Universidad de las Ciencias Informáticas

Facultad 2

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

Título: Desarrollo del módulo Gestión del Consejo de Promoción perteneciente al Sistema Informativo de la Dirección de Establecimientos Penitenciarios (SIDEP).

Autor: Javier Eloy Aparicio Vivero

Tutor: Ing. Roberto Granda Ruíz

Co-Tutor: Ing. Anabel Betancourt de los Santos

La Habana, Junio 2012

"Es poco el sacrificio que hagamos para el bien que conquistaremos."

José Luis Tassende

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.

Javier Eloy Aparicio Vivero

_____

Firma del Autor

Ing. Roberto Granda Ruíz

_____

Firma del Tutor

Ing. Roberto Granda Ruiz

- Graduado en el 2010 de la carrera Ingeniería en Ciencias Informáticas de la Universidad de las Ciencias Informáticas.
- Se vinculó a la Universidad como profesor en septiembre del 2010.
- Ha estado vinculado al proyecto productivo SIGEP desde el 2007 en el rol de programador de interfaz usuario.
- En Octubre del 2011 comenzó a trabajar en el proyecto productivo SIDEP.

Agradecimientos

A mi familia por brindarme su apoyo en todo momento, por confiar en mis posibilidades en ocasiones en las que yo mismo dudé.

A mi mamá por su incondicionalidad, preocupación y por transmitirme su optimismo cuando lo he necesitado.

A mi papá por motivarme a mejorar día a día y por sus consejos de aliento en momentos difíciles.

A mi hermana por estar siempre al tanto de lo que me sucede y por confiar siempre en mi.

A Rafael por su preocupación y por ayudarme siempre que lo he necesitado.

A mi tutor, "El Robe", por estar siempre dispuesto a ayudarme ante cualquier problema, por su preocupación, por su empeño en pos de la realización de este trabajo. El resultado obtenido es tan tuyo como mío. Muchas gracias por incitarme a investigar con el objetivo de ser un mejor ingeniero.

A mis socios René, Juanky, Ernesto (La máquina), Dariel, Damián, Leonel, Elián, Lea, Rey, Daryl y demás socios del proye y del apartamento por ayudarme cuando los he necesitado y por todos los buenos momentos que hemos compartido desde un play de fútbol hasta el medall.

A Yaidel por ayudarme siempre sin poner pretextos.

A mis niñas del aula Zulay, Yudy y Yudith por ser siempre tan delicadas y cariñosas conmigo.

A todos mis compañeros de estudio de los distintos grupos con los que he compartido.

Al MININT por darme la oportunidad de estudiar lo que siempre quise y en lo que a mí respecta en la mejor universidad del país.

A mi mamá y a mi hermana porque juntas han luchado contra todos los obstáculos que les ha impuesto la vida con el objetivo de sacarme adelante y de hacer de mí un hombre de bien.

A mi papá por ser mi ejemplo de responsabilidad y superación.

A mi abuela Amelia que siempre se enorgullecía de mis logros, estoy seguro de que esta sería otra de las ocasiones en las que con un caluroso abrazo me arroparías en señal de felicidad y orgullo.

A mi abuelo Eloy, por lo mucho que lo admiro y por ser de las personas más inteligentes que conozco.

Resumen

Con el próspero desarrollo de las tecnologías y el gran cúmulo de información que se genera actualmente en las entidades cubanas, estas han tratado de nutrirse del uso de la informática para su desarrollo interno y el mejoramiento de los procesos que se gestionan en las mismas.

La Dirección de Establecimientos Penitenciarios (DEP) en colaboración con la Universidad de las Ciencias Informáticas (UCI) ha encaminado sus esfuerzos a informatizar sus procesos. De este trabajo conjunto se obtendrá una herramienta que permita la toma de decisiones, así como el funcionamiento óptimo de los procesos en los centros penitenciarios, dando surgimiento al Sistema de la Dirección de Establecimientos Penitenciarios (SIDEP).

El presente trabajo de diploma tiene como objetivo el desarrollo del módulo Consejo de Promoción perteneciente al subsistema Tratamiento Educativo que forma parte del SIDEP. Para el desarrollo del módulo se tuvo en cuenta el cumplimiento de los requerimientos de software establecidos y priorizados en etapas precedentes del proyecto, así como el uso de las tecnologías y herramientas definidas en la arquitectura del proyecto.

La solución final de dicho módulo tiene como propósito centralizar la información referente al individuo privado de libertad. La tarea de aglutinar las particularidades de determinado individuo a lo largo de su tránsito por los centros penitenciarios será altamente valorada por los oficiales ya que podrán tener conocimiento de la conducta pasada del interno, así como el progreso actual del mismo.

Palabras Clave: consejo de promoción, herramientas, requerimientos de software, SIDEP, tecnologías.

Índice

Índice de Tablas

Índice de Figuras

Introducción

El vertiginoso desarrollo científico-técnico del mundo actual está consiguiendo hacer realidad las fantasías de hace sólo unas décadas, con un extraordinario potencial para la satisfacción de las necesidades humanas. En Cuba el progreso tecnológico en sus inicios estuvo afectado no solo por el hecho de tener que recurrir a mercados internacionales sino porque no existía una sólida base nacional de ciencia y tecnología.

El desarrollo tecnológico se caracterizaba por la ausencia en el sector estatal de equipos encargados del tratamiento de la información y por la baja calificación del personal. La ausencia en los centros penitenciarios de un sistema gestor de los procesos de control penal traía consigo que el trabajo fuera sumamente complejo. Al triunfar la Revolución se heredó un sistema penitenciario que se caracterizó por la corrupción judicial y administrativa, el crimen despiadado y la discriminación racial y social. El desarrollo del privado de libertad en este entorno culminaba con el detrimento de su integridad y dignidad humana. El gobierno cubano acometió contra estos males realizando un conjunto de transformaciones que contribuyeran al mejoramiento de la condición humana y conducta social de los internos.

La informatización del sistema penitenciario cubano, constituye una de las principales iniciativas por parte de la dirección del país en pos de agilizar los procesos que se realizan en un centro penitenciario. Tiene sus inicios en el año 1989 con el decreto orden 43/99 del Viceministro Primero (VMP) del Ministerio del Interior (MININT), teniendo como objetivo la automatización de los datos principales del recluso y ciertos aspectos de control penal. Este proceso tuvo como primera tarea el diseño e implementación del Sistema Automatizado para el Control del Recluso (SACORE), para centralizar parte de la información relacionada con los internos. (1)

Con motivo del 50 aniversario del MININT surge el plan 20 x 50, que impulsa la modernización tecnológica y la colaboración con las Universidades. En dicho plan se decide por parte de la jefatura del MININT desarrollar en conjunto con la UCI un proyecto productivo con el propósito de informatizar toda la gama de procesos que se controlan en una unidad penitenciaria, de ahí el surgimiento del SIDEP.

En el año 2008, a partir del surgimiento de los programas educativos en los centros penitenciarios, el órgano gestor encargado del otorgamiento de los beneficios dejó de llamarse Consejo de Beneficios para ser lo que hoy se conoce como Consejo de Promoción. La inserción de los programas educativos

enfocados en los internos tiene como objetivo la disminución del rigor penitenciario, lograr que el interno adopte una actitud correctiva con respecto al delito cometido, por último, reintegrar una persona a la sociedad, capaz de adaptarse y convivir regido por códigos de conducta acorde a las leyes del país.

El Consejo de Promoción es el órgano gestor del otorgamiento de beneficios, en el cual se desarrollan varios de los procesos más complejos que se llevan a cabo en los establecimientos penitenciarios, debido a que aglutina un gran número de factores e indicadores resultantes de la valoración de varios ambientes de trabajo, como por ejemplo, el análisis de la personalidad, la conducta dentro del centro y la actitud ante los programas educativos; dichos criterios son emitidos por parte del re-educador, el oficial interno y el profesor respectivamente.

El proceso de otorgamiento de beneficios encuentra como principal problema el correcto manejo de las evaluaciones cualitativas y cuantitativas:

Evaluaciones cualitativas: resaltan las valoraciones con respecto a la personalidad y el comportamiento del interno por parte de los profesionales que lo atienden.

Evaluaciones cuantitativas: controlan los períodos de sanción teniendo en cuenta cada incidencia, para de esta forma calcular cuándo un interno puede obtener determinado beneficio.

La actual solución informática (SACORE) presenta como principales problemas que afectan el correcto otorgamiento de beneficios en el sistema de gestión penitenciaria:

- En ocasiones coinciden los períodos para recibir dos beneficios sin embargo son tratados como un único beneficio, atendiendo un total de nueve beneficios.
- El proceso de elevar la decisión de un beneficio para ser evaluado según el nivel de aprobación no está implementado. Por ejemplo todas las rebajas de sanciones son analizadas única y exclusivamente por el Jefe de Unidad y el actual sistema no gestiona el proceso de notificación al nivel de aprobación correspondiente.
- No existe soporte para la obtención de los reportes emitidos por los distintos órganos, por ejemplo, para consultar las incidencias asociadas al interno y la opinión fundamentada emitida por el equipo multidisciplinario.

Las evaluaciones anteriormente mencionadas, varían progresivamente según el número de indisciplinas cometidas por el interno y en conjunto con los problemas que presenta el SACORE, afectan los procesos

de gestión del consejo de promoción. El manejo incorrecto de estas evaluaciones traería consigo que no se evalúe correctamente al interno, al no disponer de los criterios emitidos con respecto a su proceder en el centro penitenciario no se tendrá en cuenta si el comportamiento del privado de libertad es adecuado, por ejemplo, mostrando interés en el programa educativo del centro. Por último si no se controlan los períodos en los que el interno debe recibir un beneficio se estarían violando sus derechos propiciando su inconformidad y la mala actitud para afrontar las tareas.

El objetivo principal de la automatización del Consejo de Promoción es recopilar todos los criterios emitidos por el educador guía y el oficial del equipo multidisciplinario, los cuales son imprescindibles para que el gestor del consejo valore el comportamiento del interno y de esta forma, llegar a una decisión final referente al otorgamiento de beneficio.

Dada la situación problemática anterior, se plantea como **problema a resolver:**

- La actual ejecución de la gestión del Consejo de Promoción afecta el correcto otorgamiento de beneficios a los internos en el sistema penitenciario cubano.

Teniendo como **objeto de estudio:**

- La gestión de los procesos de tratamiento educativo en los sistemas penitenciarios.

Para dar solución al problema se plantea el siguiente **objetivo general:**

- Desarrollar el módulo Gestión del Consejo de Promoción perteneciente al subsistema Tratamiento Educativo para gestionar el proceso de otorgamiento de beneficios.

Se define como **campo de acción:**

- La gestión de los procesos del consejo de promoción en el sistema penitenciario cubano.

Tomando en consideración el problema planteado se propone la siguiente **idea a defender:**

El desarrollo del módulo Gestión del Consejo de Promoción contribuirá al correcto otorgamiento de beneficios en los centros penitenciarios cubanos.

Como **objetivos específicos** se trazaron los siguientes:

- Elaborar el marco teórico de la investigación.

- Realizar el modelo de diseño de los requisitos planteados para el módulo Gestión del Consejo de Promoción del subsistema Tratamiento Educativo perteneciente al SIDEP.
- Implementar el módulo Gestión del Consejo de Promoción del subsistema Tratamiento Educativo perteneciente al SIDEP.
- Validar las funcionalidades de la propuesta de solución.

Para dar solución a los objetivos planteados anteriormente se trazan las siguientes **tareas de la investigación:**

- Análisis de soluciones nacionales e internacionales con un mismo perfil.
- Descripción de las herramientas y tecnologías para dar solución al problema.
- Realización del modelo de diseño para el módulo Gestión del Consejo de Promoción.
- Realización del diagrama de componentes para el módulo Gestión del Consejo de Promoción.
- Estudio de las reglas del negocio asociadas al consejo de promoción.
- Implementación del módulo Gestión del Consejo de Promoción.
- Diseño los casos de prueba para del módulo Gestión del Consejo de Promoción.
- Diseño de las pruebas de caja negra para el módulo Gestión del Consejo de Promoción.
- Realización de las pruebas de caja negra para el módulo Gestión del Consejo de Promoción.
- Corrección de las no conformidades encontradas.

Como parte de la solución se obtendrán los siguientes **resultados:**

- Mayor organización en el proceso del otorgamiento de beneficios, al dejar bien definidas las actividades que pueden ser ejecutadas por el gestor del consejo de promoción.
- Centralizar la información referente al interno y los datos necesarios que intervienen en la toma de decisiones.
- Obtener un sistema que se base en la transparencia de sus procesos, respetando los derechos del privado de libertad y evitando que cualquier opinión injusta de un oficial afecte la progresión de interno.

Para la elaboración de esta investigación se presenta el siguiente **diseño metodológico**:

Métodos Teóricos

Analítico-sintético: Este método ha sido utilizado durante el diseño teórico de la investigación que sustenta la propuesta de solución. Se evidencia en el análisis del proceso de otorgamiento de beneficios en los distintos sistemas, estudio de las herramientas y tecnologías; sintetizando las características principales y dividiendo las partes para un mejor entendimiento.

Histórico-lógico: En la primera parte de la investigación se realiza un estudio de la problemática anunciada, revisando las características del sistema penitenciario cubano, estableciendo así una conexión entre su concepción histórica y la actualidad.

Modelación: En la confección de los diagramas que permitirán representar la propuesta de solución y a partir de estos se comenzará a implementar la herramienta.

Entrevista: Se entrevistó a un oficial de la DEP a través de una conversación planificada con el objetivo de profundizar sobre el estado del sistema penitenciario en Cuba durante los primeros años de la Revolución.

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

Capítulo 1

En este capítulo se incluyen los aspectos fundamentales relacionados con el módulo Consejo de Promoción para manifestar la necesidad de informatizarlo, así como un estudio de las tendencias internacionales relacionadas con los sistemas penitenciarios. Se realiza un estudio de la metodología de desarrollo del software, las herramientas y los lenguajes de modelado que permiten dar solución al problema que se plantea.

Capítulo 2

En este capítulo se explica cómo está estructurada la arquitectura de la aplicación que se implementará, los patrones de diseño que se utilizarán y se describen las funcionalidades por las cuales está compuesto el módulo Consejo de Promoción. Se expone una representación de los diagramas de clases e interacción, así como el diseño de la base de datos para la persistencia de la información.

Capítulo 3

En este capítulo se presenta el diagrama de despliegue de la aplicación, resaltando los requerimientos que deben tener cada uno de los nodos físicos que intervienen en el proceso. Se muestran los diagramas

de componentes, especificando la interacción del módulo con otros subsistemas así como la relación interna de sus componentes. Por último se expone el proceso de validación y pruebas al que se sometió la propuesta de solución.

Capítulo 1

Fundamentación Teórica

1.1. Introducción

En este capítulo se realiza un estudio del estado del arte de la gestión de los beneficios en los sistemas penitenciarios que existen actualmente en el mundo y en Cuba. Igualmente, se hace un estudio de la metodología, herramientas y lenguajes que se utilizarán para el desarrollo de la solución de la propuesta.

1.2. Valoración del estado del arte

Desarrollo tecnológico en Cuba

Desde el triunfo de la Revolución, el gobierno cubano ha puesto especial interés en el uso masivo de las tecnologías de la información y las comunicaciones (TIC) en la economía nacional, la sociedad y al servicio del ciudadano. La estrategia de informatización, como expresión del proceso revolucionario cubano, tiene al ciudadano como centro de sus objetivos, buscando elevar su calidad de vida en su desempeño familiar, laboral, educacional, cultural, social y político.

La DEP no estuvo exenta de estos avances tecnológicos y se valió de dicho desarrollo con el objetivo de automatizar los procesos de seguimiento y control que se desarrollan en los centros penitenciarios.

