# Universidad de las Ciencias Informáticas
## Facultad 4


# Título: Diseño e Implementación del Módulo de Información Histórica de la Sala Situacional del SIGEP

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

**Autor(es):**

Liesky Díaz González

**Tutor(es):**

Ing. Maykel Pérez Martínez

Ing. Adolfo M. Iglesias Chaviano

Junio 2008

---

## DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Facultad 4 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Liesky Díaz González

Ing. Maikel Pérez Martínez

Ing. Adolfo M. Iglesias Chaviano

## AGRADECIMIENTOS

***Le agradezco a todos los que de una forma u otra siempre me han ayudado.***

---

## DEDICATORIA

***Dedico este trabajo especialmente a la memoria de mi padre el cual lo es todo para mí, a mi querida madre, mi hermano y mis tíos Norberto y Mildrey y demás familiares. Gracias también a Nadiesda por su apoyo y entrega en todo momento, a mis compañeros y amigos.***

**RESUMEN**

En América la situación penitenciara es un punto débil en todo sentido, el deterioro de este sistema está regido producto al aumento de la pobreza, la propia explotación a la que están sometidos los pueblos, la corrupción que existe en estos países, entre otros factores directa o indirectamente. Las prisiones se han convertido en lugares de matanza, de corrupción, tráfico de drogas, descontrol y supervivencia de aquellos que por alguna razón se encuentran recluidos de su libertad. La República de Venezuela no se encuentra fuera de esta situación, sin embargo hay una voluntad política de revertir esta situación.

Este trabajo consiste en el diseño y desarrollo de de un sistema informático que muestre la información histórica del Sistema Penitenciario Venezolano para así apoyar la toma de decisiones y el análisis de la información existente. Durante el desarrollo del trabajo, se realizaron diferentes tareas que permitieron el cumplimiento de todos los objetivos planteados, utilizando para esto, herramientas y metodologías reconocidas mundialmente.

**Palabras Claves:**

Sala Situacional, Sistema Penitenciario Venezolano, Información histórica, Tablas, Gráficas, Configuración.

## TABLA DE CONTENIDOS

## ÍNDICE DE FIGURAS

## ÍNDICE DE TABLAS

---

## INTRODUCCIÓN

El sistema penitenciario de Venezuela está integrado en la actualidad por un total de 33 prisiones a lo largo de toda la geografía del país, casi un tercio de las cuáles se concentran en la zona del central (Caracas y estado Miranda). Administrativamente, el sistema está conformado por penitenciarías, cárceles nacionales o locales y colonias penitenciarias – destinadas todas ellas a albergar a condenados (población que cumple condena), en función del tipo de condena – e internados judiciales – destinados a albergar a procesados (población reclusa en espera de sentencia). En la práctica, existe una notable distorsión de la función que cumple cada centro, habiendo algunos cuya población es reflejo de su denominación oficial, y otros donde conviven los condenados y procesados con criterios laxos de separación. [2]

A principios del año 2005, se elaboró un censo nacional relacionado a la situación judicial de la población penitenciaria. Este censo permitió un acercamiento a la situación en la que se encontraba el sistema penitenciario. [2]

Aunque fueron válidas las intenciones del censo, este mecanismo carece de la posibilidad de centrar la información y sobre todo de actualizar la misma.

Después de analizar detalladamente los datos recogidos acerca del Sistema Penitenciario Venezolano, se llegó a la conclusión de que existían serios problemas con esta entidad en el país. Estos problemas se ubicaban fundamentalmente en [2]:

-La demora en el proceso, lo cual conlleva a un retraso en la imposición de la pena y en el otorgamiento de fórmulas alternativas en el cumplimiento de la pena.

-El escaso nivel de información con respecto al expediente judicial de cada individuo.

-La pobre clasificación de la población penitenciaria, es decir no existía una caracterización de un individuo que ingresara en un sistema penitenciario tales como:

- Delito
- Grado de peligrosidad
- progresividad
- Perfil psicosocial
- Estado de salud

Esto obstaculiza la ejecución de una adecuada política de ubicación dentro del establecimiento penitenciario [2]:

- Carencia de identificación biométrica de los privados de libertad, lo que favorece prácticas reprobables, como la suplantación de identidad.
- Dificultades en la formulación y seguimiento de un plan coherente de rehabilitación, que pueda ser evaluado con cierta periodicidad y admita reajustes.
- Comunicación deficitaria y ausencia de controles de gestión entre los establecimientos penitenciarios y la Dirección de Servicios Penitenciarios, lo que favorece un clima de anarquía en los establecimientos penitenciarios.
- Hacinamiento en la mayoría de los centros penitenciarios con inadecuadas instalaciones físicas dando como resultado, condiciones de vidas infrahumanas.
- Graves deficiencias en materia de servicios públicos y asistenciales.
- Impera la ingobernabilidad, la violencia, la extorsión, la corrupción, las concesiones.
- La imposibilidad de tener un conocimiento detallado de la población penitenciaria ya fuera a nivel de centros penitenciarios, de Coordinación Regional, como a nivel nacional. No existía la forma de tener de forma organizada y detallada la información pasada.
- Comunicación deficitaria y ausencia de controles de gestión entre los establecimientos penitenciarios y la Dirección de Servicios Penitenciarios, lo que favorece un clima de anarquía en los establecimientos penitenciarios.

Estas características constituyen la esencia de Sistema Penitenciario Venezolano en la actualidad, las cuales son totalmente antagónicas con los deseos del gobierno de tener a este sistema, más los esfuerzos gubernamentales que se realizan para la humanización del sistema.

La información que llega es a través de correo electrónico, teléfono, o valijas, es la que la dirección almacena, por lo que no se considera una información oportuna, ni completa en muchos de los casos, dando consigo un desconocimiento casi total de la situación histórica de los establecimientos penitenciarios e imposibilitando el control sobre el clima carcelario porque:

- No se cuenta con un mecanismo de actualización de la información.
- No se pueden establecer estrategias reales que beneficien al sistema penitenciario y con ello a los internos.
- No existe la posibilidad de realizar una evaluación constante de la progresividad de los privados de libertad.

- No se puede controlar que se cumpla con los trámites correspondientes con el otorgamiento de los beneficios definidos en la ley penitenciara venezolana.
- Resulta casi imposible realizar una valoración con exactitud del estado de salud de los reclusos, imposibilitando tomar medidas y dedicar recursos para mitigar propagación de epidemias y otras enfermedades.

En la actualidad este Sistema Penitenciario Venezolano presenta dificultades a la hora de tomar decisiones o medidas referentes al comportamiento y características de los reclusos. La información que se recibe no es la más oportuna, pues en muchos casos no es la más actualizada. Teniendo en cuenta estos aspectos no se pueden tomar decisiones de índole tácticas y estratégicas, por lo que el entorno que se puede catalogar no es un espacio para el diagnóstico, revisión de los antecedentes y valoración del contexto en el cual se ejecutan las políticas por las cuales se rige una entidad o un ente determinado.

En general, con la poca información de la que se dispone de los centros penitenciarios, la Dirección General no puede tomar una decisión objetiva a los problemas que se puedan presentar en los centros penitenciarios, lo cual, frena la misión y la visión de la Dirección General de Servicios Penitenciarios en su logro por un sistema justo y humano. Por lo tanto recobra una gran importancia tener un espacio que favorezca el análisis del entorno, pero ésta demanda la construcción y actualización en la captura, registro y análisis de la información; que faciliten las propuestas de inferencias por parte de los directivos y por las cuales se construirán las futuras políticas de la dirección.

Dada la situación existente en el Sistema Penitenciario que abarca a los diferentes niveles, nacionales, regionales y centrales surge el siguiente Problema:
¿Cómo gestionar y configurar la presentación de la información referente al sistema penitenciario venezolano en la Sala Situacional, de forma tal que esta sirva para realizar análisis del comportamiento histórico del proceso penitenciario?

Teniendo en cuenta el problema presentado, se define como Objeto de Estudio:
La gestión de la información histórica del Sistema Penitenciario Venezolano.
Por lo que se especifica el siguiente Campo de Acción: El proceso de apoyo a la recepción y análisis de la información penitenciaria en la Sala Situacional de la Dirección General de Servicios Penitenciarios Venezolanos.

Dado el estado en que se encuentra la recepción y el poco análisis de la información en la Dirección Penitenciario y las condiciones descritas, se plantea el siguiente Objetivo General: Automatizar una herramienta que muestre la información histórica del Sistema Penitenciario Venezolano, que servirá de base para la toma de decisiones de índole tácticas y estratégicas en la Dirección General de Servicios Penitenciarios (DGSP), tomando como fuente informativa la base de datos centralizada que mantiene actualizada el Sistema de Gestión Penitenciaria (SIGEP).

Objetivos específicos:

1- Diseñar el mecanismo que se utilizará para brindar información histórica.
2- Desarrollar la implementación del mecanismo que se utilizará para brindar información histórica.

Para lograr los objetivos específicos planteados se define las siguientes tareas generales:

1. Investigar sobre los distintos tipos de gráficos posibles a implementar.
2. Analizar los procesos del negocio relacionados con la presentación de la información.
3. Diseñar los procesos de la información penitenciaria en Sala Situacional.
4. Definir técnicas para mostrar la información.
5. Implementar las funcionalidades del módulo Información histórica, en correspondencia con los objetivos específicos planteados.

Todos los objetivos y las tareas que se han identificado, están en correspondencia con el avance que presenta el desarrollo del proyecto SIGEP.

Para la realización de este trabajo se aplicaron varios métodos de investigación que ayudaron a determinar el problema y el desarrollo del mismo, tales como:

- La observación.
- Histórico-Lógico.
- Modelación.
- Las entrevistas.
- Hipotético-Deductivo.

Fueron estos los métodos aplicados para la identificación de las principales informaciones que se reciben y se generan en la dirección. También se utilizó el Analítico Sintético como parte del método teórico para la propia conceptualización de la Sala Situacional.

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

## 1.1. Actualidad e importancia del trabajo

La creación de un espacio para el análisis sobre un fenómeno determinado o sobre una entidad específica, permite el control sobre su funcionamiento, este análisis de la información se concentra en la determinación de las tendencias de comportamiento, facilitando la clasificación de las áreas, regiones o sectores donde se requieren atención urgente, o regular. Contando con una cultura de análisis y uso adecuado de la información se podrían determinar los factores que influyen en situaciones normales y críticas.

En consecuencia con lo visto anteriormente, con el estado en que se encuentra la propia dirección del Sistema Penitenciario y contando con un Sistema Informático que gestione los procesos del Sistema Carcelario Venezolano, urge la necesidad de desarrollar una aplicación para que apoye el trabajo de la Sala Situacional, brindando información constante sobre el comportamiento de los principales indicadores, emitiendo diferentes tipos de alertas en caso de detectar irregularidades en su comportamiento, mostrando además, la información más actual y oportuna posible, reflejando así la realidad penitenciaria. Este sistema informático constituiría una herramienta fundamental para establecer en la dirección general un control estricto de la situación carcelaria del país, de igual manera facilitaría el mando, el análisis del entorno y la creación de políticas en víspera de un buen funcionamiento del Sistema Penitenciario.

## 1.2. Métodos, herramientas y técnicas

### 1.2.1. Plataforma de desarrollo

Una plataforma de desarrollo es el entorno común en el cual se desenvuelve la programación de un grupo definido de aplicaciones. Comúnmente se encuentra relacionada directamente a un sistema operativo, sin embargo, también es posible encontrarlas ligadas a una familia de lenguajes de programación o a una Interfaz de programación de aplicaciones o API (Interfaz de Programación de Aplicaciones) por sus siglas en inglés.

