Universidad de las Ciencias Informáticas

Facultad 2

Título: Diseño e Implementación del módulo Egreso del SIDEP.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas.

Autores: Ricardo Cárdenas Del Risco

Reinier Álamo Quesada

Tutor: Ing. Alejandro Santana Coterón

Co-Tutora: Ing. Yanay Viera Lorenzo

El pueblo más feliz es el que tenga mejor educados a sus hijos, en la instrucción del pensamiento, y en la dirección de los sentimientos.

José Martí

Declaración de Autoría

Declaro ser autor de la presente tesis y concedo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2012.

Reinier Álamo Quesada

_____

Firma del Autor

Ricardo Cárdenas Del Risco

_____

Firma del Autor

Ing. Alejandro Santana Coterón

_____

Firma del Tutor

Agradecimientos

Compartido

A nuestro tutor y co-tutora, Alejandro y Yanay por apoyarnos en todo momento y de poder contar con ellos hasta el final. También a nuestra líder y arquitecto de proyecto, Yanet y Maikel por prestarnos su valioso tiempo y paciencia. Y por último pero no por eso menos importante, a Leandro Roura por ser más que un compañero de proyecto, el cual nos ayudó incondicionalmente en todos los aspectos de este trabajo.

A todos sinceramente, muchas gracias.

De Ricardo Cárdenas Del Risco

Primeramente agradecer a una de las personas más importante en mi vida, mi madre que sin ella no hubiera sido posible hacer realidad este sueño, por dedicar su vida a guiarme por el buen camino, por haber estado a mi lado en los momentos difíciles y estar consciente de que podía lograrlo.

A mi padre y a Reina por todo su apoyo, confianza y paciencia. Por haberme sacado una sonrisa en los momentos donde me inundaba la preocupación y la tristeza.

Agradecimientos

A mi novia Adriana, que en estos cinco años a su lado me han iluminado la vida. Por ser más que una novia y compañera, por ser una gran amiga que ha sabido guardar mis secretos y junto a la cual he pasado uno de mis mejores años de mi vida.

A mis abuelos Luis, Ofelia, Agapito y Yolanda, porque siempre soñaron con verme graduado y me han demostrado todo el cariño del mundo.

A mis tías, Margara, Adela y Mirza, que siempre me demostraron que podía contar con ellas, y me quieren como si fuera un hijo más.

A mis primos, Jarol, Yusley, Li, Jonatan, José Carlos y María que más que primos son hermanos ya que me han brindado de lo poco que han tenido.

A mis amigos de la infancia Alexeis Agüero y Alexeis Morales que aunque estábamos lejos siempre estuvieron al tanto de mí.

A mis compañeros y profesores que de cierta forma han contribuido a mi formación tanto profesional como personal.

Ricardo

Agradecimientos

De Reinier Álamo Quesada

Primeramente quisiera agradecer a la Revolución por crear esta universidad, Agradecer a mis padres Ana Isabel y Cristóbal por ser el motor de mi inspiración durante estos 5 años.

A mi hermana Vilma a la que siempre estoy molestando.

A mis abuelos Luis, Alicia, Morales y Kika por su eterna preocupación por mis estudios.

A mis tías Maricela, Madeline, Ileana y Gemary por su inagotable cariño.

A mis tíos Jorge, Julio, Iván, Omar, Eduardo, Alberto por su apoyo durante estos cinco años.

A todos mis primos y primas, especialmente a Eduardito, Ivancito y Luisito por todos los momentos que pasamos juntos.

A mis amigos de la infancia Yhoandy, Juan Miguel y Alejandro por estar siempre apoyándome.

A mis amigos del politécnico Yosbel, Leonardo, Luis Alberto, Chermito, Yasmany, Ramiro y muchísimos más que no menciono por cuestión de espacio por los buenos y malos momentos vividos.

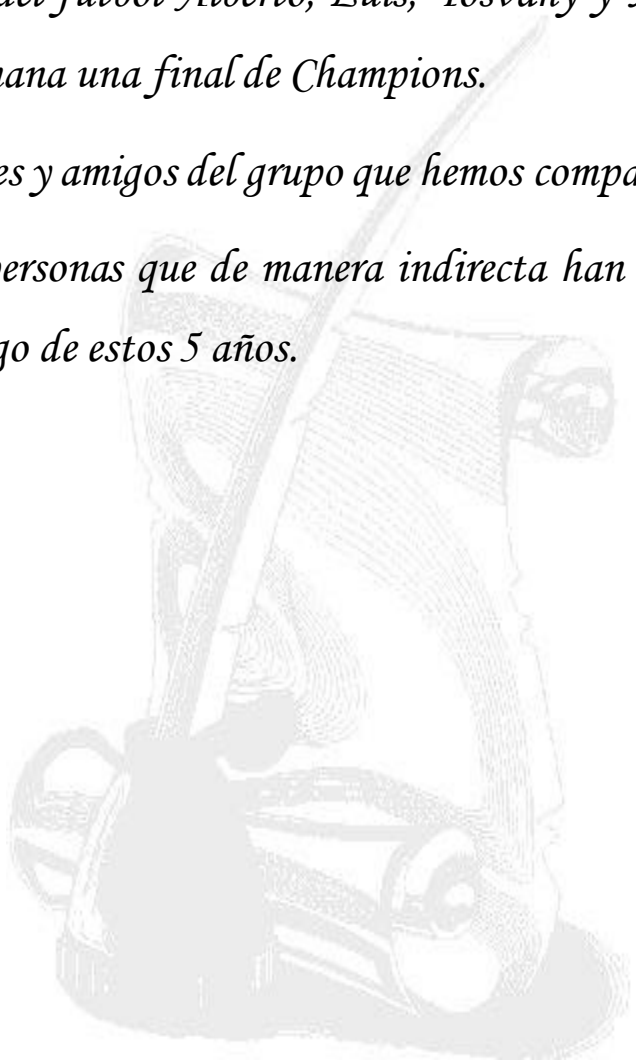
Agradecimientos

A los amigos de mi padre Reyes y Luis Orlando que fueron los que me motivaron a estudiar informática hace 12 años atrás.

A mis amigos del fútbol Alberto, Luis, Yosvany y Michelin y otros por hacer de cada fin de semana una final de Champions.

A mis profesores y amigos del grupo que hemos compartido estos cinco años.

A todas esas personas que de manera indirecta han influido positivamente en mi carrera a lo largo de estos 5 años.



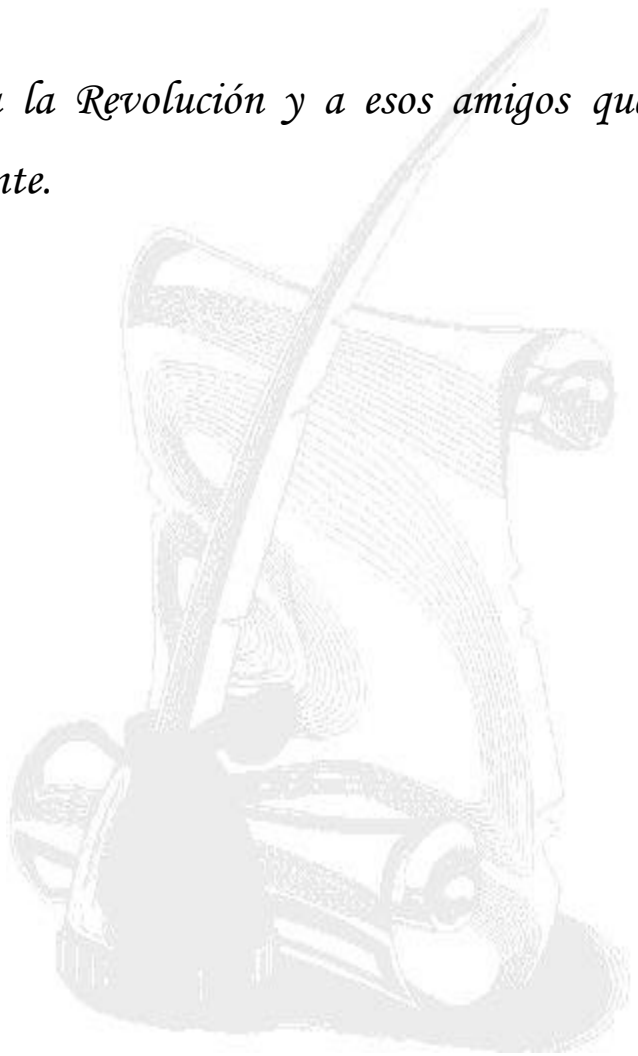
Dedicatoria

A mis padres y a la Revolución, porque son mi espíritu y fuerza.

Ricardo

A mis padres, a la Revolución y a esos amigos que siempre me han apoyado incondicionalmente.

Reinier



Resumen

El país se encuentra inmerso en el proceso de informatización de la sociedad, el cual incluye también a las instituciones del gobierno. Actualmente en el Sistema Penitenciario Nacional se hace uso del Sistema Automatizado de Control del Recluso (SACORE) para la gestión de la información penitenciaria referente a los individuos que se encuentran cumpliendo sanción. Este sistema ha sido perfeccionado por solicitud de los usuarios durante sus 9 años de explotación, pero a pesar de esto continúa presentando dificultades ya que no se encuentran informatizados procesos priorizados entre los que se encuentra el proceso de egreso. Esta situación genera una serie de problemas en la gestión de información referente al proceso de egreso, además de un trabajo complejo.

En el actual trabajo se realiza el diseño e implementación del módulo Egreso del SIDEP (Sistema Informativo de la Dirección de Establecimientos Penitenciarios). Cumpliendo durante el desarrollo las tareas y objetivos planteados en el análisis de los requerimientos de software previamente contratados con el cliente y haciendo uso de las herramientas y tecnologías de software establecidas para el proyecto.

Índice

Índice

Introducción

El Sistema Penitenciario Cubano a través de la historia ha ido cambiando sus métodos y procedimientos en la gestión y trato de los internos. Antes de 1959 estuvo caracterizado por el maltrato y el abuso a los prisioneros, pero luego del triunfo revolucionario esta situación empezó a cambiar. El gobierno revolucionario comenzó aprobar políticas y leyes para revertir la situación heredada de la tiranía de Batista, creando nuevas formas para enfrentar el delito en correspondencia con su gravedad y haciendo un reordenamiento jurídico.

En el año 1989 se dan los primeros pasos en la automatización de la gestión de datos legales sobre individuos, creando un software para el registro y control de la legalidad de los acusados, sancionados y asegurados. Este software se fue perfeccionando a través de los años hasta que en el 2002 surge el Sistema Automatizado para Control de Reclusos (SACORE), el cual brinda ciertas ventajas en la gestión y uso de la información pero constituye una solución parcial porque no integra todos los procesos legales por los que transita un interno. Se desarrollaron otros dos sistemas automatizados que complementan la información del SACORE: El Sistema Automatizado de Incidencias de la Dirección de Establecimientos Penitenciarios (SAIDEP) y el Sistema Administración de Capacidades de la Dirección de Establecimientos Penitenciarios (SACDEP).

A la Universidad de Ciencias Informáticas se le ha asignado la tarea de desarrollar el Sistema Informativo de la Dirección de Establecimientos Penitenciarios (SIDEP) con el objetivo de elevar el nivel de automatización de los procesos legales de los internos. Este sistema está compuesto por varios subsistemas entre los que se encuentra Registro Legal el cual se encarga de administrar los aspectos relacionados con la situación legal de internos y el control del cumplimiento de la sanción o medida de seguridad reeducativa de internamiento durante su tránsito por el sistema penitenciario. Este subsistema a su vez contiene siete módulos, en el cual uno de ellos es Egreso. Este módulo tiene como objetivo registrar el egreso de los internos de un centro penitenciario. El egreso sólo puede otorgársele a un interno si existen decisiones judiciales que los avalan, excepto en el caso de bajas definidas por el sistema penitenciario. El otorgamiento de los egresos influye sobre la legalidad de los internos en el Sistema Penitenciario. Actualmente el Sistema Penitenciario Cubano presenta varios problemas a la hora de realizar un proceso de egreso a un interno, ya sea este un acusado, sancionado o un asegurado:

- Existen dificultades y demoras para realizar los procesos y trámites durante el egreso de los internos.

- Puede otorgarse un egreso sin tener en cuenta los beneficios y el motivo de ingreso del interno.
- Al realizar un egreso no se tiene en cuenta la logística existente que ha dejado de usarse.
- Actualmente SACORE maneja de forma conjunta los egresos y los traslados de un centro a otro, pero son dos procesos con objetivos diferentes.

Por lo anteriormente planteado surge como **problema** de la investigación: ¿Cómo mejorar el otorgamiento de los egresos en el Sistema Penitenciario Cubano?

Como **objeto de estudio** se ha definido los procesos de egreso de los Sistemas Penitenciarios, planteándose como **objetivo general** diseñar e implementar el módulo Egreso del SIDEP. El **campo de acción** quedaría enfocado en los procesos de egreso en el Sistema Penitenciario Cubano.

Además se proponen los siguientes **objetivos específicos**:

1. Elaborar el marco teórico de la investigación.
2. Realizar el modelo de diseño del módulo.
3. Implementar la propuesta de solución de software.
4. Garantizar la calidad y el funcionamiento de la propuesta de solución.

Para el cumplimiento de los objetivos específicos y dar respuesta a los cuestionamientos planteados se han trazado las siguientes **tareas de investigación**:

1. Análisis de las soluciones nacionales e internacionales existentes para el estudio de los procesos de egreso.
2. Descripción de las herramientas y tecnologías para dar solución al problema.
3. Realización de los diagramas de clases para el módulo Egreso.
4. Realización de los diagramas de interacción para el módulo Egreso.
5. Diseño de los casos de prueba para el módulo Egreso del SIDEP.
6. Realización del diagrama entidad-relación de la base de datos para el módulo de Egreso.
7. Realización el diagrama de componentes para el módulo Egreso del SIDEP.
8. Implementación del módulo Egreso del SIDEP.
9. Realización de las pruebas de caja negra del módulo Egreso del SIDEP para garantizar la calidad del producto.

De esta manera, la **idea a defender** con la presente investigación, es que con el diseño e implementación del módulo de Egreso se mejorará el otorgamiento de los egresos en el Sistema Penitenciario Cubano. Para ello los **métodos de investigación científica** empleados son:

Métodos Teóricos:

- **Histórico-Lógico**: Se utiliza este método porque fue necesario analizar las tecnologías, herramientas y aplicaciones de gestión de información vinculadas a los Centros Penitenciarios, conociendo de esta forma la evolución que ha tenido, para así poder dar solución a las necesidades en el Sistema Informativo de la Dirección de Establecimientos Penitenciarios.
- **Analítico-sintético**: Se emplea con el objetivo de realizar la síntesis de los elementos fundamentales a abordar en el documento.
- **Modelación**: Se utiliza este método ya que se realizó el diseño de una aplicación Web y gracias a este modelado se puede entender la lógica del proceso a automatizar.

Métodos Empíricos:

- **Observación:** Se emplea para identificar algunas características en el proceso de gestión de egreso como la forma de realización, quienes intervienen, que utilizan, entre otras.

- **Entrevista:** Se realizaron entrevistas a los analistas del proyecto con el propósito de entender los procesos que se llevan a cabo en el módulo de Egreso en el SIDEP.

El documento de tesis se estructura de la siguiente forma:

Capítulo 1: Fundamentación Teórica

Aborda conceptos relacionados con los centros penitenciarios y el proceso de egreso. Se analizan las principales tecnologías y herramientas de desarrollo a utilizar en el diseño e implementación del módulo Egreso del sistema penitenciario cubano.

Capítulo 2: Diseño del sistema

Se definen las actividades que darán cumplimiento a las tareas propuestas y se describe la arquitectura usada para dar solución al problema. También se presentan los diagramas de clases e interacción.

Capítulo 3: Implementación y Prueba

Se desarrollan las funcionalidades propuestas y se termina realizando los casos de prueba para evaluar el cumplimiento de las funcionalidades.

Capítulo 1: Fundamentación Teórica

1.1. Introducción

En el presente capítulo se abordan temas relacionados con los principales conceptos y términos usados en la investigación, enfatizando los antecedentes de la necesidad de la creación de sistemas informáticos para que gestionen la información proveniente de los centros penitenciarios del país y extranjeros. Se definen conceptos asociados al problema a resolver que ayudarán a entender mejor el mismo y se analizaran las tecnologías, herramientas y metodología que serán usadas en la investigación.