Situación de los Sistemas de Gestión Penitenciaria en el mundo

Todas las sociedades del mundo tienen necesariamente que acogerse a un conjunto de leyes arbitrarias, creadas por el propio hombre, con el objetivo de garantizar la estabilidad social y un comportamiento adecuado de la sociedad. A nivel mundial existen diferentes tipos de sistemas de gestión penitenciaria en dependencia de la peculiaridad de cada país. Para el desarrollo de esta investigación, se realizó un estudio de diferentes sistemas encargados del control de internos. Seguidamente se analizan algunos de ellos.

1.2.1. Dirección General del Sistema Penitenciario (DGSP)

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

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

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

Las limitaciones se evidencian en los centros que no poseen enlace aún con la dirección central lo que debe llevar a una transformación de la red externa, reestructuración de la red interna de la sede, la dotación de internet a la institución, la actualización de toda la información de los centros penitenciarios y actualización de los equipos. El principal problema que presentan estos centros penitenciarios que no pueden consultar la dirección central es que no es posible verificar el tránsito del interno durante su privación de libertad.

El sistema penitenciario panameño es el conjunto organizado, funcional y estructurado de elementos normativos, técnicos y científicos. Sus objetivos principales son lograr la resocialización del privado de libertad sobre la base de un adecuado tratamiento penitenciario, el trabajo, la capacitación, la educación y la práctica de valores morales, de modo que puedan reincorporarse útilmente a la sociedad y ejecutar las sentencias emitidas por los tribunales de justicia y las resoluciones de las autoridades administrativas de policía.

El Departamento de Tratamiento y Rehabilitación presenta la recopilación de las actividades de resocialización más significativas del Sistema Penitenciario, además de ser el órgano encargado de

otorgar los beneficios. Los privados de libertad, para obtener los beneficios que se establecen en su Reglamento Penitenciario deben someterse a los siguientes procedimientos:

- Se reciben los expedientes de clasificación en la recepción del Departamento.
- Se recibe el expediente de clasificación cuando el privado de libertad obtenga la sentencia en firme.
- Registrar fecha y número de entrada de los expedientes por el profesional encargado del trámite.
- Lectura y revisión del expediente, con los requisitos que señala la ley (propuesta de clasificación, informes jurídico, social, psicológico y plan individual de tratamiento), basado en la discusión de la Junta Técnica. De lo contrario devolverlo al Centro Penal.
- Confeccionar resolución para firma del Director, debidamente refrendado por la jefa de tratamiento.
- Ingresar en la computadora y en el libro record (después de firmada la resolución), el nombre del privado de libertad y su respectiva Clasificación en Período.
- Entregar la resolución de clasificación a la trabajadora social encargada de las solicitudes de permiso de trabajo o estudio, en caso de tener, solicitudes de esta índole para tramitar.
- Enviar al Centro Penal la resolución de clasificación para conocimiento del equipo técnico, notificación al privado de libertad y copia al expediente. (2)

1.2.2. El Instituto Nacional Penitenciario (INPE)

Es un Organismo Público Ejecutor del Sector de Justicia, rector del Sistema Penitenciario Nacional perteneciente a la República de Perú, se rige por el Código de Ejecución Penal y su Reglamento.

Sus funciones principales son:

- Dirigir y administrar el Sistema Nacional Penitenciario.
- Realizar investigaciones sobre criminalidad y elaborar políticas de prevención del delito.
- Brindar asistencia post penitenciaria.
- Dictar normas técnicas y administrativas sobre el planeamiento y construcción de infraestructura penitenciaria. (3)

Los beneficios penitenciarios admisibles según la normatividad penitenciaria peruana son los siguientes:

- Permiso de salida, el que puede ser concedido al interno por un máximo de 72 horas, en determinados casos.
- Redención de la pena por el trabajo y la educación. Se otorga un día de reducción de pena por dos días de labor.
- Semi-libertad. Permite egresar al sentenciado que ha cumplido la tercera parte de la pena.
- Liberación condicional. Se concede al sentenciado que ha cumplido la mitad de la condena impuesta.
- Visita íntima. Otorgado por el director del establecimiento penitenciario.
- Otros beneficios. Integrado por recompensas, tales como la autorización para trabajar en horas extraordinarias, desempeñar labores auxiliares de la administración penitenciaria y concesión extraordinaria de comunicaciones y visitas. (4)

1.2.3. Sistema de Gestión Penitenciaria (SIGEP)

SIGEP es un software diseñado e implementado para el sistema penitenciario de Venezuela. En este sistema no existe un módulo gestor de los procesos de otorgamiento de beneficios. El proceso inicia cuando el individuo ha cumplido los términos legales que establecen cada una de las medidas (en unos casos se tratan de una porción de pena, en otros casos como certificados médicos).

En la sede se realiza una evaluación técnica (módulo evaluaciones técnicas). Cuando ésta se realiza y es generada por el sistema, la misma se envía al tribunal a modo de solicitud (módulo solicitudes) y cuando el tribunal tome la decisión (de otorgamiento o denegación) se registra la decisión (módulo decisiones), si es un otorgamiento implica la progresividad que será atendida por los módulos de egreso e ingreso. (5)

En el sistema penitenciario venezolano siempre que se aprueba un beneficio el interno egresa del centro en el que se encuentra para ingresar posteriormente en otro centro con distintas condiciones, de ahí que este proceso sea tratado en mayor medida por los módulos (Egreso e Ingreso).

1.2.4. Situación en Cuba

El sistema penitenciario nacional, comienza el desarrollo de la informatización a partir del 1989 con la automatización de los datos principales del recluso y algunos aspectos de control penal. A raíz del cumplimiento de la orden 43/99 del VMP se crea un Sistema Automatizado para el Control del Recluso (SACORE).

Este sistema fue diseñado y programado sobre la tecnología existente en el MININT, fue implantado a partir de enero 2003. Contiene como módulos principales Control Penal, Reeducación Penal y el Orden Interior. (1)

Cuenta con 9 años de explotación, atiende un total de nueve beneficios generando un gran cúmulo de información, la misma resulta muy difícil de procesar. Estas dificultades provocan que el análisis estadístico se convierta en un proceso engorroso y en muchas ocasiones lento, propiciando la demora en la entrega a tiempo de los informes, afectando de esta forma la toma de decisiones.

Conclusiones parciales de las soluciones existentes para los centros penitenciarios.

El análisis de las soluciones informáticas desarrolladas en otros países permitió concluir que los sistemas penitenciarios analizados comparten métodos similares para el control de reclusos. Los sistemas gestores de los establecimientos penitenciarios de otros países se ajustan a las características de esa región, presentan distintos métodos de manejar el otorgamiento de beneficios llegando a atender menos beneficios de los que se analizan en el SIDEP. Teniendo en cuenta las razones anteriores se puede concluir que muchos de los procedimientos de las soluciones internacionales no serían aplicables a Cuba y que el actual sistema SACORE no gestiona todos los procesos que intervienen en el otorgamiento de beneficio, de ahí la necesidad de desarrollar una aplicación para ello.

1.3. Proceso de Desarrollo de Software

1.3.1. Metodología de Desarrollo

Una metodología de desarrollo de software abarca los disímiles procedimientos y técnicas que deben seguirse para desarrollar un software. Concretamente "define quién está haciendo qué, cuándo hacerlo y cómo alcanzar cierto objetivo". La metodología de desarrollo de software se encarga de elaborar estrategias; están centradas en las personas o los equipos y orientadas hacia la funcionalidad y la entrega. Su objetivo es elevar la calidad del software a través de un mayor control sobre el proceso.

Como metodología de desarrollo para el SIDEP se optó por el Proceso Unificado de Desarrollo desarrollado por la empresa Rational (RUP[1]), que al ser una metodología pesada intenta obtener un objetivo común por medio del orden y el uso exhaustivo de documentación durante todo el ciclo de vida

[1] Acrónimo en inglés de Rational Unified Process

del proyecto lo que brinda la ventaja de poder efectuar un futuro mantenimiento de manera más eficiente, son recomendadas para los proyectos de grandes dimensiones y con grandes equipos de desarrollo.

Divide el proceso de desarrollo en ciclos, donde se obtiene un producto al final de cada ciclo y se apoya en el Lenguaje de Modelado Unificado (UML [2]). Cada período se divide en cuatro Fases: Inicio, Elaboración, Construcción, y Transición; cada fase concluye con un hito bien definido. (6)

La metodología RUP está definida por tres aspectos fundamentales:

Procesos dirigidos por casos de usos: Un caso de uso (CU) es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado, el conjunto de todos los casos de usos constituye el modelo de casos de uso y describe la funcionalidad total del sistema, por lo que son usados para especificar los requisitos de un sistema, para guiar su diseño, implementación y prueba, de este modo los casos de usos no solo inician el proceso sino que le proporcionan un hilo conductor.

Centrado en la arquitectura: La arquitectura es una vista del diseño completo con las características más importantes resaltadas, dejando los detalles a un lado. Esta se construye con los casos de usos más significativos del sistema que representan las funciones claves del sistema en desarrollo.

Iterativo e incremental: Como el desarrollo de un proyecto es un proceso complejo, es práctico dividir el trabajo en partes más pequeñas o mini proyectos, cada mini proyecto es una iteración que resulta en un incremento, las necesidades de los usuarios no pueden definirse completamente al principio, por lo que se refinan en iteraciones sucesivas. (6)

La aplicación de estos aspectos en el SIDEP se evidencia en la confección de los CU que constituyen la línea base utilizada por el equipo de programadores. Además se identifican las principales funcionalidades para establecer una arquitectura conformada por subsistemas y por último en la realización de iteraciones al realizar la captura de requisitos, desarrollar la propuesta de solución y validarla con el cliente.

El SIDEP es un proyecto de gran magnitud donde se gestionan los procesos del control penal. El éxito de este proyecto tendrá un impacto positivo en la sociedad, al asegurar que se respeten los derechos de los interno y reinsertarlos a la comunidad como individuos capaces de regirse por los códigos de conducta que se establecen en Cuba. Por tales motivos es imprescindible asegurar la calidad del sistema y con ese

[2] Acrónimo en inglés de Unified Model Language

objetivo se seleccionó como metodología de desarrollo a RUP porque pretende implementar las mejores prácticas de la ingeniería de software, teniendo en cuenta los plazos específicos y presupuestos predecibles para satisfacer las necesidades de los usuarios finales.

Lenguaje de modelado

UML

Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Permite la visualización, especificación, construcción y documentación de los artefactos de sistemas. Básicamente facilita a los desarrolladores visualizar los resultados de su trabajo en esquemas o diagramas estandarizados. UML ofrece un estándar para escribir un "plano" del sistema, incluyendo aspectos conceptuales tales como: procesos de negocios y funciones del sistema y aspectos concretos como: expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional o RUP). (7)

A continuación se mencionan los diagramas que contiene UML y que fueron utilizados para mostrar diferentes aspectos de las entidades representadas, durante el desarrollo de la propuesta de solución.

- Diagrama de clases
- Diagrama de componentes
- Diagrama de despliegue
- Diagrama de casos de uso

1.3.2. Herramientas

Las herramientas y tecnologías utilizadas para el desarrollo del módulo “Gestión del Consejo de Promoción” fueron inicialmente establecidas por el equipo de arquitectos del SIDEP.

Visual Paradigm Suite 5.0

Visual Paradigm para UML (VP-UML) es una herramienta CASE[3] de diseño UML. Es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Su mayor éxito radica en la capacidad de ejecutarse sobre

[3] Computer Aided Software Engineering (Ingeniería de Software Asistida por Computadora)

diferentes sistemas operativos lo que le confiere la característica de ser multiplataforma. Constituye un software privativo y sus distintas ediciones son compatibles.

Utiliza UML como lenguaje de modelado, ofreciendo soluciones de software que permiten a las organizaciones desarrollar las aplicaciones con mayor calidad de forma rápida y satisfactoria. Es fácil de usar y presenta un entorno gráfico agradable para el usuario. (8)

Durante la realización del módulo se utilizó dicha herramienta para la realización de los diagramas de clases del diseño, de interacción, de componentes y por último el modelo de despliegue.

ER/Studio 8.0

Es una herramienta CASE 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. Soporta más de 30 de las bases de datos más importantes de la industria, entre ellas Oracle. (9)

ER/Studio 8.0 ofrece un nuevo componente visual para el análisis del impacto en el rendimiento entre fuentes de datos. Esto permite a los usuarios visualizar, analizar y documentar cómo fluyen los datos a través de la organización. (10)

ER/Studio posee poderosas características de modelación que aceleran el ciclo de diseño, como son:

- Diseño físico y lógico separados.
- Transformación automática de un diseño lógico a un diseño físico para una plataforma específica de base de datos.
- Extensa validación de los modelos. Es posible validar ampliamente la calidad e integridad tanto del diseño lógico como del diseño físico del modelo. (9)

Esta herramienta fue utilizada para realizar el diseño lógico del modelo de datos referente al módulo, donde quedan plasmadas todas las tablas que intervienen y la relación entre ellas.

NetBeans 7.0

Es utilizado como ambiente de desarrollo pues trae incluido entre sus novedades más destacadas el uso de Groovy, además de permitir el empleo de módulos (en inglés plug-in) para ampliar sus funcionalidades: se utilizó el módulo Groovy, que provee soporte para el lenguaje Groovy y el framework

Grails. El IDE[4] NetBeans es un reconocido entorno de desarrollo integrado disponible para Windows, Mac, Linux y Solaris, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. El proyecto NetBeans está formado por un IDE de código abierto y una plataforma de aplicación que permite a los desarrolladores crear con rapidez aplicaciones web, empresariales, de escritorio y móviles.

Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactorización. Cuenta con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código. El IDE también hace uso de un espacio de trabajo, en este caso un grupo de metadatos en un espacio para archivos planos, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente. Puede extenderse usando otros lenguajes de programación como son C/C++, Phyton. Es un producto libre y gratuito sin restricciones de uso. (11)

Apache Tomcat 6.0

Apache Tomcat es un contenedor Web escrito en Java, por lo que funciona en cualquier sistema operativo que disponga de una máquina virtual Java y desarrollado en un ambiente participativo y abierto (12). Las versiones de Apache Tomcat 6.0 implementan el Servlet 2.5 y JavaServer Pages 2.1 especificaciones del Java Community Process. Publicado bajo la licencia Apache versión 2, siendo una marca registrada de la Fundación de Software Apache. (13)

Puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era solo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y Tomcat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad. (13)

En el caso del SIDEP se selecciona como servidor de aplicaciones debido precisamente a que la plataforma de desarrollo que se utiliza es Java.

1.3.3. Tecnologías

Grails 1.3.7

[4] Integrated Development Environment (Entorno de Desarrollo Integrado)

Grails es el framework utilizado para el desarrollo del SIDEP, es un framework web de código abierto compatible con Java. Combina principios tales como "convención sobre configuración" (Convention Over Configuration) y "No te repitas" (Don't Repeat Yourself) junto a una serie de frameworks de código abierto como Hibernate (para mapeo objeto-relacional), Spring (para inyección de dependencia) y SiteMesh (para plantillas). Todo esto, unido al lenguaje dinámico Groovy construido sobre Java. Grails pretende ser un framework altamente productivo, proporcionando un entorno de desarrollo estandarizado y ocultando gran parte de los detalles de configuración al programador.

Grails se puede ampliar a través de plugins. Por su dinamismo es capaz de acortar el ciclo de desarrollo, ahorrando tiempo de trabajo y agilizándolo. Incluye un contenedor web, base de datos, sistemas de construcción y pruebas. Esta combinación puede reducir el tiempo inicial del proyecto y el tiempo de instalación del desarrollador a minutos en lugar de horas. No hay que instalar o mantener scripts, todo lo que se necesita viene incluido en un paquete fácil de instalar.