Es una plataforma para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura distribuida por capas, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones.

Java Platform, Enterprise Edition o Java EE (anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE hasta la versión 1.4), es una plataforma de programación—parte de la Plataforma Java—para desarrollar y ejecutar software de aplicaciones en Lenguaje de programación Java con arquitectura de N niveles distribuida, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. Java EE incluye varias especificaciones de Interfaz de Programación de Aplicaciones (API), tales como Conectividad de la Base de Datos de Java (JDBC), Invocación de Métodos Remotos (RMI), e-mail, Servicios de Mensajería de Java (JMS), Servicios Web, Lenguaje de Marcas Extensible (XML), entre otros y define cómo coordinarlos. Java EE también configura algunas especificaciones únicas para Java EE para componentes. Estas incluyen Enterprise JavaBeans, servlets, portlets (siguiendo la especificación de Portlets Java), JavaServer Pages y varias tecnologías de servicios web. Esto permite al desarrollador crear una Aplicación de Empresa portable entre plataformas y escalable, a la vez que integrable con tecnologías anteriores. Otros beneficios añadidos son, por ejemplo, que el servidor de aplicaciones puede manejar transacciones, la seguridad, escalabilidad, concurrencia y gestión de los componentes desplegados, significando que los desarrolladores pueden concentrarse más en la lógica de negocio de los componentes en lugar de en tareas de mantenimiento de bajo nivel.[3]

.

### 1.2.2. Lenguaje de programación

El lenguaje de programación Java presenta características muy favorables que junto al hecho de que es un lenguaje libre, pudiéndose utilizar el compilador y la maquina virtual de forma gratuita le asegura una gran popularidad a nivel mundial. Entre sus características se encuentran: es un lenguaje simple, orientado a objetos e independiente de la arquitectura.

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsytems. El lenguaje en sí mismo tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros memoria. Entre las ventajas más comunes de este lenguaje se pueden citar las siguientes:
Orientado a objetos

La Programación Orientada a Objetos (POO) es un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas de computadora. Está basado en varias técnicas, incluyendo herencia, modularidad, polimorfismo, y encapsulamiento.

### 1.2.3. Plugins

**Spring IDE**

Spring IDE es un plugin que sirve como interfaz de usuario gráfica para la configuración de los archivos usados por Spring Framework. Permite el completamiento de etiquetas, valores de atributos y elementos en estos archivos de configuración. [3]

**Hibernate**

Hibernate Tools es una herramienta de Mapeo objeto-relacional para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones. [4]

Es un motor de persistencia, que es el componente software encargado de traducir entre objetos y registros. [4]

**Subclipse**

Subclipse es un plugin para Eclipse que adiciona integración para el control de versiones (Subversion, específicamente), permitiendo operaciones de sincronización, actualización, entre otras. Permite bloqueos a recursos para que otros usuarios no puedan modificarlo. Dispone de una vista de comparación entra el recurso local y remoto en caso que exista conflicto entre la versión del recurso local con el remoto. Muestra una vista del historial de versiones de los recursos con un conjunto de atributos de las acciones realizadas sobre el recurso.

Soporta conectarse a varios repositorios de control de versiones al mismo tiempo, permitiendo hacer operaciones sobre el repositorio directamente.

### 1.2.4. Contenedor Web

**Apache Tomcat**

Apache Tomcat es un contenedor de Servlet usado en la implementación de referencia oficial para las tecnologías Java Servlet y JavaServer Pages. Es desarrollado en un ambiente participativo y abierto. La versión 5.x es implementada a partir de las especificaciones Servlet 2.4 y JSP 2.0 y cuenta con un

mecanismo de recolección de basura perfeccionado, basado en su reducción. Posee una capa envolvente nativa para Windows y Unix para la integración de las plataformas.

### 1.2.5. Control de versiones

Es la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Los sistemas de control de versiones facilitan la administración de las distintas versiones de cada producto desarrollado, así como las posibles especializaciones realizadas. El control de versiones se realiza principalmente en la industria informática para controlar las distintas versiones del código fuente. Sin embargo, los mismos conceptos son aplicables a otros ámbitos como documentos, imágenes, sitios web, etcétera.

**Subversion**

Es un software de sistema de control de versiones de código abierto y gratuito. Soporta el manejo de ficheros y directorios a través del tiempo. Hay un árbol de ficheros en un repositorio central. El repositorio es como un servidor de ficheros ordinario, excepto porque recuerda todos los cambios hechos a sus ficheros y directorios, permitiendo recuperar versiones antiguas de sus datos, o examinar el historial de cambios de los mismos.

### 1.2.6. Sistemas Gestores de Base de Datos

**Sistemas Gestores de Base de Datos (Oracle 10g)**

Oracle es un sistema de gestión de base de datos relacional (o RDBMS por el acrónimo en inglés Relational Data Base Management System), fabricado por Oracle Corporation. Se considera uno de los sistemas de bases de datos más completos. Es un sistema gestor de base de datos robusto, tiene muchas características que garantice la seguridad e integridad de los datos; que las transacciones se ejecuten de forma correcta, sin causar inconsistencias; ayuda a administrar y almacenar grandes volúmenes de datos; estabilidad, escalabilidad y es multiplataforma.

### 1.2.7. Herramientas de modelado

UML, por sus siglas en inglés, *Unified Modeling Language* 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 de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir.

**Visual Paradigm Suite**

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

### 1.2.8. Frameworks

En general, el término framework, se refiriere a una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que se pueden añadir las últimas piezas para construir una aplicación concreta. Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo, como el uso de patrones. Los frameworks son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional.

**Spring framework.**

Spring Framework (también conocido simplemente como Spring) es un Framework de código abierto para el desarrollo de aplicaciones en la plataforma Java. Spring es el más popular de todos los frameworks de peso ligero en la actualidad, con respecto al desarrollo de aplicaciones web en Java. Spring además presenta una arquitectura muy flexible. La gran ventaja que presenta la utilización de este framework recae en la presencia de varios módulos completamente independientes unos de otros.

**Hibernate framework.**

Hibernate es una capa de persistencia objeto/relacional y un generador de sentencias SQL de código abierto. Con este framework, se puede diseñar objetos persistentes que podrán incluir relaciones, colecciones, polimorfismo, y un gran número de tipos de datos. Con Hibernate se logra una abstracción total del gestor de base de datos a utilizar. De una manera muy rápida y optimizada se puede realizar consultas contra cualquiera de los entornos soportados: Oracle, DB2, MySql, etc. Otra ventaja que existe en la utilización de este frameworks, es el propio dialecto que propone Hibernate, Hibernate Query Languaje (HQL), para desarrollar las consultas. Con HQL se obtienen los resultados de una consulta de forma objetual lo que permite desarrollar la capa de acceso a dato más rápido e intuitivo para el programador

### 1.2.9. API

Una API (del inglés Application Programming Interface - Interfaz de Programación de Aplicaciones) es un conjunto de especificaciones de comunicación entre componentes software. Se trata del conjunto de llamadas al sistema que ofrecen acceso a los servicios del sistema desde los procesos y representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general, por ejemplo, para dibujar ventanas o iconos en la pantalla. De esta forma, los programadores se benefician de las ventajas de la API haciendo uso de su funcionalidad, evitándose el trabajo de programar todo desde el principio.

**Jfreechart API**

JFreeChart es un API libre para Java que facilita la creación de gráficas de distintos tipos, que una vez creadas pueden ser utilizadas perfectamente en una pagina web o formando pare de un reporte PDF, XLS, etc.

**JasperReport API.**

JasperReports es una poderosa API libre para Java para la creación de reportes en variados formatos como PDF, HTML, XLS, CSV y XML de forma relativamente sencilla.

### 1.2.10. IDE

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI, Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios.

Eclipse IDE cuenta con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código. El IDE también hace uso de un espacio de trabajo, en este caso un grupo de metadata en un espacio para archivos plano, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente, emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la plataforma de cliente rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Adicionalmente Eclipse puede extenderse usando otros lenguajes de programación como son C/C++ y Phyton, además de trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos. La arquitectura plugin permite escribir cualquier extensión deseada en el ambiente. Se provee soporte para Java y CVS en el SDK de Eclipse.

Subclipse es un plugin para Eclipse que adiciona integración para el control de versiones (Subversion, específicamente), permitiendo operaciones de sincronización, actualización, entre otras. Permite bloqueos a recursos para que otros usuarios no puedan modificarlo. Dispone de una vista de comparación entra el recurso local y remoto en caso que exista conflicto entre la versión del recurso local con el remoto. Muestra una vista del historial de versiones de los recursos con un conjunto de atributos de las acciones realizadas sobre el recurso.

Soporta conectarse a varios repositorios de control de versiones al mismo tiempo, permitiendo hacer operaciones sobre el repositorio directamente.

### 1.2.11. Patrones de Diseño

Los patrones de diseño proponen una manera de reutilizar la experiencia de los desarrolladores, para ello se clasifican y se describen formas de solucionar problemas que ocurren frecuentemente en el desarrollo

del software. Por tanto los patrones de diseño están basados en la recopilación del conocimiento de los expertos en desarrollo de software y se puede plantear que son soluciones concretas porque proponen soluciones a problemas concretos, son soluciones técnicas porque indican resoluciones técnicas basadas en la Programación Orientada a Objetos (POO), se utilizan además en situaciones frecuentes ya que se basan en la experiencia acumulada en resolver problemas que se han convertido en reiterativos y también presenta una característica muy importante, favorecen la reutilización de código.

## 1.3. Flujo de trabajo del rol de Programador de Interfaz de Usuario.

### 1.3.1 Actividad 1: Confeccionar el mapa de navegación

Diagrama de navegación muestra las interacciones entre las vistas y el servidor, que de alguna manera reflejen las posibles peticiones y las correspondientes respuestas. De aquí se determinan los nombres lógicos de las vistas y los nombres de las URLs de las peticiones.

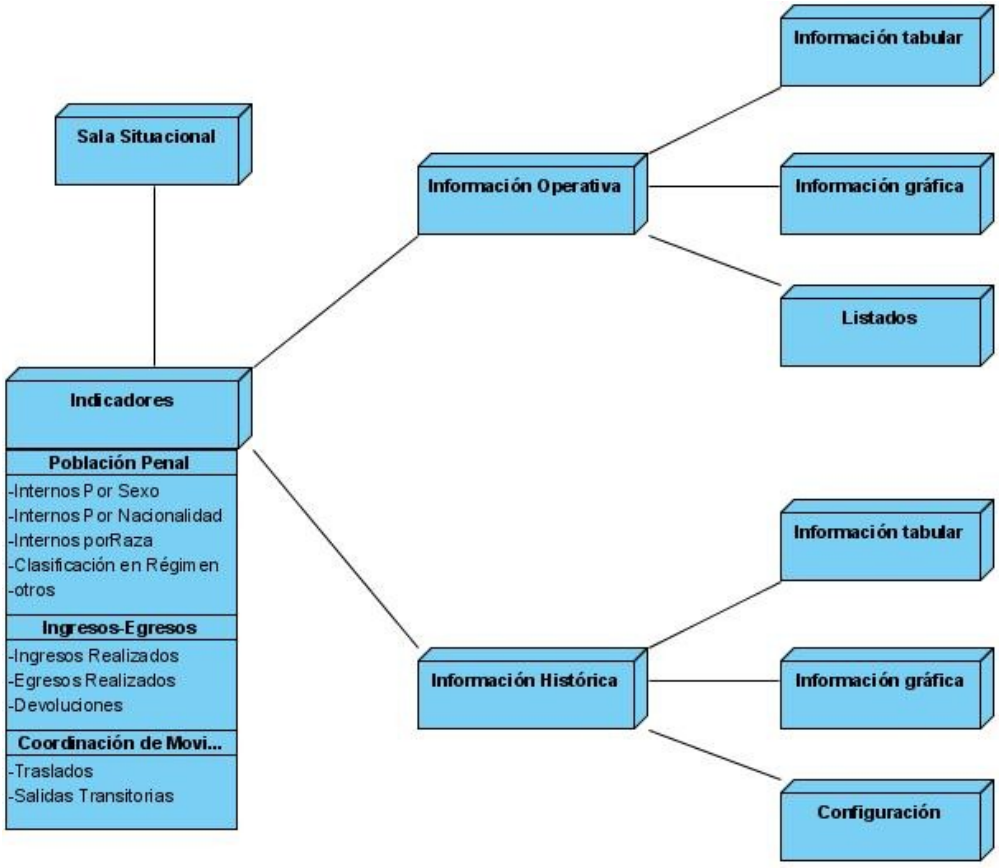


**Figure 1 Mapa de Navegación**

### 1.3.2 Actividad 2: Dado el diagrama anterior, definir tipos de controladores que se van a usar utilizando un mapa de controladores predefinidos.