1.2. Conceptos asociados al problema

Sistema penitenciario: se define como el conjunto de normas, procedimientos y dependencias dispuestas por el Estado para la ejecución del régimen penitenciario entre los que se encuentran además los principios, programas, recursos humanos, dependencias e infraestructura que se encuentran relacionadas y destinadas a este régimen. (1)

Egreso: Es la operación de liberación de un interno por cumplimiento de su condena u otros motivos. El egreso sólo puede registrarse si existen decisiones judiciales que los avalan, excepto en el caso de bajas definidas por el sistema penitenciario. En este último, para registrar el egreso definitivo deben registrarse los documentos que confirman la defunción del interno. (1)

1.3. Sistemas penitenciarios y soluciones informáticas

Cada gobierno define la estructura de su sistema penitenciario de acuerdo a la legislación y las condiciones reales que posee. Desde el propio triunfo revolucionario se comenzó un proceso de transformaciones que contribuyeron al mejoramiento de la condición humana y conducta social a los privados de libertad. Esto no solo ha ocurrido en Cuba sino también en otros países, los cuales han tenido la necesidad de modernizar sus principales sistemas informáticos para tratar de dejar atrás el precario funcionamiento de las instituciones y la estandarización de los procesos penitenciarios, característica que posee actualmente muchos de los centros penitenciarios de América Latina.

A continuación se abordan características de algunos de los sistemas informáticos implantados en centros penitenciarios en el continente, cuyo análisis y estudio ayudó al desarrollo de este trabajo.

1.3.1. Sistema de Gestión Penitenciaria en Venezuela (SIGEP)

El sistema nació de un acuerdo de amistad entre los países de Venezuela y Cuba con el objetivo de lograr un sistema informático centralizado que estandarice e implemente los procesos fundamentales en el sistema penitenciario venezolano como también tenga en cuenta el respeto a los derechos de los individuos, su actividad de rehabilitación y reinserción a la sociedad. En la actualidad SIGEP se encarga de toda la información relacionada con los internos, así como presentación ante tribunales, el expediente médico, los movimientos (salidas, traslados y visitas), entre otros.

El egreso puede ser definitivo si implica un egreso del sistema penitenciario (cumplimiento de pena, defunción) o relativo para el caso de los movimientos internos dentro del sistema penitenciario (traslados interpenales u otorgamientos de fórmulas alternativas de cumplimiento de pena o beneficios). El egreso sólo puede registrarse si existen decisiones que lo avalen, excepto en el caso de la evasión o fuga y la defunción. En este último, para registrar el egreso definitivo deben registrarse los documentos que confirman la defunción del individuo. El sistema es capaz de identificar a los individuos que les corresponde egreso en una fecha determinada a partir de la existencia de decisiones registradas en su expediente penitenciario que así lo indiquen. Existen tres tipos de Egreso en Venezuela:

Decisión Judicial: En el módulo Decisiones los funcionarios de Control Penal pueden registrar las decisiones de los tribunales y de la Dirección Nacional de Servicios Penitenciarios (DNSP), a su vez se pueden registrar decisiones en cada una de las fases de un proceso legal, estas decisiones pueden implicar el egreso del individuo del establecimiento y en algunas casos, el cierre del expediente.

Evasión o Fuga: Se considera un egreso por evasión cuando un individuo evade la custodia de la DNSP sin el uso de fuerza o violencia. El egreso por fuga se refiere a la evasión mediante el uso de fuerza o violencia. En el sistema se procede de la misma forma para ambos tipos de egreso.

Fallecimiento: Cuando fallece un individuo que se encontraba bajo la custodia de la DNSP, se debe registrar su egreso en el sistema. Se considera un egreso por defunción cuando se ha registrado como motivo de egreso “defunción” y se han registrado además los documentos legales que confirman el hecho. Si se indica como motivo de egreso “defunción” pero no se registran los documentos correspondientes se considera una presunta defunción. Al registrar el egreso de un individuo por defunción o por presunta defunción, el expediente del mismo queda inhabilitado.

1.3.2. Sistema Automatizado para el Control del Recluso (SACORE) Cuba

En el año 1989 se empieza a automatizar los procesos principales vinculados al interno y algunos aspectos del control penal, este sistema se denomina SACORE .Este garantizaba una respuesta rápida a las solicitudes de información de los diferentes órganos e instituciones del estado (Jefatura del Ministerio del Interior, Ministerio de Justicia, Ministerio de las Fuerzas Armadas, Tribunales, Fiscalías), recogía la mayor información posible de los internos en todas las especialidades existentes, permitía la recuperación dinámica a partir de una solicitud de búsqueda y realizaba el traslado automáticamente de todos los datos del interno al nivel nacional.

El sistema penitenciario cubano actualmente hace uso de SACORE el cual tiene sus ventajas y deficiencias. Entre las ventajas se poden encontrar:

- Reduce el trabajo con materiales físicos (papeles, documentos y otros).
- Automatiza muchos procesos por los que transita el interno.
- Permite gestionar la información a nivel nacional a través de la red.

Con el paso de los años surgen nuevos procedimientos y normas que son necesarias automatizar. A pesar que el sistema SACORE se ha actualizado no se ha logrado automatizar eficientemente muchas funcionalidades entre las que se encuentra el módulo de Egreso, presentando el mismo las siguientes deficiencias:

- Puede otorgarse un egreso sin tener en cuenta el motivo de ingreso.
- Al realizar un egreso entre centros penitenciarios gran parte de la información se realiza en formato físico.
- No existe una correcta gestión de información para los diferentes tipos de egreso existentes.

1.4. Limitaciones de las soluciones informáticas

Después de un análisis de los sistemas de Cuba y Venezuela se puede resumir que el sistema SACORE aún presenta deficiencias de automatización aunque este se ha actualizado en varias ocasiones, sobre todo en el módulo Egreso, presentando grandes problemas a la hora de realizar muchos de los procesos y trámites que se le realizan a los internos antes de poder egresar. El sistema SIGEP presenta una correcta gestión de documentos y datos de los internos pero no está regido por la legislación Cubana, por lo que no se gestionan muchos datos de interés para la correcta reinserción social de los internos en la sociedad.

Los egresos posibles en el sistema penitenciario venezolano, según la legislación vigente, se dividen en egresos por evasión o fuga, defunción y decisión judicial. La forma de proceder cuando se encuentran procesos legales pendientes varía de un establecimiento a otro dejando muchas veces la decisión final a manos del director del establecimiento lo que puede dar lugar a una infracción.

1.5. Metodología, tecnologías y herramientas utilizadas en el desarrollo

1.5.1. Metodología de Desarrollo

"El Proceso Unificado de Desarrollo conocido como RUP, es un proceso de desarrollo de software que permite el desarrollo de software a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad. Aunque con el inconveniente de generar mayor complejidad en los controles de administración del mismo. Sin embargo, los beneficios obtenidos recompensan el esfuerzo invertido en este aspecto.

El proceso de desarrollo constituye un marco metodológico que define en términos de metas estratégicas, objetivos, actividades y artefactos (documentación) requerido en cada fase de desarrollo. Esto permite enfocar esfuerzo de los recursos humanos en términos de habilidades, competencias y capacidades a asumir roles específicos con responsabilidades bien definidas.

Estructura del ciclo de vida del proceso de desarrollo unificado

Fase de concepción

Esta fase tiene como propósito definir y acordar el alcance del proyecto con los patrocinadores, identificar los riesgos potenciales asociados al proyecto, proponer una visión muy general de la arquitectura de software y producir el plan de las fases y el de iteraciones.

Fase de elaboración

En la fase de elaboración se seleccionan los casos de uso que permiten definir la arquitectura base del sistema y se desarrollaran en esta fase, se realiza la especificación de los casos de uso seleccionados y el primer análisis del dominio del problema, se diseña la solución preliminar.

Fase de construcción

El propósito de esta fase es completar la funcionalidad del sistema, para ello se deben clarificar los requerimientos pendientes, administrar los cambios de acuerdo a las evaluaciones realizados por los usuarios y se realizan las mejoras para el proyecto.

Fase de transición

El propósito de esta fase es asegurar que el software esté disponible para los usuarios finales, ajustar los errores y defectos encontrados en las pruebas de aceptación, capacitar a los usuarios y proveer el soporte técnico necesario. Se debe verificar que el producto cumpla con las especificaciones entregadas por las personas involucradas en el proyecto." (2)

Debido a la gran cantidad de módulos y relaciones entre subsistemas del proyecto se optó por el uso de la metodología RUP. El módulo Egreso presenta una gran complejidad en las validaciones, en las cuales se utilizan una gran cantidad de datos aportados por clases de otros módulos, por lo que se hace necesaria la generación de documentación para su posterior mantenimiento. También se utilizó esta metodología para que existiera una buena comunicación entre los miembros del proyecto. La generación de documentación en cada proceso de desarrollo contribuirá a crear un sistema de fácil mantenimiento y flexible a los futuros cambios legislativos.

1.5.2. Herramientas de Desarrollo

"NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos. Al día de hoy hay disponibles dos productos: el NetBeans IDE y NetBeans Platform. NetBeans IDE es un entorno de desarrollo - una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. También está disponible NetBeans Platform; una base modular y extensible usada como estructura de integración para crear grandes aplicaciones de escritorio. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones. Ambos productos son de código abierto

y gratuitos para uso tanto comercial como no comercial." (3)**Netbeans 7.0** ha sido el IDE seleccionado para el desarrollo del proyecto porque ofrece las siguientes características afines al proyecto:

- Incluye un cliente de control de versiones para el trabajo con los archivos de código.
- Tiene soporte para el lenguaje de programación definido en el proyecto.
- Brinda soporte para el marco de trabajo web.

Visual Paradigm para UML 5.0 es una poderosa herramienta CASE[1] multiplataforma que permite el modelado UML (Unified Modeling Language). Proporciona a los desarrolladores una plataforma que permite una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. También facilita la interoperabilidad con otras herramientas CASE y con los entornos de desarrollos más usados. Permite la generación de una amplia gama de diagramas, incluyendo diagramas de entidad-relación y la documentación. (4) Los proyectos de software según su complejidad necesitan ser modelados y documentados. Visual Paradigm ofrece una serie de diagramas que permite la documentación y modelado de los mismos. La complejidad de la secuencia de validaciones en el módulo Egreso y otros contribuye a que esta herramienta sea imprescindible para el diseño del mismo.

ER/Studio 8.0 "es una herramienta CASE muy fácil de usar, con soporte de múltiples niveles de análisis y que permite realizar tanto el modelado lógico de los datos como la implantación física en la base de datos elegida para tal fin. ER/Studio, soporta más de 30 de las bases de datos más importantes de la industria." (5) Se determinó el uso de ER / Studio como herramienta de modelado de las clases del dominio porque

Tortoise SVN 1.6 es un cliente de Subversión (SVN), implementado como una extensión Shell de Windows. Tiene una interfaz intuitiva y fácil de usar y no necesita el cliente de línea de comandos para su funcionamiento. (6) En los proyectos con muchos desarrolladores es fundamental tener una correcta gestión de la configuración. Para ello se seleccionó a Tortoise Subversión como cliente de control de versiones. La elaboración del módulo egreso fue llevada a cabo por dos desarrolladores, por lo que se hizo necesario establecer una serie de pautas y medidas para la organización del trabajo. Además de gestionar los cambios en el trabajo realizado a través del tiempo.

[1] CASE: del ingles, Computer Aided Software Engineering

Apache Tomcat 6.0 es una implementación libre y de código abierto de las tecnologías Java Servlet y Java Server Pages desarrollados bajo el proyecto Jakarta en la fundación Apache Software. Tomcat está disponible para uso comercial bajo la licencia ASF (Apache Software Foundation) desde el sitio web de Apache. (7)Se ha seleccionado Apache Tomcat debido principalmente a que el entorno de desarrollo está basado en Java y ser un servidor web libre.

1.5.3. Lenguajes

Java es un lenguaje de programación escrito por la empresa Sun Microsystems en 1991 para uso interno. Fue creado con la idea de crear un lenguaje orientado a objetos independiente de la plataforma. Inicialmente fue nombrado Oak y en 1995 pasó a nombrarse Java y se hizo público. Para hacer posible su ejecución en múltiples plataformas este hace uso de una máquina virtual llamada Máquina Virtual de Java (JVM) la cual se encuentra disponible en una gran cantidad de plataformas. Actualmente es uno de los lenguajes más difundidos y está orientado al uso en la Web. (8) Su uso en el desarrollo de la aplicación es fundamental debido a la gran cantidad de librerías que posee y su estrecha integración con el protocolo IP.

JavaScript es el lenguaje interpretado orientado a objetos desarrollado por Netscape que se utiliza en millones de páginas web y aplicaciones de servidor en todo el mundo. JavaScript es un lenguaje de programación dinámico que soporta construcción de objetos basado en prototipos. La sintaxis básica es similar a Java y C++ con la intención de reducir el número de nuevos conceptos necesarios para aprender el lenguaje. JavaScript puede funcionar como lenguaje procedimental y como lenguaje orientado a objetos. Los objetos se crean programáticamente añadiendo métodos y propiedades a lo que de otra forma serían objetos vacíos en tiempo de ejecución, en contraposición a las definiciones sintácticas de clases comunes en los lenguajes compilados como C++ y Java. (9) En el desarrollo del proyecto SIDEP su uso es imprescindible ya que a través de él se ejecutan las validaciones de los datos y se envían peticiones asincrónicas al servidor.

Groovy: Groovy es un lenguaje dinámico que se ejecuta en la JVM. Usa los mismos tipos de datos que Java y es 100% compatible con él a nivel binario, lo que le convierte en el complemento perfecto del lenguaje líder en desarrollo de software empresarial. Juntos, Groovy y Java constituyen la plataforma de nueva generación para desarrollo de software empresarial. (10) Al tener el soporte para lenguajes de

dominio específico facilita la creación de consultas complejas y dinámicas a través de Hibernate. También junto a Java es el lenguaje sobre el que fue implementado el marco de desarrollo web del proyecto.

1.5.4. Tecnologías

Ajax es el acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es la unión de varias tecnologías independientes con el objetivo permitir a través del navegador el envió y recepción de datos de manera asincrónica. Siendo posible de esta forma realizar cambios en la página sin tener que recargarla nuevamente. Las tecnologías que componen Ajax son las siguientes:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML, XSLT y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías. (11)

El uso de esta tecnología ayudaría a disminuir la carga de trabajo en el servidor ya que este no tendría que refrescar la página completa y agilizaría los tiempos de respuesta a consultas sobre la base de datos.

JSON (JavaScript Object Notation) es un formato sencillo para el intercambio de información. JSON permite representar estructuras de datos (arrays) y objetos (arrays asociativos) en forma de texto. La notación de objetos mediante JSON es una de las características principales de JavaScript y es un mecanismo definido en los fundamentos básicos del lenguaje. (12) Una de las ventajas del uso de JSON es que el marco de trabajo del proyecto permite la realización de peticiones de datos usando este formato.

Grails 1.3.7 es un marco de trabajo para aplicaciones web libre desarrollado sobre el lenguaje de programación Groovy (el cual a su vez se basa en la plataforma Java).Grails pretende ser altamente productivo siguiendo paradigmas tales como convención sobre configuración o no te repitas (DRY), proporcionando un entorno de desarrollo estandarizado y ocultando gran parte de los detalles de configuración al programador. Mas adelante en el marco de desarrollo web se hará énfasis en los paradigmas que brinda esta tecnología.

Hibernate es una librería de Mapeo Relacional de Objetos de código abierto que hace sencillo el trabajo con bases de datos relacionales. Simula al desarrollador como si la base de datos trabajara directamente con objetos, lo que lleva a que el desarrollador se preocupe por el negocio de la aplicación y no del cómo se persistirá/recuperará/eliminará la información de la base de datos. (13) Hibernate es la tecnología que

esta implementada en el marco de trabajo web Grails para el mapeo de las tablas, haciendo transparente para los desarrolladores el acceso a la base de datos. A través de HQL (Hibernate Query Language) y Groovy podemos realizar las consultas.

Dojo Toolkit 1.5 es una biblioteca JavaScript de código abierto. Resuelve problemas comunes y engorrosos en el desarrollo de aplicaciones con JavaScript como, por ejemplo, la disparidad del modelo de eventos de los distintos navegadores. Provee un conjunto de componentes de interfaz gráfica de usuario como calendarios y menús, que se pueden insertar de manera sencilla en páginas HTML. (14) Estos componentes son utilizados en las interfaces del SIDEP (Sistema Informativo de la Dirección de Establecimientos Penitenciarios), logrando agilizar el desarrollo al reutilizar código existente con un alto grado de terminación.

