Universidad de las Ciencias Informáticas

Facultad 2

Título: Clasificación Penitenciaria apoyado en Razonamiento Basado en Casos (RBC).

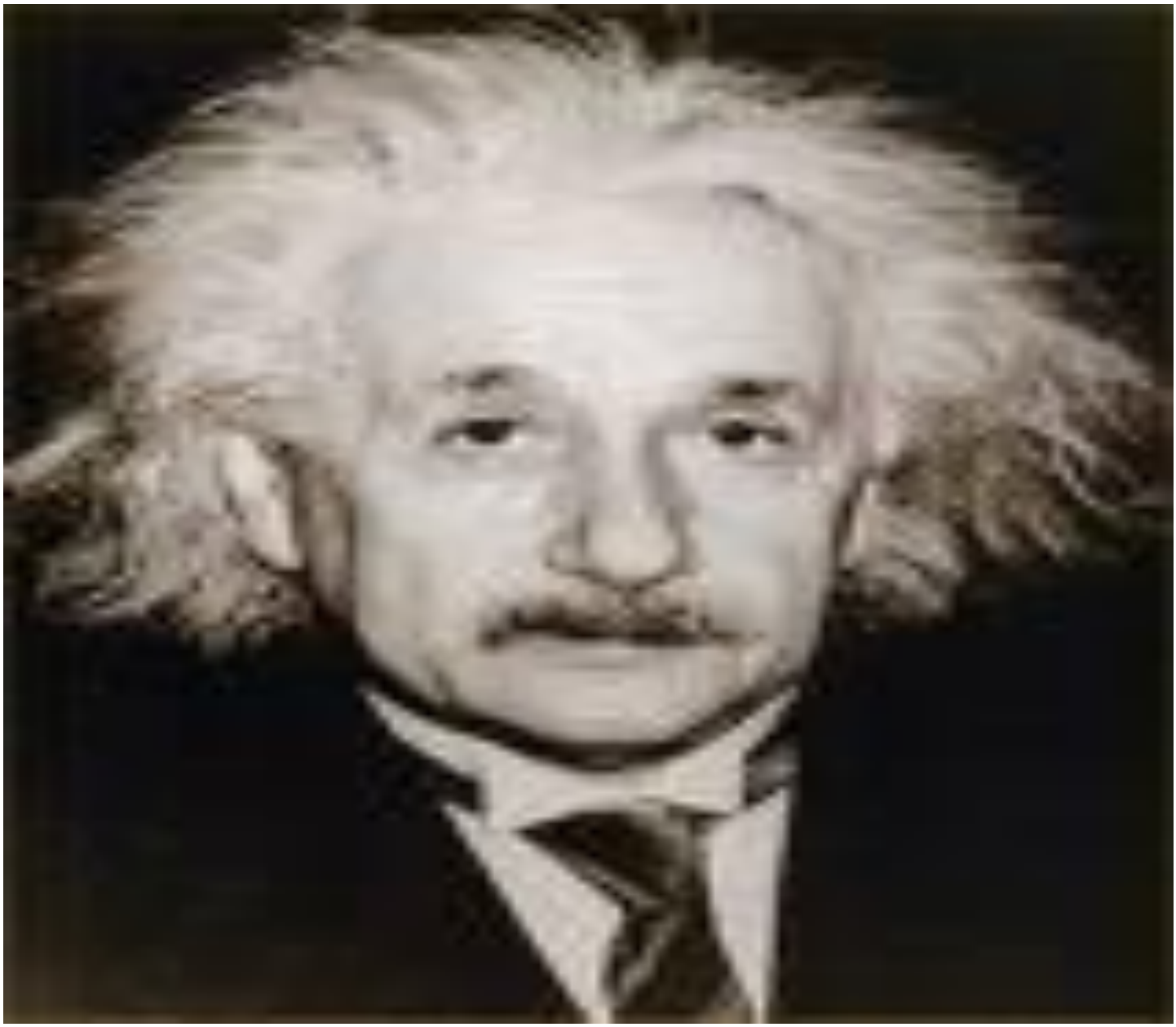
Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autor(es): Niusbel Hernández Lazo.

Tutor(es): Ing. Dasiel Cordero Morales**.**

La Habana, Junio 2013
"Año 55 de la Revolución"



Si anhelamos con seguridad y pasión la seguridad, el bienestar y el libre desarrollo del talento de todos los hombres no hemos de carecer de los medios necesarios para conquistarlos.

Albert Einstein

DECLARACIÓN DE AUTORÍA

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

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

_____ _____

Niusbel Hernández Lazo Ing. Dasiel Cordero Morales

DEDICATORIA

Dedico este trabajo sobre todo a mi mamá y a mi papá, a mi hermana, mi esposa, a mi hijo Josué que esta al nacer, a mi tía Nely , a mi sobrinita y a todos mis familiares que han sido de gran ayuda sobre todo a la hora de aconsejarme.

AGRADECIMIENTOS

Agradecerle primeramente a Dios por ayudarme a llegar hasta aquí, a mi madre, mi padre los que me enseñaron lo bueno y lo malo de la vida, a mi hermana por luchar conmigo en todo momento a mi esposa por soportarme y darme ánimos para realizar todas las cosas, a mi hijo que está por nacer el cual es el motivo de mi esfuerzo, a mi sobrina, a toda mi familia que supo ayudarme en todo momento, a la revolución por ofrecerme la oportunidad de estudiar, a mis amigos por el apoyo y a todos aquellos que de una forma u otra supieron alentarme en los momentos más difíciles.

RESUMEN

Los sistemas de clasificación del régimen de severidad, se consideran el cerebro de la gestión de un Sistema Penitenciario ya que son esenciales para proyectar las necesidades de recursos en el futuro. Un correcto funcionamiento del sistema de clasificación permite la toma de decisiones en relación a la construcción de espacios para la reclusión y establecimiento de programas de educación y rehabilitación, de acuerdo a las necesidades de los internos. El presente trabajo tiene como objetivo el desarrollo de un sistema inteligente que permita insertarse en el proceso de calificación en los sistemas penitenciarios cubanos. El diseño del sistema está enfocado en la técnica de Razonamiento Basado en Casos que permita utilizar la experiencia de expertos en el campo de la clasificación del nivel de severidad dadas las características de un sujeto. A partir del análisis de sistemas de clasificación utilizados en varios países, y la validación de expertos, fueron propuestos los rasgos para conformar los casos de la Base de Casos del sistema y la relevancia de cada una de ellos. La utilización del sistema propuesto contribuirá a agilizar el proceso de clasificación y evitar o disminuir errores durante su ejecución.

Palabras clave: Base de Casos, Clasificación penitenciaria, Sistemas Basados en Casos, Sistemas Penitenciarios

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El perfeccionamiento empresarial, social y económico a nivel mundial aumenta a una velocidad considerable en cuanto al desarrollo de las tecnologías de la informática y comunicaciones. En los últimos 10 años se han logrado muchos más avances en este aspecto que en los pasados 50 años, más acentuadamente con la aplicación de la inteligencia de negocio en la rama de la informática. Esto ha permitido el surgimiento de una nueva era donde la información y el conocimiento juegan un papel primordial para toda organización. Los volúmenes de información generados diariamente crecen de forma exponencial por lo que su correcto análisis e interpretación son fundamentales para la toma de decisiones, que constituye un recurso vital para toda organización.

Este tipo de información sólo se convierte en conocimiento cuando los individuos la contextualizan sobre un dominio específico o sobre el dominio de la entidad o empresa aplican para la resolución de un problema determinado. La inserción de las ciencias informáticas en diversos sectores de la sociedad es una tendencia en la actualidad y Cuba no queda exenta de esto, pues intenta convertirse en una de las mayores industrias productoras de software. Para esto el país ha trazado un grupo de estrategias, una de esas estrategias con la que el país intenta introducirse en el mercado del software es la creación de la Universidad de las Ciencias Informáticas (UCI) la cual mediante proyectos productivos que vinculan directamente a los jóvenes que en la misma se forman, juega un papel fundamental en el desarrollo de este proceso. La UCI promueve entre sus principios la automatización de procesos tanto de interés nacional como internacional; apoyándose en la firma de contratos con diferentes países para el desarrollo de este último. Entre muchas de las instituciones cubanas que se benefician en este proceso de automatización se encuentra el Ministerio del Interior (MININT). Dentro del MININT se encuentran los centros penitenciarios, los cuales constituyen espacios para la reclusión y hacinamiento de los sujetos con conducta social desviada.

En todos los centros penitenciarios la correcta clasificación de los internos es una de las principales formas de mantener un programa de reeducación. Esto permite la asignación de los internos a las distintas instituciones penitenciarías y centros para menores o cualquier otra prevista por la ley, sean éstos de alta, media o baja severidad, o bien a las áreas de alojamiento y convivencia dentro de una

institución penitenciaría. La clasificación permitirá una convivencia armónica y un clima de tranquilidad emocional que prepare el siguiente paso que es el tratamiento (1).

Para todos los casos la clasificación debe considerarse como una medida instrumental, de carácter temporal y revisable y no como un fin en sí misma, por lo que su aplicación está supeditada al irrestricto respeto a los principios enunciados en las leyes penales y las reglas mínimas para el tratamiento para reclusos. La clasificación debe de basarse en consideraciones pues debe mantenerse una concepción del hombre como fin y bajo ninguna circunstancia, como medio o instrumento para la consecución de otros fines. Deben garantizarse los niveles requeridos de humanidad y de certeza jurídica dentro del sistema penitenciario estatal. También debe evitarse la represividad, selectividad o estigmatización de la población penitenciaría (1).

Conforme a la importancia que tiene la clasificación para los sistemas penitenciarios, en las últimas décadas, los profesionales de los sistemas penitenciarios han trabajado para mejorar sus métodos de clasificación de los internos de acuerdo a las necesidades de custodia, trabajo y rehabilitación. Esto viene siendo impulsada por: demandas judiciales, el hacinamiento en las cárceles así como la necesidad de minimizar gastos y las preocupaciones de seguridad pública. La clasificación se ha convertido en una herramienta de gestión principal para la asignación eficiente de los escasos recursos disponibles. Los sistemas de clasificación se consideran el "cerebro" de la gestión de un sistema penitenciario, ya que son esenciales para proyectar las necesidades de recursos en el futuro. Un correcto funcionamiento del sistema de clasificación permite la toma de decisiones en relación a la construcción de espacios para la reclusión, contratación y capacitación del personal, establecimiento de programas de educación y rehabilitación, de acuerdo a las necesidades de los internos (2).

Los primeros sistemas de clasificación utilizaron como criterio la separación de los internos por edad (jóvenes y adultos), especiales (enfermedades mentales), género (masculino y femenino), primariedad penal (primario o reincidente) y otras características. El siguiente enfoque utilizado fue el modelo clínico que incluía identificar los problemas de los individuos y prescribir tratamientos individuales. Gradualmente el énfasis en diagnosticar las causas del comportamiento antisocial de los delincuentes fue sustituido por el énfasis en los tipos de programas de tratamiento disponibles. Sin embargo, los sistemas de clasificación se basaban en gran medida en criterios subjetivos o en evaluaciones clínicas que producían resultados arbitrarios y poco confiables (2).

La clasificación debe tener en cuenta no sólo la personalidad y el historial individual, familiar, social y delictivo del interno o interna, sino también la duración de la pena y medidas penales en su caso, el medio al que probablemente retornará, los recursos, facilidades y dificultades existentes en cada caso y momento para el buen éxito del tratamiento. Esto facilita que a partir del análisis del cúmulo de información de este tipo a lo largo de los años, así como la experiencia adquirida por el personal que interviene directamente en este proceso sea de gran valor para la obtención de futuras soluciones y la toma de decisiones; si se logra integrar con sistemas que posibiliten aprovechar e interpretar este conocimiento almacenado.

En la actualidad existen disciplinas capaces de modelar o simular el pensamiento humano y los procesos que ocurren en él. A la rama de las ciencias de la computación dedicada al desarrollo o uso de los ordenadores, con los que se intenta reproducir los procesos de la inteligencia humana se le denomina Inteligencia Artificial (IA) (3). La IA incluye una disciplina denominada Ingeniería de Conocimiento (IC) que proporciona los métodos y técnicas para construir sistemas computacionales denominados Sistemas Basados en Conocimiento (SBC). Un SBC es capaz de soportar una representación explícita del conocimiento en algún dominio de interés determinado y de aprovecharlo mediante los mecanismos de razonamiento apropiados para encontrar un alto rendimiento en la resolución de problemas (4). El razonamiento basado en experiencias pasadas es un procedimiento que los seres humanos emplean con mucha frecuencia para solucionar problemas, tanto en la vida diaria como en situaciones en que debe aplicarse conocimiento especializado. Los sistemas que solucionan problemas por analogía con otros se llaman a menudo Sistemas de Razonamiento Basado en Casos (SRBC) (5). Un SRBC es básicamente un modelo de razonamiento que permite resolver problemas, aprender y entender situaciones.

Los SRBC son un tipo de sistema basado en el conocimiento y que recoge experiencias pasadas, además de mejorar los métodos existentes para solucionar problemas utilizando directamente la información almacenada en una base de casos sobre los problemas o casos resueltos. Con la utilización del conocimiento almacenado, podrían darse resultados adelantados, predecir ubicaciones en centros penitenciarios así como contribuir a la reeducación de los internos e internas (3).

El funcionamiento de gran parte de los sistemas penitenciarios se sustenta en la existencia de sistemas informáticos que apoyan la gestión de los procesos asociados a los individuos: clasificación, programas de tratamiento, rehabilitación entre otros, e incluso cuando no existan estos sistemas los resultados del

proceso de clasificación se encuentran documentados. Los especialistas que participan en el proceso de clasificación tales como criminólogos y psicólogos realizan este proceso tomando como base su apreciación y experiencia profesional y los SRBC son capaces de aprovechar el conocimiento de estos expertos almacenándolo en una estructura denominada de forma general base de conocimientos.

La reutilización de experiencias previas a través de un sistema experto puede contribuir a minimizar un problema común en los sistemas penitenciarios, el de la carencia y permanencia de personal especializado, así como el de capacitación en este proceso y con esto entrenamiento en el dominio. Además, el propio funcionamiento de un SRBC garantizaría la retroalimentación necesaria al sistema de clasificación a partir del ingreso de nuevos casos a la base de casos. Brindaría además la posibilidad de clasificación automática y con esto menor utilización de recursos y una mayor velocidad en el reconocimiento del individuo así como su correcta ubicación en el establecimiento penitenciario.

En Cuba con el triunfo revolucionario en 1959, se había heredado un sistema penitenciario caracterizado por la promiscuidad, la corrupción judicial y administrativa, el crimen despiadado, la discriminación racial, social y el tratamiento brutal al hombre sancionado en detrimento de su integridad y dignidad humana. Desde el propio triunfo revolucionario se comenzó un proceso de transformaciones que contribuyeron al mejoramiento de la condición humana y conducta social a los privados de libertad. Este proceso se orientó, entre otras, en las direcciones siguientes:

- Adopción de un sistema progresivo más avanzado y justo.
- Criterios de clasificación de la población penal que aseguran mejor tratamiento colectivo e individualizado.
- Incorporación voluntaria al trabajo socialmente útil y remunerado, con fines educativos y de asistencia y seguridad social para la familia (6).

Si bien se han logrado varios cambios para mejorar la estancia de los reclusos en las distintas penitenciarías, el tratamiento a los reclusos ha de moverse dentro del respeto de los derechos humanos de dignidad, libertad, igualdad y seguridad. Con la realización de algunos experimentos, como campos de deporte, talleres, escuelas, hospitales, se ha ido más allá de la mera custodia mecánica. Entre los muchos desperfectos posibles a ocurrir, de llegar a hacerse una clasificación errónea figuran sobre todo la

agrupación de los internos sin tener en cuenta la edad, la gravedad de lo delitos y la situación personal: procesados o condenados, delincuentes primarios o reincidentes, personas sanas y personas física y mentalmente enfermas. Esto conlleva a la posible mala ubicación de los internos y que los mismos se sientan intimidados o reprimidos, además de no poseer del trato adecuado con que se le debe atender posteriormente. Todas estas circunstancias influyen desfavorablemente en los recluidos y deberían evitarse pues de ocurrir dañarían la integridad de los internos y constituiría una violación de los derechos humanos.

A pesar de que se ha perfeccionado el proceso de diagnóstico para la clasificación mediante el estudio y la caracterización individual del interno que centra su atención en el hombre y no en la sanción, el sistema penitenciario cubano ha tratado incansablemente de seguir mejorando las condiciones informatizando la gestión de los centros penitenciarios e incluyendo diversos programas de atención al interno e interna. La posibilidad de contar con un sistema que basado en técnicas de IA agilice o ayude al proceso de clasificación, sería de gran importancia para el trabajo, reeducación y reinserción social del sancionado.

Por tanto surge como **problema a resolver** ¿Cómo contribuir a la clasificación del régimen de severidad penitenciaria utilizando la experiencia y conocimiento almacenados?

Definiéndose en la investigación como **objeto de estudio** el proceso de Clasificación Penitenciaria

El **objetivo general** de este trabajo es construir un sistema que, utilizando la experiencia de resultados anteriores, contribuya a la clasificación penitenciaria en los Sistemas Penitenciarios de Cuba.

Siendo el **campo de acción** sistemas basados en el conocimiento para la clasificación del régimen de severidad penitenciaria.

El objetivo general se divide en los siguientes **objetivos específicos:**

1. Elaborar el marco teórico de la investigación identificando las principales tendencias, limitaciones y ventajas de los sistemas de software para la clasificación penitenciaria.
2. Formalizar un modelo computacional donde se apliquen técnicas de razonamiento basado en casos en la consideración de la clasificación penitenciaria como entidad principal.