La clase GestorReporteInformacionHistoricaController (MultiActionController)es un controlador de Spring, cuya funcionalidad es escuchar las peticiones de generar reportes en formatos PDF y XLS , obtener gráficas, tablas de reporte y configurar los reportes, de forma tal que se pueda formular una respuesta acorde.

### 1.3.3 Actividad 3: Confeccionar la tabla (controlador-petición-vista). Se definen los nombres de las peticiones y las respuestas por controlador

En una tabla con tres columnas, cuyos encabezados son [controlador, petición y vistas], dejar reflejados, los controladores, las peticiones a las que va a responder, y las posibles vistas a devolver en cada caso.

| Controlador | Petición | Vistas |
|---|---|---|
| GestorReporteInformacionHistoricaController | loadReportProp<br>loadTableView<br>loadGraphicView<br>configurationView<br>saveConfigurationView | mainTemplateIH<br>TableView<br>GraphicView<br>ConfigurationView |

**Tabla 1 Controlador-Petición-Vista**

### 1.3.4 Actividad 4: Declarar e Implementar controladores sin lógica.

Crear las clases de cada uno de los controladores que se determinaron según el paso anterior, además de declarar los métodos que se deberán sobrescribir o implementar en cada caso, así como declarar los atributos y dependencias con otras clases que tenga, tales como fachada de negocio, etc. Comentar, a partir de las especificaciones establecidas para generar adecuadamente los javadocs, el encabezado de cada clase a implementar, así como cada método declarado.

### 1.3.5 Actividad 5: Declarar los controladores en los ficheros XML de configuración de Spring.

Los controladores se mapean en el correspondiente fichero sigep-subsistema-modulo-servlet.xml, que se encuentra en el paquete vnz.sigep.subsistema.modulo.configuration (y también en el fichero sigep-subsistema-modulo-servlet-test.xml, que se encuentra en el mismo paquete; esto es para procurar tener un XML de configuración exclusivo a soportar y apoyar las pruebas de unidad necesarias a cada controlador posteriormente implementado). A partir de los HandlerMappings que se hayan declarado en el mismo fichero, se configurará la correspondiente petición o las peticiones con el controlador en el mismo.

### 1.3.6 Actividad 6: Implementar lógica de los controladores (posible interacción: debe estar definida las Fachadas de Negocio y las entidades, y se piden métodos necesarios en caso de que no existan.

Aquí se le da solución a la implementación de las funcionalidades necesarias para resolver cada problemática en los casos de uso.

### 1.3.7 Actividad 7: Crear las vistas, separarlas física y lógicamente a partir de las consideraciones que se hayan tomado en cuenta para crearlas previamente por parte de este y del diseñador gráfico.

**Vistas con tecnologías JSP**

Si las vistas son hechas con tecnología JSP, los correspondientes ficheros .JSP obtenidos deberán ser puestos en la carpeta /WEB-INF/jsps/[subsistema]/[modulo]/ cuyos nombres deben de alguna manera clara y breve relacionarse o describir el fin para el cuál han sido destinadas.

**Vistas con JasperReport**

En caso de que estas sean reportes, es decir, salidas en formatos no HTML, tales como PDF, XLS, RTF, etc.

**Vistas usando JFreeChart**

En este caso se usa la librería JfreeChat para generar las gráficas de línea que se expondrán, ya que brinda una gran funcionalidad a la hora de desarrollar vistas de este tipo.

### 1.3.7 Actividad 8: Machear vista con modelo y programar lógica en el cliente.

**Programar scripts (java script)**

Se debe crear un archivo .js para cada jsp que necesite de código javascript. En ese archivo se deberán programar todas las funciones y objetos que se necesite en la jsp. A continuación, algunas consideraciones sobre estos archivo js:

1. deben ser ubicados en la carpeta WebContent/js/scripts/[nombre_del_módulo] la primera línea de los archivos debe ser dojo.provide("scripts.[nombre_del_módulo].[nombre_del_archivo:

<script>

dojo.require("scripts.[nombre_del_módulo].[nombre_del_archivo]");

</script>

donde [nombre_del_archivo] es el nombre del archivo sin incluir la extensión.

**Funcionalidades comunes**

En caso de que se programen funcionalidades javascript que no sean específicas del módulo, sino que puedan ser utilizadas por otros módulos (como funciones de utilidades) se propone que se pongan en archivos .js dentro de la carpeta WebContent/js/scripts/common. Estas funcionalidades deben ser minuciosamente documentadas (haciendo uso de los comentarios javascript), para facilitar la reutilización.

**Ejecutar código javascript al cargar una página**

Si se necesita ejecutar cierto bloque de instrucciones javascript al cargar una jsp, dicho bloque debe ser puesto dentro de una función javascript y en la jsp se debe hacer la llamada a esa función.

**Ajax**

Para enviar peticiones asíncronas se propone usar la función javascript dojo.io.bind (request)

**Vistas con JasperReport**

En caso de que estas sean reportes, es decir, salidas en formatos no HTML, tales como PDF, XLS, RTF, etc.

Si el diseño lo requiere, se implementarán clases que apoyan la creación de los reportes, tales como DataSources, ChartCustomizers.

### 1.3.8 Actividad 10: Realizar pruebas a la interfaz para verificar el correcto funcionamiento e interacción de esta con el supuesto usuario final.

El programador interactúa directamente con la aplicación y con la interfaz o las interfaces asociada(s) al caso de uso y corrige los detalles que crea pertinentes para que esta muestre los datos como es debido, (supuestamente el controlador ya ha sido probado por lo que no ha de ser necesario modificar el código del mismo, a no ser que se detecte alguna falla).

### 1.3.9 Actividad 11: Implementar validaciones en el cliente.

Se usa JS para validar en el cliente la entrada incorrecta de datos

### 1.3.10 Actividad 12: Probar que el reporte es devuelto correctamente desde la web.

# CAPÍTULO 2: DESCRIPCIÓN DEL SISTEMA

## 2.1 Características generales del Sistema de Gestión Penitenciaria

En este epígrafe se describe la propuesta del software que propone la gestión de la Información Histórica en Sala Situacional del Sistema de Gestión Penitenciaria. El alcance del mismo se ajusta a lo previsto para la primera etapa del desarrollo del SIGEP que comprendes las áreas de Control Penal, Registro y Control de Visitantes, Observación e Historia Clínica. Todos estos módulos en la actualidad ya están desarrollados, lo que equivale a que exista un flujo de información, el cuál será utilizado para general todo lo relacionado con la Información Histórica de dichos módulos. [2]

La solución informática se apoyará en diferentes técnicas para mostrar los datos correspondientes, utilizando la Sala Situacional. El Sistema mostrará las informaciones en varios niveles de desagregación, es decir, de lo general a lo particular. Por otra parte se podrá filtrar la información mostrada y se generarán reportes con frecuencias determinadas.

Esta información generada se almacena en una Base de Datos localizada en la Dirección General de Servicios Penitenciarios, convirtiéndose en la fuente de información del sistema informático para la Sala Situacional y más explícitamente para el trabajo de Información Histórica de dicho módulo. Se mostrará en éste el comportamiento histórico de: Información Operativa, Novedades, Situación Jurídica, Traslados, Visitas Familiares e Institucionales entre otras a través de tablas y gráficas permitiendo a los especialistas que laboren en Sala Situacional contar con la información necesaria para ver el comportamiento de la información, así como la toma de decisiones oportunas y tácticas, y para establecer estrategias según corresponda. [2]

El Sistema de Gestión Penitenciaria (SIGEP), dará respuesta a las necesidades de gestión, información y apoyo a la toma de decisiones de la Dirección General de Servicios Penitenciarios, en tres niveles:

- Operativo: integrado por los Establecimientos penitenciarios (Internados Judiciales y Centros Penitenciarios), Centros de Tratamiento Comunitario (CTC) y Unidades Técnicas de Apoyo al Sistema Penitenciario (UTASP).
- Táctico: integrado por las Coordinaciones Regionales.
- Estratégico: integrado por la Dirección General de Custodia y Rehabilitación del recluso.

## 2.2 Solución de Software Propuesta

El módulo de información histórica se encarga de evaluar el comportamiento de un elemento determinado en un espacio de tiempo especificado. Potencia, además el análisis de la información recopilada desde todo el país en asuntos penitenciarios. Información Histórica tiene como misión fundamental:

Garantizar la continuidad del mando, mediante la recepción, control, seguimiento, canalización y análisis oportuno de las informaciones que se tramitan.

De inviolable cumplimiento son los siguientes objetivos:

1. Generar reportes y gráficas que apoyen a las decisiones de índole táctica y estratégica.
2. Apoyar a la dirección a evitar, a controlar y a monitorear irregularidades en el sistema penitenciario.

Para dar cumplimiento a los objetivos trazados y paralelamente esté en correspondencia que el alcance del sistema de gestión penitenciaria:

1. La información histórica que permita evaluar comportamiento de diferentes aspectos del sistema.

La Información Histórica no es más que acumulación de la propia información operativa que luego permitirá establecer evaluaciones entre intervalos de fechas. El sistema informático que apoya a la gestión de la Sala Situacional de la Dirección General de Servicios Penitenciarios contará además, con una sección donde se observará el comportamiento en el tiempo de disímiles informaciones del sistema penitenciario mediante diferentes tipos de reportes. Dichos reportes reflejarán la evolución de algún aspecto, tanto a nivel nacional, regional como a nivel de centro, en intervalos de tiempo según se especifique, ya sea, por años, por mes, por semanas o por días. Las informaciones que brindarán estos reportes serán configuradas por el usuario. Asociado a estos reportes se generan gráficas, las cuales facilitarán la comprensión y el análisis de los datos mostrados.

Esta sección servirá como herramienta para el control sobre las diferentes áreas del sistema penitenciario, determinando si existió o si existe alguna anormalidad de acuerdo a los datos que se muestren, además que se observarán tendencias que pueden influir en decisiones tácticas o estratégicas por parte de la dirección.