Oracle Database 11g es un sistema de gestión de base de datos objeto-relacional (o ORDBMS por el acrónimo en inglés de Object-Relational Data Base Management System), desarrollado por Oracle Corporation.

Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando en las siguientes características:

- Soporte de transacciones
- Estabilidad
- Escalabilidad
- Soporte multiplataforma.

Las últimas versiones de Oracle han sido certificadas para poder trabajar bajo GNU/Linux. (15) El sistema SIDEP se piensa desplegar en diferentes locaciones en la cuales existirá una réplica de la base de datos y se hace necesario tener sincronizadas todas las réplicas con una base datos central, además se quiere implementar un Data Mart[2]. Todas estas condiciones generan un gran flujo de información donde el indexado de los datos es primordial. Oracle es la alternativa seleccionada debido a que es el sistema más eficiente en comparación con otras alternativas como MySQL y PostgreSQL. Otra de las razones para la adopción de Oracle fue su reciente soporte para el sistema operativo Linux, ya que reduce los costos del sistema.

[2] Data Mart: Un Data Mart es la implementación de un almacén de datos con alcance restringido a un área funcional, problema en particular, departamento, tema o grupo de necesidades.

1.6. Propuesta de solución

Como resultado del estudio de los sistemas SIGEP y SACORE se determinó por parte de la dirección del proyecto la necesidad crear una aplicación Web con arquitectura n-capas basándose en el marco de trabajo Grails. Definiéndose como entorno de desarrollo integrado el NetBeans y lenguaje de programación Groovy. El proceso de implementación estará guiado por la metodología RUP utilizando al Visual Paradigm como herramienta de modelado, como servidor Web Apache Tomcat y gestor de base de datos Oracle 11g. La aplicación estará dividida en 7 subsistemas entre los cuales estará el subsistema Registro Legal y este contendrá 7 módulos entre los que se encuentra el módulo Egreso.

1.7. Conclusiones

En el presente capítulo se ha hecho un estudio sobre los dos sistemas de gestión penitenciaria, analizando en cada caso la realización de los procesos de egreso y las características de cada uno. Concluyendo que SACORE no brinda suficientes funcionalidades por lo que se hace necesario la implementación de un nuevo sistema de gestión. Se analizaron las características de las diferentes tecnologías, metodologías y herramientas utilizadas en el desarrollo de este sistema de gestión penitenciaria.

Capítulo 2: Diseño del Sistema.

2.1 Introducción

En el actual capítulo se describirá la arquitectura usada en el desarrollo del proyecto SIDEP, enfocándose en el módulo Egreso. Se hará un análisis de los patrones de diseño y casos de uso que lo componen. También se describirán las relaciones entre las clases a través de diagramas de clases del diseño.

El proceso de egreso se inicia al recibir el oficial de registro legal un mandamiento de libertad por parte del tribunal. El oficial busca el expediente al que fue otorgado el mandamiento de libertad y hace una revisión de los procesos legales. Si el interno tiene otro proceso con al menos un mandamiento de Admisión o Auto de imposición se le informa al interno que debe permanecer en prisión sino se confecciona el modelo de egreso y se envía el expediente a la DNI (Dirección Nacional de Identificación) y termina el egreso.

2.2 Análisis del Negocio

Para el diseño e implementación de una aplicación es necesaria e imprescindible haber realizado anteriormente un buen análisis del negocio, que implicaría definir los requisitos, realizar de los diagramas de CU del Sistema, explicar la relación que existe entre ellos, los actores que interviene en el proceso de negocio, entre otras funciones. El desarrollo de este trabajo de diploma parte con haber aprobado todo lo especificado en las Disciplinas de Modelación del Negocio y Requerimientos planteadas anteriormente por los analistas principales del proyecto del SIDEP.

La descripción de los CU se hizo de acuerdo a las leyes y resoluciones vigentes en ese período, habiéndose definido veintiséis tipos de egreso, facilitando una mejor organización de la información. Se hizo uso de los requisitos capturados por los analistas para el modelado de los diferentes diagramas contribuyendo al desarrollo de la aplicación.

2.3 Descripción de los Casos de Usos del módulo Egreso

En el desarrollo del documento se apreciara solamente la modelación completa(especificación de CU, Diagrama de Clases del Diseño, Diagrama de Secuencia) de dos CU, debido a la gran importancia y complejidad que poseen, los restantes Diagramas se podrán ver en los anexos. A continuación se mostrará las especificaciones de los casos de usos Egreso por Muerte y Suspensión de la Medida de Seguridad Reeducativa de Internamiento con el objetivo de que se comprenda mejor la lógica del negocio

modelada en los diagramas de Clases del Diseño y de los de Secuencias expuestos en los próximos epígrafes.

Registrar Egreso por Muerte

Objetivo	Registrar el egreso de internos que hayan muerte por la ejecución de su sanción de muerte.
Actores	Oficial de Registro Legal
Resumen	El CU se inicia cuando el sistema muestra los campos para registrar un egreso por muerte, el Oficial de Registro Legal registra el egreso del interno por muerte, adicionando el acta de Medicina Legal. Termina el CU.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	El Oficial de Registro Legal debe estar autenticado en el sistema. El interno tiene tipo de sanción “Muerte” por el proceso que tiene estado “Cumpliendo”. El interno tiene fecha de firmeza y fecha de confirmación del Consejo de Estado por el proceso que tiene estado “Cumpliendo”.
Postcondiciones	Queda registrado el egreso del Sistema Penitenciario del interno.
Flujo de eventos	
Flujo básico Registrar egreso por muerte	

	Actor	**Sistema**
1.		El sistema valida que el interno tenga la fecha de confirmación del Consejo de Estado en el proceso con estado “Cumpliendo”.

		Si no se cumple que el interno tenga en el proceso con estado "Cumpliendo" la fecha de confirmación del Consejo de Estado, ver el flujo alterno 1a. "No se puede dar egreso por muerte".
2.		El sistema muestra los siguientes campos para ser completados: • Fecha de egreso • Hora de muerte
3.	El Oficial de Registro Legal introduce los datos y oprime el botón "Aceptar".	
4.		El sistema valida los datos introducidos. Si no se introdujeron todos los datos obligatorios, ver el flujo alterno 4a. "Faltan datos obligatorios".
5.	El Oficial de Registro Legal oprime el botón "Cerrar". Si el Oficial de Registro Legal selecciona la opción de "Imprimir" para imprimir el modelo de egreso, se ejecuta el CU extendido "Generar modelo".	
6.		El sistema registra la información.
7.		El sistema asigna el estado "Cumplido" al proceso por el que

		estaba cumpliendo el interno.
8.		El sistema decrementa una capacidad en las capacidades en uso del sistema y actualiza la capacidad sin ocupar.
9.		El sistema le asigna al estado de la cama donde se encontraba residiendo cada interno, “Disponible”.
10.		El sistema asigna al estado del expediente el valor de “Cerrado”.
11.		Termina el CU.
Flujos alternos		
1a. No se puede dar egreso por muerte		
	Actor	**Sistema**
1.		El sistema muestra un mensaje “El interno no puede egresar por “Ejecución de la sanción de muerte” porque no tiene la confirmación del Consejo de Estado”.
2.		Regresa al paso 4 del flujo básico del CU Registrar egreso.
4a. Faltan datos obligatorios		
	Actor	**Sistema**
1.		El sistema muestra un mensaje “Introduzca los datos obligatorios” y

		señala los campos obligatorios que no fueron introducidos.
2.	Regresa al paso 3 del flujo básico "Registrar egreso por muerte"	

Relaciones	**CU Incluidos**	No aplicable
	CU Extendidos	Generar modelo (Ver Especificación de Casos de Uso. Comunes)
Requisitos no funcionales	No aplicable	
Asuntos pendientes	No aplicable	

Registrar Egreso por Suspensión de la Medida de Seguridad Reeducativa de Internamiento

Objetivo	Registrar el egreso de internos por suspensión de la medida de seguridad.
Actores	Oficial de Registro Legal
Resumen	El CU se inicia cuando el sistema muestra los campos para registrar un egreso por extinción de la sanción, el Oficial de Registro Legal registra el egreso del interno por suspensión de la Medida de Seguridad Reeducativa de Internamiento, el sistema valida que el interno pueda egresar por dicho motivo y termina el CU.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	El Oficial de Registro Legal debe estar autenticado en el sistema. El interno debe tener estatus legal "Asegurado". Debe haberse aprobado en el módulo Gestión del Consejo de

	Promoción un beneficio de “Suspensión de la Medida de Seguridad Reeducativa de Internamiento” por parte del Tribunal.	
Postcondiciones	Queda registrado el egreso del Sistema Penitenciario del interno.	
Flujo de eventos		
Flujo básico Registrar egreso		
	Actor	**Sistema**
1.		El sistema verifica que al interno se le haya aprobado, por el Tribunal, un beneficio de “Suspensión de la Medida de Seguridad Reeducativa de Internamiento”. Si no se le ha aprobado un beneficio de “Suspensión de la Medida de Seguridad Reeducativa de Internamiento”, ver el flujo alterno 1a. “No se puede dar egreso por variación”.
2.		El sistema muestra los siguientes campos para ser completados: • Fecha de egreso
3.	El Oficial de Registro Legal introduce los datos y oprime el botón “Aceptar”.	
4.		El sistema verifica si el interno tiene otro proceso con estado “Interrumpido”. Si el interno tiene un proceso con el

		estado de “Interrumpido”, ver el flujo alterno 4a. “Interno con proceso interrumpido”.
5.		El sistema verifica si el interno tiene otro proceso con estado “Por cumplir”. Si el interno tiene un proceso con el estado de “Por cumplir”, ver el flujo alterno 5a. “Interno con proceso por cumplir”.
6.		El sistema verifica si el interno tiene otro proceso con trámite legal pendiente “Causa pendiente”. Si el interno tiene causa pendiente con el documento “Mandamiento de Admisión”, ver el flujo alterno 6a. “Pasar a ser acusado del Tribunal”.
7.		El sistema verifica si el interno tiene otro proceso con trámite legal pendiente de “EFP pendiente”. Si el interno tiene EFP pendiente con el documento “Auto de imposición de la medida cautelar”, ver el flujo alterno 4b. “Pasar a ser acusado de Fiscalía”.
8.		El sistema valida los datos introducidos.

		Si no se introdujeron todos los datos obligatorios, ver el flujo alterno 8a. “Faltan datos obligatorios”.
9.		El sistema registra la información.
10.		El sistema muestra los datos del egreso y permite imprimir el modelo de egreso.
11.	El Oficial de Registro Legal oprime el botón “Cerrar”. Si el Oficial de Registro Legal oprime el botón “Imprimir” para imprimir el modelo de egreso, se ejecuta el CU extendido “Generar modelo”.	
12.		El sistema asigna el estado “Cumplido” al proceso por el que estaba cumpliendo el interno.
13.		El sistema decrementa una capacidad en las capacidades en uso del sistema y actualiza la capacidad sin ocupar.
14.		El sistema le asigna al estado de la cama donde se encontraba residiendo cada interno, “Disponible”.
15.		El sistema mantiene el estado del expediente “Abierto” pero “Deshabilitado”.

16.		Termina el CU.
Flujos alternos		
1a. No se puede dar egreso por variación		
	Actor	**Sistema**
1.		El sistema muestra un mensaje “El interno no puede egresar por “Suspensión de la Medida de Seguridad Reeducativa de Internamiento” porque el Tribunal no lo ha aprobado”.
2.		Regresa al paso 4 del flujo básico del CU Registrar egreso.
4a. Interno con proceso interrumpido		
	Actor	**Sistema**
1.		El sistema cambia el estatus legal del interno a Asegurado.
2.		El sistema muestra el mensaje “El interno no puede egresar de prisión por tener una medida de seguridad interrumpida” y no permite generar el modelo de egreso.
3.		El sistema asigna el estado del proceso por el que cumple “Cumplido”.
4.		El sistema asigna el estado del proceso que tenía interrumpido

		“Cumpliendo”.
5.		Se ejecuta el CU “Registrar trámite legal pendiente de rectificación de liquidación”
6.		Termina el CU.
5a. Interno con proceso por cumplir		
	Actor	**Sistema**
1.		El sistema cambia el estatus legal del interno a Sancionado.
2.		El sistema muestra el mensaje “El interno no puede egresar de prisión por tener una multa por pagar” y no permite generar el modelo de egreso.
3.		El sistema asigna el estado del proceso por el que cumple “Cumplido”.
4.		El sistema asigna el estado del proceso que tenía por cumplir “Cumpliendo”.
5.		Se ejecuta el CU “Registrar trámite legal pendiente de rectificación de liquidación”
6.		Termina el CU.
6a. Pasar a ser acusado del Tribunal		
	Actor	**Sistema**

1.		El sistema verifica el número de apartado del Mandamiento de Admisión del interno. Si el número de apartado es #4, el sistema cambia el estatus legal del interno a Acusado PJ. Si el número de apartado es #2, el sistema cambia el estatus legal del interno a Acusado FD.
2.		El sistema muestra el mensaje "El interno no puede egresar de prisión por tener una causa pendiente" y no permite generar el modelo de egreso.
3.		El sistema asigna el estado "Cumplido" al proceso por el que estaba cumpliendo el interno y "Pendiente" al proceso por el que pasa a Acusado.
4.		Termina el CU
4b. Pasar a ser acusado de Fiscalía		
	Actor	**Sistema**
1.		El sistema cambia el estatus legal del interno a Acusado PJ.
2.		El sistema muestra el mensaje "El interno no puede egresar de prisión por tener EFP pendiente" y no

		permite generar el modelo de egreso.
3.		El sistema asigna el estado “Cumplido” al proceso por el que estaba cumpliendo el interno y “Pendiente” al proceso por el que pasa a Acusado.
4.		Termina el CU.
8a. Faltan datos obligatorios		
	Actor	**Sistema**
1.		El sistema muestra un mensaje “Introduzca los datos obligatorios” y señala los campos obligatorios que no fueron introducidos.
2.	El Oficial de Registro Legal introduce los datos y oprime el botón “Aceptar”.	
3.		Regresa al paso 8 del flujo básico “Registrar Egreso por muerte”

Relaciones	**CU Incluidos**	No aplicable
	CU Extendidos	Generar modelo (Ver Especificación de Casos de Uso. Comunes) Registrar trámite legal pendiente de rectificación de liquidación (Ver especificación de casos de uso. Situación legal)
Requisitos no funcionales	No aplicable	
Asuntos	No aplicable	

pendientes	

A continuación se muestra una descripción de todos los Casos de Usos del Módulo de Egreso, con la que se podrá entender mucho mejor cual es el objetivo y función de cada uno de ellos.

CU	Nombre	Descripción
1	Registrar el egreso de internos	Permite registrar el egreso de internos en dependencia del motivo, registrando los datos y los documentos que permiten avalar ese egreso.
2	Registrar egreso por fallecimiento	Permite registrar el egreso de internos que hayan fallecido en el cumplimiento de la sanción, medida cautelar o medida de seguridad.
3	Registrar egreso por evasión	Permite registrar el egreso de internos que se hayan evadido en el cumplimiento de la sanción, prisión provisional o medida de seguridad.
4	Registrar egreso por ausencia al permiso	Permite registrar el egreso de internos que se hayan ausentado al permiso otorgado durante el cumplimiento de la sanción o medida de seguridad.
5	Registrar egreso por licencia extrapenal	Permite registrar el egreso de internos que se les haya aprobado la licencia extrapenal durante el cumplimiento de la sanción.
6	Registrar egreso por muerte	Permite registrar el egreso de internos que hayan muerto por la ejecución de su sanción de muerte.
7	Registrar egreso por cumplimiento	Permite registrar el egreso de internos que arriben a la fecha de cumplimiento.
8	Registrar egreso por libertad condicional	Permite registrar el egreso de internos que arriben a la fecha de libertad condicional.

9	Registrar egreso por sentencia absolutoria dictada en proceso de revisión	Permite registrar el egreso de internos que el Tribunal disponga por sentencia absolutoria dictada en proceso de revisión.
10	Registrar egreso por suspensión de la ejecución de la sanción	Permite registrar el egreso de internos que el Tribunal disponga por suspensión de la ejecución de la sanción.
11	Registrar egreso por sustitución de privativa por subsidiaria	Permite registrar el egreso de internos que se les sustituya la privativa de libertad por una subsidiaria.
12	Registrar egreso por indulto o amnistía	Registrar el egreso de internos que el Tribunal disponga por indulto o amnistía.
13	Registrar egreso por expulsión o traslado del territorio nacional	Permite registrar el egreso de internos que se le realice el proceso de traslado del territorio nacional.
14	Registrar egreso por pago de multa	Permite registrar el egreso de internos por pago de multa.
15	Registrar egreso por cumplimiento del término de la Medida de Seguridad Reeducativa de Internamiento	Permite registrar el egreso de internos por cumplimiento de la medida de seguridad.
16	Registrar egreso por variación de la Medida de Seguridad Reeducativa de Internamiento	Permite registrar el egreso de internos por variación de la medida de seguridad.
17	Registrar egreso por sus pensión de TCCI	Permite registrar el egreso de internos por suspensión de TCCI.
18	Registrar egreso para acusado	Permite registrar el egreso de internos para un acusado por sobreseimiento, absuelto o cumplir con la preventiva en sentencia no firme.