3. Desarrollar el modelo propuesto cumpliendo con estándares de codificación y calidad definidos en la UCI.

4. Validar la solución propuesta para la clasificación del régimen de severidad penitenciaria.

Para dar cumplimiento a los objetivos específicos anteriormente planteados se definen las siguientes **tareas de investigación:**

1. Identificación y análisis de los procesos principales de clasificación penitenciaria.

2. Caracterización de la situación existente en el mundo, en la región y en el país en particular sobre la clasificación penitenciaria y la técnica de razonamiento basado en casos para definir la posición de los investigadores.

3. Análisis de sistemas que utilicen la técnica de RBC para resolver problemas de clasificación penitenciaria.

4. Definición de una base de casos que permita almacenar el conocimiento de clasificaciones penitenciarías anteriores.

5. Definición de los componentes del mecanismo de recuperación del sistema de razonamiento basado en casos.

6. Definición de los componentes del mecanismo de adaptación del sistema de razonamiento basado en casos.

Como **preguntas de investigación**

1. ¿Cuáles son las principales tendencias, limitaciones y ventajas en los sistemas de software para la Clasificación Penitenciaria en el régimen de severidad?

2. ¿Cuáles técnicas de IA son utilizadas para solucionar los problemas existentes en el proceso de Clasificación Penitenciaria?

3. ¿Cuáles sistemas que apoyados en la técnica de RBC son utilizados para resolver problemas de Clasificación Penitenciaria?

4. ¿Cómo confeccionar la BC para el proceso de Clasificación Penitenciaria en los regímenes penitenciarios cubanos?

5. ¿Cómo crear el motor de inferencia, como parte del SRBC, para realizar el proceso de Clasificación Penitencia en los regímenes penitenciarios cubanos?

6. ¿Cuáles técnicas de IA permiten desarrollar mecanismos de aprendizaje automático?

Métodos Científicos utilizados en la investigación.

Métodos teóricos.

Analítico–Sintético: Se utilizó para el estudio a partir de fuentes bibliográficas seguras de los Sistemas Inteligentes, los Sistemas de Razonamientos Basados en Caso; así como las clasificaciones de los Sistemas Penitenciarios. Permitió además descomponer el problema de investigación en elementos por separado y profundizar en el estudio de cada uno de ellos, para luego sintetizarlos en la solución propuesta.

Inductivo-Deductivo: Se utilizó para el planteamiento del objetivo y la extracción de las ideas fundamentales para la elaboración y fundamentación del trabajo de diploma.

Histórico-Lógico: Posibilitó estudiar de forma analítica la trayectoria histórica real de los fenómenos, su evolución y desarrollo. El método permitió realizar la primera parte de la investigación, al hacer un análisis bibliográfico de los Sistemas Inteligentes, el Razonamiento Basado en Casos y técnicas de IA más utilizadas en el desarrollo de Sistemas Inteligentes. Dio paso a la exploración de trabajos realizados en el campo de los Sistemas Inteligentes y de soluciones previas existentes a problemas similares al actual. Se utilizó para determinar a través de la evaluación de la bibliografía conceptos de esta temática, que permiten conocer el estado de la evolución actual del fenómeno e identificar posibles mejoras y alternativas de solución.

Métodos empíricos.

Encuesta: Apoyará a la incorporación de conocimientos mediante las entrevistas planificadas efectuadas a los especialistas tanto en Inteligencia Artificial como en Clasificación Penitenciaria. Se seleccionó una muestra intencional no probabilística, ya que las encuestas serán dirigidas a especialistas con gran conocimiento en el tema, como por ejemplo, trabajadores vinculados a los regímenes penitenciarios, directores de centros penitenciarios, especialistas en el tema de clasificación penitenciaria, entre otros, sin tener en cuenta el por ciento que representan los mismos dentro de la población.

Organización del documento

El documento está estructurado en 4 capítulos:

Capítulo1. Fundamentación Teórica: se especifican conceptos que serán tratados a lo largo del documento y que son de vital importancia para la comprensión del mismo. Se realiza un estudio del arte de sistemas inteligentes, además del uso de los sistemas de clasificación penitenciaria a nivel mundial.

Capítulo 2. Formalización del Modelo computacional para la Clasificación penitenciaria: Está orientado a definir cómo quedará formado el modelo de clasificación, basado en las técnicas de IA, enmarcado en los SRBC.

Capítulo 3: Características y Diseño del sistema: Está orientado a la identificación de las necesidades de los clientes. Descripción general del funcionamiento del sistema. Se documenta la disciplina de diseño, incorporando los elementos definidos por la metodología de desarrollo de software XP.

Capítulo 4. Implementación y Prueba: Se procederá a abordar las HU definidas, se realizan tareas de la ingeniería que describen los procesos y funcionalidades a implementar. Se especifican la estrategia de prueba empleada según la metodología XP.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción al capítulo.

En el presente capítulo se describen los antecedentes y el estado existente en el mundo, la región y en particular el país sobre la clasificación penitenciaria. Además se aborda sobre las técnicas de inteligencia artificial como apoyo al proceso de clasificación del régimen de severidad penitenciaria. También se incluyen conceptos teóricos de las tecnologías, lenguajes y herramientas a utilizar para implementar la solución de esta investigación, así como la metodología de desarrollo empleada en el proceso de solución.

1.1 Clasificación penitenciaria

¿Qué es la clasificación penitenciaría?

Se entiende por clasificación el procedimiento mediante el cual son asignados los internos a las distintas instituciones penitenciarias y centros para menores o cualquier otra prevista por la ley, sean éstos de alta, media o baja severidad, o bien a las áreas de alojamiento y convivencia dentro de una institución penitenciaria (7). Además es el instrumento a través del que se articula el régimen penitenciario o tratamiento que se dispensa al interno, con el principal objetivo de su reinserción social.

Objetivos de la clasificación penitenciaria

El objetivo de la clasificación de los internos en los centros de readaptación es la de garantizar el derecho a una estancia digna y segura dentro de la Institución. Además se enfoca en preservar la integridad de los internos, disminuir la desadaptación, incidir en la readaptación social, permitir sentar las bases para el tratamiento técnico, reducir los efectos de la prisionalización y garantizar los derechos humanos de la persona en reclusión (7).

Ventajas de la clasificación penitenciaria

La correcta clasificación ofrece diversas ventajas, como las siguientes (7):

1. Separación adecuada de los diferentes tipos de internos.
2. Mayor supervisión y control de la custodia.
3. Mayor disciplina.
4. Mayor productividad de los internos.
5. Organización efectiva de las facilidades de tratamiento y entrenamiento.
6. Mayor continuidad de los programas de tratamiento y entrenamiento, mejores actitudes de los internos.
7. Reduce los fracasos entre los internos puestos en libertad.

Grados de clasificación

En varios países los grados de clasificación varían en correspondencia con el tipo de régimen: En España, Perú, Cuba, Venezuela, México, Estados Unidos y Canadá los grados de clasificación se denominan correlativamente, correspondiéndose con un determinado régimen de vida. Muchos de estos países adoptan tres o cuatro grados de clasificación, pero la mayoría de estos adoptan cuatro grados, ya que incluyen como último la libertad condicional:

- **1º Grado:** Régimen cerrado.
- **2º Grado:** Régimen ordinario.
- **3º Grado:** Régimen abierto.
- **4º Grado:** Libertad condicional.

Primer Grado: Será de aplicación a aquellos penados que, bien inicialmente, bien por una involución en su personalidad o conducta, sean clasificados en primer grado por tratarse de internos extremadamente peligrosos o manifiestamente inadaptados a los regímenes ordinario y abierto.

Para que proceda la clasificación en primer grado deben ponderarse la concurrencia de factores tales como (7):

1. Naturaleza de los delitos cometidos a lo largo de su historial delictivo, que denote una personalidad agresiva, violenta y antisocial.
2. Comisión de actos que atenten contra la vida o la integridad física de las personas, la libertad sexual o la propiedad, cometidos en modos o formas especialmente violentos.
3. Pertenencia a organizaciones delictivas o a bandas armadas, mientras no demuestren en ambos casos, signos inequívocos de haberse sustraído a la disciplina interna de dichas organizaciones o bandas.
4. Participación activa en motines, plantes, agresiones físicas, amenazas o coacciones.
5. Comisión de infracciones disciplinarias calificadas de muy graves o graves, de manera reiterada y sostenida en el tiempo.
6. Introducción o posesión de armas de fuego en el Establecimiento penitenciario, así como la tenencia de drogas tóxicas, estupefacientes y sustancias psicotrópicas en cantidad importante, que haga presumir su destino al tráfico.

Segundo Grado: Se corresponde con el régimen ordinario y se puede decir que suele ser el punto de partida. Es el grado inicial de clasificación para la mayoría de los penados. Serán clasificados en 2º grado los penados en quienes concurran unas circunstancias personales y penitenciarías de normal convivencia, pero sin capacidad para vivir, por el momento, en semilibertad.

En cuanto al régimen de vida, conlleva el régimen ordinario destacando los siguientes aspectos (7):

1. Dado que se refiere a unas circunstancias de normal convivencia, se aplicará a penados en 2º grado, penados sin clasificar, detenidos y presos.
2. Los principios de seguridad, orden y disciplina tendrán su razón de ser y su límite en el logro de una convivencia ordenada.
3. La separación de la población se ajustará a las necesidades del tratamiento, a los programas de intervención y a las condiciones generales del Establecimiento (sumisión del régimen al tratamiento)
4. El Consejo de Dirección establecerá un horario que contemplará el tiempo dedicado a actividades, garantizando el tiempo destinado al descanso.
5. Los internos vendrán obligados a realizar las prestaciones personales necesarias para el mantenimiento del buen orden, la limpieza y la higiene en los Establecimientos.
6. Se ha de fomentar la participación de los internos en actividades.

Tercer Grado: Esta clasificación se aplicará a los internos que por sus circunstancias personales y penitenciarías estén capacitados para llevar a cabo un régimen de vida en semilibertad, valorándose a tal fin las circunstancias tales como (7):

- Personalidad.
- Historial individual, familiar, social y delictivo.
- Duración de las penas.
- Medio social al que retorne.
- Recursos, facilidades y dificultades
- Momento para el buen éxito del tratamiento.

Cuarto Grado: Se trata del último grado y supone que el cumplimiento del resto de la pena se realice en situación de libertad, si bien con una serie de controles o condiciones que se establecen en la resolución que aprueba la misma. Para su concesión se establecen los siguientes requisitos (7):

- Que se encuentren en 3º grado de tratamiento penitenciario.
- Que hayan extinguido las ¾ partes de la condena.
- Que hayan observado buena conducta y exista respecto a los sentenciados, un pronóstico individualizado y favorable de reinserción social.

En Cuba estos gradaos de clasificación están dado de la misma forma, lo que con los nombre máxima, media o mínima severidad, sin incluir el de libertad condicional ya que el cumplimiento de este grado se da en libertad.

1.2 Inteligencia Artificial

La IA se remonta a las aspiraciones de filósofos como Ramón Llul, Descartes y Leibniz que tenían como objetivo supremo la mecanización total del razonamiento humano, o ingenieros soñadores como Babbage que proyectaron la construcción de máquinas inteligentes. El término "inteligencia artificial" fue acuñado formalmente en 1956 durante la conferencia de Darthmounth dirigida por McCarrthy, Minsky, Rochester y Shannon, donde todos participaban de una metodología en común: el uso del ordenador como analogía fructífera de cara a comprender y simular el comportamiento inteligente de las personas (8).

Se puede definir la IA como una ciencia que tiene como objetivo el diseño y construcción de máquinas capaces de imitar el comportamiento inteligente de las personas. Una rama de la Informática que investiga y produce razonamiento por medio de máquinas automáticas y que pretende fabricar artefactos dotados de la capacidad de pensar (8). La IA se enfocó inicialmente en la producción de sistemas con capacidad de asistir al ser humano en la toma de decisiones o en la búsqueda de soluciones a problemas. Se orientó al desarrollo de mecanismos que facilitaran la comunicación con el computador en lenguaje natural y se estructuraron sistemas expertos, actualmente conocidos como Sistemas Basados en Conocimiento (SBC), para almacenar la experticia humana.

Sistemas basados en el conocimiento

Un Sistemas basados en el conocimiento (SBC) puede definirse como un sistema computarizado que utiliza conocimiento específico de un dominio y se emplea para dar solución a problemas en dicho dominio. Estos sistemas son un modelo computacional formado por tres componentes esenciales: la base de conocimiento (BC) que almacena el conocimiento necesario para resolver los problemas del dominio, la máquina de inferencia (MI), que representa un procedimiento en el cual está implementado algún método de solución de problemas que utiliza el conocimiento almacenado en la BC para resolver problemas del dominio y la interfaz de usuario (5).

El conocimiento representado en los SBC es el de los expertos en el dominio, haciendo uso de las experiencias pasadas. Estas experiencias representan conocimiento informal o atajos que permiten al experto encontrar rápidamente solución a un problema sin tener que realizar un análisis detallado de la situación, gracias a que se cuenta con experiencias de los casos resueltos anteriormente o intentos

fallidos en resolver un problema similar. Éstos pueden recordar o no en detalles un análisis realizado a una situación anterior, pero pueden reconocer el enfoque dado a esta situación. Es conveniente la construcción de un SBC para no perder el conocimiento del experto, además de que este puede escasear y necesitarse en muchos lugares y se puede utilizar para mejorar la calidad del conocimiento de los expertos humanos.

Ventajas y Desventajas de los SBC

Los SBC tienen ventajas y desventajas cuando se comparan con otras soluciones como el software convencional o expertos humanos.

Ventajas

- Amplia distribución de experticia escasa.
- Fácil modificación (conocimiento explícito y accesible).
- Consistencia en las respuestas (los expertos humanos pueden diferir en sus explicaciones, incluso un mismo experto puede responder de forma diferente en momentos diferentes).
- Gran accesibilidad (los SBC trabajan las 24 horas todos los días).
- Preservación de la experticidad (constituye una memoria institucional y poseen la capacidad para adquirir nuevo conocimiento y perfeccionar el que poseen).
- Solución de problemas que incluyen datos incompletos.
- Explicación de soluciones (justifica sus conclusiones y explica por qué hace una pregunta).
- Permite evaluar el efecto de nuevas estrategias añadiendo o modificando conocimiento (3).

Desventajas de los SBC

- Las respuestas no siempre son correctas.
- Conocimiento limitado al dominio de experticidad.
- Ausencia de sentido común.
- No reconocen el límite de su conocimiento.

Arquitectura de los SBC

Los SBC pueden ser diferenciados del resto de las técnicas de la IA ya que: en primer lugar, ejecutan tareas que de alguna forma resultan complicadas o simplemente difíciles para cualquier ser humano y que están reservadas solamente a unos pocos en un dominio en particular. Para ejecutar estas tareas o resolver los problemas que solucionan los expertos, los SBC utilizan estrategias de búsqueda que comúnmente se denominan heurísticas. Además, emplean conocimiento para razonar acerca de sus

propios métodos de inferencia y por último proporcionan explicaciones o justificación de la solución obtenida. Los SBC pueden representarse de manera general de la siguiente manera:

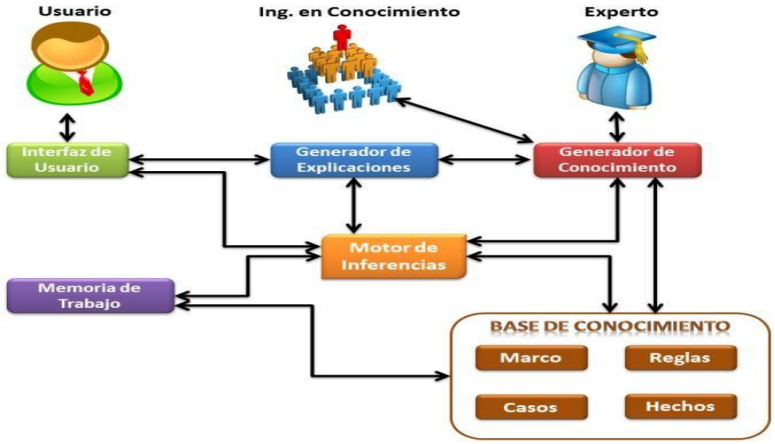


Figura 1.1: Representación general de los SBC.

Como se ilustra en la **Figura 1.1**, en la base de conocimientos se albergan los conocimientos relativos a la tarea que resuelve el experto. El motor de inferencia es el encargado de aplicar esos conocimientos e inferir las soluciones. La memoria de trabajo contiene la información de datos de entrada y conclusiones intermedias que se generan en el proceso de razonamiento. El generador de explicaciones es el módulo que proporciona una explicación coherente con la solución encontrada. El subsistema de adquisición de conocimientos es el medio que facilita tanto la inclusión como la actualización del conocimiento. Por último la interfaz del usuario permite establecer una comunicación entre el SBC y el usuario.

De manera general, los elementos fundamentales que componen la arquitectura de los SBC son:

- Interfaz
- Máquina o Motor de Inferencia (MI)
- Base de Conocimientos

La forma en que se relacionan estos elementos se muestra en la **Figura 1.2.**

Figura 1.2: Arquitectura de los SBC

La interfaz es el componente que facilita la interacción con el sistema. La máquina de inferencia es quién realiza la obtención y transformación del conocimiento previo almacenado en la base de conocimientos por un experto, para dar solución a un nuevo problema planteado.