Los reportes que se mostrarán contienen los siguientes elementos:

| REPORTES DE INFORMACIÓN HISTÓRICA | |
|---|---|
| **Informaciones sobre la población penal** | Total de internos primarios y reincidentes. |
| | Total de internos por sexo. |
| | Total de internos por raza. |
| | Total de internos por nacionalidad. |
| | Total de internos por régimen. |
| | Total de evadidos y fugados. |
| | Total de defunciones. |
| | Total de internos indocumentados. |
| | Total de internos por grupo etáreo. |
| | Total de internos procesados y penados. |
| | Total de internos por delito. |
| | Total de internos por progresividad. |

**Tabla 2 Reporte de información histórica A.**

| REPORTES DE INFORMACIÓN HISTÓRICA | |
|---|---|
| **Informaciones sobre ingresos y egresos** | Total de ingresos realizados. |
| | Total de egresos realizados. |
| | Total de devoluciones. |
| **Informaciones sobre los principales movimientos fuera de los establecimientos** | Comportamiento de las irregularidades en la ejecución de los traslados, salidas transitorias y transferencias. |
| **Información sobre el Control de visitas** | |
| **Informaciones visitas familiares** | Visitas realizadas |
| | Total de visitantes |
| | Total de internos con visita planificada |
| **Informaciones visitas institucionales** | Visitas realizadas |
| | Total de visitantes. |
| | Total de visitantes por cada una de las instituciones a las que pertenecen. |
| **Informaciones sobre las presentaciones a tribunales** | Comportamiento de las audiencias realizadas y no realizadas |

**Tabla 3 Reporte de información histórica B.**

| REPORTES DE INFORMACIÓN HISTÓRICA | |
|---|---|
| **Informaciones sobre Requisas y Decomisos** | Información sobre de requisas por centro |
| | Información sobre de decomisos de objetos |
| **Informaciones sobre Novedades** | Información sobre motines, secuestros, fugas, huelgas, túneles |
| | Información sobre hechos de agresión |
| | Información fallecidos |
| **Informaciones sobre Medidas Disciplinarias** | Sanciones impuestas |
| | Indisciplinas cometidas |
| **Informaciones sobre Capacidades** | Población penal en el tiempo |
| | Uso de capacidades penitenciarias |
| **Informaciones sobre Dotación de Internos** | Entrega de dotación |
| **Informaciones sobre Educación** | Información sobre individuos incorporados a actividades educativas |
| | Información sobre el grado de instrucción de la población penal |

**Tabla 4 Reporte de información histórica C.**

| REPORTES DE INFORMACIÓN HISTÓRICA | |
|---|---|
| **Información sobre individuos incorporados que trabajan** | Información sobre individuos incorporados que trabajan |
| **Informaciones sobre Deporte** | Información sobre individuos vinculados por modalidad |
| | Actividades por modalidad |
| **Informaciones sobre Cultura** | Información sobre individuos vinculados por manifestación |
| | Actividades por manifestación |
| **Informaciones sobre Evaluación Técnica** | Información sobre individuos evaluados |

**Tabla 5 Reporte de información histórica D.**

### 2.3 Descripción del Sistema

En el siguiente diagrama se muestran las principales funcionalidades que contiene el módulo de la aplicación Información histórica de Sala Situacional (Ver Descripción Textual Anexo A).

- Generar Reporte de Información Histórica.
- Generar Reporte Gráfico.
- Configurar Reporte.
- Generar Reporte Tabular.

Es necesario destacar que por cada uno de los reportes mencionados en el epígrafe anterior se realizarán las acciones de generar reporte gráfico, generar reporte tabular y configurar reporte.

El funcionario de Sala Situacional es el encargado de realizar todas las acciones referentes a la generación de los reportes y su respectiva configuración velando por el comportamiento de los mismos.

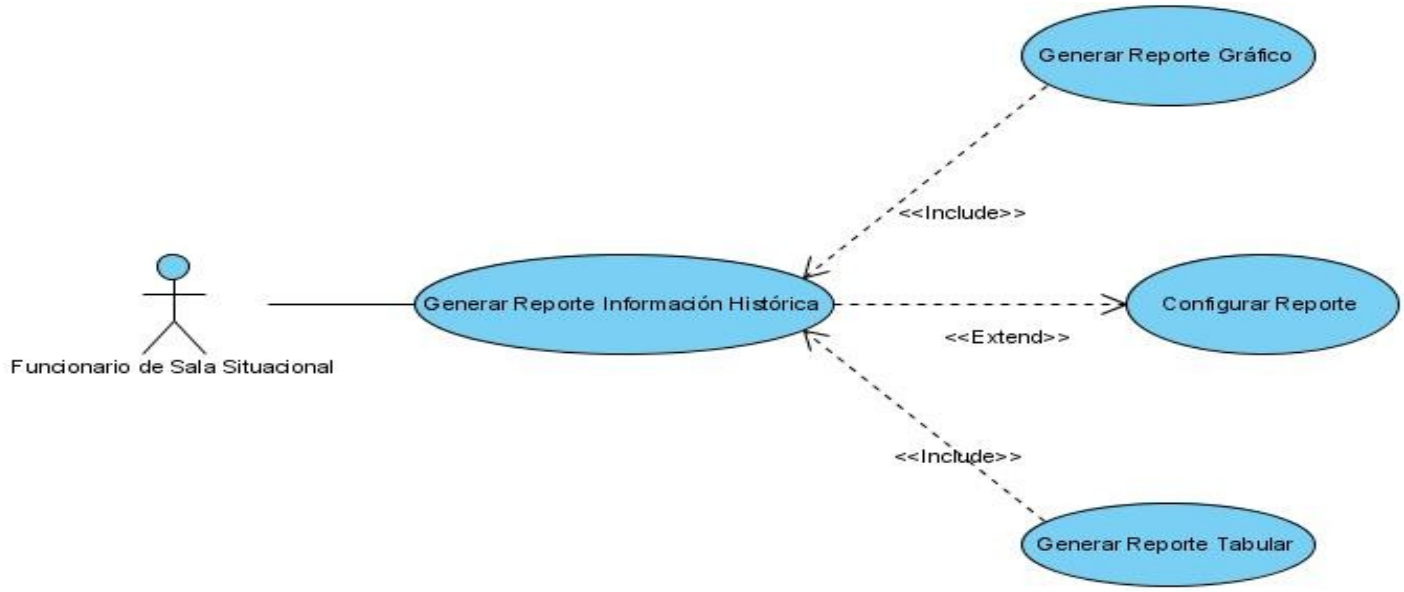


**Figure 2 Diagrama de Casos de Uso del Sistema**

## 2.4 Solución propuesta en la arquitectura definida para el desarrollo del SIGEP.

Arquitectura de tres capas: Las arquitecturas en capas son muy utilizadas para el desarrollo de aplicaciones hoy día por las grandes ventajas que proporcionan. El principal objetivo que persigue una arquitectura dividida en *n* capas es reducir dependencias entre artefactos situándolos en capas lógicas, donde cada capa depende del servicio prestado por la capa inferior y presta un servicio a la capa superior, proporcionando a los desarrolladores ventajas en cuanto al mantenimiento y reutilización de componentes o artefactos. Una arquitectura de tres capas, dividida de la forma siguiente:

### 2.4.1 Capa de presentación

Esta es la capa encargada de la interacción con los diferentes tipos de usuarios, se encarga de modelar la forma en que se mostrarán y se recogerán los datos entrados por los usuarios, así como de la apariencia visual que tendrá la aplicación. Se comunica con la capa de lógica de negocio a la cual envía todas las solicitudes de los usuarios y recibe la respuesta después de que hayan sido procesadas cada una de estas solicitudes.

- Información tabular: en este tab es donde se llama a la página en la cual se muestra la información del reporte seleccionado en forma de tabla.

- Información gráfica: en este tab es donde se llama a la página en la cual se muestra la información del reporte seleccionado en forma de gráfica siendo esta lineal.

- Configuración: en este tab es donde se llama a la página en la cual se configuran los reportes de información histórica, teniendo en cuenta intervalos de fecha, frecuencia del intervalo, como tipo de intervalo, dígase anual, mensual, semanal, diario.

Creación de componentes para la reutilización de código como son widget tanto para las tablas como para las gráficas.
Desarrollo del mapeo entre controladores y vistas usando Ajax.

### 2.4.2 Capa de Negocio

Es la encargada de modelar la lógica de negocio que dará solución a cada uno de los casos de usos de la aplicación, es en esta donde se establecen las reglas o restricciones que debe cumplir la aplicación. Se comunica con la capa de presentación para recibir las solicitudes de los usuarios y enviar las respuestas después del procesamiento.

- Definición de fachada para interactuar con la capa de negocio.
- Creación de una estructura para la generación de reportes
- Implementación de los manager

### 2.4.3 Capa de Datos

Es la encargada del manejo de datos persistentes, forma parte de la misma el servidor de base de datos como encargado de almacenamiento de datos, se comunica con la capa de lógica de negocio desde donde recibe las solicitudes de almacenar o recuperar información.

# CAPÍTULO 3: DESCRIPCIÒN DE LA SOLUCIÒN PROPUESTA

## 3.1 Diagrama de clases CU Generar reporte de Información Histórica

Los Diagramas de clases son los encargados de mostrar la estructura de un sistema mostrando sus clases, atributos y relaciones entre ellas. Es un diagrama estático.

En este diagrama se muestran las clases que colaboran para realizar el CU Generar reporte y sus relaciones.