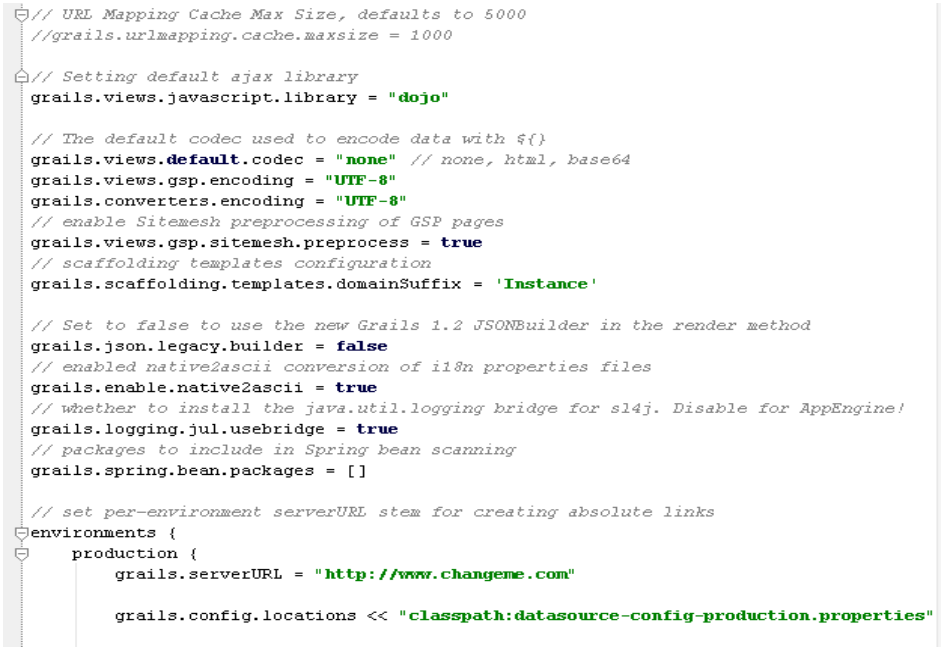
19	Registrar egreso por modificación o revocación de la medida cautelar	Permite registrar el egreso de internos para un acusado por modificación o revocación de medida cautelar de prisión provisional.
20	Registrar egreso por suspensión de la Medida de Seguridad Reeducativa de Internamiento	Permite registrar el egreso de internos por suspensión de la medida de seguridad.
21	Registrar egreso por prescripción penal	Permite registrar el egreso de internos por prescripción penal.
22	Registrar egreso por retroactividad de la ley	Permite registrar el egreso de internos por retroactividad de la ley.
23	Registrar egreso por revisión de la causa del tribunal	Permite registrar el egreso de internos por revisión de la causa del tribunal supremo.
24	Registrar egreso por aplazamiento de la sanción	Permite registrar el egreso de internos por aplazamiento de la sanción.
25	Consultar Egresos	Permite consultar los egresos otorgados por diferentes criterios de búsqueda.
26	Generar reporte previo a la libertad	Permite generar el reporte que debe enviarse para notificar que el interno se encuentra previo a la libertad.

2.4 Marco de trabajo de desarrollo web

“Grails es un marco de trabajo de desarrollo para aplicaciones web, dentro de la plataforma Java, con base en otros proyectos open-source como Spring, Hibernate y SiteMesh, que utiliza el lenguaje de programación Groovy. El objetivo de Grails es brindar al usuario un entorno de alta productividad, extensible y fácil de utilizar, ofreciendo el balance adecuado entre consistencia y funcionalidades potentes.

Grails nace como la respuesta de Java al cambio de paradigma impuesto por otros marcos de trabajo como Ruby on Rails y Django. En este cambio de paradigma, en el cuál se comenzaron a cuestionar las arquitecturas complejas para el desarrollo de aplicaciones web, aparecieron varios conceptos revolucionarios como "Convention over configuration" y "Don't repeat yourself".

Convention over configuration: Se basa en la idea de eliminar los archivos de configuración, utilizados generalmente para bases de datos y mapeo de peticiones, siendo reemplazados por convenciones pre-establecidas, tomadas directamente desde el código fuente." (16) En los entornos de trabajo basados en Java la configuración de los archivos del proyecto siempre han sido una tarea compleja, Grails pretende facilitar este proceso como se muestra en la siguiente imagen donde se muestra la configuración de una aplicación

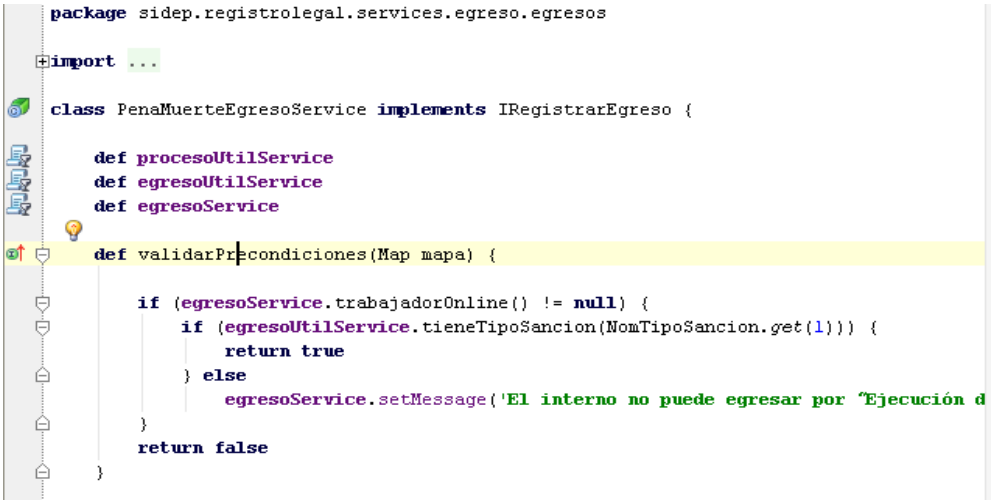


```
// URL Mapping Cache Max Size, defaults to 5000
//grails.urlmapping.cache.maxsize = 1000

// Setting default ajax library
grails.views.javascript.library = "dojo"

// The default codec used to encode data with ${}
grails.views.default.codec = "none" // none, html, base64
grails.views.gsp.encoding = "UTF-8"
grails.converters.encoding = "UTF-8"
// enable Sitemesh preprocessing of GSP pages
grails.views.gsp.sitemesh.preprocess = true
// scaffolding templates configuration
grails.scaffolding.templates.domainSuffix = 'Instance'

// Set to false to use the new Grails 1.2 JSONBuilder in the render method
grails.json.legacy.builder = false
// enabled native2ascii conversion of i18n properties files
grails.enable.native2ascii = true
// whether to install the java.util.logging bridge for sl4j. Disable for AppEngine!
grails.logging.jul.usebridge = true
// packages to include in Spring bean scanning
grails.spring.bean.packages = []

// set per-environment serverURL stem for creating absolute links
environments {
    production {
        grails.serverURL = "http://www.changeme.com"

        grails.config.locations << "classpath:datasource-config-production.properties"
```

.

Figura 1 Facilidad de configuración

"Don't repeat yourself: Más conocido como DRY por sus siglas en inglés, este principio alienta al desarrollador a implementar estructuras reutilizables, de forma de minimizar la repetición de código." (16) En la aplicación se creó la clase **EgresoUtilService** la cual contiene funciones que le son útiles a todos los servicios que registran a los diferentes tipos de egreso, cada vez que se requiera un método de esta clase al controlador o servicio que la requiera le es aplicada una inyección de dependencias.



```
package sidep.registrolegal.services.egreso.egresos

import ...

class PenaMuerteEgresoService implements IRegistrarEgreso {

    def procesoUtilService
    def egresoUtilService
    def egresoService

    def validarPrecondiciones(Map mapa) {

        if (egresoService.trabajadorOnline() != null) {
            if (egresoUtilService.tieneTipoSancion(NomTipoSancion.get(1))) {
                return true
            } else
                egresoService.setMessage('El interno no puede egresar por "Ejecución d
        }
        return false
    }
```

Figura 2 Reutilización de Código

"Como se planteó anteriormente, Grails utiliza Groovy, el lenguaje de programación creado por Guillerme Laforge, como lenguaje base. Groovy es un lenguaje dinámico que cuenta con su propia especificación (JSR-241) y compila a código intermedio (bytecode). Esto permite la total compatibilidad con Java, por lo cual se puede reutilizar cualquier librería de dicho lenguaje dentro de una nueva aplicación Grails." (16)

2.5 Arquitectura del Sistema

La arquitectura de software puede ser vista como la estructura del sistema en función de la definición de los componentes y sus interacciones.

Al hablar de arquitectura de software, se hace alusión a la especificación de la estructura del sistema, entendida como la organización de componentes y relaciones entre ellos; los requerimientos que debe satisfacer el sistema y las restricciones a las que está sujeto, así como las propiedades no funcionales del sistema y su impacto sobre la calidad del mismo; las reglas y decisiones de diseño. (17)

En la figura 1 se puede observar la arquitectura n-capas de la aplicación donde se aplica una variante del patrón Modelo Vista Controlador en las capas azules. La arquitectura que se planteó para el desarrollo de

la aplicación está basada en la arquitectura Grails, la cual incluye en la capa de la vista a los archivos gsp y las etiquetas de Grails, la capa controladora es similar a la del patrón MVC de Spring, en la capa del modelo contiene las clases de dominio con la característica que para su interacción con la base de datos utiliza Hibernate y finalmente incluye una capa de servicios donde se gestiona la parte de la lógica.

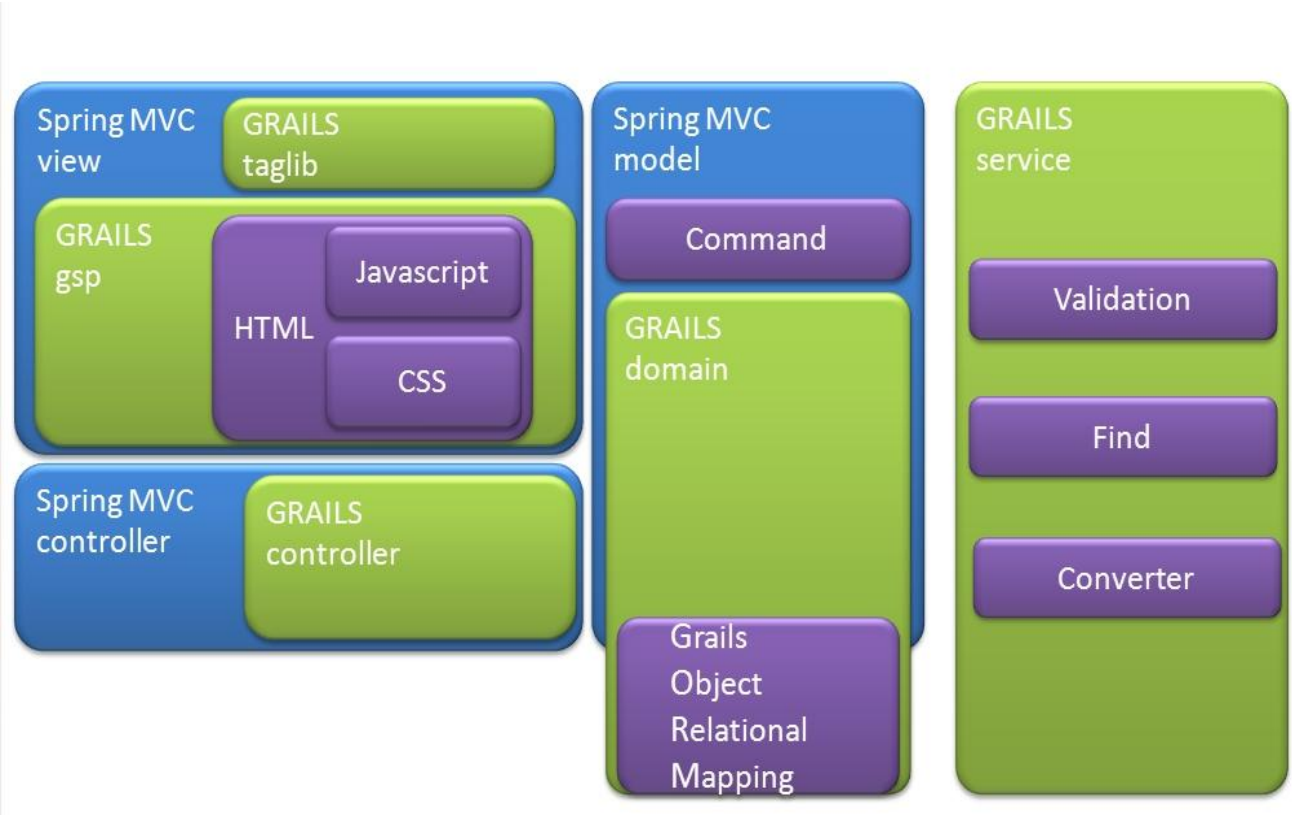


Figura 3 Ubicación de las capas del MVC usado por Grails respecto al de Spring

- **Vistas:** En esta capa se encuentran las plantillas Groovy Server Page (GSP), las cuales son las responsables de mostrar al usuario la información requerida que se encuentra en el modelo, para lo cual se apoyó mediante códigos HTML, JavaScript y Cascading Style Sheets (CSS) ,utilizando el marco de trabajo Dojo para la creación y perfeccionamiento de las vistas.
- **Modelo:** En esta capa es donde están las clases del dominio y los comandos utilizados por Spring. Una clase dominio representa datos persistentes y por defecto constituye una entidad de la BD. Grails utiliza un gestor de persistencia escrito en Groovy sobre Hibernate conocido por GORM (Grails Object

Relational Mapping) para controlar el ciclo de vida de las entidades y proporcionar una serie de métodos dinámicos que facilita la manipulación de los datos.

- **Controlador:** Es el encargado de mantener la comunicación entre las demás capas, actualizando o recibiendo peticiones de las vistas, así como interactuando con los servicios y el dominio. Grails utiliza el marco de trabajo Spring MVC e Inversión de Control para la inyección de dependencia, facilitando que el controlador solo tenga que definir una variable con el nombre del servicio que se desea emplear. Es decir, cuando el usuario realiza una petición, el Controlador es el encargado de gestionar la aplicación de la lógica del negocio sobre el modelo de dato, determinando cual es la vista a mostrar seguidamente.

