : Addisson-Wesley, 2000.

18. **Barclay, Kenneth y Savage, John.** *Groovy Programming. An introduction for java developers.* s.l. : Elsevier Inc, 2007.

19. **Apache Software Foundation.** Apache Tomcat 6.0 User Guide. [En línea] http://tomcat.apache.org/tomcat-6.0-doc/index.html.

20. **Visual Paradigm International.** Visual Paradigm for UML . [En línea] http://www.visual-paradigm.com/product/vpuml/editions/modeler.jsp.

21. **Oracle.** NetBeans. [En línea] http://netbeans.org/index_es.html.

22. —. Oracle Database 11g. [En línea] http://www.oracle.com/us/products/database/index.html.

23. **Osorio, M.** *Diccionario de Ciencias Jurídicas Políticas y Sociales.* Buenos Aires : Editorial Heliasta, 1981.

24. **Oracle.** Java Platform, Enterprise Edition. [En línea] 2010. http://www.oracle.com/us/products/middleware/application-server/050871.pdf.

25. **Ladd, Seth, y otros, y otros.** *Expert Spring Mvc And Web Flow.* s.l. : Apress, 2006.

26. **Pressman, Roger .** *Ingeniería del Software. Un enfoque práctico.* s.l. : MacGraw Hill, 2002.

27. **Somerville, Ian.** *Softweare Engineering.* s.l. : Pearson Education, 2006.

28. www.minjusticia.gob.ec. [En línea]

29. **Guerrero, José Iván.** Fundamento legal del sistema penitenciario. [En línea]

30. www.juancarlosmoral.es. [En línea]

ANEXOS

Anexo 1 Diagramas de clases del diseño del módulo Sistema de colectivos.