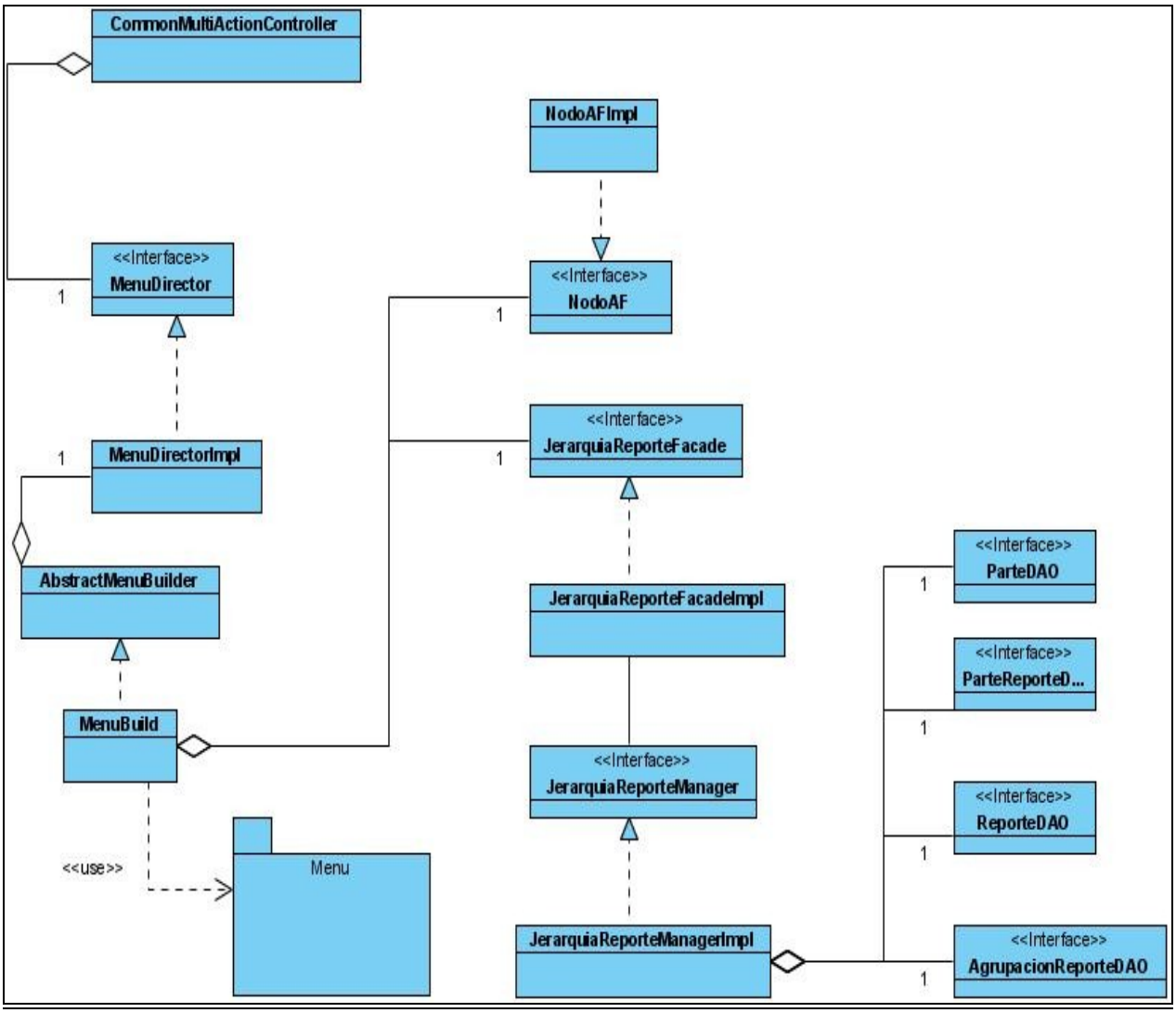


**Figure 3 Diagrama de clases CU Generar reporte de Información Histórica**

### 3.2 Descripción del paquete Menú del CU Generar Reporte Información Histórica

| **Paquete : Menú** |
| --- |
| Este paquete de utilidades tiene como objetivo representar de forma objetual el menú de la aplicación, para hacer más transparente su construcción. Dada la forma en que este se construye se decidió diseñarlo utilizando el patrón de diseño Builder, el cuál permite construir un objeto complejo por partes. |

**Tabla 6 Paquete Menú**

### 3.3 Descripción de las principales Clases del CU Generar Reporte Información Histórica

| **Nombre : CommonMultiActionController** |
| --- |
| El controlador es el responsable de responder la petición de construcción de menú. |

**Tabla 7 Clase CommonMultiActionController**

| **Nombre : MenuDirector** |
| --- |
| Esta clase es la encargada de decidir que parte del menú se mandara a construir, incluso en el orden que se hará. |

**Tabla 8 Clase MenuDirector**

| **Nombre : AbstractcMenuBuilder** |
| --- |
| Clase abstracta que define los métodos mediante los cuáles se construirán cada una de las partes del objeto menú. |

**Tabla 9 Clase AbstractcMenuBuilder**

| **Nombre : MenuBuild** |
| --- |
| Clase encargada de implementar o definir como se construye cada una de las partes del menú. |

**Tabla 10 Clase MenuBuild**

| **Nombre : NodoAF** |
| --- |
| Interfaz que define los métodos necesarios para construir cada uno de los tipos de nodo que forman al menú. |

**Tabla 11 Clase NodoAF**

| **Nombre : NodoAFImpl** |
| --- |
| Clase que se encarga de definir como se construye cada uno de los nodos del menú. |

**Tabla 12 Clase NodoAFImpl**

### 3.4 Diagrama de clases del diseño CU Generar Reporte Gráfico

Este diagrama muestra la interacción entre las clases que conforman el CU Generar reporte gráfico y sus relaciones, es válido señalar el paquete JFreeChart que contiene además un conjunto de clases que son las encargadas de crear los gráficos por los cuáles se mostrará la información.

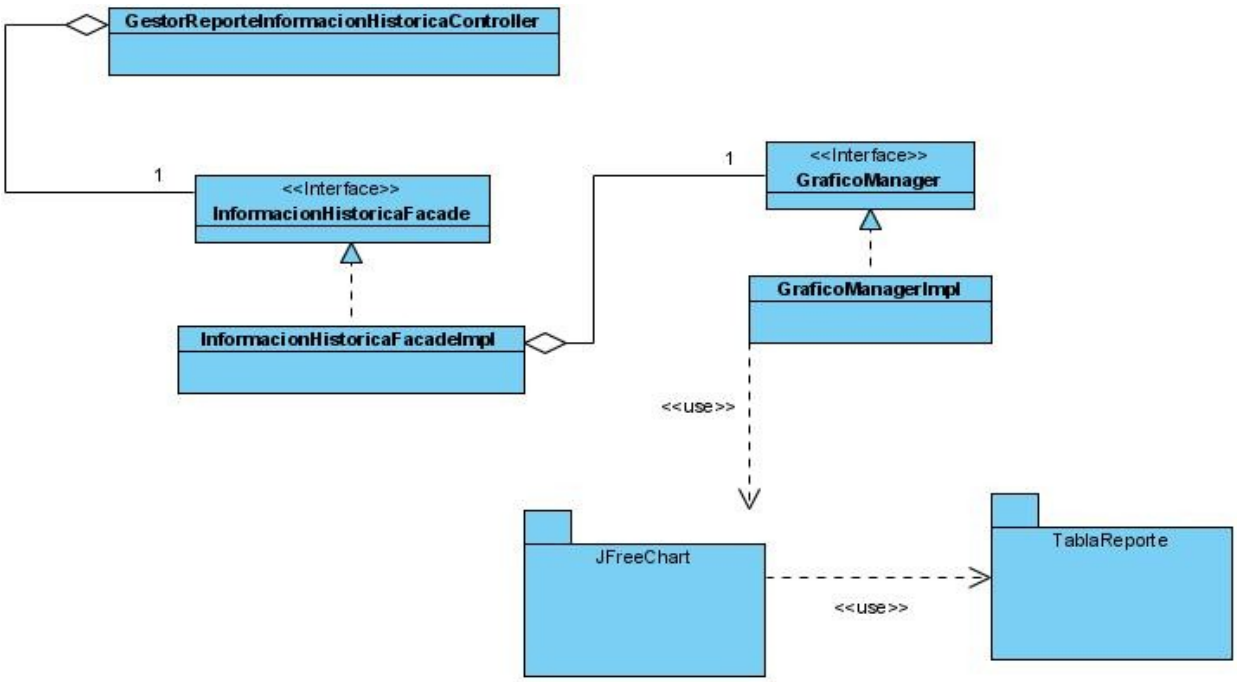


**Figure 4 Diagrama de clases del diseño CU Generar Información Gráfica**

### 3.4 Diagrama de clases del diseño CU Configurar Reporte

El diagrama que se muestra es el encargado de la estructura de las clases que colabora en el CU Configurar reporte, donde se brinda la posibilidad de redefinir los parámetros y tiempos que se desea mostrar en los reporte, los cuales se encuentran predefinidos en la BD.

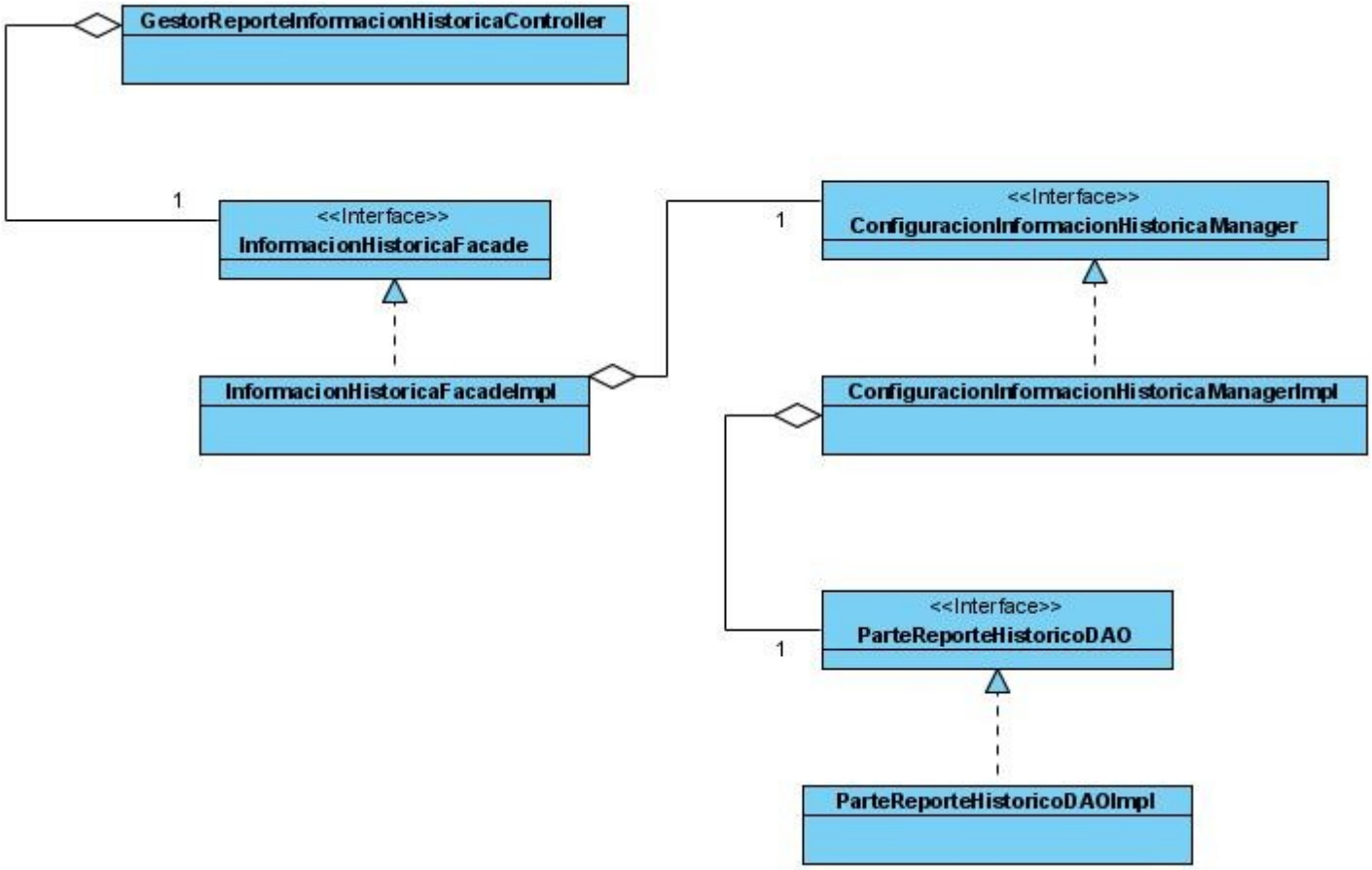


**Figure 5 Diagrama de clases del diseño CU Configurar Reporte**

### 3.5 Diagrama de clases del diseño CU Generar Reporte tabular

El CU Generar Reporte tabular es de todos el que maneja la mayor cantidad de clases debido a la estructura adoptada por el módulo.