En lugar de usar archivos de configuración XML se basa en las convenciones para hacer más fácil, sencillo y productivo el desarrollo, donde ya no es necesario configurar la aplicación mediante XML. (14)

Los plugins que utiliza el framework son:

- grails-hibernate-1.3.7
- grails-tomcat-1.3.7
- grails-dojo-1.5.0.1
- grails-spring-security-core-1.0.1

Dojo 1.5

Dojo es un framework que contiene Apis y widgets (controles) para facilitar el desarrollo de aplicaciones Web que utilicen tecnología AJAX. Contiene un sistema de empaquetado inteligente, los efectos de UI[5], drag and drop Apis, widget Apis, abstracción de eventos, almacenamiento de Apis en el cliente, e interacción de Apis con AJAX. Resuelve asuntos de usabilidad comunes como pueden ser la navegación y detección del navegador, soportar cambios de URL en la barra de URLs para luego regresar a ellas. Proporciona una gama más amplia de opciones en una sola biblioteca Java Script y es compatible con navegadores antiguos. (15)

[5] Usser Interface (Interfaz de Usuario)

Durante el desarrollo se utilizó Dojo para el trabajo con el DOM[6] y el acceso a los widgets se realiza a través de los taglibs que se han personalizado en la aplicación.

Oracle Database 11g

Es un Sistema Gestor de Bases de Datos con características objeto-relacionales, utilizando tecnología cliente/servidor. Permite la gestión de grandes bases de datos, un alto rendimiento en transacciones, disponibilidad controlada de los datos de las aplicaciones, la gestión de la seguridad y la autogestión de la integridad de los datos. Se considera a Oracle como uno de los sistemas de bases de datos más completo, destacando: estabilidad, escalabilidad y soporte multiplataforma. Además se encuentra muy difundido sobre todo en las grandes compañías, transnacionales o instituciones gubernamentales por sus elevados precios de licencias de software, soporte técnico y sus elevados requisitos de hardware. (16)

Dicho gestor de base de datos se utilizó para administrar y almacenar grandes volúmenes de datos. También se puede afirmar que constituye una herramienta de administración gráfica cómoda de utilizar y que permite el análisis de la información.

1.3.4. Lenguajes

1.3.4.1. Tecnologías del lado del cliente.

HTML

HTML es un lenguaje de composición de documentos y especificación de ligas de hipertexto que define la sintaxis y coloca instrucciones especiales que no muestra el navegador, aunque si le indica cómo desplegar el contenido del documento, incluyendo texto, imágenes y otros medios soportados.

Una de las características esenciales de este lenguaje es la universalidad, y significa que prácticamente cualquier ordenador, independientemente del sistema operativo, puede leer o interpretar una página web. Sin embargo, el aspecto de dichas páginas depende del navegador debido a que no todos muestran una página web de la misma forma. (17)

Java Script

JavaScript es un lenguaje de programación ligero y orientado a objetos que se utiliza principalmente para crear páginas web dinámicas. Con este lenguaje script se pueden generar páginas dinámicamente en

[6] Acrónimo en inglés de Document Object Model (Modelo de Objetos del Documento)

función de las preferencias del usuario, validar los datos introducidos en un formulario o modificar dinámicamente el contenido de la página. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. (18)

Durante el desarrollo del módulo se utilizó JavaScript para validar todas las entradas de datos del lado del cliente, notificando en caso de existir datos incorrectos o campos en blanco que son de carácter obligatorio.

1.3.4.2. Tecnologías del lado del servidor

Groovy 1.7.8

Es el lenguaje utilizado para el desarrollo de aplicaciones Grails .Es un lenguaje dinámico basado en la máquina virtual de Java que tiene como ventaja que su sintaxis es compatible con Java, pero añade características dinámicas y sintácticas inspiradas en lenguajes como Python, Ruby y Smalltalk que ahorran muchas líneas de código. Precisamente su similitud con Java permite que la curva de aprendizaje para los desarrolladores familiarizados con Java sea casi nula. El código fuente Groovy se compila a bytecodes igual que Java, y es posible instanciar objetos java desde Groovy. Groovy incluye mejoras sintácticas en relación con la facilidad de manejos de objetos mediante nuevas expresiones y sintaxis, distintas formas de declaración de literales. Control de flujo avanzado mediante nuevas estructuras, y nuevos tipos de datos, con operaciones y expresiones específicas. En resumen: el código Groovy es mucho más breve que el de Java. (19)

Java

El lenguaje para la programación en Java, es un lenguaje orientado a objeto, de una plataforma independiente. Fue desarrollado por la compañía Sun Microsystems a principios de los años 90, con la idea original de usarlo para la creación de páginas web. Tiene muchas similitudes con el lenguaje C y C++, pero contiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen

inducir a muchos errores, como la manipulación directa de punteros o memoria. Java se considera un software de distribución libre y de fácil aprendizaje.

Independientemente a todas las potencialidades que posee Java, puede ser un lenguaje de ejecución lenta debido al uso de la máquina virtual de Java (ésta consume muchos recursos), además el uso de un recolector de basura para eliminar de forma automática aquellos objetos no requeridos, añade una sobrecarga que puede afectar al rendimiento. Esto es en parte debido a las frecuentes declaraciones de tipos y conversiones de tipo manual. (20)

1.4. Conclusiones

Al finalizar dicho capítulo, se puede concluir que resulta de vital importancia la informatización del sistema penitenciario cubano. Se analizaron varios sistemas internacionales con un mismo perfil pero no satisfacen a plenitud las necesidades que actualmente tienen los centros penitenciarios de Cuba, por tal motivo resulta importante el desarrollo de una aplicación basada en las herramientas y tecnologías seleccionadas.

Capítulo 2

Diseño del Módulo

2.1. Introducción

En este capítulo se abordarán los temas referentes al diseño de la propuesta de solución del módulo Consejo de Promoción, los cuales se encuentran sustentados por las reglas del negocio con las que debe cumplir la propuesta de solución y los requisitos funcionales del sistema. Se muestra como está concebida la arquitectura, se realizan las descripciones de las tablas de la base de datos y se tienen en cuenta los patrones de diseño que aportan soluciones concretas a problemas específicos para lograr un diseño eficaz en la propuesta de solución.

2.2. Reglas del Negocio

Las Reglas del Negocio contienen las políticas, operaciones, definiciones y restricciones vigentes en una organización y guían los esfuerzos en pos de alcanzar los objetivos. Son aquello que usamos para operar un negocio, un medio por el cual la estrategia es implementada. Las reglas especifican - en un nivel adecuado de detalle - lo que una organización debe hacer. Son las guías que determinan como se lleva el día a día de las operaciones. Algunas de las principales reglas del negocio que respaldan el proceso de otorgamiento de beneficios son las siguientes:

A continuación se exponen los niveles de aprobación con algunos de los principales tipos de beneficios que analizan:

Jefatura Provincial y del MEIJ[7]

- Permisos especiales de Salida al Hogar.

Jefe DEP

- Clasificación y Desclasificación en régimen mayor severidad.

[7] Municipio Especial de la Isla de la Juventud

Jefe Unidad

- Todas las rebajas de sanciones: la rebaja de sanción se tiene en cuenta a partir de la fecha en que comienza a extinguir.

Tribunal

- Libertad Condicional: los sancionados a privación perpetua de libertad puede concedérsele la libertad condicional una vez que hayan transcurridos 30 años en prisión.
- Variación de la medida de seguridad

En la reunión del Consejo de Promoción son analizados un total de quince beneficios, divididos en dos grupos:

Beneficios con término: son otorgados en períodos calculables atendiendo a la sanción por la que cumple cada interno, son un total de cuatro beneficios dentro de los que figuran:

- Libertad condicional
- Rebaja de sanción por buena conducta
- Sustitución de privativa de libertad por subsidiaria
- Progresión en régimen o fase

Beneficios sin término: son otorgados en cualquier período de la sanción atendiendo al comportamiento del interno, son un total de once beneficios:

- Suspensión de rebaja de sanción
- Libertad por solicitud del tribunal
- Licencia extrapenal
- Variación de la medida de seguridad
- Rebaja de sanción por conducta excepcional
- Suspensión de TCCI[8]
- Suspensión de la medida de seguridad
- Revocación de TCCI
- Ubicación en semi-libertad

[8] Trabajo correccional con internamiento

- Tránsito anticipado a media o mínima
- Regresión de régimen o fase

2.3. Requisitos Funcionales

Los requisitos suelen expresarse como objetivos del sistema, determinan cómo funcionará y establecen las restricciones sobre su operación e implementación. Un requisito es una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado. Se aplica a las condiciones que debe poseer el sistema o uno de sus componentes para satisfacer un contrato. (21)

Teniendo en cuenta los procesos y las reglas del negocio que se identificaron anteriormente, a continuación se definen los requisitos funcionales que presentará el sistema.

Gestión del Consejo de Promoción

No.	Requisitos Funcionales
1	Registrar reunión de Consejo de Promoción
2	Otorgar libertad condicional a un interno
3	Denegar libertad condicional a un interno
4	Otorgar rebaja de sanción por buena conducta a un interno
5	Denegar rebaja de sanción por buena conducta a un interno
6	Otorgar sustitución de sanción por subsidiaria a un interno
7	Denegar sustitución de sanción por subsidiaria a un interno
8	Otorgar progresión con régimen o fase a un interno
9	Denegar progresión con régimen o fase a un interno
10	Otorgar suspensión de rebaja de sanción a un interno
11	Denegar suspensión de rebaja de sanción a un interno
12	Otorgar libertad por solicitud del tribunal a un interno
13	Denegar libertad por solicitud del tribunal a un interno
14	Otorgar licencia extrapenal a un interno
15	Denegar licencia extrapenal un interno
16	Otorgar variación de la medida de seguridad a un interno
17	Denegar variación de la medida de seguridad a un interno

18	Otorgar rebaja de sanción por conducta excepcional a un interno
19	Denegar rebaja de sanción por conducta excepcional a un interno
20	Otorgar suspensión de TCCI a un interno
21	Denegar suspensión de TCCI a un interno
22	Otorgar suspensión de la medida de seguridad a un interno
23	Denegar suspensión de la medida de seguridad a un interno
24	Otorgar revocación de TCCI a un interno
25	Denegar revocación de TCCI a un interno
26	Otorgar ubicación en semi-libertad a un interno
27	Denegar ubicación en semi-libertad a un interno
28	Otorgar tránsito anticipado a media o mínima a un interno
29	Denegar tránsito anticipado a media o mínima a un interno
30	Otorgar regresión con régimen o fase a un interno
31	Denegar regresión con régimen o fase a un interno
32	Mostrar listado de internos de suspensión de rebaja de sanción
33	Mostrar listado de internos de libertad por solicitud del tribunal
34	Mostrar listado de internos de licencia extrapenal
35	Mostrar listado de internos de variación de la medida de seguridad
36	Mostrar listado de internos de rebaja de sanción por conducta excepcional
37	Mostrar listado de internos de suspensión de TCCI
38	Mostrar listado de internos de suspensión de la medida de seguridad
39	Mostrar listado de internos de revocación de TCCI
40	Mostrar listado de internos de ubicación en semi-libertad
41	Mostrar listado de internos de tránsito anticipado a media o mínima
42	Mostrar listado de internos de regresión de régimen o fase
43	Consultar listado de permisos
44	Proponer interno para beneficio sin término
45	Cancelar propuesta de interno para beneficio sin término
46	Actualizar término para suspensión de TCCI y de medida de seguridad
47	Consultar listado de internos con beneficio analizado

48	Registrar evaluación de conducta de los internos

Tabla 1: Requisitos Funcionales del módulo Gestión del Consejo de Promoción.

2.4. Diagramas de casos de uso del sistema

Las principales funcionalidades que se pueden realizar en el Consejo de Promoción se muestran a continuación especificando el encargado de realizarlas.

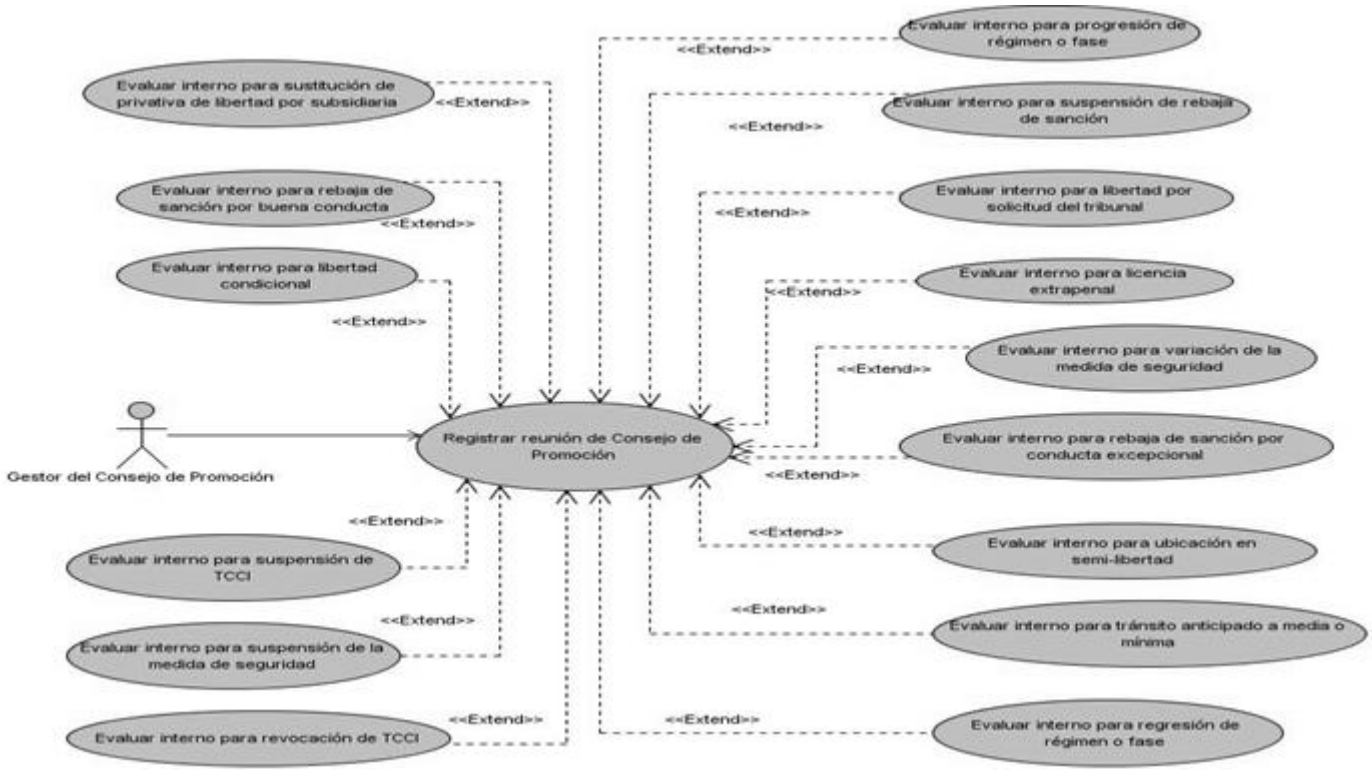

Figura 1: Diagrama del CU de Consejo de Promoción.

El Gestor del Consejo de Promoción (GCP) es el encargado de gestionar todos los procesos que intervienen en el otorgamiento de beneficios. Tiene como objetivo consultar la información sobre el tránsito del interno por el centro penitenciario, la cual servirá como referencia para la toma de decisiones.