Cuando se dice que los componentes de la capa de control "gestionan la aplicación de la lógica de negocio", se refiere a que son responsables de que ésta se aplique, lo cual no quiere decir que se implementará la lógica de los CU en los controladores. Normalmente esta lógica estará implementada en una capa:

- **Servicio:** En esta capa se encuentran la lógica de negocio de los casos de uso que componen la aplicación y es la encargada de llevar los cambios en el modelo. (18) La mayor parte de la lógica de la aplicación es llevada a cabo por los servicios con el objetivo de evitar sobrecargar la lógica de los controladores. Es importante entender el significado y la utilidad de estas capas para así poder desarrollar aplicaciones fáciles de mantener y evolucionar. El siguiente diagrama muestra el flujo recomendado para un caso de uso típico:

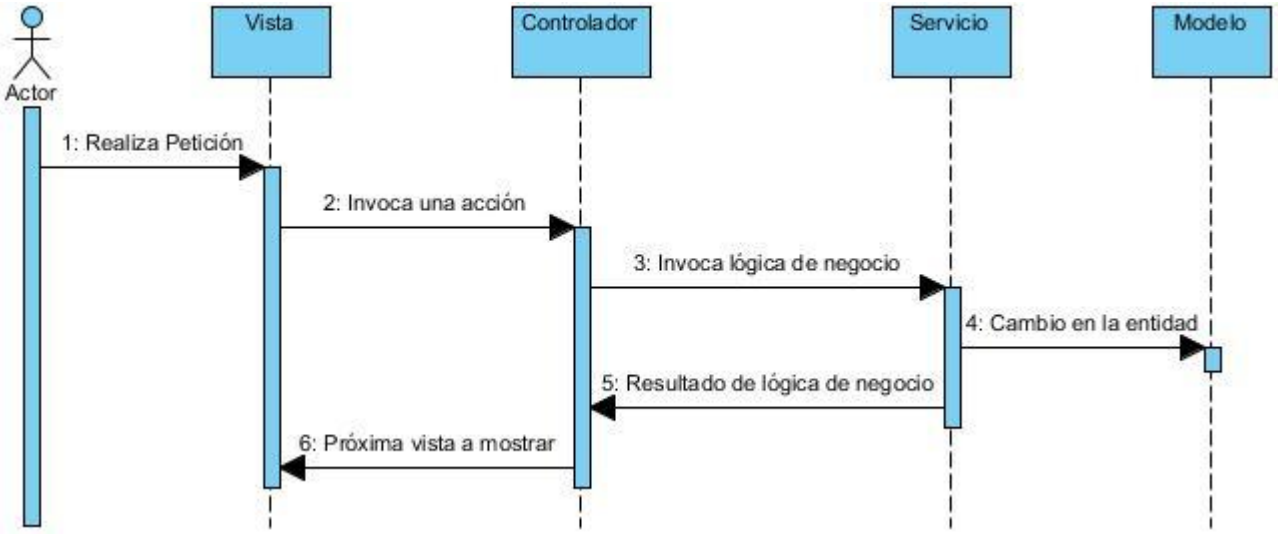


Figura 4 Ejemplo de interacción de las diferentes capas

1. El usuario realiza una petición al sistema mediante la Vista.
2. La solicitud llega al Controlador, según lo configurado en el mapeo de URLs.
3. El controlador invoca el Servicio encargado de la implementación del caso de uso.
4. En base al resultado de la invocación se realiza la modificación en el Modelo, el sistema decide cual es la próxima vista a mostrar y solicita a la capa de presentación que se la muestre al usuario.

Al utilizar este reparto de responsabilidades se consigue, por un lado componentes más pequeños y fáciles de mantener, y por otro, la posibilidad de reutilizar código en mayor medida.

2.6 Patrones de Diseño

Los patrones de diseño son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Es una solución estándar para un problema común de programación, una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios y es una manera más práctica de describir ciertos aspectos de la organización de un programa.

Un patrón de diseño es una abstracción de una solución en un nivel alto. Los patrones solucionan problemas que existen en muchos niveles de abstracción. Hay patrones que abarcan las distintas etapas del desarrollo; desde el análisis hasta el diseño y desde la arquitectura hasta la implementación. (19)

Algunos de los patrones de diseño que se utilizaron a lo largo del proceso de diseño para el desarrollo del proyecto SIDEP fueron:

Inversión de Control (IoC): La inversión de Control es otro patrón utilizado por Grails, según el cual las dependencias de un componente no deben gestionarse desde el propio componente para que este solo tenga la lógica necesaria para hacer su trabajo. (18) Este patrón viene implícito en el marco de trabajo en las capas internas siendo totalmente transparente para el desarrollador. Este es el patrón encargado de que los controladores le deleguen a los servicios, las peticiones realizadas por el usuario que tiene que ver con la gestión de la lógica del negocio. En Grails, Spring es el que se encarga de usar la Inversión de Control ya que es el que controla el flujo de trabajo y realiza la inyección de dependencia en las entidades del sistema.

Modelo Vista Controlador (Model View Controller): Para el diseño de aplicaciones con interfaces se utiliza el patrón de diseño Modelo-Vista-Controlador. La lógica de una interfaz de usuario cambia con más frecuencia que las clases del domino y la lógica de negocio. En caso de utilizar un diseño que mezcle todos los componentes de la aplicación, al tener que realizar algún cambio en la interfaz, esto afectaría directamente a los demás componentes, dificultando el cambio y aumentando la probabilidad de aparición de errores. Se trata de realizar un diseño que desacople la vista del modelo, con la finalidad de mejorar la reusabilidad. De esta forma las modificaciones en las vistas impactan en menor medida en la lógica de negocio o de datos. Este modelo ha sido aplicado a la aplicación a través del entorno de trabajo Grails, independizando a las vistas gsp, los archivos JavaScript y CSS (Cascade Style Sheet) de los controladores, los servicios y las clases del dominio permitiendo la modificación del flujo y la lógica.

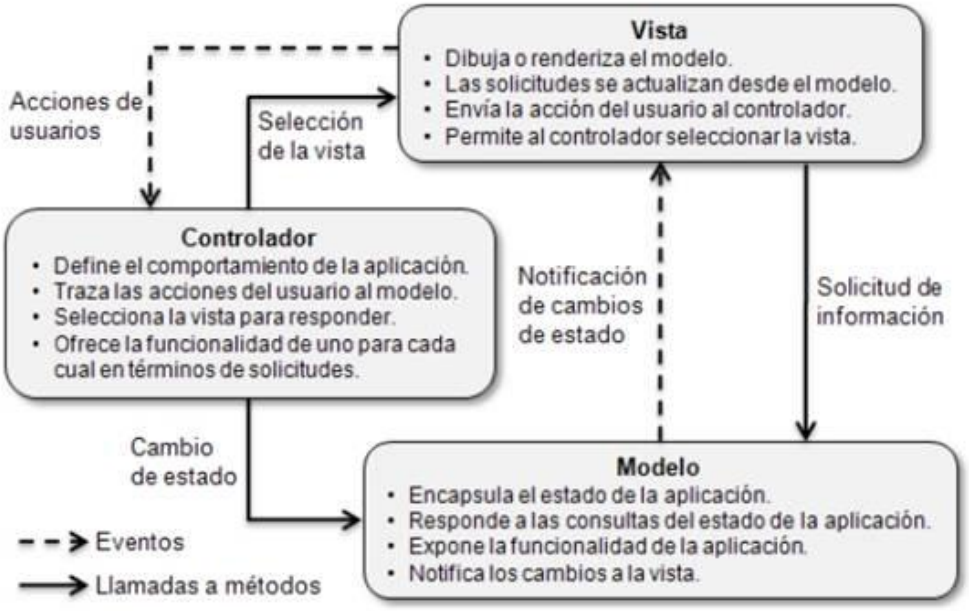


Figura 5 Patrón Modelo Vista Controlador

De los patrones básicos pertenecientes a los patrones de asignación de responsabilidades en inglés (General Responsibility Assignment Software Patterns, GRASP), se utilizaron:

Experto: Realiza la asignación de las responsabilidades a la clases que tienen la información necesaria para cumplir con la responsabilidad; es un principio básico que suele utilizarse en el diseño orientado a objetos. (19) Este patrón se esta implementado en la clase del dominio Egreso ya que esta clase es experta en el guardado de la información en la base de datos y de ella heredan los diferentes tipos de egreso. Siempre que sea necesaria la manipulación de datos de egreso esta clase será la experta.

Bajo Acoplamiento: La responsabilidad de este patrón es tener las clases lo menos ligadas entre sí, es decir, que cada clases se comuniquen con el menor número de clases que sea posible.

Alta Cohesión: La ventaja de este patrón es con él se mejora la claridad y facilidad con que se entiende el diseño. Es el responsable de asignarles a las clases responsabilidades de que trabajen sobre una misma área de aplicación y que no tengan mucha complejidad.

Controlador: Este patrón propone asignar la responsabilidad de controlar el flujo de eventos de un sistema, a clases específicas llamadas controladores. Sugiere que la lógica de negocios debe estar separada de la capa de presentación para aumentar la reutilización de código y a la vez tener un mayor control. La aplicación tiene como clase controladora a la clase EgresoController la cual es la encargada del manejo del flujo de Egreso.

El grupo de GoF (Gang of Four, Banda de los Cuatro) clasificaron los patrones en 3 grandes categorías basadas en su propósito: creacionales, estructurales y de comportamiento.

Creacionales: Crean instancias de objetos. El objetivo de estos patrones es de abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados.

Estructurales: Solucionan problemas de composición (agregación) de clases y objetos. Describen como las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades.

De comportamiento: Reduce el acoplamiento entre los objetos y define la comunicación e iteración entre los objetos de un sistema.

Singleton (instancia única): se implementa creando en determinada clase un método que cree una instancia del objeto sólo si todavía no existe alguna. Este patrón consiste en garantizar que una clase solo tenga una instancia y proporcionar un punto de acceso global a ella. Todos los servicios implementan el patrón Singleton durante el tiempo de vida del mismo. Cada vez que los mismos son referenciados en los controladores lo que hacen es llamar a la única instancia existente.

2.7 Diagrama de Clases del Diseño con estereotipos web

El Diagrama de Clases es el diagrama principal para el análisis y diseño. Un diagrama de clases presenta las clases del sistema con sus relaciones estructurales y de herencia. La definición de clase incluye definiciones para atributos y operaciones. (20)

Los diagramas de clases del diseño con estereotipos web se realizaron por CU, es decir, que se hizo por cada CU un diagrama de clases del diseño que representa las relaciones que existen entre las clases.

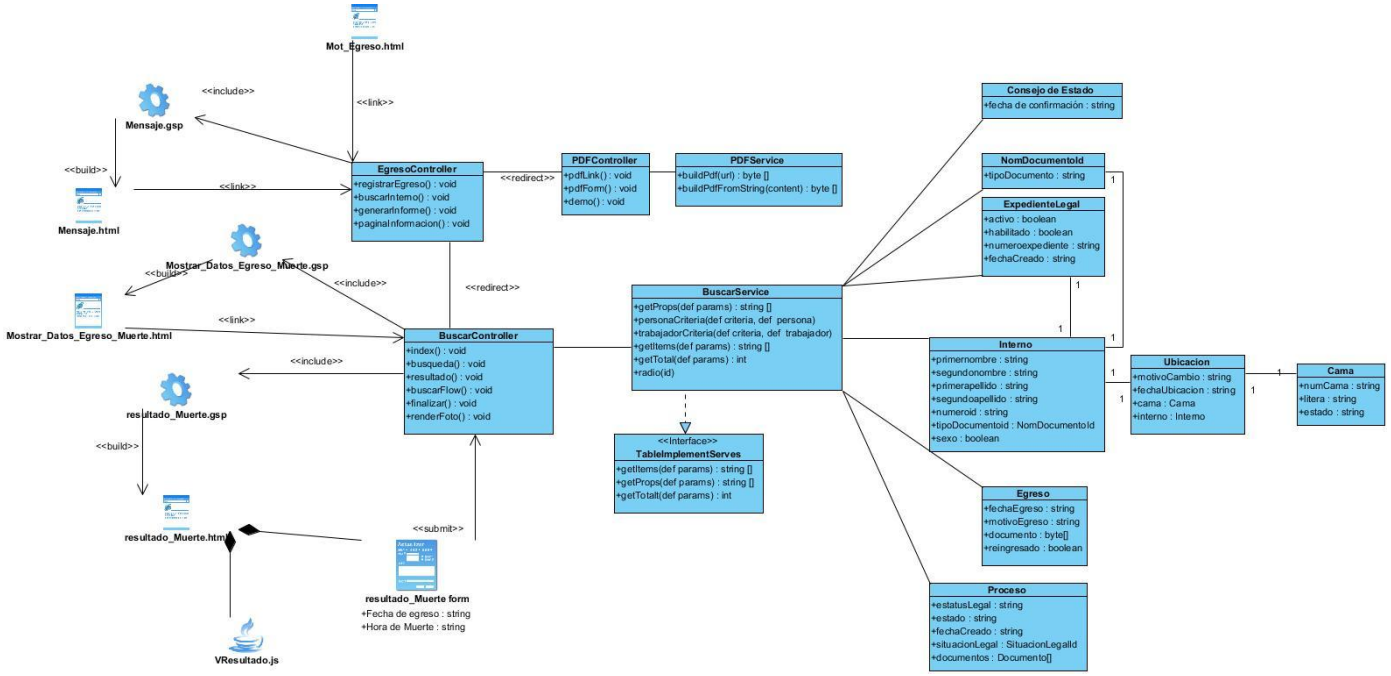


Figura 6 Diagrama de Clases del Diseño "Registrar Egreso por Muerte"

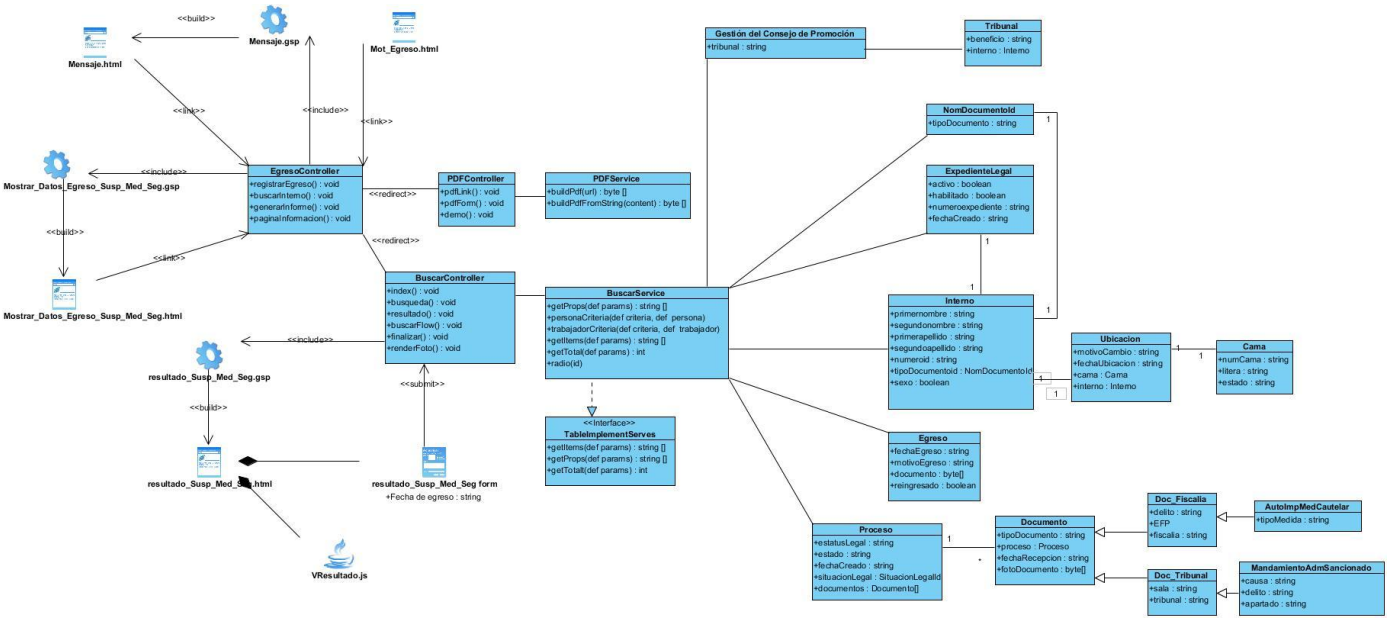


Figura 7 Diagrama de Clases del Diseño "Registrar Egreso por Suspensión de la Medida de Seguridad Reeducativa de Internamiento"

En los diagramas de clases del diseño mostrados anteriormente y los que están presentes en los anexos existen varias entidades que no pertenecen al módulo de Egreso, pero por la lógica del negocio tienen una estrecha relación con el mismo. A continuación se les explicara de forma general de que módulos proceden todas las entidades que aparecen en todos los diagramas de clases. Las entidades como Consejo de Gestión de Promoción e Incidencia pertenecen a los módulos con el mismo nombre, también están presenten las entidades Interno, Expediente Legal y Proceso, son del módulo Situación Legal. Las entidades Ubicación y Cama pertenecen al módulo de Aseguramiento Logístico, para el perfecto funcionamiento del sistema también se utilizaron métodos e información de otros módulos como Seguridad y Solicitudes. En el diagrama de componentes presentado en el siguiente capítulo se podrá observar las relaciones del módulo Egreso con otros módulos pertenecientes a su mismo subsistema o a otros.

Ver Anexos

2.8 Diagramas de Secuencia

El diagrama de secuencia describe las interacciones entre un grupo de objetos mostrando de forma secuencial los envíos de mensajes entre objetos. El diagrama puede así mismo mostrar los flujos de datos intercambiados durante el envío de mensajes. Para interactuar entre sí, los objetos se envían mensajes. Durante la recepción de un mensaje, los objetos se vuelven activos y ejecutan el método del mismo nombre.