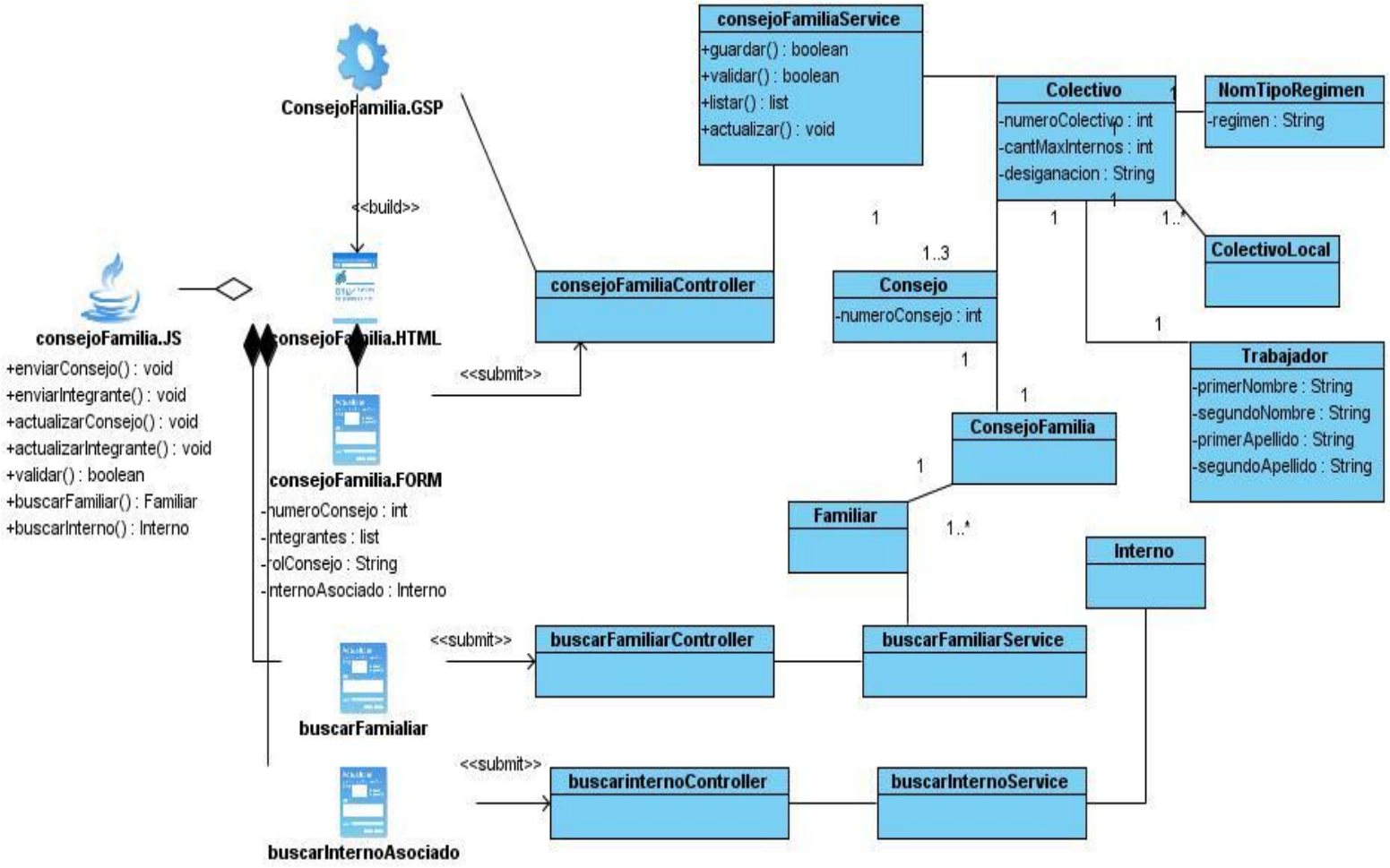


Figura 15 Diagrama de clases de diseño, CRUD-D Consejo de familia