Tipos de Sistemas Basados en el Conocimiento

Diferentes tipos de conocimiento dan lugar a diferentes SBC, entre ellos se encuentran los sistemas basados en reglas, los sistemas basados en probabilidades, sistemas basados en frames, redes expertas y los sistemas basados en casos:

- **Sistemas Razonamiento Basados en Reglas (SRBR)**

Son sistemas que utilizan para el proceso de inferencia un conjunto de reglas que constituyen la base de conocimiento del experto. Los SRBR presentan la facilidad de ser un formato muy fácil para expresar el conocimiento. Son altamente modulares pues cada regla es una unidad de conocimiento y estas pueden ser añadidas, modificadas y removidas independientemente de las otras reglas existentes. También poseen una uniformidad pues todo el conocimiento del sistema se expresa en el mismo formato. Como ventaja fundamental muestran su capacidad de interpretación y explicación de la inferencia. El proceso de solución de problemas en un SRBR es crear una cadena de inferencias que constituye un camino entre la definición del problema y su solución, mediante la utilización del encadenamiento hacia atrás y hacia adelante. Una de las facilidades que brinda es la manipulación de incertidumbre. Entre sus desventajas se encuentra el encadenamiento infinito, este se produce debido a que la mayoría de los SRBR realizan una búsqueda primero a profundidad; este método padece dicho problema, si no es implementado correctamente (5).

- **Sistemas Razonamiento Basados en Probabilidades(SRBP)**

En los Sistemas Basados en Probabilidades la adquisición del conocimiento consiste en coleccionar muestras y realizar un procesamiento estadístico que produzca las probabilidades o frecuencias que forman la base de conocimiento. No son factibles para todo tipo de dominio, pues se dificulta construir las redes con ayuda de expertos humanos cuando existen carencias de conocimiento. No son viables para explicar el razonamiento, ya que los métodos y modelos que utiliza están aún lejos de ofrecer explicaciones comprensibles. Éstos utilizan generalmente el Teorema de Bayes como método de solución de problemas (5).

- **Sistemas Razonamiento Basados en Frames(SRBF)**

Los sistemas basados en frames constituyen esencialmente sistemas contestadores a preguntas pues sus mecanismos de razonamiento resultan débiles; ellos pueden ser considerados SBC rudimentarios. Uno de los principales mecanismos de inferencia de los SRBF es la herencia. Un frame es una estructura de datos compleja que contiene un agregado de información acerca de un objeto, ofreciendo una representación estructurada de un objeto o una clase de objetos. A partir de los frames individuales se crea una organización (taxonomía) de los frames que permite al diseñador de la Base de Conocimientos describir cada clase como una especialización (subclase) de otra más genérica (más abstracta) (5).

- **Redes Expertas**

En las Redes Expertas la adquisición del conocimiento incluye la selección de los ejemplos, el diseño de su topología y el entrenamiento de la red para hallar el conjunto de pesos. Facilitan el trabajo con información incompleta y brindan algoritmos poderosos de aprendizaje para crear la base de conocimiento; pero requieren de muchos ejemplos y son cajas negras que no explican cómo se alcanza la solución. Éstas generalmente utilizan pesos como forma de representar el conocimiento y el Cálculo de niveles de activación de las neuronas como método de solución de problemas (5).

- **Sistemas de Razonamiento Basados en Casos(SRBC)**

Representa un nuevo método para resolver problemas no estructurados, en el cual el razonamiento se realiza a partir de una memoria asociativa que usa un algoritmo para determinar una medida de semejanza entre dos objetos. En este paradigma la base del comportamiento inteligente de un sistema radica en recordar situaciones similares existentes en el pasado.

Los SRBC presentan una serie de ventajas que al explotarlas darán una buena visión para mejores soluciones, ya que este utiliza las experiencias previas que almacena en forma de ejemplos concretos para llegar a nuevas soluciones o anticipar problemas. Su aprendizaje es incremental pues este asimila nuevos ejemplos a medida que vaya entrenando su conocimiento sin la necesidad de un experto y también ayuda a resolver problemas de forma más rápida que analizando el problema de forma trivial por su capacidad en derivar sus respuestas a partir de otras (5).

Después de haber realizado el análisis de las diferentes técnicas de la IA para la construcción de sistemas inteligentes, se opta por la utilización del SRBC. Sus modelos de recuperación de casos semejantes resultan adecuados para acceder a información en una situación dada y que se ajusta a la toma de decisiones. Además, no solo contienen conocimiento general del dominio de un problema o establecen asociaciones a través de un conjunto de relaciones generalizadas entre descriptores de problemas y

conclusiones, también utilizan el conocimiento específico de experiencias previas en situaciones concretas.

Por otra parte el grado de veracidad de la solución encontrada se obtiene directamente a partir del grado de semejanza entre el problema y el caso recuperado, no es necesario tener siempre como en los SBR, un procedimiento de manipulación de la incertidumbre. Otra diferencia importante con otros planteamientos de IA radica en que SRBC incorporan una concepción incremental y continua del aprendizaje, puesto que cada vez que un problema es resuelto se retiene si es relevante como nueva experiencia para que está disponible en la resolución de futuros problemas. De este modo, el aprendizaje tiene lugar de una forma natural, como un subproducto del propio método de resolución aplicado, mediante la actualización continua de la base de casos.

Es importante destacar que la rapidez de respuesta de los SRBC está dada, entre otros factores, por el hecho de que generalmente no se realizan retrocesos (backtracking), sino que se recuperan soluciones previas completas en un solo paso; lo cual acerca esta clase de razonamiento al realizado por los humanos. Es conveniente el uso de esta técnica cuando el conocimiento disponible es escaso, además los mismos sirven para la transferencia de conocimiento entre expertos. La utilización de una base de casos evita tener que derivar de nuevo soluciones ya obtenidas y recordar problemas previos evita seguir caminos equivocados. A diferencia de los métodos clásicos de solución de problemas de la Inteligencia Artificial la búsqueda de la solución a un problema no se inicia a partir de los datos o el objetivo, por lo que el camino se acorta considerablemente.

1.3 Herramientas que realizan Gestión Penitenciaria en Cuba y en el mundo

Tanto a nivel nacional como mundial existen varias herramientas que se encargan de la gestión penitenciaria. Muchas de estas herramientas manejan datos confidenciales, por lo que no es posible conocer su definición, objetivos y funcionamiento. Entre las más conocidas a nivel mundial se encuentran:

Sistema integrado de Gestión Penitenciaria (SIGEP).Está implantado en Venezuela. Tiene la capacidad de almacenar toda la información relacionada con el funcionamiento de los sistemas penitenciarios como: situación jurídica, cualidad del interno, es decir si es penado o procesado además de todo lo relacionado con los servicios al interno.

Law Enforcement Automated Data Repository (LEADR). Es un sistema de código abierto que está compuesto por un conjunto de herramientas de recogida y de intercambio de información (9). Fue diseñado inicialmente por varias administraciones locales para compartir información policial y judicial

sobrepasando fronteras y en la actualidad está siendo utilizado por cientos de administraciones del sudeste de los Estados Unidos.

El Consorcio Nacional de Sistemas de Gestión Penitenciaria (el Consorcio). Coalición mixta organizada con el propósito de desarrollar, mantener y mejorar un sistema global de bases de datos electrónicas (sistema de delincuentes) para la gestión de todos los aspectos relacionados con el encarcelamiento de delincuentes, la supervisión de los mismos y su rehabilitación (10).

A nivel nacional encontramos:

El Sistema Automatizado para el Control del Recluso (SACORE).Surge para dar cumplimiento a la Orden 43/99 del Vice-Ministro Primero del Ministerio del Interior de Cuba y tiene las siguientes características:

Garantiza respuestas inmediatas a las solicitudes de información de los diferentes órganos e instituciones del estado como son: Jefatura del MININT, Ministerio de Justicia, Tribunales, Fiscalías, MINED, INDER, FMC, MINFAR. Recoge prácticamente la totalidad de la información de los reclusos en todas las especialidades.

Sistema Automatizado para el control de Capacidades (SACDEP). Sistema que se crea con el objetivo de poseer información sobre las capacidades de los establecimientos penitenciarios, permitiendo registrar información de todos los locales del centro y para dar soporte a una mejor ubicación de los internos. Entre sus principales funcionalidades se encuentran: Diagnóstico de la infraestructura de las edificaciones. Capacidad de dormitorios en el área de reclusión. Clasificación, donde registra información importante sobre la construcción del establecimiento, datos generales sobre el estado del centro y sobre su estructura interna.

1.4 Herramientas que utilizan Razonamiento Basado en Casos en Cuba y el Mundo

A nivel nacional existen herramientas como:

Sistema Inteligente de Selección de Información (SISI): Sistema híbrido que combina Redes Neuronales Artificiales y RBC. Desarrollado en la Universidad Central de Las Villas en el año 1996. SISI es un SRBC del tipo interpretativo en tareas de diagnóstico.

Herramienta para la elaboración de sistemas de enseñanza inteligentes (HESEI): creada por el grupo de Informática Educativa e Inteligencia Artificial de la Universidad Central de Las Villas. Facilita la elaboración de SBC para el proceso de enseñanza-aprendizaje.

A nivel internacional:

CHEF: Es un planificador basado en casos, cuyo dominio son las recetas de cocina. CHEF crea las nuevas recetas a partir de otras ya conocidas, como respuesta a unos requisitos (sub metas) de platos con ingredientes específicos.

PROTOS: Implementa clasificación y adquisición de conocimiento basado en casos, dada una descripción de una situación u objeto la clasifica por su tipo. Cuando clasifica un elemento de forma incorrecta, su consultor experto le informa del error y del conocimiento que debe usar para clasificarlo correctamente.

SHYSTER: sistema jurídico basado en casos. Utiliza técnicas estadísticas para cuantificar la similitud entre los casos, y elige los casos sobre la base de esa medida de similitud.

Después de analizar cada una de estas herramientas se concluye que ninguna de estas aplicaciones es factible para dar solución al problema planteado. Las mismas no se adaptan a las características del dominio del problema, además de que no incluyen o tratan los procesos relacionados con la Clasificación Penitenciaria. Asimismo de que no se pueden tener en cuenta para la confección de la BC, ya que esto depende del tipo de información que va a almacenar. Estas herramientas no relacionan la clasificación penitenciaria con la técnica de Razonamiento Basado en Casos, además de ser sistemas privativos

1.5 Tecnologías y Herramientas

Metodología de desarrollo de software

Para asegurar el éxito durante el desarrollo de software no es suficiente contar con notaciones de modelado y herramientas, hace falta un elemento importante: la metodología de desarrollo, la cual provee de una dirección a seguir para la correcta aplicación de los demás elementos. Una metodología de desarrollo de software es un conjunto de pasos y procedimientos que deben seguirse para desarrollar software, indicando quién debe hacer cada actividad, cuándo hacerla y qué debe hacer (9). Puede seguir uno o varios modelos de ciclo de vida, es decir, el ciclo de vida indica qué es lo que hay que obtener a lo largo del desarrollo del proyecto pero no cómo hacerlo.

Actualmente las metodologías se clasifican en metodologías pesadas y metodologías agiles. Las metodologías ágiles, tienen como común denominador un modelo de desarrollo incremental para producir tempranamente pequeñas entregas en ciclos rápidos, y predisposición para el cambio y la adaptación continua; según sea la conformidad o no de lo producido, y las modificaciones propuestas por los usuarios. Estas metodologías por lo general se centran en desarrollar productos funcionales más que en conseguir una buena documentación (10). Son metodologías centradas en la implementación, que evitan cualquier tipo de documentación fuera del código. Por otra parte las Metodologías Pesadas están orientadas al control de los procesos, estableciendo rigurosamente las actividades a desarrollar,

herramientas a utilizar y notaciones que se usarán. Se decide optar por una metodología ágil para el desarrollo del software debido al poco tiempo de desarrollo y el reducido grupo de trabajo. Los principales objetivos que persigue esta metodología son (11):

- Valorar al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas.
- Desarrollar software que funcione más que conseguir una buena documentación.
- La colaboración con el cliente más que la negociación de un contrato.
- Responder a los cambios más que seguir estrictamente un plan.

Programación Extrema / Extrme Programin (XP).

La programación extrema es una metodología de desarrollo ligera basada en una serie de valores, principios y una docena de prácticas que propician un aumento en la productividad a la hora de generar software (11). XP es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. XP se basa en retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Entre las características que se basa la metodología XP están (11):

- Desarrollo iterativo e incremental.
- Pruebas unitarias continuas.
- Integración del equipo de programación con el usuario.
- Corrección de todos los errores.
- Refactorización del código.
- Propiedad del código compartida.

Entre las ventajas que posee XP se tienen:

- La comunicación, entre los usuarios y los desarrolladores.
- La simplicidad, al desarrollar y codificar los módulos del sistema.
- La retroalimentación, concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales

Lenguaje de modelado

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y

conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, configurar, mantener, y controlar la información sobre tales sistemas (12). Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. Ha sido diseñado por la combinación de estándares de tres metodologías procedentes de tres desarrolladores J. Rumbaugh, G. Booch e I. Jacobson. UML ayuda al usuario a entender la tecnología en realidad y la posibilidad de mostrar una idea antes de invertir en proyectos que no estén seguros en su desarrollo. Entre las propiedades que tiene este importante lenguaje se encuentra que modela estructuras complejas, soporta estructuras orientadas a objetos, como clases, componentes y nodos. Es un lenguaje distribuido y adaptado a las conectividades actuales y futuras y se define el comportamiento del sistema: casos de uso y diagramas de interacción se utilizara la versión 2.0.1.

Herramienta Case

Como herramienta Case se propone la utilización del Visual Paradigm en su versión 5.0. La misma es una herramienta que es soportada por cualquier sistema operativo. Es una herramienta visual de ingeniería de software para el modelado, que tiene un entorno de trabajo que muestra colección de menús, barra de herramientas y ventanas que permite realizar diagramas. Puede ser utilizada por una gran variedad de usuarios como analistas de sistemas, analistas de negocios e ingenieros de software. Provee soporte para la generación de código y la ingeniería inversa.

Lenguajes de programación

Los lenguajes de programación permiten crear programas y software. Estos facilitan la tarea de programación, ya que disponen de formas adecuadas que permiten ser leídas y escritas por las personas y a su vez resultan independientes del modelo de computador a utilizar. Para la futura implementación del sistema se propone el uso de Groovy en su versión 2.0.

Groovy

Groovy es un lenguaje de programación con una sintaxis análoga a Java que compila para Java Bytecode y corre en la Java Virtual Machine (JVM). Se integra completamente con Java, le permite mezclar y acoplar al código Groovy y Java con un esfuerzo mínimo (15).

Groovy es un lenguaje dinámico, es decir, todo ocurre en tiempo de ejecución, incluida la lógica de gestión de llamadas a métodos y acceso a propiedades. Ha sido influenciado por lenguajes como Ruby, Python, Perl, y Smalltalk, así como también Java. A diferencia de otros lenguajes que se adaptaron para la JVM, Groovy fue diseñado para la JVM, así es que no existe incompatibilidad (15).

La sintaxis de Groovy es más flexible y poderosa que la de Java. Las docenas de líneas de código en Java pueden acortarse a pocas líneas de código en Groovy ganando en legibilidad, mantenibilidad y eficiencia. Algunas personas se refieren a Groovy como un lenguaje scripting, aunque es mucho más que eso. Es un lenguaje orientado a objeto. Entre las características que distinguen a Groovy incluyen el tipado estático y dinámico, closures, sobrecarga de operadores, sintaxis nativa para listas, mapas, expresiones regulares, expresiones embebidas dentro de strings.

Este lenguaje es el seleccionado para darle solución final al sistema ya que como lenguaje se aprovechan todas las ventajas de la tecnología Java y se agregan algunas frescas como: el tipado dinámico y estático, el soporte nativo a listas, mapas, expresiones regulares. Además de que es un lenguaje dinámico, que disminuye cantidad de código respecto al lenguaje Java e incluye muchas nuevas funcionalidades en su GDK (Análogo al JDK de Java). El framework que utiliza este poderoso lenguaje, basado y construido para la madura y robusta tecnología Java es Grails.

Framework

La palabra framework define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar. Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones (16). Como framework se propone Grails en su versión 2.0.1

Grails

Es un framework de desarrollo web dinámico para la plataforma Java. Utiliza la flexibilidad de Groovy para proporcionar un dominio específico del lenguaje (DSL) para el desarrollo web. El objetivo es desarrollar aplicaciones web con el mínimo esfuerzo, sin tener que repetirse. Grails proporciona un entorno coherente y fiable entre todos sus proyectos (14). Las aplicaciones desarrolladas con este framework utilizan el patrón MVC (Modelo-Vista-Controlador). Grails está construido sobre sólidos proyectos de código abierto, como Spring, SiteMesh, GORM/Hibernate, todo sobre la máquina virtual de Java.

Es un framework open source para aplicaciones Web que potencializa el lenguaje groovy y complementa el desarrollo Java web. Se puede usar Grails como un entorno de desarrollo independiente que esconde todos los detalles de la configuración o integra la lógica del negocio hecha en Java (14). Grails apunta a hacer el desarrollo tan simple como sea posible y con esto sería muy atractivo para un amplio rango de desarrolladores no solo para aquellos que estén en la comunidad Java. Además no centra su desarrollo en reinventar lo bueno, sino mejorarlo: trata de tomar lo mejor de las tecnologías y adaptarlas mucho más

eficientes. Como ejemplo se puede citar que grails crea un simple DSL, que simplifica el trabajo con hibernate, dejando fuera las complicadas configuraciones de mapeo en archivos XML.

La integración de groovy y grails permite a los desarrolladores Web la construcción de sistemas robustos, bajo un entorno coherente y fiable, con un mínimo de esfuerzo. Donde se emplean conceptos como la reutilización de código. Y se obtienen sistemas modulares desarrollados bajo el popular patrón MVC. Soporta Ajax. Es multiplataforma. Cuenta con una gran comunidad de desarrolladores que lo mantiene en constante avance. Además una de las potencialidades de Grails es su bajo costo de instalación ya que solo necesita que estén instalados dos elementos: la máquina virtual de java (JVM) y el framework grails 1.1 o versiones superiores.