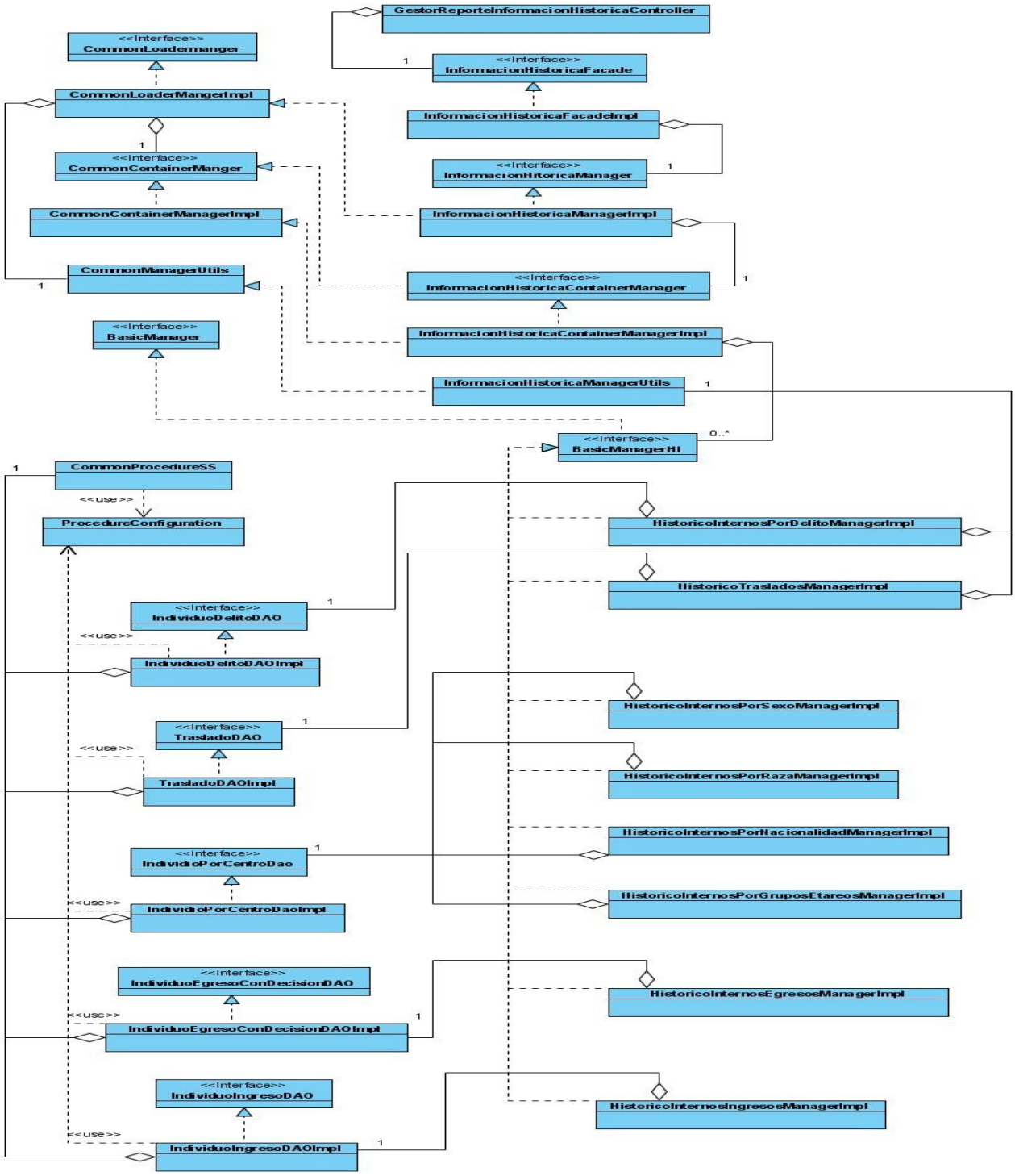

**Figure 6 Diagrama de clases del diseño CU Generar Reporte tabular**

## 3.6 Descripción de los paquetes y clases significativa

En esta sesión se describen los principales paquetes y clases utilizadas en los diagramas de clases anteriores.

| **Paquete: TablaReporte** |
|---|
| **Este es un paquete de utilidades cuyo objetivo es dar la capacidad de generar tablas para reportes.** |

**Tabla 13 Paquete Tabla Reporte**

| **Paquete: factoriaJfreeChart** |
|---|
| Este es un paquete de utilidades cuyo objetivo es dar la capacidad de generar gráficas para reportes. |

**Tabla 14 Paquete FactoriaJFreeChart**

| **Nombre : GestorReporteInformacionHistoricaController** |
|---|
| La clase GestorReporteInformacionHistoricaController es un controlador de Spring, cuya funcionalidad es escuchar las peticiones de generar reportes, obtener gráficas y tablas de reporte , configurar los reportes y formular una respuesta acorde, para esto cuenta con los siguientes métodos:<br>• buildTable()<br>• buildTable()<br>• buildReportName()<br>• configuracionIH()<br>• salvarConfiguracionInformacionHistorica() |

**Tabla 15 Clase GestorReporteInformacionHistoricaController**

| **Nombre : InformacionHistoricaFacadeImpl** |
| --- |
| La clase InformacionHistoricaFacadeImpl pertenece a la capa de lógica de negocio, su objetivo es brindar las funcionalidades que se implementan en esta capa en cuanto a: generar tablas y gráficos de reportes, a la capa de presentación, para ello cuenta con los siguientes métodos:<br>• obtenerGraficos()<br>• buildTable()<br>• salvarConfiguracionInformacionHistorica() |

**Tabla 16 Clase InformacionHistoricaFacadeImpl**

| **Nombre : BasicManager** |
| --- |
| La clase BasicManager pertenece a la capa de lógica de negocio, es la interfaz de una parte de los procesos necesarios para satisfacer a los casos de uso generar reporte grafico y generar reporte tabular. |

**Tabla 17 Clase BasicManager**

| **Nombre : InformacionHistoricaManagerUtils** |
| --- |
| La clase InformacionHistoricaManagerUtils pertenece a la capa de lógica de negocio su funcionalidad es brindar comodidad en el trabajo de los intervalos de fechas para sacar los reportes , dividiendo y reconcomiendo que tipo de intervalo es ,para esto presenta los métodos :<br>• getDates()<br>• getYears()<br>• getMonths()<br>• getWeeks()<br>• copyInto() |

**Tabla 18 Clase InformacionHistoricaManagerUtils**

| **Nombre : ConfiguracionInformacionHistoricaManagerImpl** |
| --- |
| La clase ConfiguracionInformacionHistoricaManagerImpl pertenece a la capa de lógica de negocio su funcionalidad es salvar la configuración que el usuario realiza, para esto presenta el método :<br>• salvarConfiguracion ( ) |

**Tabla 19 Clase ConfiguracionInformacionHistoricaManagerImpl**

| **Nombre : GraficoManagerImpl** |
| --- |
| La clase GraficoManagerImpl pertenece a la capa de lógica de negocio, esta clase presenta gran funcionalidad a la hora de trabajar con los reportes gráficos, para esto presenta los métodos :<br>• getFactoriaDatasetLineaHistorica()<br>• obtenerGrafico()<br>• obtenerGraficoHistorico() |

**Tabla 20 Clase GraficoManagerImpl**

### 3.7 Descripción de elementos dinámicos. Componentes construidos a fin de su reutilización.

A fin de reutilizar código JS que se utiliza en las vistas JSP encargadas de mostrar los reportes tabulares y gráficos se construyeron los widgets:

- **tablaHistorica.JS**:

  Este widget se encarga de la modelación de las tablas de Información Histórica, utilizando para esto la representación en JSON, dicha tabla es obtenida a través de una petición asíncrona.

- **graficaHistorica.JS**

  Este widget se encarga de la modelación de las gráficas de información histórica, utilizando para esto una representación en JSON de las direcciones físicas de las imágenes contenidas en las gráficas.

### 3.8 Diagrama de clases de diseño con extensiones web

Este Diagrama representa el desarrollo de las vistas que contendrá el módulo, así como su interacción con la página servidora, la cual se encarga de la funcionalidad, es decir, de responder a las peticiones que se realizan en cada una de las páginas. Contiene 5 páginas clientes incluyendo la página inicial, la cual contiene el menú de módulo, las otras son las referentes a la información tabular, información gráfica y configuración, esta última contiene archivos java script para su funcionalidad. La página servidora es la encargada de dar respuesta a todas las peticiones que se realicen desde el cliente, construyendo tanto la tabla como la gráfica del indicador seleccionado.

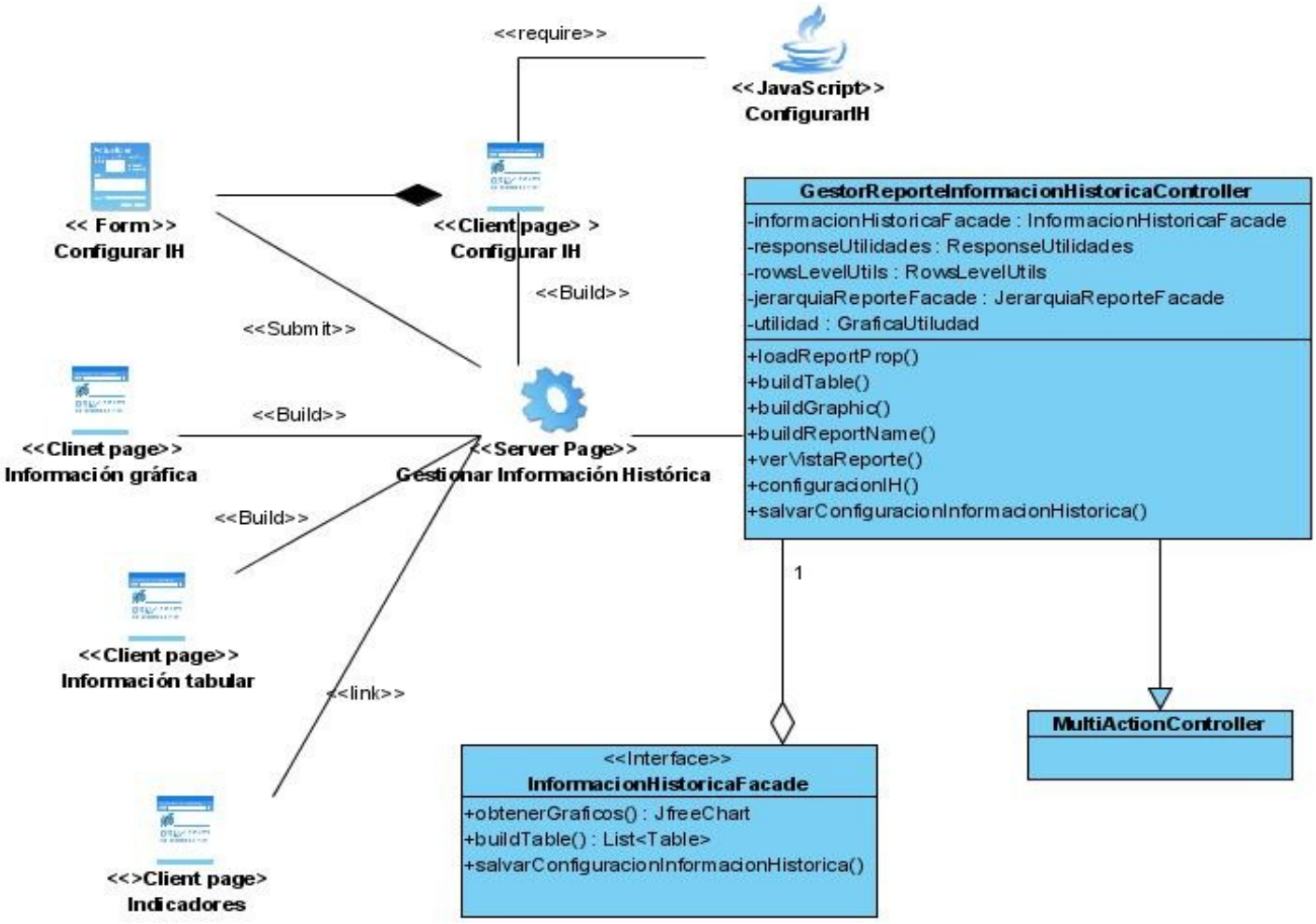


**Figure 7 Diagrama de Extensiones Web**

### 3.9 Modelo de datos

El modelo de datos que a continuación se muestra describe la estructura de las entidades relacionadas con Información Histórica y sus relaciones. En la descripción de las entidades se definen, el tipo de variables de los atributos, el objetivo de cada uno y el objetivo que encierra la entidad en sí.