A continuación se muestra el resto de las operaciones que debe realizar el GCP:

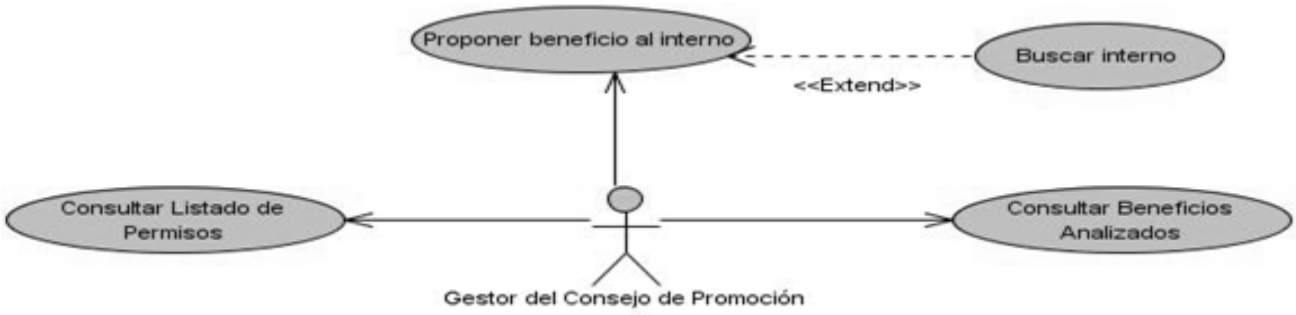

Figura 2: Diagrama del CU de Consejo de Promoción.

Consultar Listado de Permisos

El GCP consulta los listados de permisos con el objetivo de verificar cuáles son los internos que se analizarán posteriormente con el objetivo de aprobar o revocar el permiso que tiene asignado.

Proponer beneficio al interno

El GCP realiza las propuestas que serán analizadas en la reunión del consejo de promoción. Una propuesta está compuesta por la asignación de un tipo de beneficio (sin término) asociado a un interno. Para agilizar el proceso de realización de una propuesta se utiliza el buscador de internos, el cual brinda la opción de filtrar la búsqueda por varios aspectos: nombre, apellidos, alias, entre otros.

Consultar Beneficios Analizados

El GCP consulta los beneficios que han sido analizados en reuniones anteriores con el objetivo de:

- Consultar los detalles del análisis del beneficio, indicando la decisión emitida y el tribunal que intervino, así como el resto de los datos que fueron necesarios registrar para emitir la evaluación.
- Registrar la decisión del tribunal, la cual sólo se puede emitir sobre el beneficio que tenga como evaluación "Aprobado".

A continuación se exponen las restantes actividades realizadas en el módulo Gestión del Consejo de Promoción.

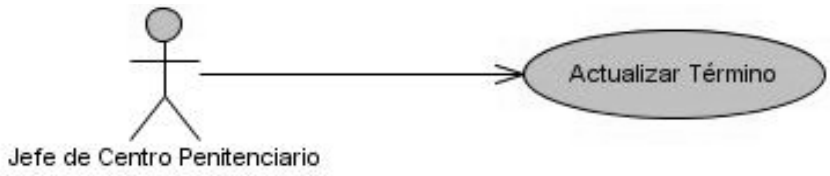

Figura 3: Diagrama del CU de Actualizar Término.

El Jefe de Centro Penitenciario actualiza el término del análisis de los beneficios. Por ejemplo, en caso de que el término tenga como valor 1/3, al interno se le analizará el beneficio una vez cumplido un 1/3 del período de su condena.

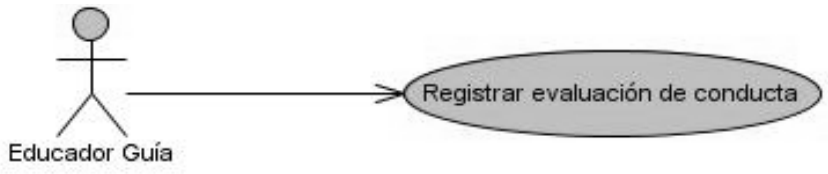

Figura 4: Diagrama del CU de Registrar evaluación de conducta.

El Educador Guía registra la evaluación de conducta asociada al interno para que posteriormente sea consultada durante la reunión del consejo de promoción.

2.5. Descripción de los Casos de Uso del módulo Gestión del Consejo de Promoción.

Las funcionalidades anteriormente expuestas a través de los diagramas son descritas a continuación para el mejor entendimiento de los procesos que sustentan el otorgamiento de beneficios.

CU	Nombre	Descripción
1	Registrar reunión de Consejo de Promoción	Permite registrar las decisiones tomadas en el Consejo de Promoción, además de todos los beneficios referentes a un interno sobre los cuales el tribunal ha emitido una decisión.
2	Evaluar interno para libertad condicional	El sistema genera un listado con los internos que han sido propuestos para el beneficio “Libertad Condicional”, también permite tomar una decisión sobre dicho beneficio.
3	Evaluar interno para rebaja de sanción por buena conducta	El sistema genera un listado con los internos que han sido propuestos para el beneficio “Rebaja de sanción por buena conducta”, también permite tomar una decisión sobre dicho beneficio.
4	Evaluar interno para sustitución de privativa de libertad por subsidiaria	El sistema genera un listado con los internos que han sido propuestos para el beneficio “Sustitución de privativa de libertad por subsidiaria”, también permite tomar una decisión sobre dicho beneficio.
5	Evaluar interno para progresión en régimen o fase	El sistema genera un listado con los internos que han sido propuestos para el beneficio “Progresión en régimen o fase”, también permite tomar una decisión sobre dicho beneficio.

6	Evaluar interno para suspensión de rebaja de sanción	El sistema genera un listado con los internos que han sido propuestos para el beneficio “Suspensión de rebaja de sanción”, también permite tomar una decisión sobre dicho beneficio.
7	Evaluar interno para libertad por solicitud del tribunal	El sistema genera un listado con los internos que han sido propuestos para el beneficio “Libertad por solicitud del tribunal”, también permite tomar una decisión sobre dicho beneficio.
8	Evaluar interno para licencia extrapenal	El sistema genera un listado con los internos que han sido propuestos para el beneficio “licencia extrapenal”, también permite tomar una decisión sobre dicho beneficio.
9	Evaluar interno para variación de la medida de seguridad	El sistema genera un listado con los internos que han sido propuestos para el beneficio “variación de la medida de seguridad”, también permite tomar una decisión sobre dicho beneficio.
10	Evaluar interno para rebaja de sanción por conducta excepcional	El sistema genera un listado con los internos que han sido propuestos para el beneficio “rebaja de sanción por conducta excepcional”, también permite tomar una decisión sobre dicho beneficio.
11	Evaluar interno para suspensión de TCCI	El sistema genera un listado con los internos que han sido propuestos para el beneficio “suspensión de TCCI”, también permite tomar una decisión sobre dicho beneficio.
12	Evaluar interno para suspensión de la medida de seguridad	El sistema genera un listado con los internos que han sido propuestos para el beneficio “suspensión de la medida de seguridad”, también permite tomar una decisión sobre dicho beneficio.
13	Evaluar interno para revocación de TCCI	El sistema genera un listado con los internos que han sido propuestos para el beneficio “revocación de TCCI”, también permite tomar una decisión sobre dicho beneficio.
14	Evaluar interno para ubicación en semi-libertad	El sistema genera un listado con los internos que han sido propuestos para el beneficio “ubicación en semi-libertad”, también permite tomar una decisión sobre dicho beneficio.
15	Evaluar interno para tránsito anticipado a media o mínima	El sistema genera un listado con los internos que han sido propuestos para el beneficio “tránsito anticipado a media o mínima”, también permite tomar una decisión sobre dicho beneficio.
16	Evaluar interno para regresión de régimen o fase	El sistema genera un listado con los internos que han sido propuestos para el beneficio “regresión de régimen o fase”, también permite tomar

		una decisión sobre dicho beneficio.
17	Consultar listado de permisos	El sistema muestra el listado de todos los permisos que han sido asignados a los internos.
18	Proponer beneficio a interno	El sistema permite hacer la propuesta de internos para analizar el otorgamiento de beneficio sin término.
19	Actualizar término	El sistema permite modificar el término para la suspensión de TCCI y de la medida de seguridad.
20	Consultar beneficios analizados	El sistema permite consultar el listado de los internos que ya han sido analizados en el Consejo de Promoción y poder incluir la decisión del tribunal para cada interno.
21	Registrar evaluación de conducta	El sistema permite registrar la evaluación de conducta de los internos de un colectivo que van a ser analizados en el Consejo de Promoción.

Tabla 2: Casos de Uso del módulo "Gestión del Consejo de Promoción".

2.6. Marco de trabajo de desarrollo web.

Como la mayoría de los frameworks de desarrollo web, Grails está basado en una especialización del patrón Modelo Vista Controlador (MVC). En Grails los modelos son tratados como clases de dominio que permiten a la aplicación mostrar los datos en la vista. Las clases de dominio de Grails son entidades persistentes. Los controladores por su parte, permiten gestionar las peticiones a la aplicación y organizar los servicios proporcionados. Por último, la vista por defecto en Grails son las Groovy Server Pages (GSP) y habitualmente muestran el contenido en formato HTML. (14)

Grails es un framework para desarrollo de aplicaciones web construido sobre cinco fuertes pilares:

- **Groovy** para la creación de propiedades y métodos dinámicos en los objetos de la aplicación.
- **Spring** para los flujos de trabajo e inyección de dependencias.
- **Hibernate** para la persistencia.
- **SiteMesh** para la composición de la vista.
- **Ant** para la gestión del proceso de desarrollo. (14)

Sus principales características son:

- **Alta productividad:** Grails tiene tres características que intentan incrementar su productividad comparándolo con los Framework Java tradicionales:
  - ✓ Inexistencia de configuración XML.
  - ✓ Entorno de desarrollo preparado para funcionar desde el primer momento.
  - ✓ Funcionalidad disponible mediante métodos dinámicos.
- **Scaffolding:** El scaffolding es una característica de determinados frameworks que permite la generación automática de código para las cuatro operaciones básicas de cualquier aplicación, que son: la creación, lectura, edición y borrado, lo que en inglés se conoce como CRUD (create, read, update and delete). El scaffolding en Grails se consigue escribiendo muy pocas líneas de código, solo se deben especificar las propiedades, comportamientos y restricciones de nuestras clases de dominio.
- **Integración con la plataforma Java:** La mejor característica que Grails ofrece en este ámbito es una integración transparente con clases mapeadas mediante el Framework Hibernate. Esto significa que aplicaciones existentes que utilicen Hibernate pueden utilizar Grails sin recompilar el código o reconfigurar las clases, aprovechando los métodos de persistencia.
- **Persistencia:** El modelo de datos en Grails se graba en la base de datos utilizando GORM (Grails Object Relational Mapping). Las clases de dominio se guardan en el directorio grails-app/domain.
- **Plugins:** Grails no siempre es la solución a cualquier problema que se nos pueda plantear en el desarrollo de aplicaciones web. Para facilitar el trabajo, Grails dispone de una arquitectura de plugins para seguridad, AJAX, testeo, búsqueda, informes y servicios web. Este sistema de plugins hace que añadir complicadas funcionalidades a la aplicación se convierte en algo muy sencillo

2.7. Arquitectura del Sistema

La arquitectura del sistema está fundamentada por la arquitectura N-Capas, basada en el patrón Modelo-Vista-Controlador (MVC), el cual contiene como capas lógicas principales: Capa de Presentación o Vistas, Capa de Control o Controladores y Capa de Datos o Modelo , en el caso particular de Grails se añade una Capa de Servicios para la lógica de negocio. (22)

La integración de las capas se evidencia de la siguiente forma:

Los controladores se encargan de mostrar las vistas y controlar el flujo de datos entre las vistas y las entidades, recogen la información introducida por el usuario, la envían a los servicios encargados de implementar la lógica de negocio, para que posteriormente persistan en la base de datos.

Figura 5: Arquitectura del sistema.

A continuación se exponen las principales características de las capas que componen la arquitectura del sistema:

Capa de Presentación

En esta capa se encuentran las Vistas, según la arquitectura que propone Grails estarán enmarcadas sus clases en el paquete View donde se encontrarán las Groovy Server Pages o más conocidas como GSP. Grails implementa el patrón MVC, en el que la lógica del negocio se separa de la presentación de la aplicación. Esto permite cambiar fácilmente el aspecto de la aplicación, sin modificar su comportamiento.

Grails utiliza para la interacción con el usuario la tecnología JSP, pero basada en una implementación mediante GSP. Además los desarrolladores pueden mezclar etiquetas de lenguajes de marcas tradicionales como HTML con código Groovy para producir vistas dinámicas. Las Vistas son los recursos que junto al modelo generado por los controladores le permiten al cliente visualizar la información, estos pueden ser páginas HTML, documentos en formato PDF, hojas de cálculo, entre otras. Las mismas están representadas en el paquete de clase Views. (14)

La capa de presentación agrupa los siguientes componentes:

- Clases de presentación(gsp)
- Taglib
- JavaScript
- CSS

Figura 6: Componentes que intervienen en la Capa de Presentación.

Capa de Controladores

En esta capa se encuentra la Lógica de Presentación, las cuales según la arquitectura que propone Grails estarán enmarcadas las clases controladoras en los paquetes Controllers. Un controlador de Grails es una clase responsable por el manejo de los pedidos provenientes de la aplicación. El controlador recibe la petición, realiza un trabajo potencial con la misma y finalmente decide que sucederá a continuación, lo cual puede incluir algunos de los siguientes flujos.

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

Proveen la entrada principal para cualquier aplicación de Grails, coordinando los pedidos entrantes, delegando hacia los servicios o clases de dominio para la lógica de negocios y mostrar las vistas. (14)

Figura 7: Paquete de Controladores.

Capa de Servicios

En esta capa se encapsula toda la lógica de la aplicación, las funcionalidades implementadas en estas clases podrán ser utilizadas por el resto de las clases controladoras y los servicios. Sus clases radicarán según la arquitectura propuesta por Grails en el paquete Services.

Figura 8: Paquete de Servicios.

Capa de Datos

Maneja los objetos de acceso a datos abstrayéndolos del mecanismo de persistencia usado; a través de interfaces que exponen las operaciones de persistencia. Grails para evitar trabajar directamente con un gestor de base de datos y sus tablas, permite trabajar con objetos utilizando Hibernate como una herramienta ORM pero esta vez dada la naturaleza dinámica de Grails y la adopción del convenio sobre la configuración, crea sobre una versión superior de una nueva implementación de Hibernate llamado Grails objeto mapeo relacional (GORM) que simplifica el trabajo con Hibernate y elimina cualquier configuración externa. (19)

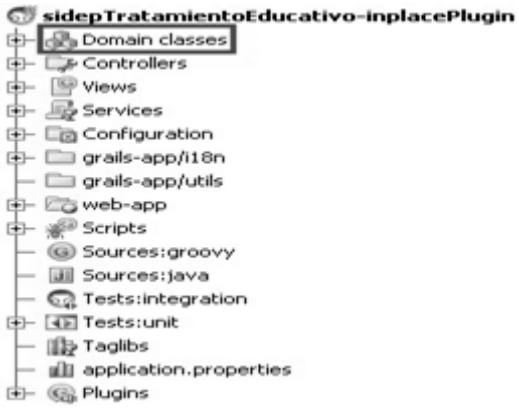

Figura 9: Paquete de Clases del Dominio.

2.8. Patrones de diseño

Un patrón es un par problema/solución con nombre que se puede aplicar en nuevos contextos, con consejos acerca de cómo aplicarlo en nuevas situaciones y discusiones sobre sus compromisos. Un patrón es una descripción de un problema bien conocido que suele incluir:

- Descripción.
- Escenario de Uso.
- Solución concreta.
- Las consecuencias de utilizar este patrón.
- Ejemplos de implementación.
- Lista de patrones relacionados. (7)

Alta cohesión

La cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.

Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo. No conviene este tipo de clases pues presentan los siguientes problemas:

- Son difíciles de comprender, reutilizar y conservar.
- Las clases con baja cohesión a menudo representan un alto grado de abstracción o han asumido responsabilidades que deberían haber delegado a otros objetos. (7)

Durante el desarrollo del software dicho patrón se manifiesta especialmente en el proceso de otorgamiento de beneficios. Teniendo en cuenta que son quince los beneficios a los que el interno tiene derecho y la gran parte de ellos son tratados de forma distinta, resulta indispensable la utilización de un controlador y un servicio asociado a él para atender cada uno de los beneficios. De esta manera el proceso de análisis de un beneficio se divide en varias clases, todas con el único objetivo de otorgar un beneficio al interno.

Bajo acoplamiento

Asigna una responsabilidad para mantener bajo acoplamiento. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras. (7)

A lo largo del proceso de desarrollo del módulo “Gestión del Consejo de Promoción” se ha puesto en práctica dicho patrón al asociar cada controlador a un servicio, en situaciones exclusivas podría estar relacionado un controlador a un máximo de cuatro servicios.

Singleton

Por defecto todos los servicios son singleton, ya que solo existe una instancia de la clase que se inyecta en todos los artefactos que declaran la variable correspondiente. Este criterio tiene como inconveniente que no se puede guardar información que sea “privada” de una petición en el servicio porque todos los controladores verían la misma instancia y el mismo valor. Este comportamiento se puede modificar utilizando la variable ”scope” asignándole el valor “session”, permitiendo crear una instancia nueva del servicio para cada sesión de usuario. (14)

Controlador

Al utilizar el patrón arquitectónico Modelo-Vista-Controlador se obtiene una capa específica designada para los controladores. Las clases controladoras son un ejemplo de estas y cada una de ellos son las encargadas de manejar los eventos y mensajes que se generan en la aplicación por ejemplo entre la capa de presentación y el modelo. (7)

Experto

Asigna una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad (7).

Este es uno de los patrones que implementa el framework Grails al utilizar GORM para el mapeo de objetos de la base de datos. Al generar las clases para la gestión de las entidades según las tablas de la base de datos con las responsabilidades debidamente asignadas, pues cada una de estas clases cuenta con un conjunto de funcionalidades relacionadas directamente con la entidad que representan.

Creador

Este patrón plantea asignarle a la clase B la responsabilidad de crear una instancia de la clase A (7). Dicho patrón se evidencia en las clases controladoras o servicios que se utilizan en los módulos o funcionalidades de la aplicación. Estas clases se encargan de la creación de las instancias de las clases que describen los objetos que en ellos se manejan, ya que estos o agregan, contienen, registran o utilizan estos objetos. Esto favorece la reutilización y favorece el bajo acoplamiento.

2.9. Diagrama de Clases

2.9.1. Descripción de las clases

Clases del Dominio

Nombre de la entidad	ConsejoPromocion	
Descripción de la entidad	Almacena los datos correspondientes a una reunión de Consejo de Promoción.	
Nombre del atributo	**Descripción**	**Tipo**
fechaConPromocion	Este campo guarda la fecha en que se realiza la reunión	Date

numeroActa	Este campo guarda el número de acta que asigna el sistema	double
anno	Este campo guarda el año en que se realiza la reunión	int

Tabla 3: Entidad ConsejoPromocion.groovy.

Nombre de la entidad	Beneficio	
Descripción de la entidad	Almacena los datos correspondientes a un Beneficio.	
Nombre del atributo	**Descripción**	**Tipo**
numeroDecision	Este campo guarda el número de la decisión que constituye la aprobación o denegación de dicho beneficio.	int
tipoBeneficio	Este campo guarda el tipo de beneficio que se le propone al interno.	NomTipoBeneficio
consejoPromocion	Este campo guarda el Consejo de Promoción al que está asociado dicho beneficio.	ConsejoPromocion
expediente	Este campo guarda el número de expediente legal al que está asociado dicho beneficio.	Expedientelegal

Tabla 4: Entidad Beneficio.groovy.

Nombre de la entidad	Propuesta	
Descripción de la entidad	Almacena los datos correspondientes a la propuesta de un beneficio a un interno.	
Nombre del atributo	**Descripción**	**Tipo**
tipoBeneficio	Este campo guarda el tipo de beneficio que se le propone al	NomTipoBeneficio

	interno.	
expedientelegal	Este campo guarda el número de expediente legal al que se le propone el beneficio.	Expedientelegal

Tabla 5: Entidad Propuesta.groovy.

Clases controladoras

Nombre de la clase	RegistrarConsejoController
Descripción de la clase	Contiene todas las acciones para registrar una reunión de Consejo de Promoción.
Acciones	**Descripción**
index()	Muestra la vista registrarconsejo.gsp
comenzarConsejo()	Crea una instancia de la clase ConsejoPromocion.groovy

Tabla 6: Clase controladora RegistarConsejoController.groovy.

Nombre de la clase	PropuestaBeneficioController
Descripción de la clase	Contiene todas las acciones para la propuesta de un beneficio a un interno.
Nombre del atributo	**Descripción**
propuestaBeneficioService	Constituye una instancia de la clase PropuestaBeneficioService.groovy.
Acciones	**Descripción**
inicio()	Muestra la vista index.gsp. La cual contiene propuestabeneficio.gsp y consultarpropuestas.gsp.
mostrarDatosInterno()	Obtiene el identificador del expediente del interno seleccionado, posteriormente realiza una búsqueda del interno utilizando como criterio el identificador, una vez encontrado el interno se muestran sus atributos: Nombre(s) y Apellidos y el número de expediente legal.

Tabla 7: Clase controladora PropuestaBeneficioController.groovy.

Componentes de interfaz de usuario

Nombre de la vista	registrarconsejo.gsp
Descripción de la vista	Muestra los campos referentes a un Consejo de Promoción.
Componentes	**Descripción**
consejopromocionForm	Este formulario agrupa los datos referentes a la entidad ConsejoPromocion.
consejo.anno	Este campo guarda el año en que se realiza la reunión.
consejo.fecha	Este campo guarda la fecha en que se realiza la reunión.
consejo.acta	Este campo guarda el número de acta que asigna el sistema.
radiosBeneficios	Este componente contiene los tipos de beneficios.
benSinTermino	Este campo contiene todos los beneficios que pertenecen a la categoría "Sin Término'.
benConTermino	Este campo contiene todos los beneficios que pertenecen a la categoría "Con Término'.
tablaInternos	Muestra los internos que corresponden al beneficio seleccionado anteriormente.

Tabla 8: Clase de presentación registrarconsejo.gsp.

Nombre de la vista	propuestabeneficio.gsp
Descripción de la entidad	Muestra los campos referentes a una Propuesta.
Componentes	**Descripción**
propuestabeneficioForm	Este formulario agrupa los datos referentes a la entidad Propuesta.
interno.nombre	Este campo guarda el nombre completo del interno.
expedientelegal.numeroexpediente	Este campo guarda el número de expediente del interno.
tipobeneficio.id	Este campo guarda el valor del Tipo de Beneficio seleccionado.
internos.tabla.id	Es una tabla la cual lista todos los internos.

Tabla 9: Clase de presentación propuestabeneficio.gsp.

Nombre del fichero	PropuestaBeneficio.js
Descripción del	Valida los campos referentes a una Propuesta.

fichero	
Componente	**Descripción**
mostrarDatos(params)	Le asigna valores a los campos según los datos del interno.
guardar()	Envía los datos del formulario.
validar()	Es la función que valida los campos correspondientes a una Propuesta, exigiendo que no se envíen en blanco y que se seleccione algún valor en el select tipo de beneficio. En cualquier caso, ordena poner los campos obligatorios en rojo hasta que sean completados y muestra un mensaje diciendo que los campos en rojo son requeridos.

Tabla 10: Fichero Java script PropuestaBeneficio.js.

Clase de servicios

Nombre de la entidad	RegistrarConsejoService.groovy	
Descripción de la entidad	Almacena en la base de datos las instancias de la clase ConsejoPromocion.	
Atributos	Tipo	Descripción
scope = "session"		Se crea una instancia nueva de RegistrarConsejoService para cada sesión del usuario.
numeroDecision	int	Toma el valor de la longitud de la lista de Beneficios más uno.
listaBeneficios	List	Almacena los beneficios sobre los cuales el tribunal ha emitido una decisión.
consejoPromocion	ConsejoPromocion	Toma el valor del Consejo de Promoción que se encuentra en ejecución.
solBeneficioService	SolBeneficioService	Constituye una instancia de la clase SolBeneficioService, creada con el objetivo de llamar al método donde se crea la solicitud del beneficio.
Acciones	Tipo	Descripción
crearConsejo ()	void	Crea una instancia de la clase ConsejoPromocion, la

		valida y la almacena en la base de datos.
valorDecision()	int	Devuelve el número de beneficios analizados más uno, asignándoselo al beneficio que será próximamente atendido.
terminarConsejo()	void	Guardar el objeto de tipo Consejo de Promoción.
getNumeroActa()	int	Devuelve el número de consejos realizados en un mes más uno, asignándoselo al consejo que será próximamente efectuado.
guardarBeneficio()	void	Se comprueba si el beneficio fue anteriormente analizado, en caso de existir y se está valorando nuevamente, la segunda decisión suprime a la ya existente.

Tabla 11: Clase de Servicio RegistrarConsejoService.groovy.

2.9.2. Diagrama de diseño con estereotipos web.

El Diagrama de Clases es el diagrama principal de diseño y análisis para un sistema. En él, la estructura de clases del sistema se especifica, con relaciones entre clases y estructuras de herencia. Durante el análisis del sistema, el diagrama se desarrolla buscando una solución ideal. Durante el diseño, se usa el mismo diagrama y se modifica para satisfacer los detalles de las implementaciones. (23)

A continuación se muestran todas las entidades que intervienen en el proceso de registrar el consejo de promoción indicando las relaciones entre ellas. El CU consiste en registrar dicha reunión además de todos los beneficios sobre los cuales el tribunal emitió una decisión.

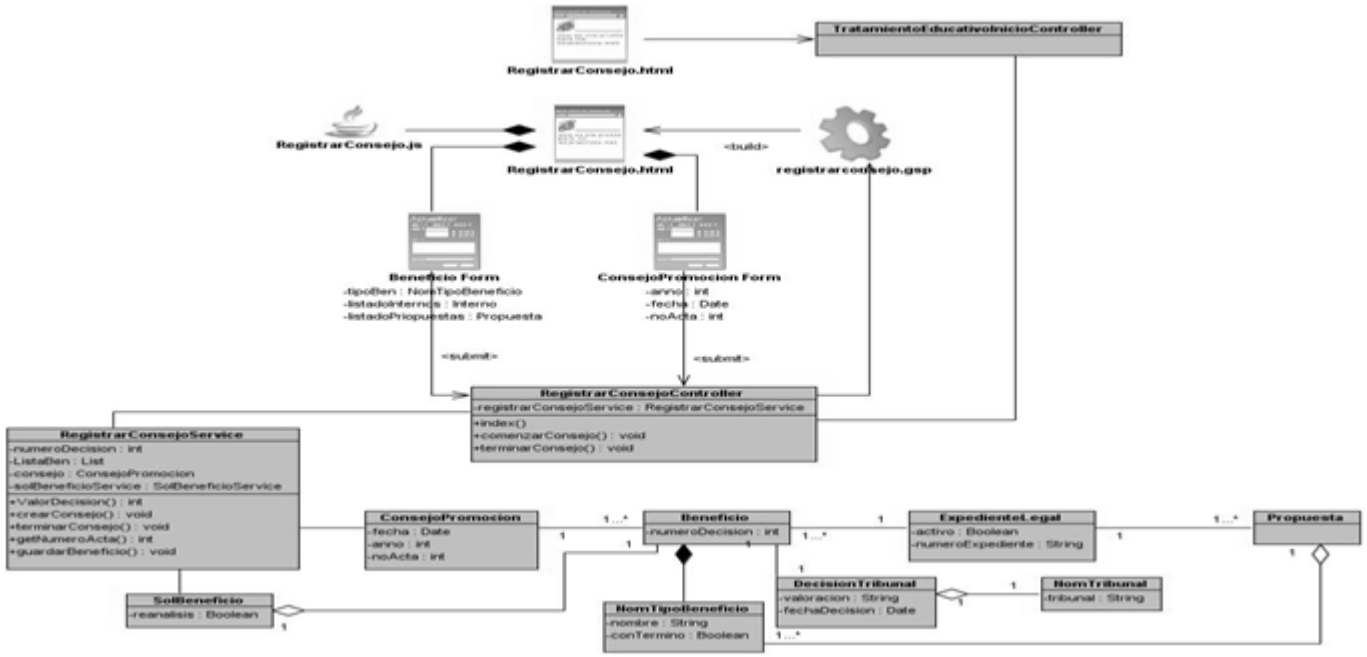

Figura 10: Diagrama de clases del caso de uso Diseño Registrar reunión de Consejo de Promoción.

A continuación una breve descripción de la relación que existe entre las entidades que intervienen en el proceso de registrar la reunión del consejo de promoción:

Todos los beneficios analizados en la reunión esta asociados a un único Consejo de Promoción **(ConsejoPromocion)**. Un beneficio **(Beneficio)** está asociado a un expediente legal **(ExpedienteLegal)** el cual contiene los datos del interno, los beneficios pueden ser de dos tipos **(NomTipoBeneficio)**: con término, cuando la fecha para otorgar el beneficio es calculable y sin término, cuando puede recibir el beneficio en cualquier período de su condena. Por último un beneficio tiene asociado a él una decisión de tribunal **(DecisionTribunal)** registrándose la evaluación emitida y el tribunal que interviene. También se encuentra implicada la entidad **Propuesta**, la misma está compuesta por un expediente legal y un tipo de beneficio asociado a él.

La entidad **SolBeneficio** guarda las solicitudes emitidas sobre un beneficio; según el tipo de beneficio y el interno que esta propuesto para recibirlo, debe obtener la evaluación de un nivel de aprobación particular, por ejemplo, todas las rebajas de sanciones son evaluadas por el Jefe de Unidad mientras que la libertad condicional es atendida por el Tribunal.

2.10. Diagramas de Interacción

Muestran una interacción, que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que puedan ser realizados entre ellos. Son importantes para modelar los aspectos dinámicos de un sistema y para construir sistemas ejecutables a través de ingeniería hacia adelante e inversa.

Tienen como objetivo visualizar, especificar, construir y documentar los aspectos dinámicos de una sociedad particular de objetos, o pueden ser usados para modelar un flujo particular de control de un caso de uso.

Los diagramas de interacción agrupan los diagramas de secuencia y de colaboración, siendo este último utilizado para reflejar el flujo de datos en cada uno de los casos de uso.

A continuación se expone el diagrama de colaboración referente al caso de uso “registrar reunión del Consejo de Promoción”.