Gestor de Base de Datos

Los Sistemas Gestores de Bases de Datos (SGBD), son software que permiten la administración y mantenimiento de los ficheros y datos de una base de datos o de varias. Proporcionan funcionalidad añadida al sistema de ficheros para facilitar la gestión de datos. Además proporcionan valor añadido, como seguridad en cuanto a acceso, copias de respaldo, entre otras. En la actualidad existen disímiles SGBD. Para la implementación de la solución se propone el uso de PostgreSQL en su versión 9.1.

PostgreSQL

Dentro de los gestores de bases de datos existentes, se nombra como uno de los más distintivos. Está diseñado para soportar volúmenes masivos de datos, sin que ello afecte en lo absoluto su rendimiento. Ofrece una fortaleza adicional al incorporar cuatro conceptos básicos: clases, herencia, tipos y funciones. Cuenta además con características que aportan potencia y flexibilidad adicional: restricciones, disparadores, reglas, integridad transaccional.

PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Soporta operadores, funciones métodos de acceso y tipos de datos definidos por el usuario. Soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos. Posee soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Corre en la casi totalidad de los principales sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows, etc. Soporta el protocolo de comunicación encriptado por SSL. Posee utilidades para limpieza de la base de datos (Vacum) y para el análisis y optimización de Query's (15).

1.6 Conclusiones de capítulo

En este capítulo se realizó un estudio del estado del arte del proceso de clasificación penitenciaria Este estudio arrojó como resultado que los sistemas estudiados no satisfacen las necesidades del problema planteado en cuanto a la clasificación penitenciaria, por lo cual se concluyó que la solución adecuada es el desarrollo de un sistema inteligente de clasificación penitenciaria. Se seleccionó XP para guiar el proceso de desarrollo de software y en cuanto a la implementación se determinó utilizar como lenguaje de programación Groovy en su versión 2.0. Para facilitar el desarrollo de la propuesta se usa el framework Grails 2.01 el cual está basado en el lenguaje Groovy. Finalmente se seleccionó PostgreSQL 9.1 como sistema gestor de base de datos para el almacenamiento y manejo de los mismos.

CAPÍTULO 2: FORMALIZACIÓN DEL MODELO COMPUTACIONAL PARA LA CLASIFICACIÓN PENITENCIARIA

Introducción al capítulo

En el presente capítulo se especifican los componentes de la base de casos. Se define y seleccionan la estructura de los casos así como la organización de la memoria. Se realiza la selección de los rasgos que influyen en la clasificación de los internos. Además se describe la composición de los rasgos, definiendo el mecanismo de recuperación con las respectivas funciones de comparación, así como la función de evaluación de la semejanza de casos. También se define el mecanismo de adaptación y la estrategia de aprendizaje del sistema.

2.1 Técnicas para la adquisición del conocimiento

Para lograr identificar los rasgos predictivos y objetivos se hace necesario aplicar métodos tales como:

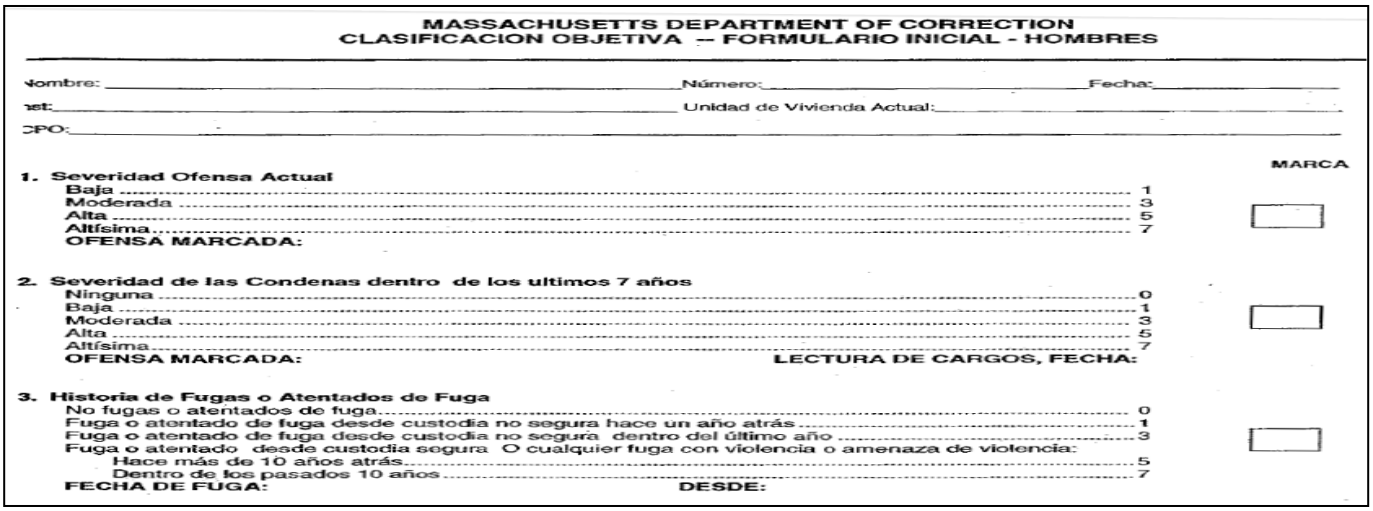
- Conocimiento documentado: consiste en realizar la extracción de conocimiento de diversas fuentes implicadas (libros, artículos de revistas, manuales, informes y documentos de ayuda), en formato digital o impreso, ya que cualquier documento podía ser útil para la introducción de conocimiento en la base de conocimiento.
- Observación directa: mediante pláticas y la visualización del trabajo desempeñado por los expertos en el tema, observación de las actividades que realiza el experto humano, para así encontrar pistas acerca del conocimiento necesario y la forma en que éste se aplica en la clasificación de los proyectos y además comprobar la coherencia de lo observado con lo dicho por los expertos humanos en las entrevistas.
- Consulta a Expertos: es una técnica que consiste en realizar una encuesta a expertos de distintas especialidades, para llegar a un acuerdo final sobre la veracidad de lo que se quiere validar o comprobar, posteriormente se realiza un tratamiento estadístico para determinar si se aceptó el criterio de los expertos.
- Entrevista a especialistas: para validar la información contenida en la base de conocimientos a utilizar, es necesaria la aplicación de una entrevista a especialistas en el tema.

Para la confección de la solución se tuvieron en cuenta 2 de las técnicas antes mencionadas. Fue empleado el conocimiento documentado, en aras de la obtención de las posibles clasificaciones de los internos teniendo en cuanta el criterio de la solución propuesta en el capítulo. Después de realizada la búsqueda en la documentación se propone el uso de consulta de expertos para la validación de los rasgos identificados u otras clasificaciones.

2.2 Selección de los criterios para la Clasificación

Para la selección de los criterios de clasificación o rasgos (para un Sistema Basado en Casos) se realizó un análisis documental de los sistemas de clasificación utilizados en diferentes países. Se consideró para el estudio el sistema de clasificación objetiva en Los Estados Unidos de América (EEUU) (16), el sistema de clasificación en los sistemas penitenciarios de Nicaragua (17), España (15) (18), Cuba (6), y el Protocolo de clasificación en Venezuela (19). Como resultado del análisis se identificaron las principales áreas de evaluación de los internos y los criterios que se deben considerar en cada una de ellas. Se destacan como elementos comunes los antecedentes delictivos, tipo y gravedad del o los delitos cometidos, el estado de salud mental del interno, características de su período escolar, relaciones sociales, estabilidad laboral, capacidad criminal así como la influencia familiar. Sobre la base de este análisis, los resultados esperados de la clasificación estarían de acuerdo con el régimen de severidad requerido para la ubicación del interno en el establecimiento penitenciario definido en Cuba y agrupando las cuatro clasificaciones existentes; y de esta forma distribuirlos según máxima severidad, mayor severidad, media severidad y severo.

Los sistemas de clasificación objetiva se basan esencialmente en establecer los criterios significativos para la determinación de la clasificación y un sistema de puntuación para los posibles valores de cada uno de ellos. Para determinar la clasificación de un individuo, éste se evalúa por cada uno de estos rasgos y se calcula la puntuación total obtenida. Una vez calculada la puntuación total, se asigna la clasificación, de acuerdo a su pertenencia a determinados rangos. En la **Figura 2.1** se muestra un fragmento de un formulario de clasificación objetiva utilizado por el departamento de correccionales de Massachusetts, EEUU. Lo distintivo de este método es la necesidad de un nivel o sistema de puntuación para la obtención de la clasificación, por lo que se necesita una base o distribución de puntos por preguntas que bien pueden volverse obsoletas o demasiado variables, motivo por el cual podrían falsear la clasificación.



MASSACHUSETTS DEPARTMENT OF CORRECTION
CLASIFICACION OBJETIVA – FORMULARIO INICIAL - HOMBRES

Nombre: _____ Número: _____ Fecha: _____
Set: _____ Unidad de Vivienda Actual: _____
CPO: _____

MARCA

1. Severidad Ofensa Actual
Baja 1
Moderada 3
Alta 5
Altísima 7
OFENSA MARCADA:

2. Severidad de las Condenas dentro de los ultimos 7 años
Ninguna 0
Baja 1
Moderada 3
Alta 5
Altísima 7
OFENSA MARCADA: LECTURA DE CARGOS, FECHA:

3. Historia de Fugas o Atentados de Fuga
No fugas o atentados de fuga 0
Fuga o atentado de fuga desde custodia no segura hace un año atrás 1
Fuga o atentado de fuga desde custodia no segura dentro del último año 3
Fuga o atentado desde custodia segura O cualquier fuga con violencia o amenaza de violencia:
Hace más de 10 años atrás 5
Dentro de los pasados 10 años 7
FECHA DE FUGA: DESDE:

Figura 2. 1 Fragmento de un formulario de Clasificación Objetiva. Se definen tres aspectos: Severidad del Delito Actual, Severidad de las Condenas dentro de los últimos 7 años e Historial.

La validación de los rasgos identificados se realizó a través de criterios de expertos, con la que se pretende tener estimaciones de precisión razonables sobre la validez de la selección. Para que una persona sea considerada experta debe poseer un conocimiento profundo de la tarea o actividad que será analizada y estar familiarizado con la misma (20).

Para la selección de los expertos se utilizó el método Delphi. Se definió como primera condición que los especialistas tuvieran experiencia en los procesos de clasificación en sistemas penitenciarios. Se le aplicó una encuesta de manera individual para determinar el coeficiente de competencias y medir el nivel de conocimiento que poseía esa persona para ser considerado como experto. Después de realizada la encuesta (Ver Anexo 1), se procedió a determinar el coeficiente de competencias (K) según define el método mediante la fórmula: $K = (K_c + K_a) / 2$ siendo K_c el coeficiente de conocimientos y K_a el coeficiente de argumentación (21). En la encuesta se incluyen preguntas que permiten obtener valores para K_a y K_c y solo se tuvieron en cuenta los expertos con $K>=0.5$, porque se consideran expertos con competencia media o alta (Ver Anexo 2).

A partir de la determinación de los expertos se definió y aplicó una encuesta a estos con el fin de determinar el grado de factibilidad de cada criterio por separado para contribuir a la clasificación de manera general y además el grado de factibilidad de cada criterio para el área de evaluación (Ver Anexo 3). De esta manera se determinaron los rasgos a tener en cuenta para la definición de los casos del SRBC, y a partir del grado de factibilidad o el criterio de expertos según la encuesta propuesta de cada

rasgo, se estableció su peso (Ver Anexo 4). En la Tabla 2.1 se muestran los rasgos seleccionados y sus posibles valores de dominio, agrupados por áreas. La Tabla 2.2 muestra el rasgo objetivo y sus posibles valores.

Tabla 2. 1 Rasgos predictores para la clasificación y sus posibles valores, agrupados por áreas

Rasgo predictor	Posibles valores	Dominio
Área Delictiva		
Debut antisocial	Temprano (antes de los 12 años), Medio (entre los 12 y los 20 años), Tardío (Después de los 20 años)	Ordinal
Antecedentes delictivos	Primario, Reincidente, Multi-reincidente	Ordinal
Desviaciones de la conducta	Ninguna, Menores, Moderadas, Graves, Críticas	Ordinal
Tipos de delitos	Imprudentes, Codiciosos , Violentos	Ordinal
Tiempo permanecido en estado de reclusión	Ninguno, Bajo (hasta 5 años), Medio (entre 5 y 10 años), Alto (más de 10 años)	Ordinal
Gravedad del delito	Leve, Moderada, Grave	Ordinal
Versatilidad delictiva	Ninguna, Baja, Media, Alta	Ordinal
Área de Salud Mental		
Estado de salud mental	Favorable, Desfavorable	Ordinal
Área Escolar		
Fugas escolares	Preuniversitario, Primaria, Secundaria	Ordinal
Abandono escolar	No, Tardío (De 9no a 12vo grado), Temprano (de 6to a 9no grado), Muy temprano (antes 6to grado)	Ordinal
Escuela especial	Sí, No	Binaria
Retardo en el desarrollo Psíquico	Sí, No	Binaria
Retraso Mental	Sí, No	Binaria
Repitió grados	Sí, No	Binaria
Riñas frecuentes	Sí, No	Binaria
Falta de respeto	Sí, No	Binaria
Área de Relaciones Interpersonales		
Relaciones con elementos de conducta antisocial	Ninguna, Ocasional, Estable	Ordinal
Adaptabilidad social	Buena, Dificultosa, Inadaptado	Ordinal
Área Laboral		

Estabilidad Laboral	Estable, Inestable	Ordinal
Area de Personalidad		
Capacidad criminal	Insensible, Frío, Egocéntrico, Violento, Manipulador, Descontrolado, Vengativo, Impulsivo.	Ordinal
Estilo de vida	Adecuado, Inadecuado	Ordinal
Area Familiar		
Funcionalidad familiar	Funcional, Disfuncional	Ordinal
Influencia de Familiares recluidos o con antecedentes delictivos	No tiene, Baja, Alta	Ordinal

Tabla 2. 2 Rasgo objetivo para la clasificación y sus posibles valores, agrupados por áreas

Rasgo objetivo	Posibles valores
Clasificación propuesta	Mínima severidad, Media severidad, Mayor severidad, Severo

2.3 Razonamiento Basado en Casos (RBC)

El proceso que realiza el RBC es análogo al razonamiento humano. Cuando una persona se enfrenta a un problema por simple que sea, empieza por buscar en su memoria experiencias similares a esta, combinando una serie de semejanzas y diferencias que ayude a obtener la nueva solución. Este es un proceso intuitivo que las personas lo realizan prácticamente sin darse cuenta. En este principio se basa el RBC. La elaboración de un sistema que emplea el RBC presenta dos problemas principales: el primero, saber cómo almacenar la experiencia de tal forma que esta pueda ser recuperada en forma adecuada y el segundo, conseguir utilizar la experiencia previa en un problema actual (22).

2.3.1 Estructura de los casos

Según Koloder un caso es una pieza contextualizada de conocimiento que representa una experiencia que enseña una lección fundamental para alcanzar los objetivos del razonador (25). Es por eso que una vez que se adquiere el conocimiento, es necesario encontrar una representación simbólica, clara, precisa y completa del mismo.

Para la representación de los casos se propone el uso de la estructura atributo-valor. En esta estructura los casos se representan como dos conjuntos desestructurados de pares atributo-valor que representan el problema y las características de la solución. En esta representación cada dato se representa con un

número fijo de atributos, junto con la valoración de esos atributos para ese dato. Es decir, los atributos representan las características del problema. Esta representación posee gran sencillez de implementación y mediante la utilización de esta estructura se puede simplificar considerablemente la complejidad de la solución.

La forma de representar y almacenar las características de los internos para obtener una clasificación en los que le permita una estancia favorable en los centros penitenciarios, se representa a través de casos.

Un caso es constituido por dos elementos generales: los rasgos predictores y los rasgos objetivos **(Figura 2.2**). La especificación de los primeros dará lugar a la salida o solución del problema, representado por el segundo. Cuando se encuentra un caso declarado en la BC se da por hecho que este poseerá un cierto grado de riqueza en la información que contiene, es decir, la experiencia está perfectamente descrita y se procura que no falte información en su descripción, además es necesario que toda esta información que contiene posea un cierto grado de organización que permita su rápido entendimiento y sobre todo llegar a la información necesaria rápidamente, minimizando acceder a información innecesaria.

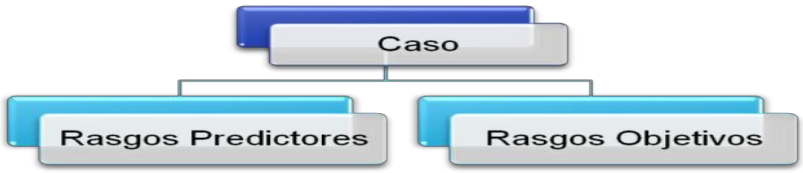


Figura 2. 2 Elementos generales de un caso

La calidad de un RBC está dada por (22):

1. La experiencia que tiene.
2. La habilidad para entender situaciones nuevas en términos de experiencias pasadas.
3. Su capacidad de adaptación.
4. Su habilidad para integrar nuevas experiencias en su memoria adecuadamente.

Arquitectura y Componentes

Los SRBC contienen dos componentes fundamentales para su funcionamiento: una Base de Casos (BC) y un componente encargado de solucionar los problemas. La BC es quien contiene todas las descripciones que han sido obtenidas previamente; cada caso representa las características específicas de un interno. En el ciclo de solución del problema se recupera un caso semejante al nuevo y la solución del problema que fue recuperado es propuesta como solución potencial del nuevo problema, luego el nuevo caso

obtenido podrá formar parte de la BC. Esto se deriva de un proceso de adaptación en el cual se indexa la vieja solución a la nueva situación.