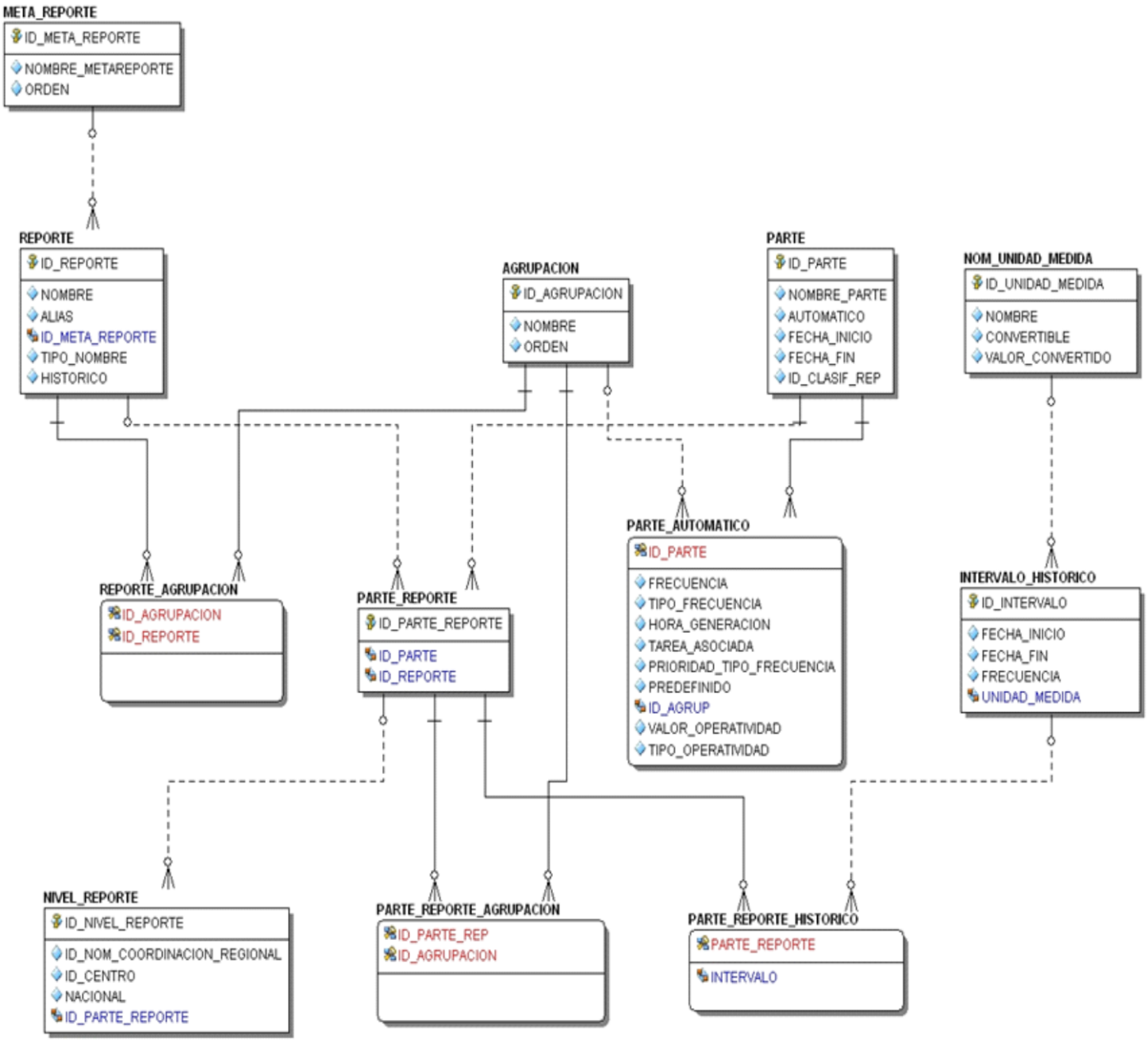


**Figure 8 Modelo de Datos de Información Histórica**

| **Entidad: REPORTE** | | |
|---|---|---|
| Esta entidad representa todos los reportes definidos para la sala situacional. De ellos se mostrará el nombre del reporte y el id que lo identifica. | | |
| Nombre | Tipo | Descripción |
| ID_REPORTE | varchar(20) | Este atributo es el identificador de cada reporte definido. |
| NOMBRE | varchar(200) | Este atributo representa el nombre de los reportes. |
| HISTORICO | NUMBER(1) | Este atributo es representado por 1 si es un reporte de tipo histórico, de lo contrario por 0 |

**Tabla 21 Entidad Reporte**

| **Entidad: AGRUPACION** | | |
|---|---|---|
| Esta entidad representa las diferentes agrupaciones que existen con los reportes definidos para la Sala Situacional. | | |
| Nombre | Tipo | Descripción |
| ID_AGRUPACION | varchar(20) | Este atributo representa la llave primaria de la entidad |
| NOMBRE | varchar(200) | Este atributo representa el nombre de la agrupación definida para cada reporte o tipo de reporte. |
| ORDEN | NUMBER(3) | Este atributo contiene los 3 valores posibles de cada agrupación |

**Tabla 22 Entidad Agrupación**

| **Entidad: META_REPORTE** | | |
|---|---|---|
| **Esta entidad representa una nueva agrupación que poseen los reportes definidos para la Sala Situacional.** | | |
| Nombre | Tipo | Descripción |
| ID_META_REPORTE | varchar(20) | Este atributo es la llave primaria de la entidad |
| NOMBRE_METAREPORTE | varchar(100) | Este atributo representa el nombre de cada Meta_Reporte definido. |
| ORDEN | NUMBER(3) | Define el orden de los Meta_Reporte |

**Tabla 23 Entidad Nueva Agrupación**

| **Entidad: PARTE** | | |
|---|---|---|
| En esta entidad se van a almacenar todos los partes que sean configurados, permitiendo las funcionalidades básicas necesarias para operar sobre los mismos (inserta, eliminar, actualizar), de estos partes se mostrará el Id que lo identifica, nombre, si es automático o no, y las fecha de inicio y fin. Esta entidad posee una relación de uno a muchos con la entidad Parte_Reporte. | | |
| Nombre | Tipo | Descripción |
| ID_PARTE | varchar(20) | En este campo se va a almacenar el identificador de cada parte, posee un valor único dentro de la entidad y constituye la llave primaria de la misma. |
| NOMBRE_PARTE | varchar(200) | En este atributo se almacena el nombre de cada parte. |
| AUTOMATICO | bit(1) | Este es una atributo booleano que especifica si el parte es automático (true / 1) o si no es automático (false / 0). |

| | | |
|---|---|---|
| FECHA_INICIO | date(0) | Este atributo junto con el atributo FECHA_FIN forman el intervalo de tiempo que se tendrá en cuenta a la hora de generar el parte. |
| FECHA_FIN | date(0) | Este atributo junto con el atributo FECHA_INICIO forman el intervalo de tiempo que se tendrá en cuenta a la hora de generar el parte. |
| ID_CLASIF_REP | VARCHAR2(10) | Este atributo dice si es reporte de información operativa, aviso, alerta o información histórica |

**Tabla 24 Entidad Parte**

| **Entidad: PARTE_REPORTE** | | |
|---|---|---|
| En esta entidad se almacena la configuración correspondiente a cada uno de los reportes que se incluyen dentro del parte que se esté configurando, además constituye la relación muchos a muchos que existe entre la entidad Parte y la entidad Reporte, posee una relación de uno a muchos con la entidad Nivel_Reporte. | | |
| Nombre | Tipo | Descripción |
| ID_PARTE | varchar(20) | En este atributo se almacena el id del parte de la relación, por lo que constituye la llave foránea con la entidad Parte. |
| ID_REPORTE | varchar(20) | En este atributo se almacena el id del reporte de la relación, por lo que constituye llave foránea con la entidad Reporte |

| | | |
|---|---|---|
| ID_PARTE_REPORTE | varchar(20) | En este atributo se almacena un identificador para la relación parte-reporte, constituye la llave primaria de la entidad. |

**Tabla 25 Entidad Parte Reporte**

| **Entidad: INTERVALO_HISTORICO** | | |
|---|---|---|
| En esta entidad se almacena los diferentes intervalos en los cuáles se evaluará la información histórica | | |
| Nombre | Tipo | Descripción |
| ID_INTERVALO | varchar(20) | En este atributo se almacena el id del parte de la relación, por lo que constituye la llave foránea con la entidad Parte. |
| FECHA_INICIO | DATE | En este atributo se almacena la fecha inicial del intervalo |
| FECHA_FIN | DATE | En este atributo se almacena la fecha final del intervalo |
| UNIDAD_MEDIDA | varchar(20) | En este atributo se almacena la unidad del intervalo, es decir si es anual, mensual, semanal o diario. |

**Tabla 26 Entidad Intervalo Histórico**

| **Entidad: NIVEL_REPORTE** | | |
|---|---|---|
| En esta entidad se almacenan todos los niveles (Nacional, Regiones y Centros) que estarán visibles en cada una de las configuraciones de los reportes que aparecen en la entidad Parte_Reporte. | | |
| Nombre | Tipo | Descripción |
| ID_NIVEL_REPORTE | varchar(20) | En este atributo se almacena un identificador para cada uno de los niveles configurados, constituye la llave primaria en esta entidad. |
| ID_NOM_COORDINACION_REGIONAL | varchar(20) | En este atributo se va a almacenar el identificador de la región en caso de que exista en la configuración, constituye una llave foránea con la entidad NOM_COORDINACION_REGIONAL. |
| ID_CENTRO | varchar(20) | En este atributo se almacena el identificador del centro en caso de que exista en la configuración, constituye una llave foránea con la entidad Centro |
| NACIONAL | bit(1) | Este atributo booleano especifica si el nivel Nacional fue configurado (true / 1) o no (false / 0) para este reporte. |
| ID_PARTE_REPORTE | varchar(20) | En este atributo se almacena el identificador de la configuración de reporte a la que pertenece cada uno de los niveles que se configuran, constituye la llave |

| | | |
|---|---|---|
| | | foránea con la entidad Partre_Reporte. |

**Tabla 27 Entidad Nivel Reporte**

| **Entidad: PARTE_REPORTE_HISTORICO** | | |
|---|---|---|
| En esta entidad se almacena el INTERVLO_HISORICO que se le asigna a cada PARTE_REPORTE | | |
| Nombre | Tipo | Descripción |
| PARTE_REPORTE | varchar(20) | En este atributo se almacena el PARTE_REPORTE al cuál se le asignará el intervalo |
| INTERVALO | varchar(20) | En este atributo se almacena el intervalo asignado |

**Tabla 28 Entidad Parte Reporte Histórico**

## CONCLUSIONES

Con el resultado del trabajo realizado se le dio cumplimiento a los objetivos inicialmente planteados: Automatizar una herramienta que muestre la información histórico del Sistema Penitenciario Venezolano, que servirá de base para la toma de decisiones de índole tácticas y estratégicas en la Dirección General de Servicios Penitenciarios (DGSP), tomando como fuente informativa la base de datos centralizada que mantiene actualizada el Sistema de Gestión Penitenciaria (SIGEP).