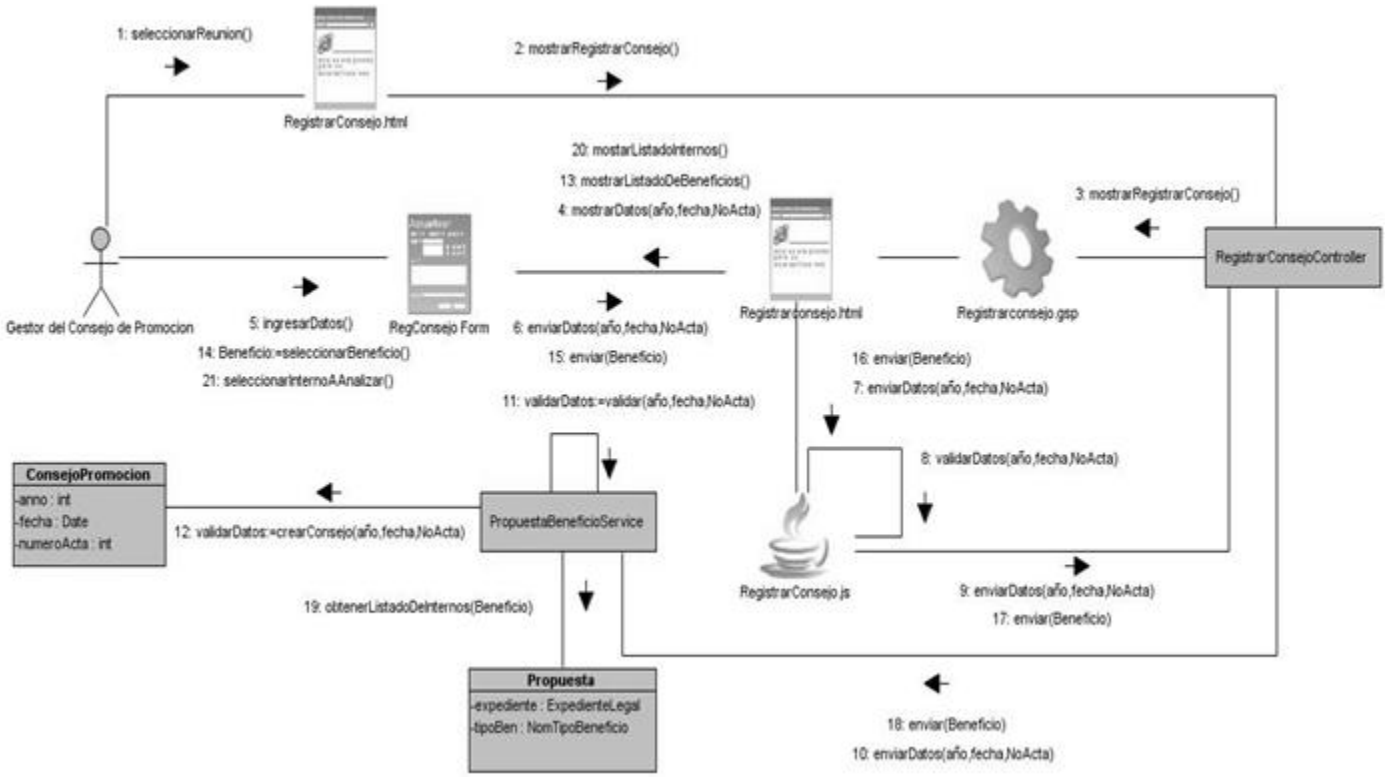

Figura 11: Diagrama de Colaboración del caso de uso Registrar reunión de Consejo de Promoción.

A continuación la descripción del flujo de datos que intervienen para registrar un nuevo consejo de promoción.

Inicialmente se selecciona la opción Registrar Consejo dentro del menú Consejo de Promoción, una vez que se muestra la interfaz el usuario debe introducir los datos del consejo, después de ser validados los datos en el cliente, son enviados al controlador **RegistarConsejoController** para posteriormente enviarlos al servicio **RegistrarConsejoService** siendo en esta calse validados nuevamente en el servidor y posteriormente se crea una instancia de la entidad **ConsejoPromocion**. Una vez creado el consejo se selecciona el tipo de beneficio que será analizado y posteriormente se muestran los internos propuestos para recibir dicho beneficio, finalmente se selecciona el interno y comienza su evaluación.

2.11. Diseño de la Base de datos

Las Bases de Datos necesitan una definición de su estructura que le permitan almacenar datos, reconocer el contenido, y recuperar la información. El artefacto resultante de esta actividad será el Modelo de Datos, que describe la representación lógica y física de los datos persistentes.

La imagen que se muestra a continuación contiene el diseño de la base de datos para módulo Gestión del Consejo de Promoción.

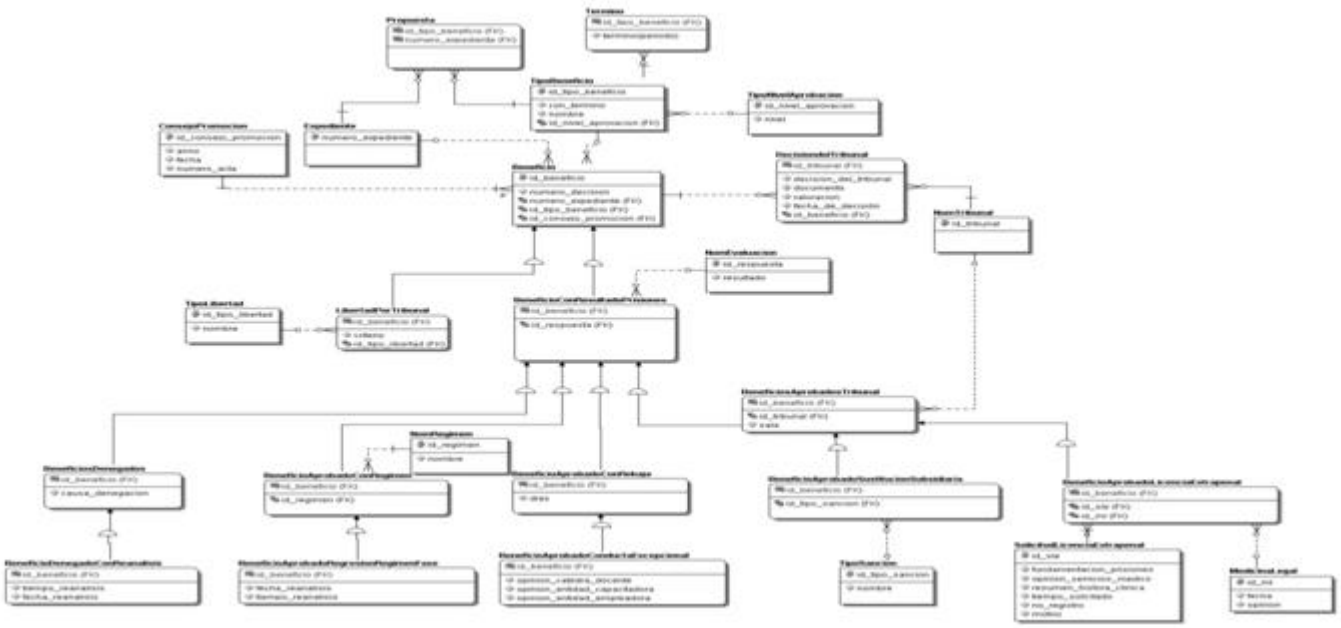

Figura 12: Diseño de la Base de Datos.

En la figura anterior se pueden apreciar las tablas que pertenecen al módulo “Gestión del Consejo de Promoción” siendo un total de 23.

En el presente modelo de datos se denota la herencia asociada a los beneficios, ya sea para su aprobación o para la denegación. La entidad **BenResultadoPrisiones** contiene la evaluación emitida sobre el beneficio. Para guardar los datos de la denegación del beneficio tenemos la entidad **BeneficiosDenegados** la cual contiene la causa de la denegación y la entidad **BenDenReanalisis** atiende a los beneficios que pueden ser reanalizados. Los beneficios asociados a las rebajas y al régimen son atendidos por las entidades **BenAprobRebaja** y **BenAprobRegimen** respectivamente. Por último la entidad **BenAprobTribunal** contiene los datos del tribunal que emite la decisión sobre el beneficio.

2.11.1. Descripción de las tablas

A continuación se presentan las tablas que intervienen en el CU Registrar reunión del Consejo de Promoción.

Nombre de la tabla	CONSEJO_PROMOCION	
Descripción de la entidad	Almacena los datos correspondientes a una reunión de Consejo de Promoción.	
Nombre del atributo	**Descripción**	**Tipo**
FECHA_CON_PROMOCION	Este campo guarda la fecha en que se realiza la reunión.	DATE
NUMERO_ACTA	Este campo guarda el número de acta que asigna el sistema.	FLOAT
ANNO	Este campo guarda el año en que se realiza la reunión.	FLOAT

Tabla 12: Tabla de la Base de Datos CONSEJO_PROMOCION.

Nombre de la tabla	BENEFICIO	
Descripción de la entidad	Almacena los datos correspondientes a un Beneficio.	
Nombre del atributo	**Descripción**	**Tipo**
NUMERO_DECISION	Este campo guarda el número de	NUMBER(10,0)

	la decisión que constituye la aprobación o denegación de dicho beneficio.	
TIPO_BENEFICIO_ID	Este campo guarda el tipo de beneficio que se le propone al interno.	NUMBER(19,0)
CONSEJO_PROMOCION_ID	Este campo guarda el Consejo de Promoción al que está asociado dicho beneficio.	NUMBER(19,0)
EXPEDIENTE_ID	Este campo guarda el número de expediente legal al que está asociado dicho beneficio.	NUMBER(19,0)

Tabla 13: Tabla de la Base de Datos BENEFICIO.

Nombre de la tabla	PROPUESTA	
Descripción de la entidad	Almacena los datos correspondientes a la propuesta de un beneficio a un interno.	
Nombre del atributo	**Descripción**	**Tipo**
TIPO_BENEFICIO_ID	Este campo guarda el tipo de beneficio que se le propone al interno.	NUMBER(19,0)
EXPEDIENTELEGAL_ID	Este campo guarda el número de expediente legal al que se le propone el beneficio.	NUMBER(19,0)

Tabla 14: Tabla de la Base de Datos PROPUESTA.

2.12. Conclusiones

Con la realización de este capítulo, se arribó a la conclusión que los artefactos propuestos por los analistas permitieron adquirir una mejor comprensión de todo lo relacionado con las funcionalidades a implementar. Además se realizaron un conjunto de descripciones de las principales clases y funcionalidades que fueron utilizadas.

Se obtuvo el diseño de una propuesta de solución tangible, que gestionará el proceso de otorgamiento de beneficios a un interno y ayudará a los oficiales encargados de la gestión del consejo de promoción en la toma de decisiones y en las operaciones de seguimiento y control.

Capítulo 3

Implementación y Prueba

3.1. Introducción

Teniendo en cuenta el análisis de la propuesta de solución del sistema así como el diseño de la misma, a continuación se procede a documentar el flujo de implementación desarrollado y posteriormente las validaciones realizadas al sistema. En este capítulo se presentan los diagramas de despliegue y de componentes para la implementación del sistema. Se definen los tipos de pruebas y los casos de prueba que permitirán evaluar y valorar la calidad del producto.

3.2. Implementación

El flujo de trabajo implementación comienza con el resultado del diseño y se realiza el sistema en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue. Los diagramas de despliegue y componentes conforman lo que se conoce como un modelo de implementación ya que describen los componentes a construir, su organización y dependencia entre nodos físicos en los que funcionará la aplicación.

El objetivo principal de esta etapa es desarrollar la arquitectura y el sistema como un todo, de la forma más específica. Teniendo en cuenta los propósitos de la implementación:

- Definir la organización del código.
- Planificar las integraciones del sistema necesario en cada iteración.
- Implementar las clases que componen el módulo generado durante el diseño.

El modelo de Implementación es la unión de varios artefactos que se generan de gran valor, pues denota la implementación actual del sistema en términos de componentes y subsistemas de implementación, es natural mantener el mismo a lo largo de todo el ciclo de vida del software.

3.2.1. Diagrama de Despliegue

El Modelo de Despliegue describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos y dónde el sistema será puesto en funcionamiento. Las estaciones de

trabajo, dispositivos y procesadores son reflejados como nodos y su estructura interna puede ser representada adicionando otros nodos o artefactos. El modelo de despliegue se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño.

A continuación se muestra el diagrama de despliegue referente al módulo Gestión del Consejo de Promoción, teniendo en cuenta los distintos nodos físicos que intervienen y especificando la infraestructura necesaria, la inclusión de estos términos en el diagrama fue propuesto por el equipo de dirección del SIDEP.

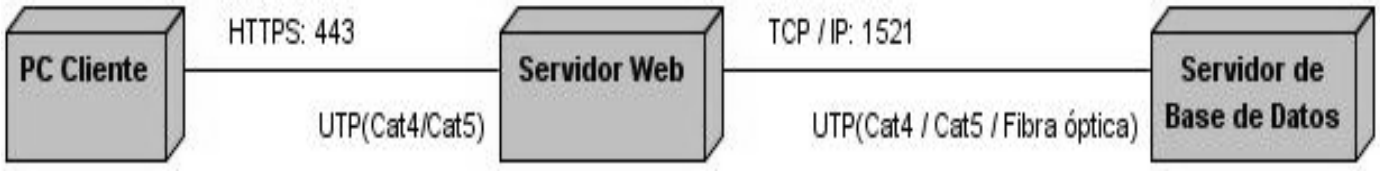

Figura 13: Modelo de Despliegue del módulo "Gestión del Consejo de Promoción".

Para el despliegue del módulo el cliente cuenta con una estructura tecnológica, donde radican el Servidor de aplicaciones y el de Base Datos.

Servidor de Aplicación:

- Trabajará sobre la distribución de Linux Suse Enterprise Server 10.0 SP2 y el servidor web Apache Tomcat 6.0.3.
- Microprocesador: 4 núcleos, 3 GHz
- RAM: 4 GB
- Espacio necesario para la instalación: 250 MB
- Espacio libre requerido: 250 GB
- JDK 1.6

Servidor de Bases de datos:

- Microprocesador: 4 núcleos, 3 GHz
- RAM: 4 GB
- Espacio necesario para la instalación: 2 GB
- Espacio libre requerido: 1 TB

- JDK 1.6
- Se utilizará Oracle 11G Enterprise Edition como sistema gestor de base de datos.

El cliente hará uso de la aplicación a través de clientes ligeros con sistema operativo Windows XP SP3 y navegador Mozilla Firefox 3.6.

3.2.2. Diagrama de Componentes

Los diagramas de componentes son usados para estructurar el modelo de implementación en término de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. Los diagramas de componentes muestran los elementos de diseño de un sistema de software. Permiten visualizar la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan. (24)

El uso más importante de estos diagramas es mostrar la estructura de alto nivel del modelo de implementación, especificando:

- Las organizaciones y las dependencias entre tipos de componentes.
- Organización de los subsistemas de implementación en capas.

A continuación se representan los paquetes de componentes que se relacionan con el módulo "Gestión del Consejo de Promoción". La interacción entre los distintos subsistemas y el módulo en desarrollo es de vital importancia, teniendo en cuenta que varios de los procesos que se realizan en el Consejo de Promoción son respaldados por la información que brindan otros módulos.

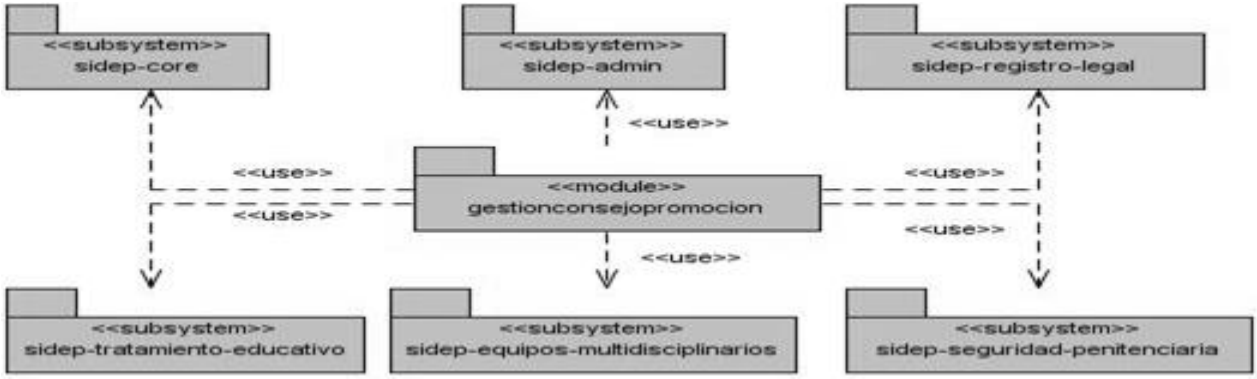

Figura 14: Diagrama de los subsistemas que se relacionan con el módulo "Gestión del Consejo de Promoción".