El funcionamiento del RBC parte del principio de simular el razonamiento humano detallado anteriormente y para ello comprende cuatro actividades principales.

- Recuperar los casos más parecidos: esta recuperación tendrá lugar en la BC donde está almacenada la experiencia obtenida en forma de casos.
- Reutilizar el o los casos para tratar de resolver el nuevo problema: esta tarea está centrada en dos aspectos: las diferencias entre el caso pasado y el actual, y qué parte o partes del caso recuperado pueden ser transferidas al nuevo caso.
- Revisar y adaptar la solución propuesta, en caso de ser necesario: Una vez analizada la tarea de reutilización, la solución del nuevo problema tiene que ser probada; este proceso de prueba se hace durante la tarea de revisión.
- Almacenar la nueva solución como parte de un nuevo caso: El nuevo caso es almacenado con vista a un posible uso futuro. Durante este proceso de aprendizaje, el sistema tiene que seleccionar qué información del caso almacenar y cómo indexar el caso en la estructura de memoria para una posterior recuperación.

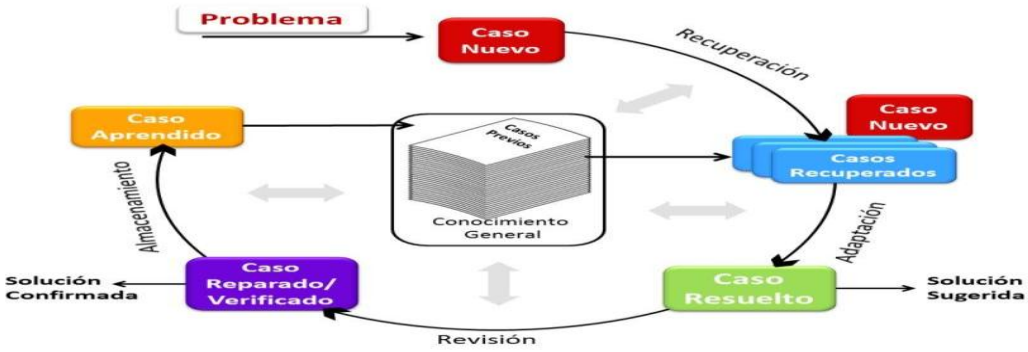


Figura 2. 3 Ciclo básico de un SRBC

En la **Figura 2.3** se puede observar el ciclo básico de un SRBC. La descripción de un nuevo problema da lugar a la aparición de un nuevo caso que se desea resolver. Este pasa a la fase de recuperación donde se obtienen los casos más similares a él, registrados en la BC. Luego en una fase de adaptación el nuevo caso es combinado con él/los casos recuperados de la BC para dar solución al nuevo problema planteado. En una fase de evaluación se verifica el éxito de solución, la mayoría de las veces esta etapa es llevada a

cabo por un agente humano. Durante el almacenamiento, la experiencia útil se guarda para una futura reutilización y el caso base se actualiza como un nuevo caso aprendido, o se modifican algunos casos existentes, evitando así, tener datos redundantes en la BC.

2.4 Representación de la Base de casos

El componente principal de un SRBC, es la base de casos, la cual constituye la memoria del sistema. La Base de Casos (BC) es el componente encargado de almacenar y organizar todos los casos disponibles y su estructura es crucial para la fase de recuperación de casos similares(24).

El conjunto de problemas resueltos (casos) forman la BC o memoria permanente del sistema, a partir de los cuales se hacen las inferencias. En la BC se registran los casos resueltos; esta debe poseer una organización que le permita mostrar cualidades similares a la memoria humana, es decir, debe ser ilimitada, en la medida que la memoria crezca no se puede hacer más lenta y debe permitir buscar directamente los elementos de memoria que sean relevantes a un problema.

2.4.1 Estructura de la memoria

Dado que los casos constituyen el elemento principal de todo SRBC, la manera de almacenarlos repercutirá directamente en el rendimiento del mismo. Tradicionalmente se han propuesto dos estructuras principales para la el almacenamiento de casos:

Memoria Plana

Los casos son unidades de información completas que se disponen en forma secuencial en una lista, arreglo o fichero. Este tipo de memoria presenta la ventaja de que añadir nuevos casos resulta muy “rápido y fácil de implementar”. No ocurre así con la recuperación de casos, ya que, aunque el procedimiento de recuperación de los casos más similares es sencillo, resulta muy lento cuando el número de casos en la base es alto. (26)

Memoria Jerárquica

Los casos están colocados en nodos de un árbol o un grafo acíclico dirigido. El grafo subdivide los casos de acuerdo a los atributos que comparten. Esto se hace localizando atributos comunes en los nodos internos del grafo. Todos los casos que comparten los valores se sitúan debajo de dicho nodo. Este modelo tiene la ventaja de mejorar la eficiencia en el acceso, pero muestra varias desventajas (3):

- Añadir casos requiere modificar la estructura.
- Mantener la red en forma óptima es costoso.

En la **figura 2.4** se muestran las estructuras de memoria plana y jerárquica

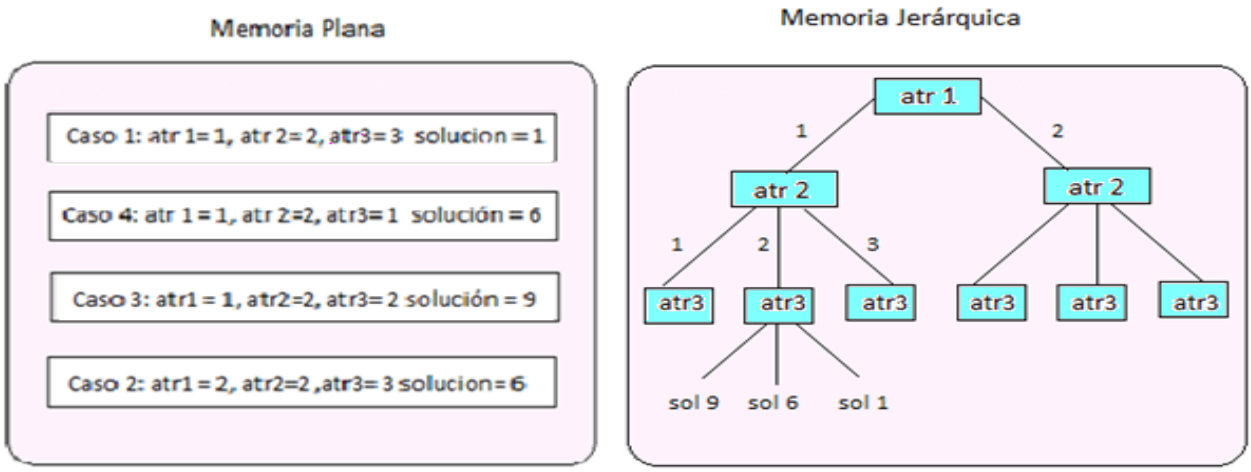


Figura 2. 4 Estructura plana y jerárquica.

En la selección de la estructura que formará la BC hay que tener en cuenta aspectos fundamentales como: el tamaño que se aspira que tenga la BC, los requerimientos que se imponen en el dominio específico en el que se trabaja y la inserción eficiente de nuevos casos (3). Teniendo en cuenta estos aspectos se seleccionó la memoria plana para la representación de la BC. Con esta estructura se logrará realizar un análisis exhaustivo sobre los casos para recuperar los mejores. A diferencia de las memorias jerárquicas estas no almacenan datos auxiliares para realizar la búsqueda minimizando el espacio que ocupan los casos (3). Además con el desarrollo de las tecnologías en la informática en los últimos años, el uso de la memoria ya no es un problema, por lo que si la BC fuese de gran tamaño, el mecanismo de recuperación de la misma no se vería afectado, ni el rendimiento de la aplicación.

2.5 Recuperación de los casos

El proceso de recuperación es responsable de la obtención de los casos más semejantes al nuevo caso, donde los rasgos predictores se valorizan por uno de los métodos establecidos. El procedimiento de recuperación consta de dos etapas: acceso y recuperación. En la etapa de acceso se seleccionan los casos potenciales para el proceso de recuperación.

2.5.1 Algoritmo de recuperación

Los algoritmos de recuperación más extendidos son: k-vecinos más cercanos, aplicable a las estructuras de memorias planas y árboles de decisión, aplicable el acceso jerárquico.

K-vecinos más cercano

El caso recuperado es aquel en el que la suma de los pesos de las características que se ajustan con las del caso nuevo es mayor que la de otros casos que ajustan. Esto quiere decir, que si los pesos de las características son iguales, un caso que ajusta con n características será recuperado antes que otro caso que empareja con k características, siendo k<n. Se puede asignar un peso mayor a las características consideradas más importantes. El vecino más cercano es una técnica muy simple que proporciona una medida de cuán similar es un caso objetivo a un caso de la BC. Pero su principal desventaja es la velocidad de recuperación ya que para encontrar el caso que mejor ajusta, el caso nuevo debe compararse con cada caso de la BC (27).

Árboles de decisión

Los árboles de decisión representan una estructura de datos que organiza eficazmente los descriptores; dichos árboles son construidos de forma tal que en cada nodo se realiza una prueba sobre el valor de los descriptores y de acuerdo con la respuesta se va descendiendo en las ramas, hasta llegar al final del camino donde se encuentra el valor del objetivo (28).

Para la recuperación del sistema se utilizó el algoritmo de recuperación K-Vecinos más cercano, ya que el mismo está más orientado a las estructuras de memorias plana, por su rápido acceso, además de adaptarse a las características que presenta la misma.

2.6 Funciones de comparación de rasgos

Para cada rasgo fue definida una función de comparación que expresa el grado de semejanza entre dos valores del dominio. Para la definición de funciones de comparación entre rasgos, se encontraron dos casos:

Rasgos que toman valores Binarios

Las variables binarias tienen dos valores, 1 o 0, donde el primero indica que el rasgo está presente y el segundo lo contrario. Para los casos en que la presencia o ausencia del rasgo en ambos objetos son igualmente relevantes, se utiliza la siguiente función de comparación.

$$\delta_i(O_o, O_t) = \begin{cases} 1, X_i(O_o) = X_i(O_t) \\ 0, \text{e. o. c.} \end{cases} \quad [1]$$

Donde:

$\delta_i(O_o, O_t)$: define la función de comparación entre los casos O_o y O_t,

$X_i(O_o)$: el valor del rasgo i para el caso O_o y

$X_i(O_t)$: es el valor del rasgo i para el caso O_t.

Rasgos que toman valores Ordinales

Estos rasgos expresan medidas o cualidades que no pueden ser calculadas de manera objetiva, tal es el caso del rasgo ***Versatilidad Delictiva***, que puede tomar los valores Baja, Media, Alta. En estos casos, no son esenciales los valores en sí, sino su orden relativo. Los valores ordinales son normalizados y luego comparados utilizando la siguiente función, basada en la distancia de Manhattan:

$\delta_i(O_o, O_t) = 1 - \left|\frac{X_i(O_o) - X_i(O_t)}{r_{máx} + r_{mín}}\right|$ [2]

Donde:

$\delta(O_o, O_t)$: define la función de comparación entre los casos O_o y O_t,

$r_{máx}$ y **$r_{mín}$**: los valores de rango máximo y mínimo del conjunto de valores existentes respectivamente y

$X_i(O_o)$: el valor del rasgo i para el caso O_o y $X_i(O_t)$: es el valor del rasgo i para el caso O_t.

Para normalizar los valores se prosigue de la siguiente manera: Se define M_i como la cantidad de estados ordenados que puede tomar el rasgo i. Se hace corresponder a cada valor del dominio, según su orden, un número entero entre 1 y **M_i**. Por ejemplo, para el caso de la Versatilidad Delictiva, **M_i = 3; Baja = 1, Media = 2, Alta = 3**. Luego a cada valor del dominio, se hace corresponder un valor real entre 0 y 1, siguiendo la siguiente fórmula (29):

$X_i = \frac{v-1}{M_i - 1}$ [3]

Donde:

v: representa el valor del número entero asignado al valor del rasgo y

M_i: la cantidad de estados ordenados.

Función de semejanza entre casos

Una vez definidas las funciones de comparación entre rasgos, se define la función de semejanza entre casos de la siguiente manera y siguiendo lo propuesto en el algoritmo k-vecinos más cercanos (3):

$\beta(O_o, O_t) = \frac{\sum_{i=1}^{n} p_i * \delta_i(O_o, O_t)}{\sum_{i=1}^{n} p_i}$ [5]

Donde:

$\beta(O_o, O_t)$: define la semejanza entre los casos O_o y O_t

p_i: define los pesos de los rasgos,

$\delta(O_o, O_t)$: define la función de comparación entre los casos O_o y O_t .

2.7 Valor umbral

El umbral es la cota inferior de los posibles valores de semejanza existente para cada uno de los casos recuperados con respecto al nuevo caso. Su verdadera función consiste en restringir el intervalo en caso de que se quiera lograr una mayor precisión en la obtención de los casos más semejantes (30). Este umbral podrá ser definido por el usuario que interactúe con el sistema y este puede variarlo en medida de qué tan exigente pueda ser con el resultado. Un umbral muy elevado puede representar la ventaja de que solo se tengan en cuenta en la fase de adaptación los casos con un alto nivel de semejanza, pero puede presentar la desventaja de que ninguno de los casos recuperados lleguen o sobrepasen este valor, lo que traería consigo no tener al menos un caso para poder realizar la adaptación.

El valor de umbral se definirá por el experto, teniendo en cuenta que para tener un menor margen de error a la hora de llevar a cabo la clasificación el valor de umbral debe ser elevado, lo más cerca posible a uno que es el mayor grado de semejanza.

Si el usuario presenta problemas con la definición del umbral, el módulo de recuperación le brindará el más adecuado según los datos que tiene registrados en la BC. Para ello se construye una matriz cuadrada donde la fila y las columnas están representadas por los casos que se encuentran almacenados en la BC y en la intersección está el valor de semejanza (**β**).

Tabla 2. 3 Matriz de semejanza entre casos

	Caso 1	**Caso 2**	**Caso 3**	**Caso n**
Caso 1	β (C1,C1)	β (C1,C2)	β (C1,C3)	β (C1,Cn)
Caso 2	β (C2,C1)	β (C2,C2)	β (C2,C3)	β (C2,Cn)
Caso 3	β (C3,C1)	β (C3,C2)	β (C3,C3)	β (C3,Cn)
Caso n	β (Cn,C1)	β (Cn,C2)	β (Cn,C3)	β (Cn,Cn)

Luego de construida la matriz, mediante el cálculo de μ se obtiene el valor umbral (fórmula 6) que dependerá de los valores de semejanza entre los casos contenidos en dicha matriz (30).

$$\mu = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \beta(O_{o,} O_t) \text{ [6]}$$

Donde:

m: número de casos

i: valor que recorrerá las filas

j: valor que recorrerá las columnas

$\beta(O_{o,} O_t)$: semejanza entre el caso 0 y el taso t

Luego de definido el umbral y comparar los valores de semejanza de los casos recuperados, solamente pasan a la fase de adaptación aquellos casos que sean iguales o estén por encima de dicho umbral.

2.8 Adaptación

Después de realizar la búsqueda y selección de las clasificaciones candidatas resultantes de la recuperación (como primer elemento de entrada al ciclo de razonamiento), se procede a la adaptación de la solución, paso siguiente en el razonamiento basado en casos. Una técnica propuesta para el sistema desarrollado lo constituye la adaptación por **ajuste de parámetros**, la cual en términos generales consiste en extraer las diferencias y para cada diferencia aplicar una técnica de ajuste especializada.

Existen varias definiciones de métodos de adaptación los cuales están agrupados en dos colecciones, los métodos de sustitución y los de transformación (30). Estos pueden aplicarse de manera individual o aprovechar las facilidades que brindan más de uno y hacerlas converger para mejorar el resultado.

Los métodos de sustitución dan valores apropiados para la nueva situación dependiendo de los valores de la solución previa. Se puede sustituir sólo un componente de la solución, varios de ellos, o todos los componentes de una solución. Estos son usados cuando se cuenta con una BC bastante sólida. Cuando no se cuenta con una BC lo suficientemente sólida los métodos de transformación son una alternativa muy viable, pues generan una nueva solución basándose en las restricciones y características de la solución requerida.

La técnica más simple es no realizar adaptación y transferir la solución vieja al nuevo problema directamente. Si para todos los elementos seleccionados los valores de los rasgos desconocidos en el nuevo problema son iguales, entonces se toman esos valores como solución para el problema, en otro caso habrá que elegir entre las diferentes posibilidades. Una variante es elegir los valores del caso con mayor semejanza.

Agrupados dentro de los métodos de sustitución se pueden encontrar la reinstanciación y el ajuste de parámetros, dos de los principales métodos de adaptación.

La reinstanciación o adaptación nula es flexible a la hora de definir un criterio de selección, ya que no especifica un criterio estándar. Esta adaptación se pone en práctica en la mayoría de los sistemas de clasificación. Para establecerlo se necesita una BC que tenga una gran cantidad de casos almacenados, ya que la variedad garantizará la precisión del resultado del problema. La solución es un proceso derivacional en tres pasos (3):

1. Abstraer el marco del caso previo y su solución.
2. Calcular la correspondencia entre los atributos de los dos problemas.