Sala Situacional, constituye un importante módulo dentro del Sistema de Gestión Penitenciaria que facilita el control de las políticas establecidas en el Sistema Penitenciario de Venezuela. También, es válido destacar que Información Histórica provee al sistema de una herramienta para medir el comportamiento de determinados indicadores y su evolución en el tiempo.

Se implementó el módulo correspondiente a Información Histórica, mostrando todos los reportes relacionados con población penal, ingreso, egreso, coordinación de movimientos, visitas, entre otros.

## RECOMENDACIONES

1. Debido a la amplitud de las funcionalidades de Sala Situacional se recomienda hacer extensiva la experiencia a otros proyectos productivos que generan reportes en sus aplicaciones.
2. Mantener la retroalimentación de la información mostrada en Información Histórica.
3. Implementar los módulos de Alertas y Avisos correspondientes a Información Histórica.

## BIBLIOGRAFÍA

[1] VISION, G. C. *Sala Situacional*, 2005.

[2] ARIAS, O. Y. A. *"Proyecto Técnico de Asesoría Especializada, Colaboración Médica Odontológica, Comunicación Institucional y Solución Tecnológica para apoyar la modernización del Sistema Penitenciario de la República Bolivariana de Venezuela"*, 2006.

[3] WIKIPEDIA. *"The Free Enciclopedia"*, 2006.

[4] HIBERNATE, G. *Hibernate Reference Documentation* 2006.

ANÓNIMO. *Sala Situacional*, 2004.

JACOBSON, I.; G. BOOCH*, et al. El proceso unificado de desarrollo de software*. 2000. p. 84-7829-036

BAUER, C. and G. KING. *Hibernate In Action*, 2005.

CORRADI, P. and G. ELEICEGUI. *Tableros de comando*, 2002.

DAVISON, D.; S. DEVIJVER*, et al. Expert Spring MVC and Web Flow*, 2006.

GILBERT, D. *The JFreeChart Class Library*, 2004.

*Jasper Report*. 2007. [Disponible en: http://jasperforge.org/sf/projects/jasperreports

JOHNSON, R. *Expert One on one J2EE Development Without EJB*, 2004.

LARMAN, C. *UML y Patrones*. 1999. p. LÓPEZ, C.

PRESSMAN, R. S. *"Ingeniería del Software. Un Enfoque Práctico".* 1998. p.

RUP. *"Rational Unified Process".* 2003.

WALLS, C. and R. BREIDENBACH. *Spring in Action*, 2005.

**ANEXO A**

**Especificación del caso de uso: Generar Reporte Información Histórica**

| **Caso de Uso del sistema** | **Generar Reporte Información Histórica** |
|---|---|
| **Actores** | Funcionario de Sala Situacional |
| **Propósito** | Generar Reporte Información Histórica determinados en el formato seleccionado por el Funcionario de la Sala Situacional, tanto en Adobe Reader (fichero pdf), ó en HTML. |
| **Resumen:**<br>Este caso de uso se inicia cuando el Funcionario de la Sala Situacional desee generar reportes de información histórica, entrando en el subsistema de Sala Situacional. Luego de seleccionar una de las opciones, el sistema buscará las configuraciones correspondientes según lo seleccionado y la muestra. | |
| Precondiciones: | El usuario debe estar autentificado<br>Debe existir al menos un parte configurado |
| **Garantías (Poscondiciones)** | |
| Mínimas: | Se debe notificar un mensaje en caso de no poder generar el reporte |
| De éxito: | Se genera el reporte tabular y el reporte gráfico |
| Acción que inicia el caso de uso: | El usuario desea generar reporte |

| **Acción del actor** | **Respuesta del sistema** |
|---|---|
| 1) El Funcionario de Sala Situacional selecciona generar reporte de información histórica.<br>. | 2) El Sistema muestra las opciones de los posibles partes a generar:<br>a) Internos por Delito<br>b) Traslados<br>c) Internos por Sexo<br>d) Internos por Raza<br>e) Internos Indocumentados<br>f) Internos por Régimen<br>g) Internos por Nacionalidad<br>h) Internos por Grupos Etáreos<br>i) Internos Egreso<br>j) Internos Ingreso<br>k) Internos Procesados Penados<br>l) Internos por Devoluciones<br>m) Internos Progresividad<br>n) Internos Salidas Transitorias<br>o) Internos Visitas Familiares |

| | |
|---|---|
| 3) El Funcionario de Sala Situacional selecciona el reporte que desea mostrar. | 4) El sistema permite mostrar las posibles acciones:<br>a) Reporte tabular Ver CUI Generar Reporte Tabular<br>b) Reporte gráfico Ver CUI Generar Reporte Gráfico<br>c) Configurar reporte Ver CUE Configurar Reportes |
| 5) El Funcionario de Sala Situacional selecciona el tipo de reporte que desea consultar y culmina el caso de uso. | 6) El sistema muestra el reporte correspondiente.<br>a) Muestra el reporte correctamente<br>7) Finaliza el caso de uso. |

| **FLUJO ALTERNO** | |
|---|---|
| **Acción del actor** | **Respuesta del sistema** |
| | 6a.El sistema muestra un error en el |

| | |
|---|---|
| | reporte seleccionado |

**Especificación del caso de uso: Generar Reporte Tabular**

| **Caso de Uso del sistema** | **Generar Reporte Tabular** |
|---|---|
| **Actores** | Funcionario de Sala Situacional |
| **Propósito** | Generar reportes determinados en el formato seleccionado por el Funcionario de la Sala Situacional, tanto en Adobe Reader (fichero pdf), ó en HTML. |
| **Resumen:** Este caso de uso se inicia cuando el Funcionario de la Sala Situacional desee generar reportes de un parte. Luego de seleccionar una de las opciones, el sistema buscará las configuraciones correspondientes según lo seleccionado y la muestra. | |
| Precondiciones: | El usuario debe estar autentificado<br>Debe existir al menos un parte configurado |
| **Garantías (Poscondiciones)** | |
| Mínimas: | Se debe notificar un mensaje en caso de no poder generar |

| | |
|---|---|
| De éxito: | Se genera el reporte tabular y el reporte gráfico |
| Acción que inicia el caso de uso: | El usuario desea generar reporte |

| **Acción del actor** | **Respuesta del sistema** |
|---|---|
| 1) El Funcionario de Sala Situacional selecciona generar reporte de información histórica. . | |
| | 2) El Sistema busca la configuración del reporte a generar. |
| | 3) El sistema muestra el reporte tabular del parte seleccionado. |
| 4) El Funcionario decide imprimir el reporte generado. | |
| | 5) El sistema muestra opciones para impresión |

| | |
|---|---|
| 6) El funcionario oprime la opción de imprimir y culmina el caso de uso | 7) El sistema imprime el reporte y lo muestra en formato pdf.<br>a) Imprime correctamente. |

| **FLUJO ALTERNO** | |
|---|---|
| **Acción del actor** | **Respuesta del sistema** |
| | 7 a. El sistema muestra un error al imprimir y muestra nuevamente la opción de imprimir |

**Especificación del caso de uso: Generar Reporte Gráfico**

| **Caso de Uso del sistema** | **Generar Reporte Gráfico** |
|---|---|
| **Actores** | Funcionario de Sala Situacional |
| **Propósito** | Generar reportes gráficos determinados en el formato seleccionado por el Funcionario de la Sala Situacional, gráficas de líneas en este caso |

| | |
|---|---|
| **Resumen:**<br><br>Este caso de uso se inicia cuando el Funcionario de la Sala Situacional desee generar reportes gráficos, generando así la información gráfica de dicho reporte. | |
| Precondiciones: | El usuario debe estar autentificado<br>Debe existir al menos un parte configurado |
| **Garantías (Poscondiciones)** | |
| Mínimas: | Se debe notificar un mensaje en caso de no poder generar |
| De éxito: | Se genera la gráfica |
| Acción que inicia el caso de uso: | El usuario desea generar reporte gráfico |
| **Acción del actor** | **Respuesta del sistema** |

| | |
|---|---|
| 1- El Funcionario de la Sala Situacional selecciona generar reporte de un parte, para así ver los reportes gráficos de dicho parte.<br>. | |
| | 2) El Sistema busca la configuración del reporte a generar. |
| | 3) El Sistema genera el reporte del parte en el formato indicado |
| 4) El Funcionario decide imprimir el reporte generado. | |
| | 5) El sistema muestra opciones para impresión |
| | 6) El sistema imprime el reporte y lo muestra en formato pdf.<br>a) Imprime correctamente. |
| 7) El funcionario oprime la opción de imprimir y culmina el caso de uso. | |

| **FLUJO ALTERNO** | |
|---|---|
| **Acción del actor** | **Respuesta del sistema** |
| | 6a. El sistema muestra un error al imprimir y muestra nuevamente la opción de imprimir |

**Especificación del caso de uso: Configurar Reporte Histórico**

| **Caso de Uso del sistema** | **Generar Reporte Gráfico** |
|---|---|
| **Actores** | Funcionario de Sala Situacional |
| **Propósito** | Configurar los reportes de Información histórica para obtener la información conveniente, determinados en el formato seleccionado por el Funcionario de la Sala Situacional ya sean tablas ó gráficas de líneas. |
| **Resumen:**<br><br>Este caso de uso se inicia cuando el Funcionario de la Sala Situacional desee configurar un reporte de Información Histórica. | |

| | |
|---|---|
| Precondiciones: | El usuario debe estar autentificado<br>Debe existir al menos un parte configurado |
| **Garantías (Poscondiciones)** | |
| Mínimas: | Se debe notificar un mensaje en caso de no poder generar el reporte |
| De éxito: | Se configura la Información |
| Acción que inicia el caso de uso: | El usuario desea configurar reporte histórico |

| **Acción del actor** | **Respuesta sistema** |
|---|---|
| 1- El Funcionario de la Sala Situacional configura el parte de Información Histórica, reajustando tanto el intervalo de fechas, la frecuencia y el tipo de intervalo (anual, mensual, semanal, diario).<br>. | 2- Guarda la configuración modificada.<br>a) Se guarda correctamente |

| | |
|---|---|
| 3- El funcionario selecciona un reporte | |
| | 4- Sistema genera el reporte del parte en el formato indicado ,y a su vez la gráfica, de dicho reporte modificado |
| 5- Culmina el caso de uso | |

| **FLUJO ALTERNO** | |
|---|---|
| **Acción del actor** | **Respuesta del sistema** |
| | 2a. El sistema muestra un error al guardar la configuración. |

**GLOSARIO**

**SIGEP:** Sistema de Gestión Penitenciaria.

**DGSP:** Dirección General de Servicios Penitenciarios.

**CTC:** Centro de Tratamiento Comunitario.

**UTASP:** Unidades Técnicas de Apoyo al Sistema Penitenciario.

**CU:** Caso de Uso.

**API:** Interfaz de Programación de Aplicaciones.

**JDBC:** Base De Datos en Java.

**RMI:** Invocación de Métodos Remotos.

**JMS:** Servicio de Mensajería en Java.

**XML:** Lenguaje de Marcas Extensible.

**POO:** Programación Orientada a Objetos.

**RDBMS:** Relational Date Base Management System.

**UML:** Lenguaje Unificado de Modelado.

**HQL:** Hibernate Query Language.

**IDE:** Ambiente de Desarrollo Integrado.