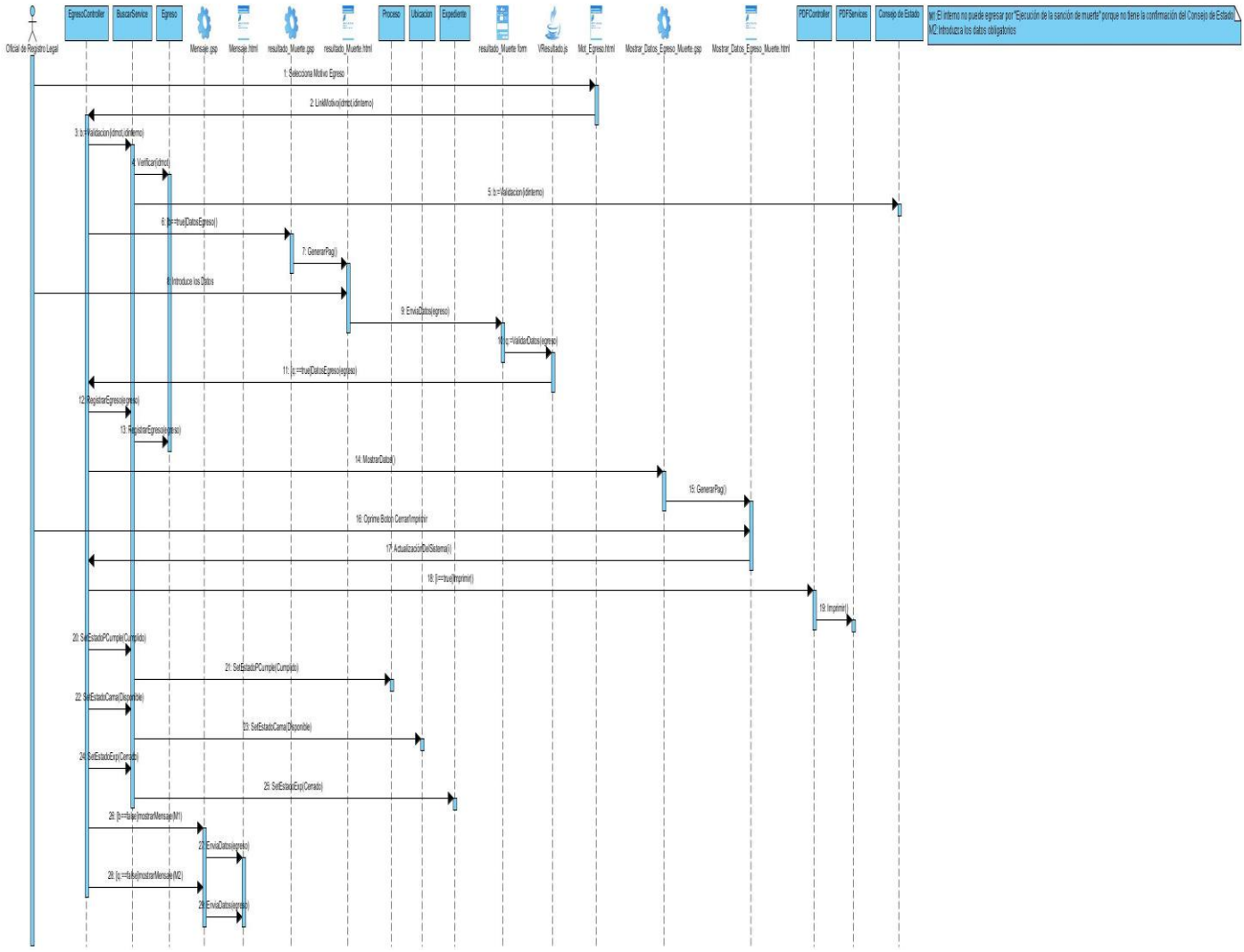


Figura 8 Diagrama de Secuencia "Registrar Egreso por Muerte"

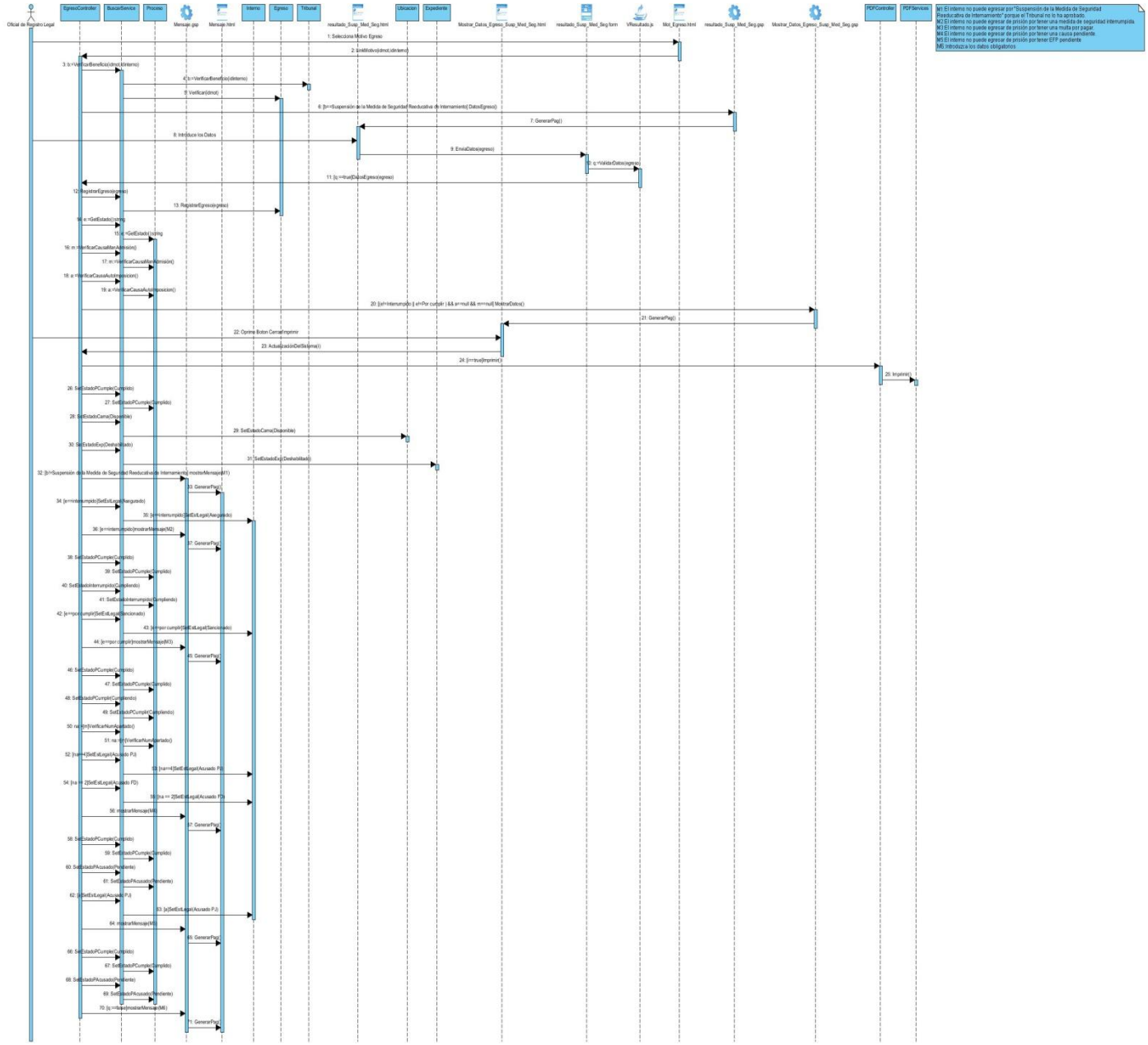


Figura 9 Diagrama de Secuencia "Registrar Egreso por Suspensión de la Medida de Seguridad Reeducativa de Internamiento"

[Ver Anexos]

2.9 Diseño de la Base de Datos

Las Bases de datos necesitan de una definición de su estructura que le permitan almacenar datos, reconocer el contenido, y recuperar la información. Es por ello que uno de los pasos cruciales en la construcción de una aplicación que maneje una base de datos, es sin duda, el diseño de la misma. Esto permite que las tablas sean definidas apropiadamente y se garantice que tenga eficiencia. Al diseñar una base de datos, se refleja la estructura del problema en el mundo real, evita el almacenamiento de información redundante y permite además representar todos los datos esperados, incluso con el paso del tiempo.

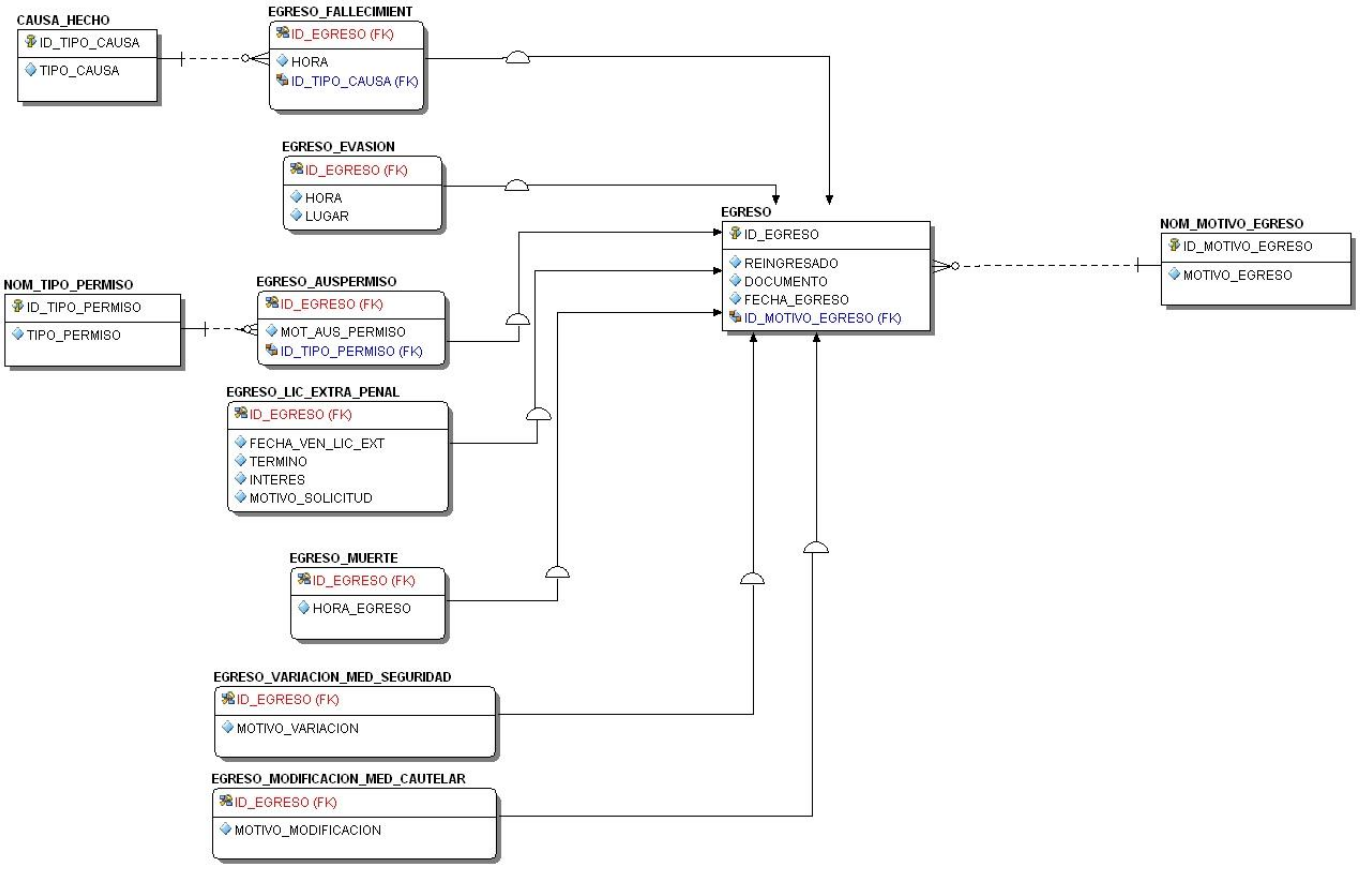


Figura 10 Diagrama Entidad Relación del Módulo de Egreso.

2.10 Descripción de las tablas

A continuación se muestra una breve descripción de las tablas Egreso y Pena de muerte. Con el objetivo de no sobrecargar el documento las restantes tablas serán mostradas en los anexos.

Columna	Tipo de Dato	Descripción
EGRESO_MUERTE_ID	NUMBER(19,0)	Identificador de Egreso
HORA_EGRESO	VARCHAR2(255 BYTE)	Cadena que define la hora de muerte
MOTIVO_MUERTE_ID	NUMBER(19,0)	Identificador del motivo de la muerte

Tabla Egreso 1: Pena de Muerte

Columna	Tipo de Dato	Descripción
EGRESO_ID	NUMBER(19,0)	Identificador del Egreso
DOCUMENTO	RAW	Arreglo de bytes que contiene un formato binario
FECHA_EGRESO	DATE	Fecha en que egreso un interno
MOTIVO_EGRESO_ID	NUMBER(19,0)	Motivo por el que egresa un interno
REINGRESADO	NUMBER(1,0)	Valor booleano que indica si es reingresado

Tabla Egreso 2: Egreso

Ver Anexos

2.11 Conclusiones

En el presente capítulo se analizaron los procesos de egreso y a partir de estos se diseñaron las clases del diseño y el modelo de datos. Para los diferentes tipos de egreso se describió su interacción a través de los diagramas de secuencia. También se hizo un estudio sobre el entorno de trabajo web y su arquitectura. La correcta utilización de los patrones de diseño como el MVC de Grails ayudó al desarrollo de la solución propuesta, formalizó un vocabulario común entre el equipo de desarrollo y proporcionó elementos reusables en el diseño del sistema de software. Se obtuvo como resultado un robusto diseño de clases que permitirá la posterior implementación y desarrollo del módulo Egreso del sistema SIDEP.

Capítulo 3: Implementación y Prueba

3.1 Introducción

En este capítulo se describen los elementos necesarios para la implementación, partiendo del resultado obtenido del diseño. Se muestra la distribución del sistema en nodos mediante el diagrama de despliegue y la organización de los componentes y las relaciones lógicas entre ellos a través del diagrama de componentes. A continuación se muestran algunas de las características tenidas en cuenta durante la fase de implementación.

3.2 Implementación

Seguridad

Con el objetivo de garantizar la seguridad de los datos y el correcto funcionamiento del módulo se implementaron validaciones de datos en ambos lados (cliente y servidor). En el lado del cliente la validación se hizo usando los componentes creados en el proyecto y JavaScript. Mientras que en el servidor se hizo uso de las restricciones que incluyen las clases de dominio para la validación de los datos antes de ser escritos en la base datos y otras funciones implementadas manualmente que se corresponden con las reglas del negocio. En el caso de ocurrencia de un error en el servidor o en las páginas clientes se le informa al usuario a través de mensajes. Los servicios hacen uso de transacciones para evitar la inserción parcial de datos o la perdida de los mismos. También se definió el acceso a los diferentes módulos del proyecto por roles.