3. Instanciar el marco del caso previo y su solución basándose en esas correspondencias.

Ajustes de Parámetros es un tipo de técnica de adaptación estructural la cual se utiliza previa o posteriormente a un proceso de reinstanciación. Se tiene el problema de que los parámetros en el problema actual son diferentes a los parámetros en el caso previo, el método consiste en extraer las diferencias y para cada una aplicar una técnica de ajuste especializada. Las heurísticas de ajuste contemplan las relaciones entre los atributos del problema y los atributos de la solución. Las descripciones de los problemas se comparan a partir de un conjunto de parámetros especificados y la diferencia entre ellos se utiliza para modificar los parámetros de la solución en la dirección oportuna. Para obtener los parámetros de una nuevo caso es necesario extraer primero las diferencias entre este y el caso reusado. Las heurísticas de ajuste capturan las relaciones entre características del problema y parámetros de la solución (3).

Como se especificó anteriormente la técnica seleccionada para realizar la adaptación es el ajuste de parámetros, en esta técnica se pueden ajustar las características o los pesos asociados a estas. Para resolver el problema se ajustaran las características, teniendo en cuenta las que el experto consideró que eran más relevantes (las de mayor peso) para dar el resultado de la clasificación. Después de ajustarlas, al nuevo caso formado se le compara con el caso entrado y si el valor de semejanza es mayor que la de los casos recuperados se guarda en la BC, ya que el caso ajustado es nuevo y tiene un mayor grado de similitud. Este método de adaptación se considera también un mecanismo de aprendizaje, pues su funcionamiento hace que los casos ajustados puedan ser aprendidos.

2.9 Taza de acierto del clasificador propuesto

Para evaluar la taza de acierto del clasificador propuesto, se emplearon cuatro bases de datos publicadas en el Machine Learning Repository de la Universidad de California (Center for Machine Learning and Intelligent Systems):

- Iris, suministrada por Michael Marshall, 1988. Base de datos que contiene cuatro atributos numéricos (anchura y longitud del sépalo y pétalo) y uno nominal (tipo de iris) que determina el tipo de planta iris. El conjunto de datos contiene tres clases, cada una de cincuenta instancias las cuales refieren características de uno de los tipos de planta iris.
- Car Evaluation (Evaluación de automóviles), suministrada por Marko Bohanec, 1997. Base de datos derivada de un simple modelo de decisión jerárquico, compuesto por seis atributos

nominales (precio de venta, tipo de mantenimiento, puertas, personas, zapata de arranque y seguridad) y uno nominal que determina el tipo de característica del auto.

- Wine Quality (Calidad del vino blanco), suministrada por Paulo Cortez, Universidad de Minho, Guimarães, Portugal, 2009. Base de datos compuesta por once atributos numéricos (acidez fija, acidez volátil, ácido cítrico, azúcar residual, cloruros, dióxido de azufre libre, dióxido de azufre total, densidad, pH, sulfatos y alcohol) y otro numérico a inducir la calidad del vino en una escala del uno al diez.
- Abalone, suministrada por Warwick J Nash, Tracy L Sellers, Simon R Talbot, Andrew J Cawthorn y Wes B Ford, 1995. Base de datos sobre una especie de caracol marino llamado Haliotis, presenta siete atributos numéricos (longitud, diámetro, ancho, peso total, peso extraído, peso de viseras y peso shell), uno entero (cantidad de anillos) como rasgos descriptores y uno nominal a inducir que identifica el sexo.

A forma de resumen se muestra en el (Anexo5) los resultados obtenidos en la evaluación del clasificador propuesto en base a la tasa de acierto. En este caso se presenta el clasificador, empleando para datos numéricos la función de Manhattan ajustada. Como resultado se concluye lo siguiente: se prueba que estadísticamente el clasificador es correcto, ya que en cada uno de los casos la taza de acierto supera el 90% y la taza de acierto promedio es de 92.8%.

2.10 Conclusiones parciales

En el presente capitulo se definieron las técnicas para la adquisición del conocimiento. Se conformaron los rasgos de los casos con sus posibles valores. Se definió la estructura de la BC, las funciones de semejanza entre casos y las funciones de comparación de los rasgos atendiendo a la cantidad de valores de dominio que pudieran tomar, se precisó el mecanismo de adaptación concluyendo que este mecanismo o técnica, es también un mecanismo de aprendizaje.

CAPÍTULO 3: CARACTERISTICAS Y DISEÑO DEL SISTEMA

Introducción al capítulo

En este capítulo se describirá el Sistema de Clasificación Penitenciaria, analizando sus funcionalidades y elementos principales. Además se describen las fases de Exploración y Planificación, propias de la metodología de desarrollo utilizada. Se muestran las Historias de Usuario (HU) más importantes para cada iteración definida. Se identifican las clases arquitectónicamente significativas para las funcionalidades del sistema y patrones de arquitectura y de diseño utilizados en el desarrollo de la aplicación web.

3.1 Objeto de estudio

El objeto de estudio sobre el cual se enmarca este trabajo de diploma es el proceso de Clasificación Penitenciaria.

3.2 Problema y situación problemática.

En todos los centros penitenciarios la correcta clasificación es una de las principales formas de mantener un programa de reeducación. Antes del triunfo de la revolución, en Cuba existía el maltrato al interno dentro de las instituciones penitenciarías. Con el transcurso de la revolución se fueron tomando medidas para mejorar la estancia de los internos dentro de las instituciones penitenciarias, logrando así una mejora en el tratamiento a los internos dentro de los centros penitenciarios. De llegar a hacerse una clasificación errónea de los internos, podría suceder que se ubiquen a los mismos sin tener en cuenta la edad, la gravedad de lo delitos y la situación personal: procesados o condenados, delincuentes primarios o reincidentes, personas sanas y personas física y mentalmente enfermas. Esto conlleva que los internos se sientan intimidados o reprimidos, además de no poseer del trato adecuado con que se le debe atender posteriormente. Todas estas circunstancias influyen desfavorablemente en los recluidos y deberían evitarse.

3.3 Información que se maneja

La información que se maneja para el desarrollo de este trabajo está relacionada con las clasificaciones más comunes que pueden presentarse o que se han presentado en los diferentes Centros Penitenciarios de nuestro país. Además, se utilizará información referente a las características propias de los sistemas de clasificación de los centros penitenciarios. Por lo que los elementos identificados como rasgos surgen del estudio de sistemas de clasificación de todo el mundo. Esto conlleva a que no se maneja información específica ni sensible de los centros penitenciarios, sino lo más general que permite de forma simple de inferir una posible clasificación del régimen de severidad.

3.4 Solución propuesta

La solución propuesta, Sistema de Clasificación Penitenciaria apoyada en RBC, está diseñada para que todas las instituciones penitenciarías del país hagan uso de la misma y se logre una mejor clasificación de los internos dentro de los sistemas penitenciarios. El sistema tiene como objetivo principal: clasificar en términos generales, sin la asistencia de un profesional, poder obtener datos adelantados a cualquier análisis y saber previamente las posibles ubicaciones de los internos en vías de disminuir errores en la ubicación.

El siguiente diagrama muestra el proceso más importante del negocio: realizar clasificación, donde se muestra el proceso de desarrollo de los mismos.

3.4.1 Modelación de proceso del sistema

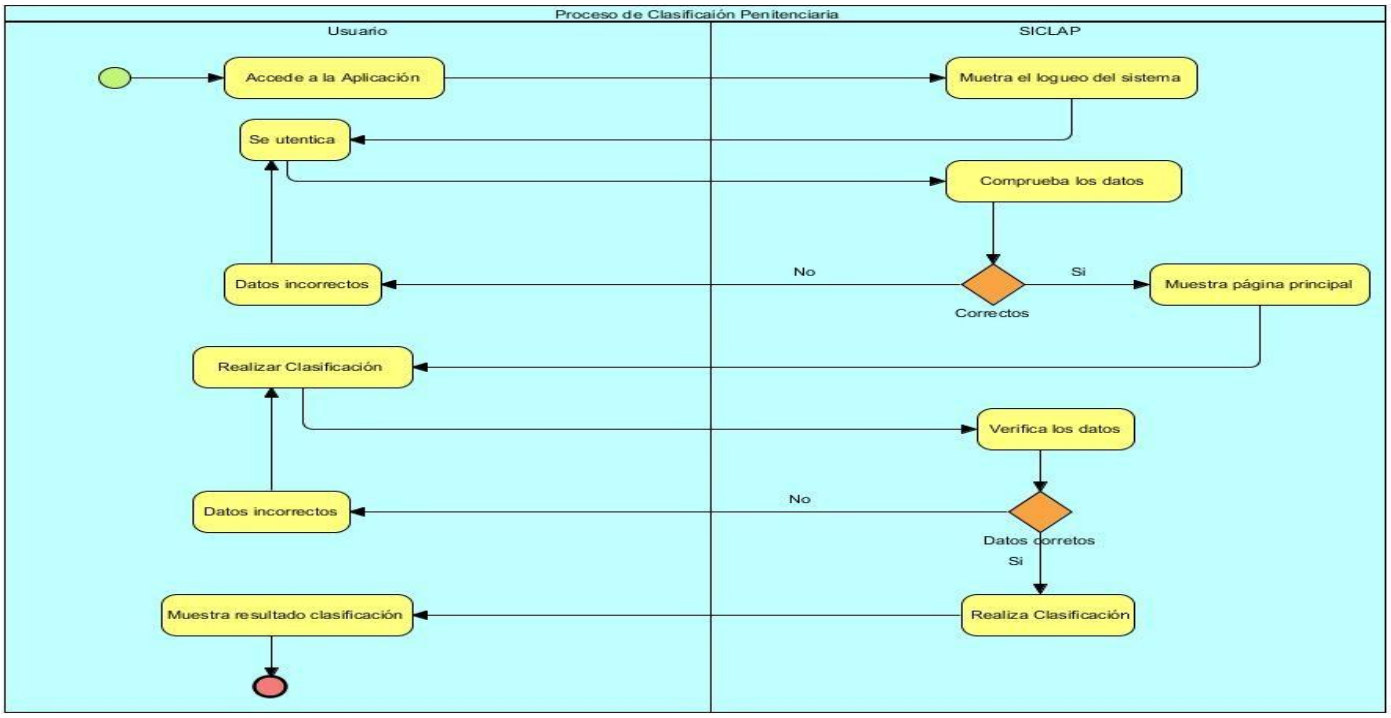


Figura 3. 1 Proceso de Clasificación Penitenciaria

3.5 Funcionalidades del sistema de Clasificación Penitenciaria

El sistema debe ser capaz de permitir:

Autenticar usuario

Gestionar Usuario

- Crear Usuario

- Modificar Usuario
- Eliminar Usuario
- Mostrar Usuario

Gestionar Régimen de Severidad

- Crear Régimen de Severidad
- Modificar Régimen de Severidad
- Eliminar Régimen de Severidad
- Mostrar Régimen de Severidad

Gestionar Caso

- Crear Caso
- Modificar Caso
- Eliminar Caso
- Mostrar Caso

Gestionar Rol

- Crear Rol
- Modificar Rol
- Eliminar Rol

Realizar Clasificación

Desconectar usuario

3.6 Requisitos no funcionales del sistema

Los requisitos no funcionales son fundamentales en el éxito del producto. Normalmente están vinculados a requerimientos funcionales. Se han definido los siguientes requisitos no funcionales.

Requisitos de Apariencia o interfaz externa:

- El sistema brindará una interfaz amigable para sus usuarios, con predominio de color azul en diversas tonalidades. Las letras serán legibles con formato Verdana, Arial, Helvetica, Sans-Serif, de 15 píxeles azules para los links y 13 píxeles negras para los textos.
- Los errores que sean visibles al usuario, indicarán las posibles causas y sus soluciones. Los mismos se mostraran el parte superior de la página de color rojo.

Requisitos Software.

- La PC donde se ejecute la aplicación debe tener instalado Firefox 3.X o superior, Chrome 5.X o superior o Safari, debido a que los estándares de CSS3 no son reconocidos por muchas de las versiones de Internet Explorer (6.X, 7.X). Debe tener alojado el SGBD PostgreSQL 9.1 o contar con dicho servidor externo.
- En caso de ser externo el servidor de base de datos, debe tener instalado PostgreSQL 9.1.
- La Computadora que sirva como servidor de la aplicación, debe tener instalado el Apache Tomcat como servidor Web, preferentemente una versión superior a 6.X y la Máquina Virtual de Java en su versión 7.3.

Requisitos de Hardware.

- Es necesario contar con una computadora con un Microprocesador con velocidad superior a los 1.8 GHz, y con una memoria RAM superior a 1 Giga Bytes, si en ella no se encuentra el servidor de Base de Datos.
- Si el servidor de base de datos es externo, debe estar alojado en una computadora con más de 1 Giga Bytes de RAM, 500 Giga Bytes HDD y tener la red disponible con una tarjeta de 10/100/1000 Mb/s.

Requisitos de diseño.

- El sistema implementado será una aplicación web.
- El sistema se implementará usando la plataforma JAVA.
- El sistema estará basado en un estilo arquitectónico en capas. Las capas estarán distribuidas de la siguiente manera:
    - ✓ Capa web: Encapsula las vistas y la lógica de presentación. En la lógica de presentación se maneja todo el flujo web utilizando la implementación del patrón Modelo-Vista-Controlador que brinda el framework Grails. Las vistas son los recursos que junto al modelo generado por los controladores le permiten al cliente visualizar la información.
    - ✓ Capa de dominio: Contiene las entidades que persistirán en la base de datos, además encapsula toda la lógica relacionada con el dominio que abarca el negocio. Esta brinda las funcionalidades mediante fachadas de servicio que son utilizadas por

los controladores en la capa web y se implementan procesos de negocio que perduran en el tiempo.

- ✓ Capa de datos: Encapsula la lógica de acceso a datos abstrayendo a las capas superiores del mecanismo de acceso a la base de datos. Dicha abstracción se realiza a través del framework de persistencia GORM, que implementa Grails, éste agrega métodos de forma dinámica a las entidades del dominio para que estas sean capaces de interactuar con el gestor de base de datos por sí mismas.

Requisitos de Seguridad.

- Seguridad y control a nivel de usuarios y contraseñas, garantizando el acceso de los mismos sólo a los niveles establecidos de acuerdo a la función que realicen. Las contraseñas sólo podrán ser cambiadas por el propio usuario o por el administrador informático del sistema.

3.7 Fase de exploración

La metodología de desarrollo XP establece como primera fase la Fase de Exploración, permitiendo al equipo de desarrollo una familiarización con las herramientas y tecnologías. Además se define el alcance real del proyecto. En esta fase se llevan a cabo las Historias de Usuario (HU) definiendo las verdaderas necesidades del cliente.

3.7.1 Historias de usuarios

Las historias de usuarios son la técnica utilizada en XP para especificar los requisitos del software, lo que equivale a los casos de uso en el proceso unificado. Las mismas son escritas por los clientes como las tareas que el sistema debe hacer y su construcción depende principalmente de la habilidad que tenga el cliente para definirlas. Son escritas en lenguaje natural, no excediendo su tamaño de unas pocas líneas de texto. Las historias de usuario guían la construcción de las pruebas de aceptación, elemento clave en XP (deben generarse una o más pruebas para verificar que la historia ha sido correctamente implementada) y son utilizadas para estimar tiempos de desarrollo. En este sentido, sólo proveen detalles suficientes para hacer una estimación razonable del tiempo que llevará implementarlas. En el momento de implementar una historia de usuario, se debe detallar a través de la comunicación con el cliente. Estas son la base para las pruebas funcionales.

En esta plantilla los campos de puntos estimados y puntos reales se llenan, luego del desarrollo de la actividad de estimación de esfuerzo, donde se decide qué tiempo se le dedicará a cada historia de usuario definida con anterioridad. Las historias de usuario proporcionan ventajas ya que:

- Están escritas en lenguaje del cliente, por lo que es muy fácil su compresión.
- Especifican cada uno de los requisitos del sistema, sin necesidad de documentaciones extensas.
- Reflejan todas las características del sistema.
- Si se definen correctamente, guían el proceso de implementación.

3.7.1 Clasificación de las historias de usuario

La prioridad en el negocio:

Alta: Se le otorga a las HU que resultan funcionalidades fundamentales en el desarrollo del sistema, a las que el cliente define como principales para el control integral del sistema.

Media: Se le otorga a las HU que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación sobre el sistema que se esté desarrollando.

Baja: Se le otorga a las HU que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura, y no tienen nada que ver con el sistema en desarrollo.

El riesgo en su desarrollo:

Alta: Cuando en la implementación de las HU se consideran la posible existencia de errores que lleven la inoperatividad del código.

Media: Cuando pueden aparecer errores en la implementación de la HU que puedan retrasar la entrega de la versión.

Baja: Cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto.

Las HU serán representadas mediante tablas divididas por las siguientes secciones:

- **Número:** número de la historia de usuario incremental en el tiempo.
- **Nombre de HU:** el nombre de la historia de usuario sería para identificarlas mejor entre los desarrolladores y el cliente.
- **Modificación de HU:** si la historia de usuario sufrió alguna modificación anterior.

- **Usuario:** involucrados en el desarrollo de la HU.
- **Iteración Asignada:** número de la iteración.
- **Prioridad en negocio**: Alta, Media o Baja.
- **Riesgo en Desarrollo*:*** Alta, Media o Baja.
- **Puntos estimados**: tiempo estimado que se demorará el desarrollo de la HU.
- **Puntos Reales**: tiempo que se demoró en realidad el desarrollo de la HU.
- **Descripción**: breve descripción de la HU.
- **Observaciones**: señalamiento o advertencia del sistema.
- **Prototipo de interfaz**: Prototipo de interfaz si aplica.