A continuación se exponen las dependencias anteriormente mencionadas:

Módulo del que depende / Subsistema	Descripción
Permisos / Tratamiento Educativo	Se determina el tipo de permiso que se le otorgará a un interno. Así como poder consultar el listado de permisos que se han otorgado.
Trabajo / Tratamiento Educativo	Se determina el tipo de beneficio relacionado con el trabajo de los internos, teniendo en cuenta el régimen.
Solicitudes / Registro Legal	Se necesita para el envío de solicitudes de algunos de los beneficios que son elevados a otros niveles para ser revisados y emitir una decisión.
Comunes / Core	Se necesita buscar los internos para la propuesta de los beneficios sin términos.
Incidencias / Seguridad Penitenciaria	Se necesita controlar las incidencias en las que incurren los internos que serán analizados en el Consejo de Promoción.
Indisciplinas y medidas disciplinarias / Seguridad Penitenciaria	Se necesita controlar las indisciplinas y las correspondientes medidas disciplinarias en las que estén relacionados los internos que serán analizados en el Consejo de Promoción.
Decisiones / Registro Legal	Se deben controlar las decisiones relacionadas con el Consejo de Promoción.
Opinión Fundamentada / Equipos Multidisciplinarios	En algunos casos es necesario tener en cuenta la Opinión Fundamentada de acuerdo al interno que se analizará en el Consejo

de Promoción.

Seguidamente se representan los componentes que se relacionan dentro del módulo “Gestión del Consejo de Promoción”.

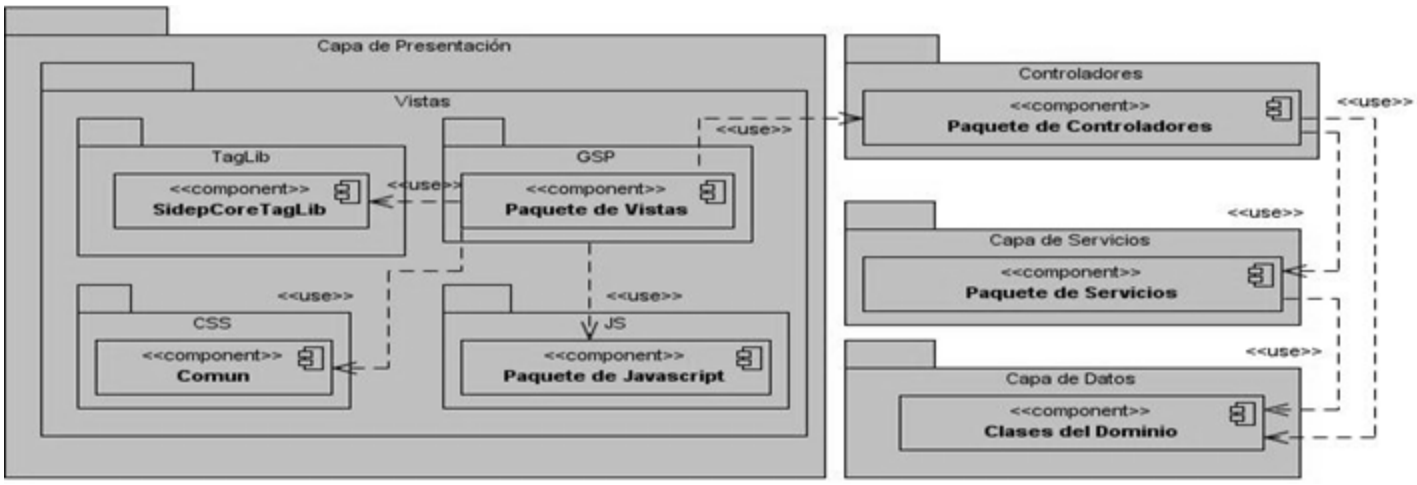

Figura 15: Diagrama de componentes que intervienen en el módulo “Gestión del Consejo de Promoción”.

La distribución de componentes se realizó por paquetes coincidiendo con las capas lógicas que caracterizan la arquitectura de la aplicación.

- Capa de Presentación
- Capa de Controladores
- Capa de Servicios
- Capa de Datos

Posteriormente se representan los componentes que intervienen en el caso de uso “Registrar reunión de Consejo de Promoción”.

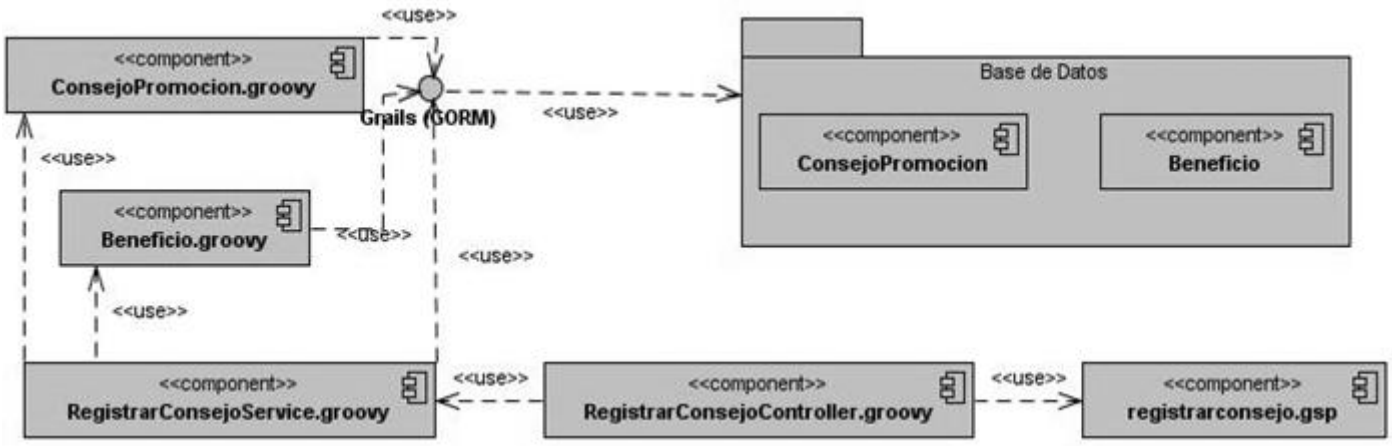

Figura 16: Diagrama de componentes que intervienen en el Caso de Uso "Registrar reunión de Consejo de Promoción".

La clase controladora **RegistarConsejoController.groovy** tiene como principal objetivo mostrar la vista **registarconsejo.gsp** además de gestionar todas las tareas realizadas en la reunión del Consejo de Promoción. La clase **RegistarConsejoService.groovy** obtiene los datos de la clase controladora para posteriormente registrarlos en la base de datos mediante las entidades del dominio, las cuales a su vez interactúan con una interfaz que brinda Grails encargada de todos los procesos de consultas y persistencia de datos que intervienen entre las entidades y las tablas de la base de datos.

Seguidamente se exponen los aspectos fundamentales de la implementación de cada una de las capas que componen la arquitectura del sistema:

3.2.3. Implementación de la Capa de Presentación

Vistas

La vista es la responsable de mostrar al usuario el estado del sistema y las acciones que tiene a su disposición. En Grails desarrollamos las vistas mediante **Groovy Server Pages**, es un archivo con extensión gsp(son los únicos componentes de Grails que no son Groovy ni Java) que reside en la carpeta **grails-app/views** del proyecto (14). La composición de las vistas depende de la inclusión de etiquetas, las cuales pueden ser:

- Etiquetas HTML.
- Etiquetas definidas por Grails.
- Etiquetas personalizadas por el equipo de desarrollo.

Etiquetas para crear vínculos

Entre las librerías de etiquetas GSP encontramos varias que nos permiten crear la conexión a ficheros Java Script.

```
<g:javascript src="${resource(dir: 'js/sidep/tratamientoeducativo/gestionconsejopromocion',
                              file: 'ConsultarPropuestas.js')}" />
```

Figura 17: Fragmento de la vista consultarpropuestas.gsp donde se realiza la conexión con el fichero ConsultarPropuestas.js.

Etiquetas para formularios

Grails incluye multitud de etiquetas relacionadas con formularios, para crear campos de todo tipo desde los básicos incluidos en HTML:

- **form** y **uploadForm**: Ambas crean una etiqueta **form**. La diferencia es que la segunda añade **enctype="multipart/form-data"** para permitir el envío de archivos al servidor.

```
<form name="consultarBenForm" id="consultarBenForm.id" enctype="multipart/form-data"
    action="${createLink(controller: 'consultarBeneficios', action: 'guardarDatosTribunal')}" class="contorno" ">
```

Figura 18: Utilización de la etiqueta form en la vista consultarbeneficios.gsp

Etiquetas lógicas y de iteración

Permiten la inclusión opcional de código en las vistas:

```
<g:if test="${nivelAprobacion}">
    <s:input name="det_nivelAprobacion"... />
    <s:dateBox name="det_fechaNivelAprob"... />
    <s:input name="det_evalNivelAprob"... />
    <s:clear/>
</g:if>
```

Figura 19: Uso de la etiqueta g: if en la vista detbentribunal.gsp

La etiqueta **g:if** comprueba el valor de la variable "nivelAprobacion" enviada desde el controlador para posteriormente mostrar varios campos.

Internacionalización

Internacionalizar una aplicación significa diseñarla de tal forma que la interfaz de usuario soporte distintos idiomas sin que sea necesario modificar el código fuente.

Con grails existe soporte para la internacionalización (o i18n, como dicen en Estados Unidos, poniendo la i, la n y el número de letras que hay en medio, 18) por medio de archivos de recursos almacenados en la carpeta grails-app/i18n. Los archivos de recursos son ficheros de java en los que se guardan las distintas versiones de cada mensaje en todos los idiomas soportados por la aplicación. Cada idioma está definido en un fichero distinto que incluye en su nombre el Local del idioma, por ejemplo:

Messages_es_AR.properties para los mensajes en español de Argentina.
Messages_en.properties para los mensajes en inglés.
Messages_es_es.properties para los mensajes en español de España.
Desde el momento en que se crea el proyecto grails coloca en la carpeta correspondiente archivos de mensajes para 11 idiomas distintos.
Con escribir en el navegador el comando ?lang=es se cambia el idioma de los componentes de la aplicación a español. (14)

En el SIDEP solo se efectuó la internacionalización de los mensajes en español teniendo en cuenta que el cliente es el MININT.

3.2.4. Implementación de la Capa de Controladores

Los controladores son los responsables de recibir las solicitudes del usuario y decidir la vista que se debe mostrar a continuación. Podemos generar controladores mediante el comando **grails create-controller**: En los controladores utiliza el método **render** para enviar respuestas al cliente. De los parámetros que se pueden enviar a la vista los más empleados durante el desarrollo del software fueron:

- **View**: La vista que queremos procesar para generar la petición.
- **Model**: Un Map[9] con el modelo para usar en la vista.
- **Converter**: Un objeto grails.converters.* para generar la respuesta.

[9] Una lista de objetos en las que se establece el par propiedad / valor.

```
def index = {
    render view: "/gestionconsejopromocion/consejodepromocion/registrarconsejo",
           model: [listaBenSinTerm: tipoBeneficioService.listaBenSinTermino(),
                   listaBenConTerm: tipoBeneficioService.listaBenConTermino(),
                   numeroActa: registrarConsejoService.getNumeroActa()
           ])
}
```

Figura 20: Ejemplo de utilización de los parámetros view y model.

```
if (registrarConsejoService.crearConsejo(cp))
    json["respuesta"] = true;
else
    json["respuesta"] = false;
render json as JSON
```

Figura 21: Ejemplo de utilización del parámetro converter.

3.2.5. Implementación de la Capa de Servicios

Según la convención que sigue grails, un servicio es una clase cuyo nombre termina en Service y que se aloja en la carpeta garils-app/services. Podemos generar servicios mediante el comando **grails create-services**.

En tiempo de ejecución, Grails usará las funcionalidades del contenedor Spring para hacer que todas las clases que declaren una variable con el mismo nombre que el servicio, tengan una instancia a su disposición. Declarando una variable con el mismo nombre del servicio pero comenzando con minúscula creamos una instancia del servicio en nuestro controlador.

```
class RegistrarConsejoController {
    def registrarConsejoService
```

Figura 22: Fragmento donde se evidencia la creación de instancia del servicio.

Por defecto todos los servicios son singleton: sólo existe una instancia de la clase que se inyecta en todos los artefactos que declaren la variable correspondiente.

Podemos modificar este comportamiento declarando una variable “scope” en el servicio con el valor:

- session: Se creará una instancia del servicio para cada sesión del usuario. (14)

Por defecto todos los servicios son transactional: esto quiere decir que si ocurre una excepción antes de que se complete el método no se realizará ningún cambio en la base de datos, preservando la integridad de los datos. Durante el desarrollo del módulo se utilizaron las transacciones específicamente en el caso de uso “Registrar reunión del Consejo de Promoción”, todos los beneficios que son analizados están asociados a un único consejo por lo que cuando se decida registrar una instancia de la entidad **Beneficio** debe haber persistido anteriormente de forma correcta la instancia de la entidad **ConsejoPromocion**.

3.2.6. Implementación de la Capa de Datos

Implementación de las clases del Dominio

Las clases del dominio se implementan en el paquete grails-app\domain, obteniendo así el modelo de datos. Grails utiliza GORM para controlar el ciclo de vida de las entidades y proporcionar una serie de métodos dinámicos que facilitan las búsquedas. GORM está construido sobre Hibernate, una herramienta de mapeo objeto-relacional que se encarga de relacionar las entidades de las clases con tablas de la base de datos y las propiedades de las entidades con las columnas de las tablas de las base de datos. (14)

Crear Entidades

Todas las clases groovy que se encuentran en la carpeta grails-app\domain serán tratadas por GORM como entidades, se definen sus propiedades y GORM se encarga de generar la tabla correspondiente en la base de datos con los campos necesarios para almacenar cada propiedad y proporcionar los métodos de búsqueda y modificación que permiten manipular la entidad. También inserta un identificador automáticamente a la tabla. (14)

Validaciones

GORM permite restringir los valores que pueden asignarse a las propiedades a través de la palabra reservada constrainst (“restricciones”), la cual define las reglas de validación para cada atributo.

```
class ConsejoPromocion {
    Date fechaConPromocion
    double numeroActa
    double anno

    static hasMany = [beneficios: Beneficio]

    static constraints = {
        fechaConPromocion(blank: false)
        numeroActa(blank: false, min: (double) 0)
        anno(blank: false, min: (double) 2008)

    }
    static mapping = {
        id column: 'consejoPromocion_id'
    }
}
```

Figura 23: Clase del Dominio "ConsejoPromoción.groovy"

Dentro de la propiedad estática **constraints** se pueden restringir los atributos de la clase del dominio por ejemplo:

- **fechaConPromocion (blank: false)**: La propiedad **blank: false** valida que este atributo siempre tenga valor para posteriormente persistir esta entidad del dominio en la base de datos.
- **anno (blank: false, min (double) 2008)**: La propiedad **min** restringe que el valor mínimo de la variable anno sea 2008.