Reutilización de código

Otro de los aspectos importantes de la implementación es la reutilización de código. Debido a que en el desarrollo de muchos casos de usos se hizo uso frecuente de funciones repetitivas, se creó el servicio *EgresoUtil* en el cual se agruparon las funciones más solicitadas por estos y se aplicaron para cada tipo de egreso. Una de las funcionalidades en la que se hizo aplico el principio de la reutilización de código fue **guardarEgreso,** la cual tiene como objetivo guardar un egreso. Inicialmente a este método se le pasaba como único parámetro el egreso a guardar en la base de datos y no se modificaba el expediente. Pero existen egresos los cuales modifican el expediente por lo cual fue necesario crear otra función que permitiera guardar el egreso y modificara el expediente. Se creó una función en la cual se le pasa el egreso y una variable booleana que es la que determina si se cambia el expediente. Con esta función se hizo posible guardar todos los tipos de egresos y se elimino la anterior función.

```
def guardarEgreso(Egreso egreso, boolean finalizarEgreso) {
    if (egreso.validate()) {
        if (egreso.save()) {
            Proceso p = this.getProcesoCumpliendo()
            if (p != null) {
                if (new EgresoProceso(proceso: p, egreso: egreso).save(flush: true)) {
                    try {
                        procesoUtilService.actualizarIngresos(egreso)
                    } finally {
                        this.finalizarEgreso(finalizarEgreso)
                        egresoService.idegreso = egreso.id
                        return true

                    }
                }
            } else {
                egresoService.addError("No tiene proceso cumpliendo")
                return false
            }
        }
    }
    else {
        println "Tiene errores"
        egreso.errors.each {
            println it
        }
        return false
    }
    return false
}
```

Figura 11 Implementación

Implementación de las clases del dominio

Las clases del dominio y sus relaciones fueron implementadas de acuerdo al diseño propuesto en el capítulo anterior. En el caso de la clase Egreso se hizo necesario que esta implementará las interfaces Serializable y Comparable porque en los casos de uso Consultar Egreso y Generar Reporte Previo se hicieron uso de listas de Egreso ordenadas. Se mapearon estáticamente todas las tablas de Egreso para acortar sus nombres debido a una restricción del gestor de base datos Oracle que limita a 30 caracteres el nombre de las tablas y si se permite que Grails nombre las tablas, al existir nombre de las mismas que sean compuestos estas serán nombradas añadiéndole el carácter “_” entre los nombres. Provocando que puedan surgir problemas al generar consultas entre las tablas. También se implementan métodos e interfaces que permitan el ordenamiento y la socialización de las clases del dominio.

```
class Egreso implements Serializable, Comparable {

    Date fechaEgreso
    NomMotivoEgreso motivoEgreso
    Boolean reingresado
    byte[] documento



    static constraints = {
        fechaEgreso blank: false
        motivoEgreso blank: false
        documento nullable: true, maxSize: 5*1024*1024
    }
    static mapping = {
        id column: "egreso_id"
        tablePerHierarchy false
    }

    String toString() {
        "Egreso realizado ${fechaEgreso}"
    }
```

Figura 12 Implementación

Implementación de la lógica del negocio

La lógica del negocio esta implementada orientada al trabajo en los servicios, quedando los controladores relegados a un papel secundario donde sirven como control de direccionamiento de las vistas y validación de los datos. En los servicios se desarrolla la mayor parte del negocio y cada tipo de egreso tiene uno. Todos los servicios de egreso implementan la interfaz **IRegistrarEgreso** la cual tiene la siguiente estructura:

```
public interface IRegistrarEgreso {
    def validarPrecondiciones(Map mapa)

    def validar(Map mapa)

    def save(Map mapa)
}
```

Figura 13 Implementación

Esta interfaz esta implementada para adaptarse a los casos de usos ya que estos se componen de tres aspectos básicos: validación de precondiciones, validación de los datos introducidos y persistencia o notificación de error de los datos.

Implementación de la interfaz de usuario

En la interfaz de usuario además de las etiquetas HTML y el marco de trabajo Dojo se utilizan las etiquetas Taglib las cuales permiten la generación de código dinámico HTML. En el caso de Egreso se hace uso de la etiqueta <*s:table*> en el muestreo de resultados de búsquedas y en la generación de reportes. Ejemplo:

primernombre	segundonombre	primerapellido	segundoapellido	nroidentidad	Motivo de Egreso	fechaEgreso	detalles
Ernesto		Villa		83010225453	Libertad condicional	2012-04-24 00:00:00	Ver detalles
MAria		Gutierrez		87111614011	Suspensión de la Medida de Seguridad Reeducativa de Internamiento	2012-04-24 00:00:00	Ver detalles
Acusado Ernesto	Olivia	Benitez		87050505051	Absueltos	2012-05-02 00:00:00	Ver detalles
Carla		Primera		44042444444	Sentencia absolutoria dictada en proceso de revisión	2012-05-10 00:00:00	Ver detalles
Solid	Solid	Snake	Snake	88070432980	Licencia extrapenal	2012-03-20 00:00:00	Ver detalles

Page 6 of 9 Showing 26 - 30 of 42

To Consultar

Figura 14 Implementación

También se utilizan otras etiquetas como las siguientes:

- Etiqueta <*s:title*> Crea una sección de título en la página
- Etiqueta <*s:dateBox*> Proporciona un selector para la fecha
- Etiqueta <*s:button*> Crea un botón
- Etiqueta <*s:buttonBar*> Permite la organización de los botones en la página

Una de las particularidades de la implementación de las vistas es que la programación de las acciones de los botones y la vista están separadas. También en todas las vistas de egresos usaron de plantillas de código que no son más que una etiqueta definida por Grails que permite el uso código HTML anidado en una misma página. Ejemplo:

```
<body>
<form action="${createLink(controller: "pagoMultaEgreso", action: "registrar")}" name="egres
      id="egreso.pagoMulta.form">
    <s:title title="Motivo:Pago de multa"/>
    <br/>
    <g:render template="/registrolegal/egreso/egreso"/>
    <s:file name="egreso.documento" label="Mandamiento de libertad"/>

    <br/>
    <hr/>
    <s:clear/>
    <s:buttonBar><s:button label="Cerrar" id="egreso.pagoMulta.aceptar.id"/></s:buttonBar>
    <s:clear/>
</form>
```

Figura 15 Implementación

3.3 Diagrama de Componentes

El diagrama de componentes ilustra los componentes del software que serán usados para conformar el sistema. Se utilizan para modelar la vista estática del sistema y muestra la organización y las dependencias lógicas entre los componentes software, sean éstos componentes de código fuente, librerías, binarios o ejecutables. “Los componente son paquetes estereotipados en subsistemas. Estos organizan la vista de realización de un sistema y pueden contener componentes y otros subsistemas.” (19)

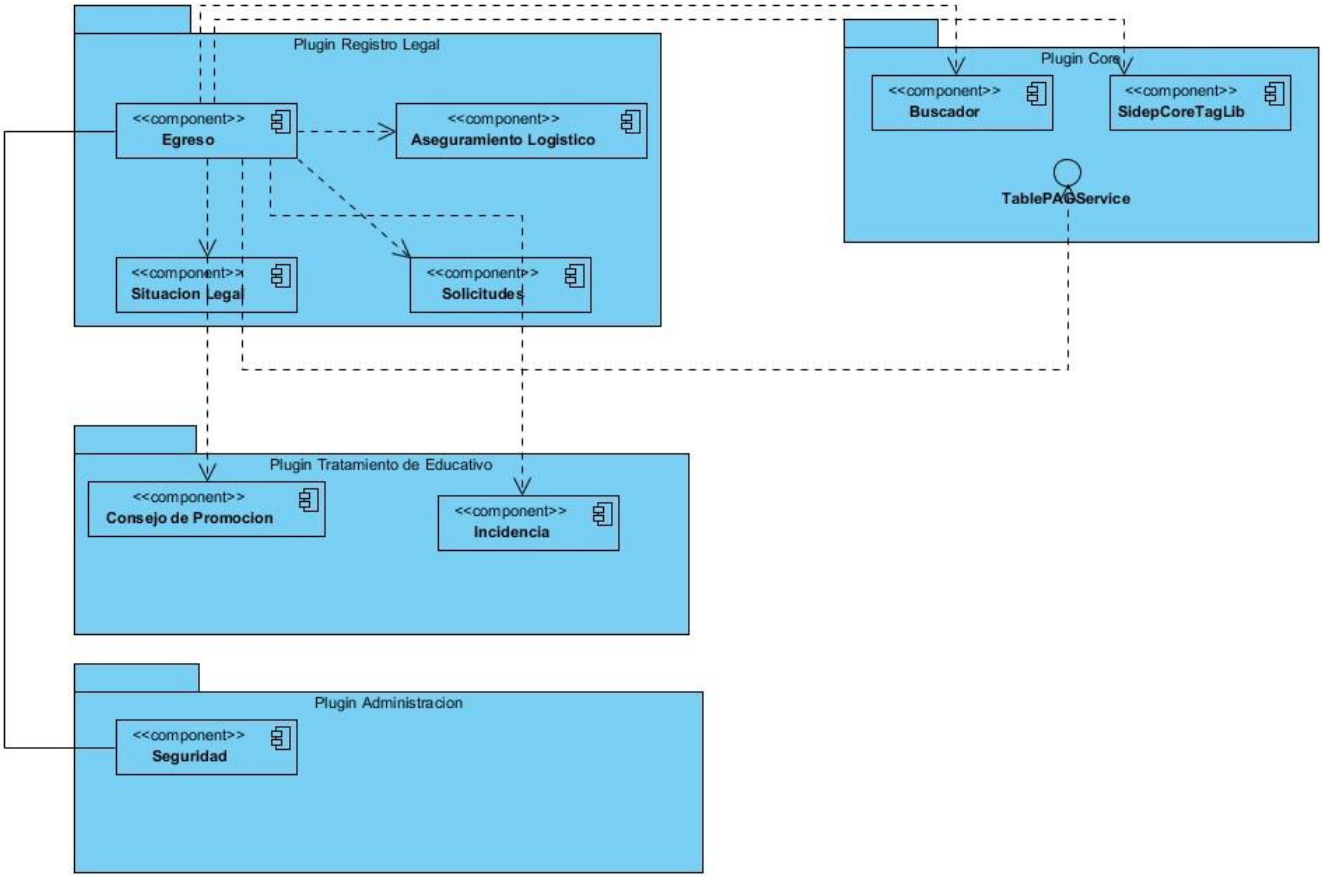


Figura 16 Diagrama de Componentes de las Relaciones generales con Egreso

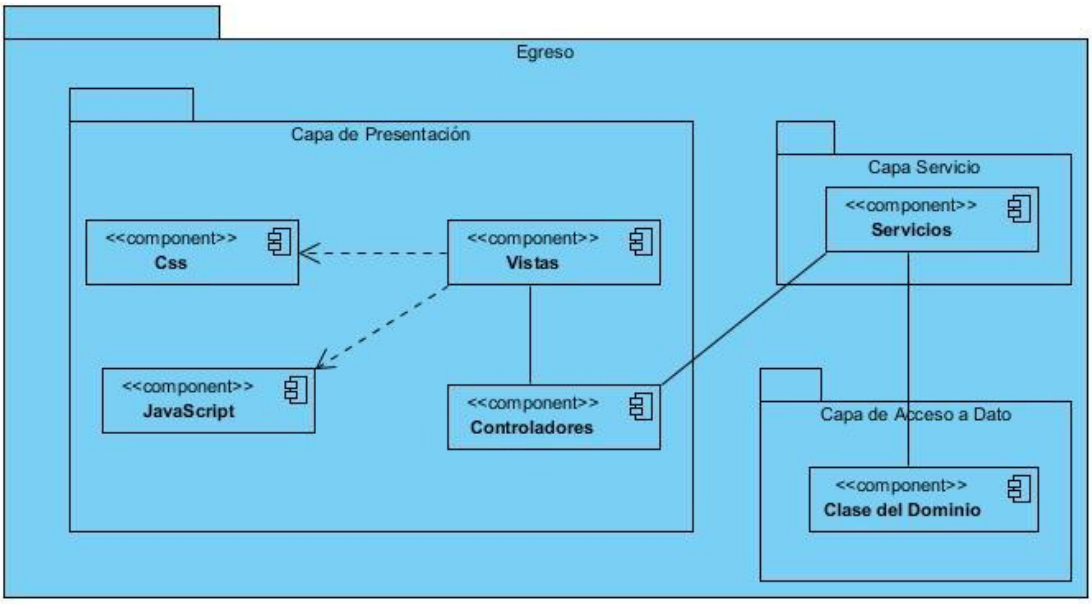


Figura 17 Diagrama de Componentes de las Relaciones internas de Egreso

3.4 Prueba del sistema propuesto

El único instrumento adecuado para determinar el status de la calidad de un producto software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos.

El objetivo principal de las pruebas realizadas al módulo de Egreso es encontrar el mayor número de errores en la implementación y de probar todas las funcionalidades descritas en los casos de uso. El método de prueba empleado fue el de Caja Negra, en el cual se llevan a cabo las pruebas de la interfaz del software. Esta prueba examina algunos aspectos del modelo fundamentalmente del sistema sin tener en cuenta la estructura interna del software.

Figura 18 Método de Prueba de Caja Negra

La técnica utilizada para la realización de las pruebas fue Particiones Equivalentes, la cual emplea los Casos de Prueba. Los Casos de Prueba son el conjunto de entradas con datos de prueba, unas condiciones de ejecución, y unos resultados esperados con el propósito de identificar y comunicar las condiciones que se llevarán a cabo en la prueba.

Id del escenario	Escenario	Fecha de egreso	Hora de muerte	Respuesta del Sistema
EC 1	Escenario 1: "Registro exitoso"	V 09/05/2012	V 17:53	El sistema muestra un mensaje informándole al funcionario que ya ha sido efectuado exitosamente el egreso del interno.
EC 2	Escenario 2: "Faltan datos obligatorios"	I V 03/07/2012	V 20:11 I	El sistema emite un mensaje para que se complete los campos obligatorios y señala cuales no fueron introducidos.

		I	I	
EC 3	Escenario 3: "No se puede dar egreso por muerte"	V 09/05/2012	V 17:53	El sistema muestra un mensaje en el cual informa que no se pudo lograr el egreso porque el interno no tiene la confirmación del Consejo de Estado.

Matriz de Casos de Prueba 1: Registrar Egreso por Muerte

Id del escenario	Escenario	Fecha de egreso	Respuesta del Sistema
EC 1	Escenario 1: "Registro exitoso"	V 09/05/2012	El sistema muestra un mensaje informándole al funcionario que ya ha sido efectuado exitosamente el egreso del interno.
EC 2	Escenario 2: "Faltan datos obligatorios"	I	El sistema emite un mensaje para que se complete los campos obligatorios y señala cuales no fueron introducidos.
EC3	Escenario 3: "No se puede dar egreso por variación"	V 09/05/2012	El sistema muestra un mensaje informando al funcionario que no se pudo lograr el egreso porque Tribunal no lo ha aprobado.

EC4	Escenario 4:" Interno con proceso interrumpido"	V 09/05/2012	El sistema muestra un mensaje donde informa que el interno no puede egresar de prisión por tener una medida de seguridad interrumpida y no permite generar el modelo de egreso.
EC5	Escenario 5: "Interno con proceso por cumplir"	V 09/05/2012	El sistema muestra un mensaje donde informa que el interno no puede egresar de prisión por tener una multa por pagar y no permite generar el modelo de egreso.
EC6	Escenario 6:" Pasar a ser acusado del Tribunal"	V 09/05/2012	El sistema emite un mensaje en el cual informa que el interno no puede egresar de prisión por tener una causa pendiente y no permite generar el modelo de egreso.
EC7	Escenario 7:" Pasar a ser acusado de Fiscalía"	V 09/05/2012	El sistema emite un mensaje en el cual informa que el interno no puede egresar de prisión por tener EFP pendiente y no permite generar el modelo de egreso.

Matriz de Casos de Prueba 2: Registrar Egreso por Suspensión de la Medida de Seguridad Reeducativa de Internamiento

Ya diseñados los casos de prueba faltaría probar contra la aplicación, y verificar si se cumplieron los requisitos establecidos en los casos de usos. El nivel de las pruebas a realizar en este caso fue el de desarrollador, el cual indica los aspectos de diseño e implementación de las pruebas más adecuadas que debe llevar a cabo el equipo de desarrolladores.

Los elementos que se tuvieron en cuenta para la satisfacción de las pruebas fueron:

- Interfaz sencilla.
- Cumplimiento de las funcionalidades presentadas.
- Rapidez en la obtención de la información.

Para el encuentro de defectos y dificultades se hizo indispensable realizar tres iteraciones de pruebas, en las que a medida que se pasaba de una a otra se, iban corrigiendo los errores hallados. A continuación se le muestra un resumen de las 43 deficiencias encontradas en la primera iteración de las pruebas.