A continuación se muestran las Historias de usuarios más importantes:

Tabla 3. 1 Historia de Usuario Nro. 1: Autenticar usuario

Historia de Usuario	
Número: 1	**Nombre Historia de Usuario:** Autenticar usuario
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Niusbel Hernández Lazo	**Iteración Asignada:** 3
Prioridad en Negocio: Alta	**Puntos Estimados:** 1/2
Riesgo en Desarrollo: Medio	**Puntos Reales:** 1/2
Descripción: El usuario introduce su nombre de usuario y contraseña para acceder a las funcionalidades de la aplicación. De acuerdo a los permisos que tenga podrá interactuar con el sistema.	
Observaciones: El sistema tiene los roles administrador, experto y usuario.	
Prototipo de interfaz:	

Tabla 3. 2 Historia de Usuario Nro. 5: Crear Régimen de severidad

Historia de Usuario	
Número: 5	**Nombre Historia de Usuario:** Crear Régimen de severidad
Modificación de Historia de Usuario Número: Ninguna	

Usuario: Niusbel Hernández Lazo	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:** 1/4
Riesgo en Desarrollo: Bajo	**Puntos Reales:** 1/4
Descripción: El experto del sistema introduce los datos del régimen de severidad: - Nombre - Descripción	
Observaciones: El experto del sistema debe estar previamente autenticado.	
Prototipo de interfaz : 	

Tabla 3. 3 Historia de Usuario Nro. 8: Crear Caso

Historia de Usuario	
Número: 8	**Nombre Historia de Usuario:** Crear Caso
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Niusbel Hernández Lazo	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:** 2/4
Riesgo en Desarrollo: Baja	**Puntos Reales:** 2/4
Descripción: El experto del sistema introduce los datos del caso: - Nombre	

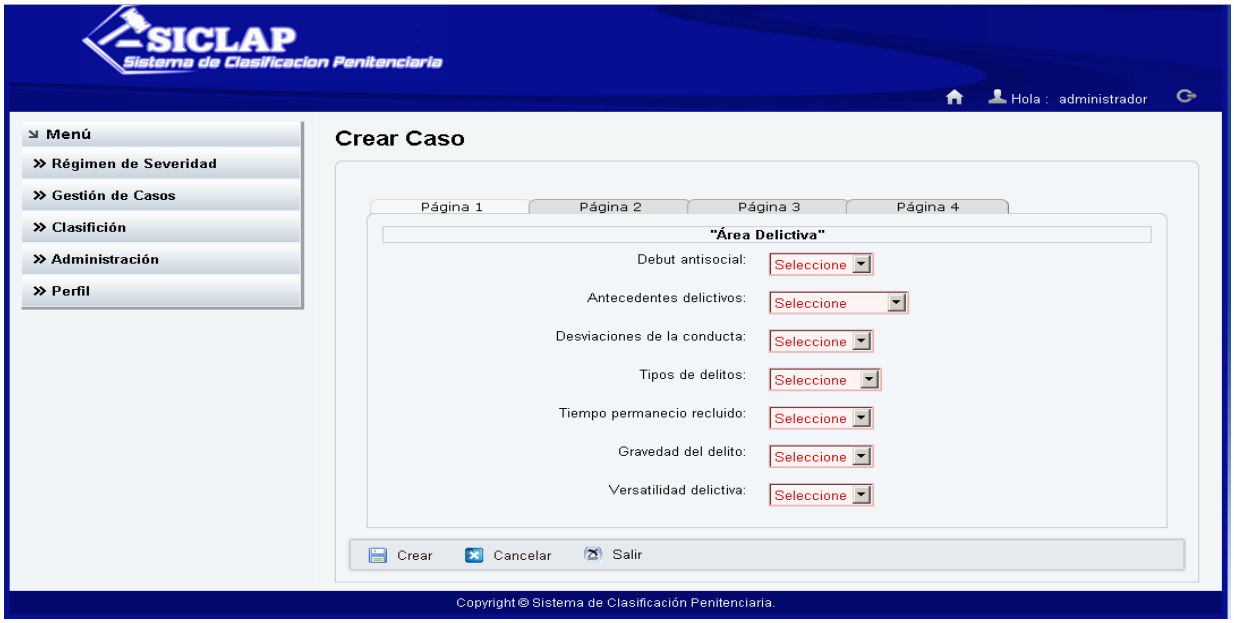
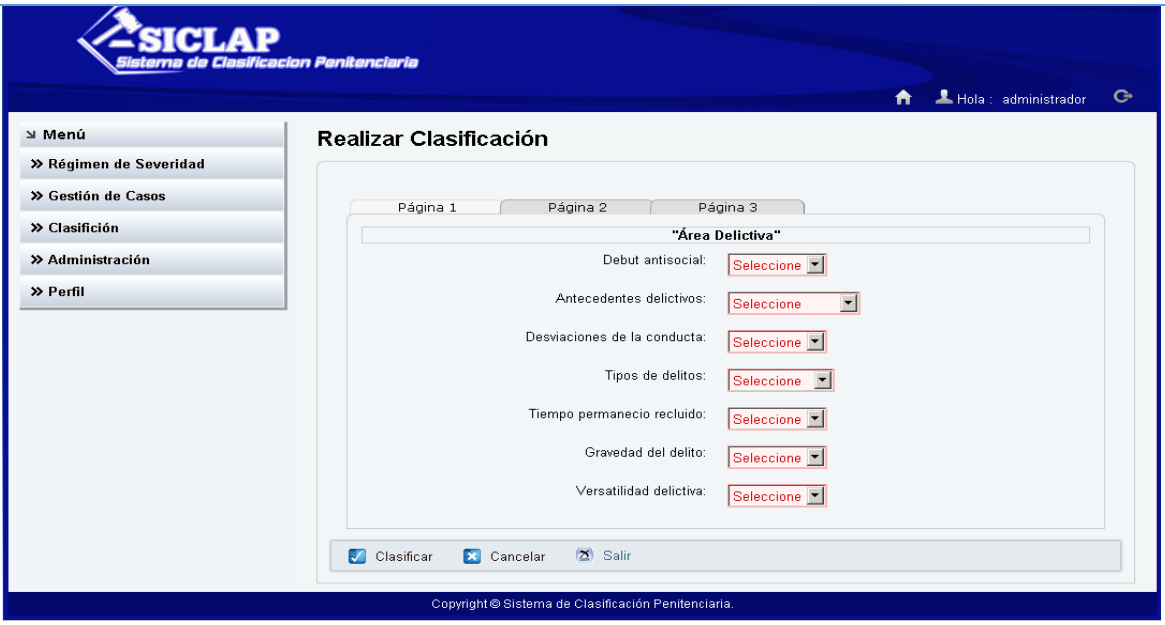
- Lista de rasgos
Observaciones: El experto del sistema debe estar previamente autenticado.
Prototipo de interfaz : 

Tabla 3. 4 Historia de Usuario Nro. 11: Realizar Clasificación

Historia de Usuario	
Número: 11	**Nombre Historia de Usuario:** Realizar Clasificación
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Niusbel Hernández Lazo	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados: 3**
Riesgo en Desarrollo: Alta	**Puntos Reales:** 3
Descripción: Cualquiera de los usuarios del sistema puede introducir los datos para realizar la clasificación: - Lista de rasgos	
Observaciones: Los usuarios del sistema, deben estar previamente autenticado.	
Prototipo de interfaz :	



Las demás HU definidas se encuentran en el (Anexo 7)

3.8 Fase de planificación

En la fase de planificación se realiza la estimación del esfuerzo que costará la implementación de cada historia de usuario, como en la metodología ágil XP las métricas son libres, puede utilizarse cualquier criterio para medir el desempeño del proyecto en cuestión. Una de las métricas más utilizadas en este tipo de metodología es la medida de puntos; un punto en esta métrica es considerado como una semana de trabajo, donde los miembros de los equipos de desarrollo trabajan sin interrupciones.

3.8.1 Estimación de esfuerzo por historia de usuario

En la siguiente tabla se muestra la estimación del esfuerzo por cada HU propuesta para el desarrollo de la aplicación:

Tabla 3. 5 Estimación de esfuerzo por HU

Historia de Usuario	Puntos de estimación
Gestionar Usuario - Crear Usuario - Modificar Usuario - Eliminar Usuario	1

- Mostrar Usuario	
Gestionar Régimen de Severidad - Crear Régimen de Severidad - Modificar Régimen de Severidad - Eliminar Régimen de Severidad - Mostar Régimen de Severidad	1
Gestionar Caso - Crear Caso - Modificar Caso - Eliminar Caso - Mostrar Caso	2
Gestionar Rol - Crear Rol - Modificar Rol - Eliminar Rol	1
Realizar Clasificación	3
Autenticar Usuario	1
Desconectar Usuario	1

Para el desarrollo de la aplicación, se planificó un total de 10 semanas, distribuyendo la mayor carga en las funcionalidades de mayor complejidad y las más críticas en el sistema.

3.8.2 Plan de iteraciones

Después de ser identificadas HU y estimar el esfuerzo dedicado a la realización de cada una de ellas, se procede a la planificación de la fase de implementación estableciendo una división de tres iteraciones, estas iteraciones están definidas por la importancia y el peso en el negocio, de cada una de las HU.

- **Iteración 1**

En la iteración 1 se implementarán las HU: 2, 5, 8, 11 y la 14 ya que intervienen directamente con el funcionamiento de la aplicación y su prioridad en el negocio es alta.

- **Iteración 2**

En la iteración 2 se implementarán las HU: 3, 4, 6, 7, 9, 10, 12, 13, 15, 16 y 17 (ver Anexo 7)ya que intervienen directamente con el funcionamiento de la aplicación y su prioridad en el negocio es Media o Baja.

- **Iteración 3**

En la iteración 3 se implementarán las HU: 1 y 18. Al complementarse esta iteración dan al traste una versión 1.0 del producto final, de este modo el sistema se pondrá en funcionamiento para evaluar su desempeño.

Tabla 3. 6 Plan de duración de las iteraciones

Iteración	Orden de las HI a implementar	Duración
1	Crear Usuario Crear Régimen de Severidad Crear Caso Crear Rol Realizar Clasificación	6 semanas
2	Modificar Caso Eliminar Caso Modificar Régimen de Severidad Eliminar Régimen de Severidad Modificar Usuario Eliminar Usuario Modificar Rol Eliminar Rol Mostrar Usuario Mostrar Caso Mostrar Régimen de Severidad	3 semanas
3	Autenticar Usuario Desconectar Usuario	1 semana

3.8.3 Plan de entregas

Se presenta el plan de entrega estimado para la fase de implementación, detallando la fecha de fin de cada iteración, los productos obtenidos, así como el sistema sobre el cual se está implementando.

Tabla 3. 7 Plan de entrega del SICLAP: Sistema de Clasificación Penitenciaria

Sistema	Fin de la Iteración 1 8 de Abril de 2013	Fin de la Iteración 2 15 de Mayo de 2013	Fin de la Iteración 3 22 de Mayo de 2013
SICLAP	SICLAP v0.1	SICLAP v0.2	SICLAP v1.0

3.9 Patrones utilizados

Los patrones son una estructura de implementación que logra una finalidad determinada a un lenguaje de programación de alto nivel (11). En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Existen diferentes tipos de patrones: patrones arquitectónicos y patrones de diseño. Un patrón arquitectónico especifica un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes. Un patrón de diseño está relacionado con los aspectos del diseño de los subsistemas. Es una solución estándar para un problema común de programación y una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios.

3.9.1 Patrones arquitectónicos

Son los que definen la estructura de un sistema de software, los cuales a su vez se componen de subsistemas con sus responsabilidades, también tienen una serie de directivas para organizar los componentes del mismo sistema, con el objetivo de facilitar la tarea del diseño de tal sistema (11). **Modelo Vista Controlador (MVC):** Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón el cual define un estilo arquitectónico de llamada y retorno, se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. El modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio, y el controlador es el responsable de recibir los eventos de entrada desde la vista. . En la **figura 3.2** se muestra cómo interactúa cada una de estas capas.

Figura 3. 2 Modelo-Vista-Controlador.

Grails se vale de las bondades del patrón **MVC** brindando las vistas en forma o extensión gsp. Este tipo de páginas conforman la capa de interacción con el cliente. Los controladores, encargados de mantener el flujo de comunicación entre las vistas y el modelo, están definidos por los groovy controllers o controladores groovy, los cuales especifican cómo acceder a los datos y el envío de las respuestas a las vistas. El modelo, definido por clases de dominio de groovy que se mapean en la base de datos utilizada, complementan la tercera capa y permiten el manejo y almacenamiento de los datos. De esta forma, queda definida la utilización del patrón **MVC** en la interacción y desarrollo con el framework. . Por su parte, Grails define los Servicios como otro componente del negocio, el cual separa la lógica de negocio de los controladores y la incluye en ella.

3.9.2 Patrones de diseño.

Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores (11). Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Inversión del control (IoC): La inversión de control es un patrón utilizado en Grails, según el cual las dependencias de un componente no deben gestionarse desde el propio componente para que éste sólo contenga la lógica necesaria para hacer su trabajo.

Al utilizar servicios en Grails, solo se escribe una variable con su nombre, sin tener en cuenta la instanciación del servicio y su configuración. Grails configura Spring para que gestione su ciclo de vida y sus dependencias. El objetivo de esta técnica es mantener los componentes tan sencillos como sea posible a la vista del programador, incluyendo únicamente código que tenga relación con la lógica de negocio. Así, la aplicación será más fácil de comprender y mantener.

3.9.2.1 Patrones GRASP

Son patrones generales de software para asignación de responsabilidades, es el acrónimo de Patrones de Software de Asignación de Responsabilidades Generales (General Responsibility Assignment Software Patterns). Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software (11).

Patrón Controlador: Sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases

según el método llamado. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto para aumentar la reutilización de código y a la vez tener un mayor control.

Los controladores en Grails, son los encargados de posibilitar una capa intermedia entre las vistas y el modelo. Además controla el flujo de información referente a la lógica de negocio.

Alta cohesión: Se refiere a que la información que almacena una clase debe de ser coherente y debe estar (en la medida de lo posible) relacionada con la ella misma.

Bajo acoplamiento: Se refiera a tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

Experto: La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. De esta manera se garantiza una alta cohesión y un bajo acoplamiento.

3.9.2.2 Patrones GOF (Gang of Four)

Singleton: Garantiza que una clase sólo tenga una instancia, y proporciona un punto de acceso global a ella. Se utiliza en la instanciación de los servicios en los controladores. Por defecto en Grails todos los servicios son Singleton. En la **figura 3.3** se muestra un ejemplo del código donde se evidencia el uso de este patrón.

```
class CasoController {

    static allowedMethods = [save: "POST", update: "POST", delete: "POST"]

    def casoService    // instancia de el servicio  Caso Servicio
    def riesgoService    // instancia de el servicio  Riesgo Servicio
    def formulaClasificarService     // instancia de el servicio  Formula Servicio

    //------------------ Salvar Caso ----------------------------------------------------
    @Secured(['ROLE_EXPERTO','ROLE_ADMINISTRADOR'])
    def save()
    {
                def casoexiste = new Caso(params)
                def b = casoService.Search(params)
                def a = casoService.CasoIgual(b,params)
                 if(a == "No")
                 {
                     def casoInstance = casoService.Salvar(params)
                     if (casoInstance.hasErrors())
                     {
                         render(view: "create", model: [casoInstance: casoInstance])
                     }
                     else
                     {
                         flash.message = message(code: 'default.created.message', args: [message(code: 'caso.label', defa
                         redirect(action: "create")

                     }
                 }
```

Figura 3. 3 Ejemplo del patrón Singleton

Decorador / Decorator: Añade responsabilidades adicionales a un objeto dinámicamente, proporcionando una alternativa flexible a la especialización mediante herencia, cuando se trata de añadir funcionalidades. El uso de este patrón se centra en la utilización de SiteMesh integrado a la arquitectura

del framework. El mismo permite a partir de una plantilla inicial de la página incorporar modificaciones a cada vista en particular.

3.10 Tarjetas Clase Responsabilidad Colaborador (CRC)

Como parte de la fase del diseño de la metodología XP, el uso de las tarjetas CRC permiten al programador centrarse y apreciar el desarrollo orientado a objetos. Además de que constituyen un modelo simple de organizar las clases más relevantes para las funcionalidades del sistema. Este modelado CRC utiliza tarjetas, con el objetivo de desarrollar una representación organizada de las clases.

Un modelo CRC es en realidad una colección de tarjetas índices estándar que representan clases. Las tarjetas se dividen en tres secciones. A lo largo del borde superior de la tarjeta se escribe el nombre de la clase. En el cuerpo de la tarjeta se listan las funcionalidades (responsabilidades) de la clase a la izquierda y aquellas clases que se implican en cada funcionalidad (colaboración), a la derecha (9).

A continuación se describen las tarjetas CRC correspondientes a las clases más relevantes para las funcionalidades del sistema.

Tabla 3. 8 Tarjeta CRC UsuarioController

Clase: UsuarioController	
Descripción: Clase encargada de gestionar la información del usuario.	
Responsabilidad	**Colaborador**
Gestionar Usuario	UsuarioService Usuario

Tabla 3. 9 Tarjeta CRC LoguinController

Clase: LoguinController	
Descripción: Clase encargada de controlar el acceso de la aplicación.	
Responsabilidad	**Colaborador**
Autenticación al sistema	Usuario Rol securytiService

Tabla 3. 10 Tarjeta CRC CasoController

Clase: CasoController
Descripción: Clase encargada de gestionar la información de los casos y realizar la clasificación.