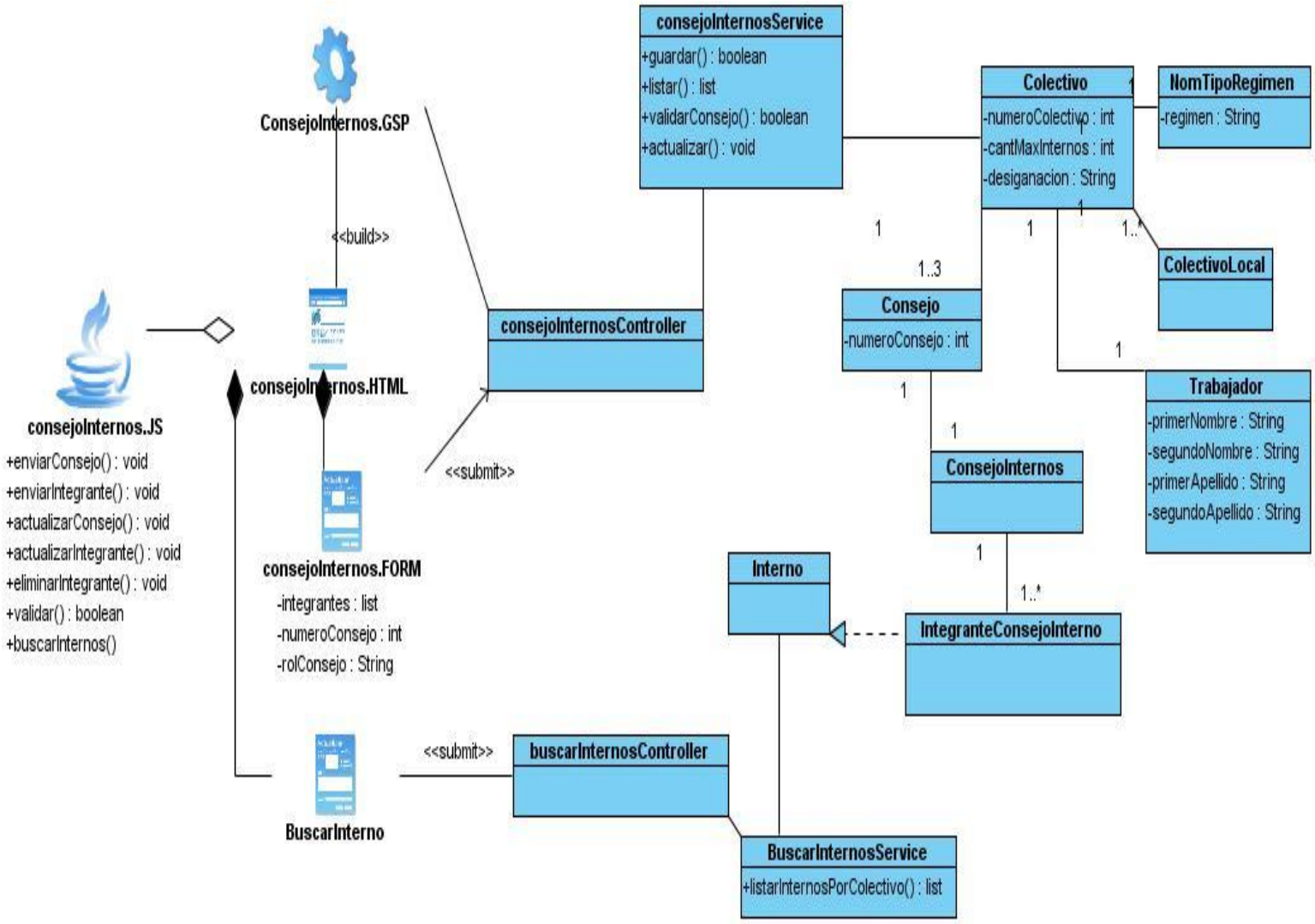


Figura 16 Diagrama de clases de diseño, CRUD-D Consejo de internos.