No	No Conformidad	Ubicación	Etapa de detección	Clasificación	Respuesta del Equipo Desarrollo
4	Cambiar el mensaje: ”Faltan datos obligatorios“ por: “Los campos en rojo son requeridos”	Aplicación/(En todas las interfaces)	Iteración 1	Recomendación	Resuelta
8	Mensaje de error de incidencia en evasión: dice de fallecimiento, cambiar por evasión	Aplicación/ Interfaz de Evasión	Iteración 1	Significativo	Resuelta
11	Todos los botones para finalizar el egreso se llaman Cerrar	Aplicación/(En todas las interfaces)	Iteración 1	Recomendación	Resuelta
15	En el motivo de egreso Sentencia absolutoria dictada en	Aplicación/ Interfaz de Sentencia absolutoria dictada en	Iteración 1	Significativo	Resuelta

	proceso de revisión, cuando se van a registrar los datos muestra el motivo de egreso: Suspensión de la Ejecución de la Sanción	proceso de revisión			
22	Poner en los resultados de la consulta del reporte previo, una columna de la tabla que sea Nombre (s) y apellidos	Aplicación/ Interfaz de Reporte previo	Iteración 1	Recomendación	Resuelta
26	Cambiar mensaje de cuándo se va a egresar por variación de la medida reeducativa de internamiento que dice suspensión en vez de variación	Aplicación/ Interfaz de Variación de la medida reeducativa de internamiento	Iteración 1	Significativo	Resuelta
28	Error de validación del egreso por Prescripción penal	Aplicación/ Interfaz de Prescripción penal	Iteración 1	Significativo	Resuelta
32	Falta el motivo de egreso de Retroactividad de la ley en el nomenclador	Aplicación/ Interfaz de Retroactividad de la ley	Iteración 1	Significativo	Resuelta
36	Falta el motivo de egreso de Revisión de la	Aplicación/ Interfaz de Revisión de la	Iteración 1	Significativo	Resuelta

	causa del tribunal supremo en el nomenclador	causa del tribunal supremo			
40	Poner en los resultados del consultar egresos, una columna de la tabla que sea Nombre (s) y apellidos	Aplicación/ Interfaz de Consultar Egreso	Iteración 1	Recomendación	Resuelta

Dificultades detectadas en las pruebas

3.5 Diagrama de Despliegue

El diagrama de despliegue describe como una aplicación se despliega a través de una infraestructura. Muestra la disposición física de los distintos nodos que componen el sistema y el reparto de los componentes sobre dichos nodos, los cuales representan los objetos físicos con recursos computacionales que generalmente tienen algo de memoria y, a menudo, capacidad de almacenamiento. Las conexiones establecidas son asociaciones de comunicación entre los nodos, que no son más que el protocolo de comunicación o la red utilizada.

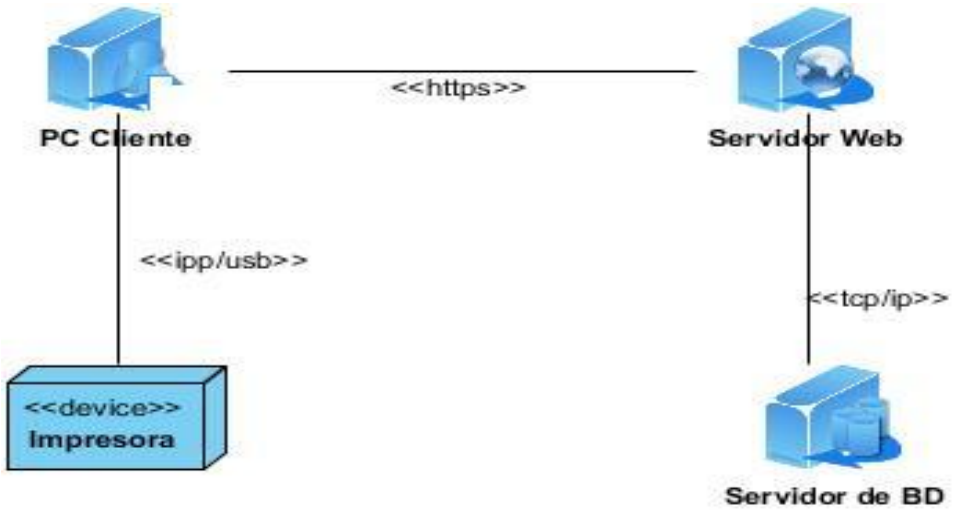


Figura 19 Diagrama de Despliegue

Para el despliegue del módulo es necesario:

Puestos de trabajo (PC usuario)

- RAM: 512 MB
- Navegador web Mozilla Firefox 3.6.*

Servidor de la aplicación

- Microprocesador: 4 núcleos, 3 GHz
- RAM: 4 GB
- Espacio necesario para la instalación: 250 MB
- Espacio libre: 250 GB
- JDK 1.6
- Apache Tomcat 6.0.2

Servidor de Bases de datos

- Microprocesador: 4 núcleos, 3 GHz

- RAM: 4 GB
- Espacio necesario para la instalación: 2 GB
- Espacio libre: 1 TB
- JDK 1.6
- Oracle 11g

Tipos de Cables Requeridos y Puertos de Inicialización:

Para la conexión entre el Servidor Web y el de Base de Datos es preferible que el tipo de cable sea Fibra Óptica ya que garantiza el potencial necesario para transportar la información por su gran ancho de banda y en cuanto a la seguridad es extremadamente difícil intervenir una fibra, y virtualmente imposible hacer la intervención indetectable, porque se nota en el debilitamiento de la energía luminosa en recepción. De no ser posible la Fibra Óptica, entonces el cable pudiera ser UTP (Cat4 | Cat5), en cualquiera de los dos caso, el protocolo de comunicación será TCP/IP y el puerto de inicialización será 1521.En el caso del tipo de cable a utilizar entre el Servidor Web y la PC Cliente seria UTP (Cat4 | Cat5) con HTTPS como protocolo de comunicación y con 443 como puerto de inicialización.

3.6 Conclusiones

En este capítulo se presentaron los elementos más significativos en la implementación del módulo Egreso. Se generaron los artefactos necesarios para la realización de pruebas del sistema. Se diseñaron los casos de prueba y se aplicaron en las diferentes iteraciones de la fase de prueba con el objetivo de garantizar la calidad y el correcto funcionamiento. Se detallaron las características del despliegue del sistema final. Al finalizar este capítulo ya se tiene la solución construida, probada y funcionando correctamente.

Conclusiones Generales

En la confección del presente trabajo de diploma se le dio cumplimiento al objetivo general planteado contribuyendo al mejoramiento del módulo Egreso del SIDEP. Se hizo un estudio de las herramientas y tecnologías utilizadas en el diseño e implementación del trabajo. Se logra reducir las demoras en los procesos y trámites. Durante el proceso de egreso se tienen en cuenta los beneficios otorgados al interno. Se logró una correcta identificación los procesos de egreso de cada uno de los motivos de egreso. Las actividades de diseño e implementación fueron mostradas a través de las relaciones de los diagramas de clases y componentes. Con el proceso de prueba realizado se garantiza que el producto tiene una gran calidad y funciona correctamente ya que cumplió satisfactoriamente con todos los parámetros definidos.

Recomendaciones

Se recomienda la realización de un monitoreo constante del rendimiento del módulo en la medida en que la cantidad de datos que el sistema maneja vaya creciendo. Debido a la amplitud de las funcionalidades de Egreso hacer extensiva la experiencia a otros proyectos productivos que tengan que ver con el desarrollo de aplicaciones de Gestión Penitenciarias.

Bibliografía

1. Lorenzo, Yanay Viera. Glosario de Término. 2010.

2. Zaragoza, María de Lourdes Santiago. Universidad Tecnologica del Valle de Mezquital. [En línea] [Citado el: 12 de 2 de 2012.] http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm.

3. NetBeans.org. *NetBeans.org.* [En línea] Sun Microsistems. [Citado el: 2 de Abril de 2012.] http://netbeans.org/index_es.html.

4. Visual Paradigm International. Visual Paradigm. *Visual Paradigm.* [En línea] [Citado el: 14 de enero de 2010.] http://www.visual-paradigm.com/support/documents/vpumluserguide/12/13/5963_aboutvisualp.html.

5. CASE, Computación Avanzada y Sistemas Empresariales SA de CV. [En línea] 2012. [Citado el: 5 de Abril de 2012.] http://www.casenet.com.mx/index.php?option=com_content&view=article&id=58&Itemid=59.

6. SourceForge. SourceForge. [En línea] 2012. [Citado el: 22 de Abril de 2012.] http://sourceforge.net/projects/tortoisesvn/.

7. Tomcat WebSite. [En línea] 2012. [Citado el: 15 de Abril de 2012.] http://tomcat.apache.org/.

8. CiberAula. *CiberAula.* [En línea] 2010. [Citado el: 4 de Junio de 2012.] http://java.ciberaula.com/articulo/que_es_java.

9. Developer Mozilla. [En línea] 2012. [Citado el: 14 de 2 de 2012.] https://developer.mozilla.org/es/JavaScript/Acerca_de_JavaScript.

10. Groovy, Escuela de. Escuela de Groovy. [En línea] 2012. [Citado el: 2 de Mayo de 2012.] http://www.escueladegroovy.com/informacion/groovy-grails.

11. Pérez, Javier Eguíluz. Sunshine. *Sunshine.* [En línea] [Citado el: 2012 de Junio de 5.] http://sunshine.prod.uci.cu/gridfs/sunshine/books/introduccion_ajax_2caras.pdf.

12. —. Libros.es. [En línea] [Citado el: 20 de 3 de 2012.] http://www.librosweb.es/ajax.

13. Epidata Consulting. *Epidata Consulting.* [En línea] [Citado el: 4 de Junio de 2012.] http://www.epidataconsulting.com/tikiwiki/tiki-pagehistory.php?page=Hibernate&diff2=21&diff_style=sideview.

14. The Dojo Foundation. DojoTolkit. *DojoTolkit.* [En línea] http://www.dojotoolkit.org/.

15. Oracle. Sitio Web Java. [En línea] [Citado el: 22 de Noviembre de 2011.] http://www.java.com/es/download/faq/helpful_concepts.xml.

16. Bertolami, Leandro. [En línea] 2011. [Citado el: 10 de 2 de 2012.] http://www.1024.com.uy/revista/index.php/galileo-galilei/110-grails-el-santo-grial-de-java.

17. Erika Camacho, Fabio Cardeso y Gabriel Núñez. *Arquitecturas de software.* 2004.

18. Brito, Nacho. *Manual de Desarrollo Web con GRAILS.* 2009. ISBN: 978-84-613-2651.

19. Larman, Craig. *UML y Patrones. Introducción al análisis y diseño orientado a objeto.* México : s.n., 1999.

20. Kauffman, Morgan. *Modelado de Sistemas con UML.* 2011.

21. Marblestation. *Marblestation.* [En línea] http://www.marblestation.com/?p=644.

22. JSON. Java Script Object Notation. [En línea] http://www.json.org/json-es.html.

23. The Dojo Foundation. DojoTolkit. [En línea] http://www.dojotoolkit.org/.

24. López, Diego Jiménez. Sunshine. [En línea] Octubre de 2009. [Citado el: 15 de Marzo de 2012.] http://sunshine.prod.uci.cu/gridfs/sunshine/books/Buscador_Semntico_sobre_una_Red_Social_de_preguntas_de_examen.pdf.

25. Definicion.de. Definicion.de. [En línea] 2012. [Citado el: 25 de Abril de 2012.] http://definicion.de/sql/.

26. Pimentel, Iosev Pérez y Luis Alberto. *Arquitectura base sobre la WEB.* UCI : s.n., 2007.

27. Osorio, Manuel. *Diccionario de Ciencias Jurídicas, Políticas y Sociales.* s.l. : Heliasta, 1997.

28. Gómez, Yadira Benavides y Juan Carlos. *Diseño e implementación de los módulos Decisiones y Egresos del Sistema Gestión Penitenciaria de la República Bolivariana de Venezuela.* Ciudad de la Habana: UCI : s.n., 2008.

29. Escalona, Nora Koch y María José. *Ingeniería de Requisitos en Aplicaciones para la Web-Un estudio comparativo.* 2002.

30. Pressman, Roger. *Ingeniería del Software, un enfoque práctico.* 2002.

31. Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides. *GOF.Design Patterns: Elements of Reusable Object-Oriented Software.* s.l. : Addison-Wesley, 1995.

32. Jesús García Molina, Ana Moreira y Gustavo Rossi. *UML: el lenguaje estándar para el modelado de software.* s.l. : Ediciones SA, 2004.

33. Schnuller, Joseph. *Aprendiendo UML en 24 horas.* 2000.

34. Buschmann, Frank. *Pattern-Oriented Software Architecture.* s.l. : Wiley, 2001.

35. Cornejo, Jóse Enrique González. Arquitectura en Capas. Un camino hacia los procesos distribuidos. [En línea] 2001. http://www.docirs.cl/arquitectura_tres_capas.htm.

36. Méndez, Néstor Darío Duque. Conceptos de arquitectura Cliente/Servidor. [En línea] 2008. http://www.it.uc3m.es/mcfp/docencia/si/material/1_cli-ser_mcfp.pdf..

37. Patrón MVC. [En línea] 2008. http://www.proactiva-calidad.com/java/patrones/mvc.htm..

38. JAVA, Sitio Oficial. Sitio Oficial JAVA. [En línea] http://java.com/es..

39. Library, Oracle Database Documentation. Oracle Database Documentation Library. 2005.

40. Crawford, W. & Kaplan, J. J2EE Design Patterns. California. s.l. : O'Reilly & Associates, 2003.

41. Susana, Hugo Michael y Nancy. *Diagrama de Despliegue.* 2003.

42. Software., Gestión de Calidad y Pruebas de. Gestión de Calidad y Pruebas de Software. [En línea] 2005. http://www.pruebasdesoftware.com/.

43. Cuba Minrex. *Cuba Minrex.* [En línea] [Citado el: 4 de Junio de 2012.] http://www.cubaminrex.cu/CDH/61cdh/Derechos%20Humanos%20en%20Cuba/Sistema%20Penitenciario.htm.

44. Batista, Yanet Del Risco. Proyecto Tecnico. *Proyecto Tecnico.* 2012.

45. NetBeans.org. [En línea] [Citado el: 2 de Abril de 2012.] http://www.netbeans.org.

46. Lorenzo, Yanay Viera. *Entrevista sobre funcionamiento del Sistema Penitenciario.* [entrev.] Reinier Alamo Quesada.

Glosario de términos

Evasión: Acción de fuga del lugar de internamiento.

Acusado: la persona a quien se le haya decretado la medida cautelar de prisión provisional.

Sancionado: la persona ejecutoriamente sancionada a privación de libertad o a Trabajo Correccional Con Internamiento (TCCI).

Asegurado: la persona a quien se le hubiere impuesto una medida de seguridad reeducativa de internamiento.

ORACLE: Sistema gestor de base de datos utilizado para almacenar los datos manejados por la aplicación.

UML: Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido en la actualidad.

RUP: Proceso unificado de desarrollo de software. Metodología utilizada para elaborar la documentación del software.

EFP: Expediente de Fase Preparatoria.

Acusado FD: Acusado Falta de documento.

Acusado PJ: Acusado Pendiente a Juicio.

TCCI: Trabajo Correccional con Internamiento.

TCSI: Trabajo Correccional sin Internamiento.

DNI: Dirección Nacional de Identificación.

MINED: Ministerio de Educación.

INDER: Instituto Nacional de Deportes, Educación Física y Recreación.

FMC: Federación de Mujeres Cubanas.

MINFAR: Ministerio de las Fuerzas Armadas Revolucionarias.

TLP: Trámite Legal Pendiente.
MVC: Patrón de diseño Modelo Vista Controlador.

USB (Universal Serial Bus): es un estándar industrial desarrollado en los años 1990 que define los cables, conectores y protocolos usados en un bus para conectar y comunicar ordenadores, periféricos y dispositivos electrónicos. Está diseñado para estandarizar la conexión de periféricos, como mouse, teclados, escáneres, cámaras digitales, teléfonos móviles, reproductores multimedia, impresoras y muchos otros más.

UTP (Unshielded Twisted Pair- Par trenzado no blindado): es un tipo de cable que no está blindado y que suele utilizarse en las telecomunicaciones.

IPP (Internet Printing Protocol): define un protocolo de impresión y gestión de los trabajos a imprimir, el tamaño del medio, la resolución, etc. Soporta el control de acceso, la autenticación y el cifrado, siendo así una solución de impresión más capaz y segura que otras más antiguas.

HTTP (Hypertext Transfer Protocol - Protocolo de transferencia de hipertexto): es el protocolo usado en cada transacción de la Web. Define la sintaxis y la semántica que utilizan los elementos software de la arquitectura web (clientes, servidores, proxies) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

HTTPS (Hypertext Transfer Protocol Secure- Protocolo seguro de transferencia de hipertexto): es un protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto, es decir, es la versión segura de HTTP.

TCP/IP (Transmission Control Protocol/Internet Protocol): describe un conjunto de guías generales de diseño e implementación de protocolos de red específicos para permitir que un equipo pueda comunicarse en una red.

Cat 4 (Cable de Categoría 4): es una descripción no estandarizada de cable que consiste en 4 cables UTP con una velocidad de datos de 16 Mbit/s y un rendimiento de hasta 20 MHz.

Cat 5(Cable de Categoría 5): es uno de los grados de cableado UTP descritos en el estándar EIA/TIA 568B, puede transmitir datos a velocidades de hasta 10000 Mbps a frecuencias de hasta 100 MHz.