Operaciones sobre el modelo de datos

Actualizaciones: Todas las entidades en la aplicación tienen métodos de instancia que facilitan su manipulación: dentro de los que figuran save () y delete ().

- **save():** Se utiliza para insertar el registro en la base de datos o para actualizarlo si ya existía.

```
def terminarConsejo(ConsejoPromocion cp) {
    boolean respuesta = false
    cp = Consejo()
    if (cp.validate()) {
        respuesta = cp.save()
        listaBeneficios.each {
            it.save()
        }

    }
    listaBeneficios = []
    respuesta
}
```

Figura 24: Método para guardar la entidad ConsejoPromocion.groovy

- **delete():**Se utiliza para eliminar un registro.

```
def eliminarPropuesta(Propuesta propuesta) {
    if (propuesta) {
        propuesta.delete()
        return true
    }
    return false
}
```

Figura 25: Método para eliminar una instancia de la entidad Propuesta. groovy.

Relaciones Uno-A-Muchos

Una relación de uno a muchos es cuando una clase, por ejemplo ConsejoPromocion.groovy, tiene varias instancias de otra clase, por ejemplo Beneficio.groovy. Con Grails se puede definir con la propiedad **hasMany**.

Uno

La forma de representar esta relación en el modelo de datos es la siguiente:

```
class ConsejoPromocion {
    Date fechaConPromocion
    double numeroActa
    double anno

    static hasMany = [beneficios: Beneficio]

    static constraints = {
        fechaConPromocion(blank: false)
        numeroActa(blank: false, min: (double) 0)
        anno(blank: false, min: (double) 2008)

    }
    static mapping = {
        id column: 'consejoPromocion_id'
    }
}
```

Figura 26: Clase del dominio ConsejoPromocion.groovy

La propiedad **hasMany** indica que cada Consejo de Promoción tiene asociado a él uno o varios beneficios.

Muchos

```
class Beneficio {
    int numeroDecision
    NomTipoBeneficio tipoBeneficio
    ConsejoPromocion consejoPromocion
    DecisionTribunal decisionTribunal

    }
}
```

Figura 27: Clase del dominio Beneficio.groovy

Todos los beneficios están asociados a un único Consejo de Promoción.

Métodos estáticos

Grails contiene un conjunto de métodos que facilitan las operaciones de consultas sobre los datos almacenados en la tabla o tablas correspondientes.

- **count()**: Se utiliza para contar los elementos de una entidad.

```
//cantidad de consejos efectuados en el mes
def cantidadConsejos = ConsejoPromocion.countByFechaConPromocionBetween(primerDia - 1, ultimoDia + 1)

if (cantidadConsejos == 0)
    return 1
else
    return cantidadConsejos + 1
```

Figura 28: Uso del método count() aplicado a la entidad ConsejoPromocion.groovy.

La expresión señalada en la imagen tiene como objetivo contar las instancias de la entidad ConsejoPromocion.groovy cuya fecha este comprendida entre dos días especificados por parámetros.

Cuando no conocemos el identificador de la instancia o instancias que buscamos, necesitamos hacer consultas, que pueden enfocarse de maneras distintas en función de la complejidad que necesitemos.

A continuación se exponen algunos de los principales métodos para personalizar las búsquedas.

Dynamic Finders("localizadores dinámicos")

En los servicios se usaron los métodos **findBy()** y **findAllBy(),** los cuales son personalizados resaltando el atributo de la entidad, por ejemplo, **findByAtributo()**. La diferencia es que el primero solo devuelve una instancia que cumpla la condición de búsqueda, mientras que el segundo devolverá una lista con todos los resultados que correspondan. Estos localizadores son muy potentes para realizar consultas básicas

sobre el modelo de datos, pero no permiten hacer búsquedas avanzadas que incluyen muchos campos y comparaciones complejas. Para esto se recomienda construir consultas propias mediante criteria. (14)

```
def tipoBen = NomTipoBeneficio.findById('2'.toLong())
```

Figura 29: Fragmento donde se evidencia la utilización de los dynamic finders.

Criteria

Permite hacer búsquedas avanzadas que incluyen varios campos y comparaciones complejas.

```
void buscarBenConsejoPromocion() {
    listaBen = []
    listaBen = Beneficio.createCriteria().list() {
        and {
            consejoPromocion {
                eq("fechaConPromocion", cp?.fechaConPromocion)
                eq("numeroActa", cp?.numeroActa)
                eq("anno", cp?.anno)
            }
            tipoBeneficio {
                eq("id", tipoben?.id)
            }
        }
    }

}
```

Figura 30: Criteria aplicado a la entidad Beneficio.groovy

3.3. Pruebas

Objetivos de las pruebas

Las pruebas de software tienen como objetivos detectar errores no encontrados hasta el momento en la aplicación. Se puede hablar entonces del éxito de las pruebas siempre y cuando se hallen errores en el software. Con las pruebas se puede además observar hasta qué punto el software parece funcionar en concordancia con los requisitos funcionales descritos para el sistema; aunque no pueden asegurar la ausencia de defectos, sólo puede mostrar que existen.

Tipos de Prueba

Dentro de los tipos de prueba definidos para ser aplicados a la propuesta de solución se encuentran:

Funcionalidad

- **Función: E**stas pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso.
- **Seguridad:** Asegurar que los datos o el sistema solamente es accedido por los actores deseados.

Usabilidad

- **Usabilidad:** Prueba enfocada a factores humanos, estéticos, consistencia en la interfaz de usuario, ayuda sensitiva al contexto y en línea, asistente documentación de usuarios y materiales de entrenamiento.

Metas

Validar que la aplicación:

- Cumpla con los requisitos funcionales especificados en el diseño de la solución por medio de casos de uso.
- Cumpla con los requisitos no funcionales especificados en el diseño de la solución.
- Cumpla con las restricciones de entrada y salida de la información especificada en el diccionario de datos.
- Cumpla íntegramente con la estructura referencial especificada en el mapa de navegación.

Niveles de Prueba

- **Prueba de desarrollador:** Es la prueba diseñada e implementada por el equipo de desarrollo, en esta se definen los aspectos de diseño e implementación de las pruebas que debe llevar a cabo el equipo de desarrolladores.

Métodos de prueba

- **Pruebas de Caja Negra**

Como método de pruebas a aplicar se seleccionaron las pruebas de caja negra -también denominadas pruebas de comportamiento- porque se centran en los requisitos funcionales del software. Permiten al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La realización de estas pruebas permite encontrar:

- Funciones incorrectas.
- Errores de interfaz.

- Errores en las salidas.
- Errores en el acceso a los datos.

Técnica de Prueba

- **Casos de Prueba**

Son un conjunto de condiciones y variables que prueban las funcionalidades del sistema, en estos se incluyen la descripción de la funcionalidad que es probada, la cual es tomada del caso de uso y la respuesta del sistema en cada caso. Los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene. A continuación se muestra el diseño de casos de prueba para el caso de uso Registrar reunión del Consejo de Promoción.

Resumen: El caso de uso (CU) se inicia cuando el GCP desea registrar un nuevo Consejo de Promoción. El sistema registra la información y termina el CU.

Precondiciones: El GCP debe estar autenticado en el sistema.

Post condiciones: Queda registrado un nuevo Consejo de Promoción.

Escenario	Descripción	Año	Fecha	Número de Acta	Respuesta del sistema
EC 1.1 Registrar Consejo de Promoción	Permite registrar los datos referentes a un Consejo de Promoción.	V	V	V	Registra el nuevo Consejo de Promoción y los beneficios analizados.
EC 1.2 Campos Vacíos	Indica que existen campos vacíos.	I V I	V I I	V V V	El sistema muestra el mensaje “Introduzca los datos obligatorios” indicando los campos obligatorios que faltaron por llenar.
EC 1.3 Los datos son incorrectos	Indica que se han introducido datos incorrectos.	I	V	V	El sistema muestra el mensaje “Corrija los datos erróneos introducidos” y señala los campos erróneos introducidos.

EC 1.4 Datos de la incidencia	Se muestran los datos de las incidencias asociadas al interno seleccionado.	NA	NA	NA	El sistema muestra los datos de la incidencia: • Tipo de hecho • Noticia • Número de parte • Fecha de ocurrencia • Resumen de la incidencia • Medidas tomadas
EC 1.5 Documento Evaluación de Conducta	Se muestra la evaluación de conducta referente al interno seleccionado.	NA	NA	NA	El sistema muestra los datos de la "Evaluación de Conducta" emitida por el equipo multidisciplinario

Tabla 15: Caso de prueba asociado al CU "Registrar reunión del Consejo de Promoción".

Resultados de la aplicación de las pruebas de caja negra.

Para validar el software desarrollado, fue sometido a pruebas de caja negra por el grupo de desarrollo del proyecto SIDEP, haciendo uso de 21 casos de prueba correspondientes al módulo Gestión del Consejo de Promoción, siendo evaluadas todas sus funcionalidades. Como resultado de la revisión se encontraron once no conformidades en la primera iteración siendo resueltas en la segunda iteración.

No	Ubicación	No Conformidades	Estado
1	Propuesta de Beneficio/ Insertar Propuestas	Una vez que se guarda la propuesta no se desmarca el campo de selección única "Opinión Fundamentada".	Resuelta
2	Propuesta de Beneficio/ Consultar Propuestas	No están internacionalizadas las columnas.	Resuelta
3	Registrar Consejo	El botón cerrar permanece habilitado aunque no se haya analizado ningún beneficio.	Resuelta

4	Registrar Consejo	No se bloquea el botón aceptar mientras espera por la respuesta del servidor.	Resuelta
5	Registrar Consejo	No se valida que el campo de texto “tiempo de reanálisis” solo admita números y no más de 5 caracteres.	Resuelta
6	Consultar Beneficios	Los campos de texto mantienen sus valores después de guardar.	Resuelta
7	Consultar Beneficios	Se mantienen visibles los campos de texto luego de guardar la información.	Resuelta
8	Consultar Beneficios	Luego de guardar no se notifica al usuario si se realizó con éxito.	Resuelta
9	Evaluación de Conducta	No se muestra un mensaje indicando si se guardó con éxito o no.	Resuelta
10	Evaluación de Conducta	No se limpian los campos de texto al guardar.	Resuelta
11	Actualizar Término	Luego de guardar no se notifica al usuario si se realizó con éxito.	Resuelta

Tabla 16: Tabla de las principales funcionalidades del módulo “Gestión del Consejo de Promoción”.

3.4. Conclusiones

En este capítulo se detallaron los principales aspectos que intervienen en el proceso de implementación y pruebas. Quedan expuestas las principales funcionalidades empleadas que contribuyeron a la correcta implementación del sistema obteniendo un sistema funcional y acorde a los requisitos. Se generaron un conjunto de artefactos que contribuyen al mejor entendimiento de los flujos de procesos que se desarrollan en el módulo Gestión del Consejo de Promoción. Con el desarrollo de este capítulo se pudo apreciar la importancia de la realización de las pruebas en el proceso de desarrollo de software. Se

desarrollaron las pruebas de caja negra, las cuales de manera general arrojaron resultados satisfactorios, siendo corregidos todos los errores encontrados por las mismas.

Conclusiones Generales

- Se realizó un estudio de las tecnologías y herramientas a utilizar en el diseño e implementación del módulo así como del análisis de los requisitos de software y del modelo de negocio correspondientes.
- Como resultado del trabajo realizado se diseñó e implementó el módulo Gestión del Consejo de Promoción a partir de los requerimientos de software establecidos con el cliente y haciendo uso de la arquitectura y las tecnologías definidas por el proyecto.
- Las actividades de diseño e implementación fueron ejemplificadas a través de diagramas de clases y fragmentos de código fuente cumpliendo así con los objetivos propuestos para este trabajo.
- Se validaron las funcionalidades de la solución propuesta con la realización de pruebas, demostrándose que cada una de las funcionalidades responde apropiadamente a los requisitos funcionales definidos.

Recomendaciones

- Despliegue de la aplicación en una estación penitenciaria para que sea sometido a pruebas.
- Agregar la funcionalidad de imprimir las evaluaciones de los beneficios.
- Documentar todo el código para que sea comprendido por futuros desarrolladores.

Referencias Bibliográficas

1. **Borrego, Jaen Fernández.** *Informe de factibilidad para el proyecto de desarrollo estratégico del Sistema Penitenciario Nacional.* 2009.

2. Dirección General del Sistema Penitenciario. [En línea] 2007. [Citado el: 20 de 11 de 2011.] http://www.sistemapenitenciario.gob.pa.

3. Instituto Nacional Penitenciario (INPE). [En línea] 7 de 2009. [Citado el: 15 de 11 de 2011.] http://www.inpe.gob.pe.

4. **INPE, Instituto Nacional Penitenciario.** *Codigo de Ejecución Penal,Decreto Legislativo #654.* 2010.

5. **Martínez Cabrera, Jorge Ernesto y Benavides Zaila, Yadira.** *Descripción de cambios a módulos del SIGEP v2.0.* 2009.

6. **Jacobson, Ivar, Booch, Grady.** *El proceso Unificado de Desarrollo de Software.* Madrid : Addison Wesley, 2000.

7. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* 1999. págs. 189-424. Vol. 1ra Edicion.

8. **Visual Paradigm International.** Visual Paradigm. [En línea] http://www.visual-paradigm.com/product/vpuml/.

9. CASE Computación Avanzada y Sistemas Empresariales. [En línea] [Citado el: 3 de 6 de 2012.] http://www.casenet.com.mx.

10. [En línea] 1990. [Citado el: 3 de 6 de 2012.] http://www.codegear-shop.com/epages/62042259.sf/es_ES/?ObjectPath=/Shops/62042259/Products/%22Embarcadero%20ER/Studio%22.

11. DosIdeas. [En línea] [Citado el: 4 de 6 de 2012.] http://www.dosideas.com/wiki/NetBeans.

12. SOWRE. [En línea] 2011. [Citado el: 5 de 6 de 2012.] http://www.sowre.es/tecnolog%C3%ADas/servidores-de-aplicaciones/apache-tomcat.html.

13. The Apache Software Foundation. [En línea] [Citado el: 4 de 6 de 2012.] http://tomcat.apache.org/tomcat-6.0-doc/index.html.

14. Manual de desarrollo web con Grails. [En línea] 9 de 7 de 2009. http://www.manual-de-grails.es.

15. The Dojo Foundation. DojoTolkit. [En línea] [Citado el: 10 de 12 de 2011.] http://www.dojotoolkit.org/.

16. Oracle Database. [En línea] [Citado el: 7 de 11 de 2011.] http://www.oracle.com/es/products/database/index.html.

17. **Kemedy, Chuck Musciano y Bill.** *HTML La guia completa.* s.l. : McGRAW-HILL INTERAMERICANA EDITORES, 1999.

18. Librosweb.es. [En línea] [Citado el: 5 de 6 de 2012.] http://www.librosweb.es/javascript/capitulo1.html.

19. Groovy. A dynamic language for the Java platform. [En línea] SpringSource and the Groovy Community, 2009. [Citado el: 15 de 12 de 2011.] http://www.groovy.codehaus.org/.

20. **Eckel, Bruce.** *Thinking in Java.* 2000. 2nd edition.

21. [En línea] [Citado el: 4 de 6 de 2012.] http://www.infor.uva.es/~mlaguna/is1/apuntes/2-requisitos.pdf.

22. **Abdul-Jawad, Bashar.** *Groovy and Grails Recipes.* 2009. pág. 221.

23. *Modelo de Clases.* [En línea] [Citado el: 10 de 5 de 2012.] http://www.dcc.uchile.cl/~psalinas/uml/modelo.html.

24. *MSDN.* [En línea] [Citado el: 11 de 5 de 2012.] http://msdn.microsoft.com/es-es/library/dd409390.aspx.