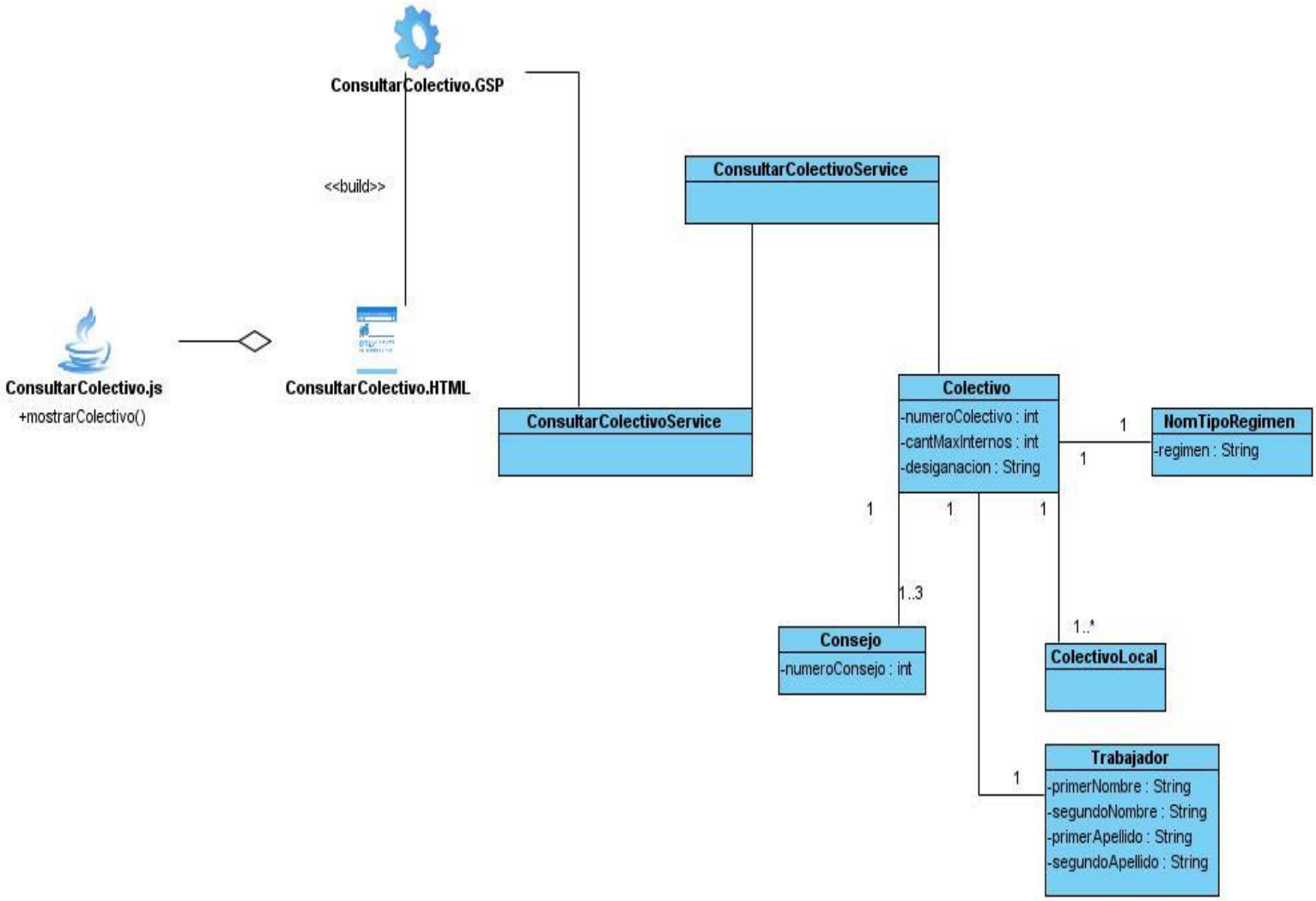


Figura 17 Diagrama de clases de diseño, CRUD-D Colectivo

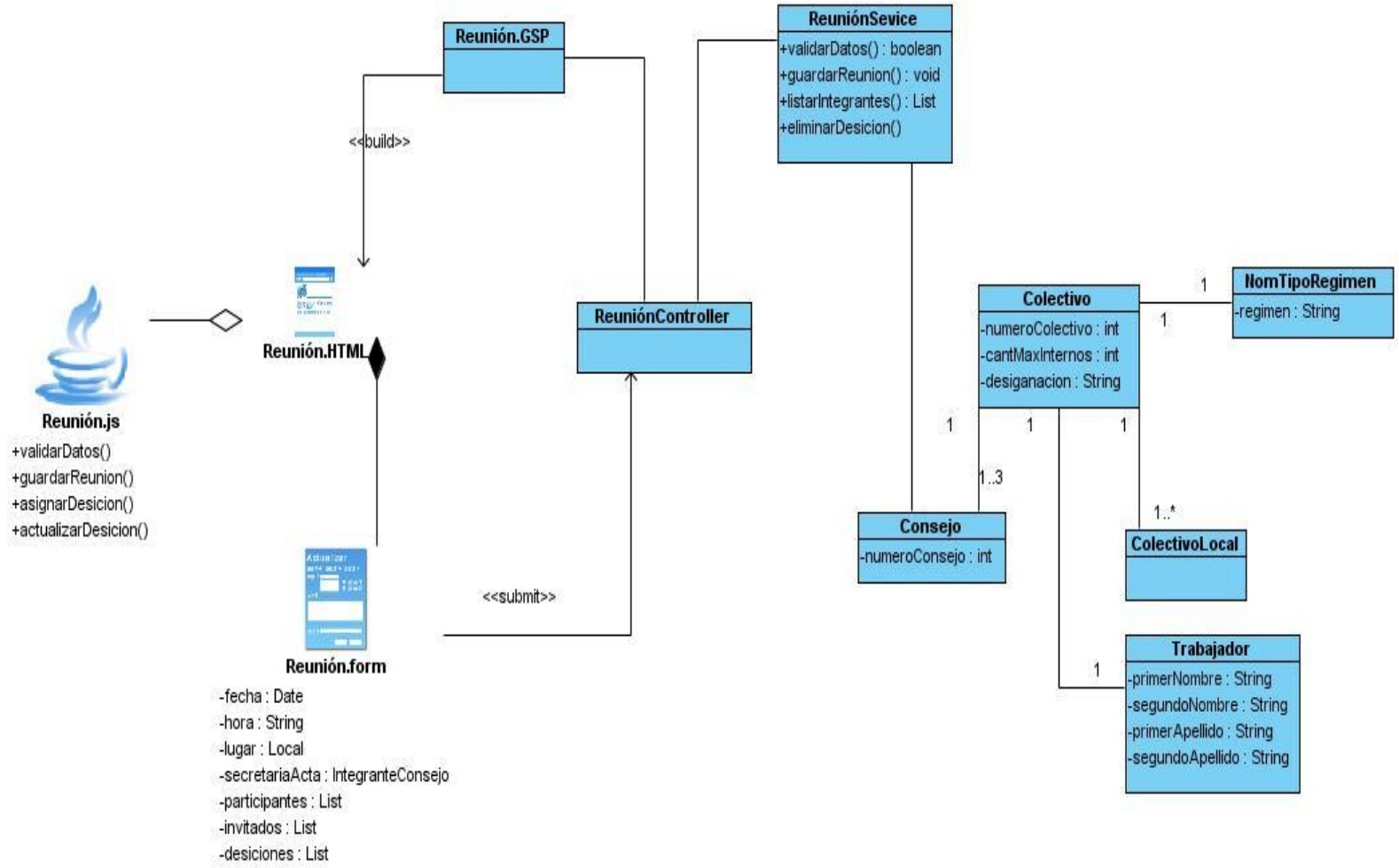


Figura 18 Diagrama de Clases de diseño, Registrar reunión