Responsabilidad	Colaborador
Gestionar Caso	CasoService Caso FormulaClasificacionService RegimenSeveridad

Tabla 3. 11 Tarjeta CRC RegimenSeveridadController

Clase: RegimenSeveridadController	
Descripción: Clase encargada de gestionar la información del régimen de severidad.	
Responsabilidad	**Colaborador**
Gestionar RegimenSeveridad	RegimenSeveridadService RegimenSeveridad

Tabla 3. 12 Tarjeta CRC LogoutController

Clase: LoguinController	
Descripción: Clase encargada de controlar el cierre de sesión de la aplicación.	
Responsabilidad	**Colaborador**
Desconexión del sistema	Usuario Rol securityService

3.11 Modelo físico la base datos

En la **figura 3.4** se muestra el modelo físico de la base de datos perteneciente al sistema de Clasificación Penitenciaria.

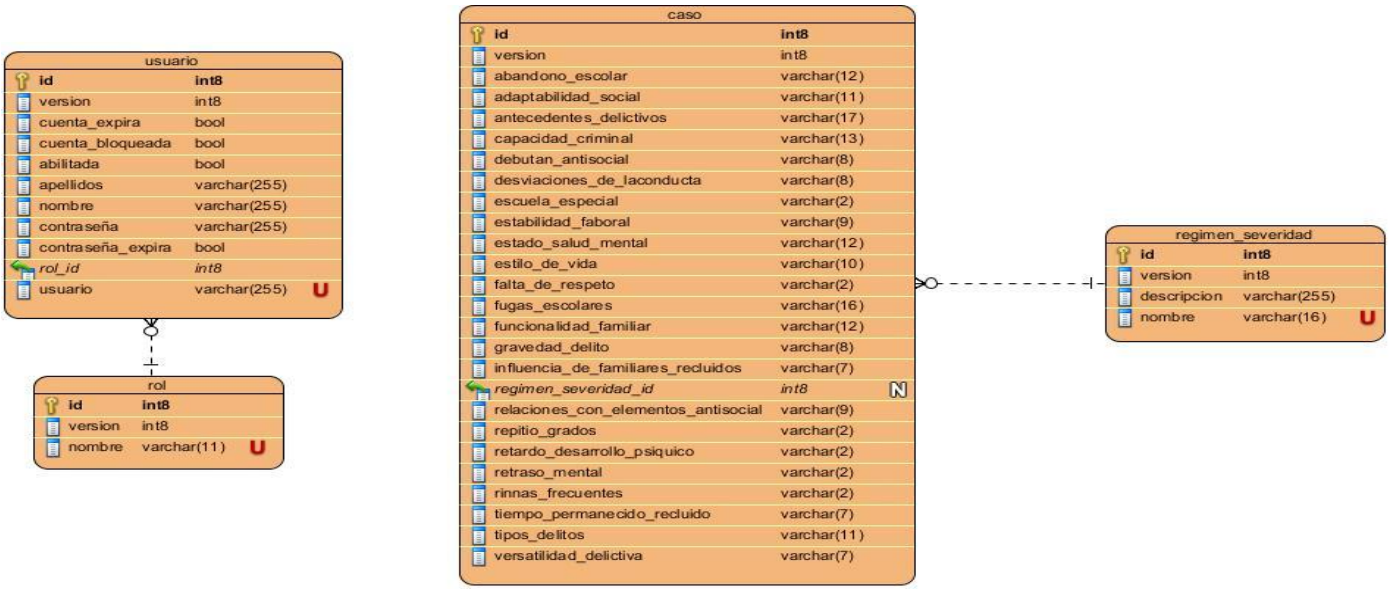


Figura 3. 4 Modelo físico de la base de datos

Descripción de la base de datos

Inicialmente la base de datos del Sistema de Clasificación Penitenciaria cuenta con 5 tablas, de las mismas resultan críticas: caso y régimen_severidad, las cuales guardan los datos que tienen que ver con la clasificación, estas se van llenando a medida que los expertos del sistema introduzcan los casos anteriores. La tabla régimen_severidad es la encargada de almacenar el tipo de clasificación que se utiliza en el proceso de clasificación penitenciaria. La tabla caso almacena todas las características utilizadas para dar la clasificación. La tabla usuario, rol y usuario_rol son las que almacenan los datos que corresponden con los usuarios, para la seguridad del sistema.

3.12 Conclusiones parciales

En el presente capítulo se describió la propuesta del sistema a desarrollar. Se identificaron las funcionalidades y requisitos que el sistema debe cumplir, así como la descripción de las HU divididas por iteraciones y la planificación del esfuerzo dedicado a la realización de cada una de ellas en el orden en que se les dará cumplimiento según las necesidades del cliente. Además se realizó un mayor acercamiento al diseño del sistema; mediante el estudio y asignación de patrones tanto de arquitectura como de diseño, se logró una mayor organización en el trabajo para la implementación del sistema de clasificación penitenciaria.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

Introducción

La concepción de la propuesta del sistema y los artefactos generados y presentados en el capítulo anterior, ayudan al programador a entender las funcionalidades que busca el cliente y, por tanto, ser capaz de llevarlas a código interpretable por la computadora, o lo que es lo mismo: implementar el sistema.

En el presente capítulo se exponen las tareas de la ingeniería definidas. Además, se realizarán las pruebas al software, las cuales se derivan de las Historias de Usuario y tareas de la ingeniería que se han implementado como parte de la realización del software.

4.1 Tareas de ingeniería

La metodología XP plantea que la implementación de un software se hace iterativamente, que al culminar cada iteración se obtiene un producto funcional, que debe ser probado y mostrado al cliente. Durante el transcurso de las iteraciones, se realiza la implementación de las HU definidas por el cliente y descritas por el equipo de desarrollo en la fase de exploración. Como parte de este plan, se descomponen estas HU en tareas de la ingeniería las cuales son asignadas a los programadores para ser implementadas durante la iteración correspondiente.

Las tareas de la ingeniería serán representadas mediante tablas divididas por las siguientes secciones:

- **Número Tarea:** los números deben ser consecutivos.
- **Número Historia de Usuario:** número de la historia de usuario a la que pertenece la tarea.
- **Nombre Tarea:** nombre que identifica a la tarea.
- **Tipo de Tarea:** las tareas pueden ser de: Desarrollo, Corrección, Mejora, Otra (Especificar)
- **Puntos Estimados:** tiempo estimado en semanas que se le asignará a su desarrollo.
- **Fecha Inicio:** fecha en que inicia el desarrollo de la tarea.
- **Fecha Fin**: fecha en que finaliza el desarrollo de la tarea.
- **Programador Responsable:** nombre y apellidos del programador.
- **Descripción:** breve descripción de la tarea.

A continuación se muestran las tareas en correspondencia con aquellas Historias de Usuario de prioridad más alta:

Tabla 4. 1 Tarea de ingeniería Nro. 1: Crear usuario

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 2
Nombre Tarea: Crear usuario	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1/4
Fecha Inicio: 10/1/2013	**Fecha Fin:** 13/1/2013
Programador Responsable: Niusbel Hernández Lazo	
Descripción: El administrador del sistema registra un nuevo usuario en el sistema, pudiendo entrar los datos: - Usuario - Nombre - 1er Apellido - Contraseña - Rol - Habilitada	

Tabla 4. 2 Tarea de ingeniería Nro. 2: Autenticar usuario

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario: 1**
Nombre Tarea: Autenticar usuario	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1/2
Fecha Inicio: 16/5 /2013	**Fecha Fin:** 20/5 /2013
Programador Responsable: Niusbel Hernández Lazo	
Descripción: El usuario anteriormente registrado en el sistema por el administrador, procede a entrar al sistema mediante su usuario y contraseña asignada, ambos como campos obligatorios.	

Tabla 4. 3 Tarea de ingeniería Nro. 3: Crear Régimen de Severidad

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario: 5**
Nombre Tarea: Crear Régimen de Severidad	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1/4
Fecha Inicio: 2/2 /2013	**Fecha Fin:** 4/2 /2013
Programador Responsable: Niusbel Hernández Lazo	
Descripción: El experto del sistema registra un nuevo régimen de severidad en el sistema, pudiendo entrar los	

datos: - Nombre - Descripción

Tabla 4. 4 Tarea de ingeniería Nro. 4: Crear Caso

Tarea de Ingeniería	
Número Tarea: 4	**Número Historia de Usuario: 8**
Nombre Tarea: Crear Caso	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2/4
Fecha Inicio: 9/2 /2013	**Fecha Fin:** 12/2 /2013
Programador Responsable: Niusbel Hernández Lazo	
Descripción: El experto del sistema registra un nuevo caso en el sistema, pudiendo entrar los datos: - Régimen de Severidad - Lista de rasgo	

Tabla 4. 5 Tarea de ingeniería Nro. 5: Realizar Clasificación

Tarea de Ingeniería	
Número Tarea: 5	**Número Historia de Usuario:** 11
Nombre Tarea: Realizar Clasificación	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 16/2 /2013	**Fecha Fin:** 2/3 /2013
Programador Responsable: Niusbel Hernández Lazo	
Descripción: Los usuarios previamente autenticados pinchan la opción clasifica del sub-menú Clasificación y entran los datos requerido: lista rasgo , posteriormente pincha el botón clasificar	

Las demás tareas de ingeniería se encuentran en el (Anexo 8)

4.2 Pruebas

Las pruebas que se le realizan a un software, son uno de los instrumentos capaces de medir el estado de calidad de un producto. El proceso de pruebas se dirige fundamentalmente a componentes del software o al sistema de software en general, con el objetivo de medir si el software cumple las funcionalidades establecidas por el cliente. Las pruebas del software verifican y revelan la calidad de un producto, así como que son utilizadas para identificar posibles errores en la implementación y permiten evitar efectos colaterales no deseados a la hora de realizar modificaciones.

Como elemento propio de la metodología utilizada, durante el desarrollo de software se lleva a cabo la Fase de Prueba. XP establece probar tanto como sea posible, permitiendo así, un aumento de la calidad del sistema desarrollado. XP divide las pruebas del sistema en dos grupos: pruebas unitarias y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida, diseñadas por el cliente final. Este tipo de pruebas, constituyen uno de los mayores avances en la programación desde la aparición de la orientación a objetos y aunque son recomendadas, su uso o no, queda a merced de los desarrolladores.

4.2.1 Pruebas unitarias

Las pruebas unitarias son las encargadas de verificar el código y son diseñadas por los programadores para garantizar que las funcionalidades exigidas por el cliente estén siendo implementadas correctamente. No generan artefactos y se realizan en el transcurso de la implementación de un software. Estas pruebas fueron realizadas haciendo uso del sistema de Testing o Prueba que brinda el propio Framework utilizado.
Los test unitarios son aquellos en los que se verifica que un método se comporta como debería, sin tener en cuenta su entorno. Esto significa que cuando se ejecutan las pruebas unitarias de un método Grails no inyectará ninguno de los métodos dinámicos con los cuenta la aplicación cuando se está ejecutando.
Durante el proceso de desarrollo, podemos probar el estado de la aplicación mediante el comando:

grails test-app

Ver (Anexo 9)

A partir de esto, se ejecutarán todas las pruebas unitarias y de integración del proyecto, y generará un informe al que se puede recurrir en caso de fallos. El informe se genera en texto y HTML. Para crear un test unitario de un Servicio se ejecuta el comando:

grails create-unit-test<nombre de la prueba>

Creándose una nueva batería de pruebas en la carpeta test/unit del proyecto. La clase generada hereda de GrailsUnitTestCase, lo que hace que se tenga a disposición una serie de métodos para que facilite la tarea de probar los servicios definidos. Este test (prueba) se ejecutará sin levantar el contexto Grails, de forma que las entidades no disponen de los métodos dinámicos inyectados por GORM, ni se realiza la inyección de dependencias. Para esto, GRAILS aprovecha el soporte nativo en.

La clase GrailsUnitTestCase pone a disposición métodos para que los objetos se comporten en pruebas como lo harían en ejecución. Gracias a estos métodos, se pueden probar los artefactos y usar los métodos dinámicos de Grails sin necesidad de levantar el contexto de ejecución.

4.2.2 Pruebas de aceptación

Las pruebas de aceptación o funcionales son la mejor forma de probar la aplicación de extremo a extremo: desde la petición realizada por un navegador hasta la respuesta enviada por el servidor. Las pruebas funcionales prueban todas las capas de la aplicación: el sistema de enrutamiento, el modelo, las acciones y las plantillas. En realidad, son muy similares a lo que se hace manualmente cada vez que se añade o modifica una acción y se prueban dichos cambios en el navegador para comprobar que todo funciona bien al pulsar sobre los enlaces y botones y que todos los elementos se muestran correctamente en la página. En otras palabras, lo que se hace es probar un escenario correspondiente a la historia de usuario que se acaba de implementar en la aplicación.

Las pruebas de aceptación correspondiente a cada una de las funcionalidades del Sistema de Clasificación Penitenciaria, serán representadas mediante tablas que incluyen los siguientes campos:

Código caso de prueba: que contiene el identificador de caso de prueba; en el caso de las presentes, se utiliza el identificador “P” y un número “1.0” consecutivo empezando de uno.

Nombre historia de usuario: que contiene el nombre de la HU correspondiente a este caso de prueba.

Nombre de la persona que realiza la prueba: contiene el nombre del responsable que realiza la prueba.

Descripción de la prueba: que contiene una breve descripción de la prueba realizada.

Condiciones de ejecución: se incluyen las condiciones necesarias para que se pueda realizar la prueba.

Entrada/Pasos de ejecución: contiene una serie de pasos enumerados para lograr realizar la prueba de esta HU.

Resultado esperado: contiene la descripción de lo que se espera luego de realizar la prueba (cumplimiento de las restricciones del producto).

Evaluación de la prueba: muestra si la prueba fue satisfactoria o insatisfactoria.

Tabla 4. 6 Caso de aceptación #2: HU Crear Caso

Caso de Prueba de Aceptación

Código caso de prueba: P2.3	**Nombre historia de usuario:** Crear Caso
Nombre de la persona que realiza la prueba: Ing. Dasiel Cordero Morales	
Descripción de la prueba: Se insertan los datos de un caso, inicialmente se insertarán incorrectamente para verificar las validaciones del sistema, luego de forma correcta para comprobar que los datos sean almacenados.	
Condiciones de ejecución: El usuario debe tener los permisos suficientes para realizar esta operación.	
Entrada/Pasos de ejecución: 1. Autenticarse en la aplicación. 2. Seleccionar del sub-menú caso; crear caso. 3. Entrar datos incorrectos. 4. Entrar datos correctos. 5. Verificar que el nuevo caso este registrado	
Resultado esperado: El sistema debe alertar al usuario cuando se inserten datos en blanco en los campos obligatorios (Nombre, Lista de Rasgos); además debe alertar cuando ya exista un caso con el mismo nombre y los mismos rasgos, almacenado en la base de datos. Cuando se inserten los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrarlos en el listado.	
Evaluación de la prueba: Satisfactoria	

Tabla 4. 7 Caso de aceptación #7: HU Realizar Clasificación

Caso de Prueba de Aceptación	
Código caso de prueba: P7.4	**Nombre historia de usuario:** Realizar Clasificación
Nombre de la persona que realiza la prueba: Ing. Dasiel Cordero Morales	
Descripción de la prueba: Se insertan los datos de los rasgos, inicialmente se insertarán incorrectamente para verificar las validaciones del sistema, luego de forma correcta para comprobar que los datos sean almacenados.	
Condiciones de ejecución: El usuario debe tener los permisos suficientes para realizar esta operación.	
Entrada/Pasos de ejecución: 1. Autenticarse en la aplicación. 2. Seleccionar del sub-menú clasificación: realizar clasificación. 3. Entrar datos incorrectos.	

4. Entrar datos correctos. 5. Verificar que muestre el resultado
Resultado esperado: El sistema debe alertar al usuario cuando no ha culminado de insertar todos los rasgos para la clasificación, además debe alertar si selecciono dos rasgos iguales. Cuando se llenan los datos correctamente, el sistema debe dar un resultado de clasificación.
Evaluación de la prueba: Satisfactoria

Las demás pruebas de aceptación realizadas se encuentran en el (Anexo 10)

Las pruebas unitarias se realizaron después de haber concluido cada iteración, obteniendo en la primera iteración dos errores de los cinco HU a probar, en la segunda iteración dos errores de las trece HU a probar y en la tercera iteración todos las HU probadas todos dieron resultados satisfactorios. Con los errores detectados se realizaron otras iteraciones hasta que fueron mitigados. Para realizar las pruebas de aceptación se realizó una encuesta a los expertos donde se solicitaba que llenaran la misma con valores reales para poder realizar las pruebas (Anexo 11). De la encuesta se obtuvieron 22 casos para probar de los mismos se tomaron 5 para probar y los demás se introdujeron en la base de casos para realizar las pruebas. Este paso se realizó en 3 iteraciones, tomando 5 casos diferentes cada vez que se realizaba una iteración y los demás casos se introducían en la base de casos, arrojando los resultados siguientes:

En la primera iteración de prueba de los 5 casos a probar 4 resultaron satisfactorios y 1 insatisfactorio, en la segunda iteración de los 5 a probar los 5 resultaron satisfactorios y en la tercera iteración se obtuvieron los mismos resultados que en la primera iteración, arrojando a la conclusión de que el clasificador es correcto. Los casos clasificados incorrectamente se deben a que la base de casos es pequeña por lo que se tendría que introducir más casos para que este devuelva los resultados más adecuados para la clasificación.

4.3 Conclusiones parciales

El presente capítulo estuvo enfocado en darle cumplimiento a los requisitos funcionales y no funcionales definidos en el capítulo anterior, lo cual se comprobó con la realización de las pruebas, conociendo la necesidad de las mismas en una aplicación para demostrar su correcto funcionamiento, su calidad, fiabilidad y justificación de tecnologías utilizadas, así como sus clasificaciones y usos, definiendo para ello una estrategia que integra pruebas de aceptación y unitarias, arrojando como resultado pruebas satisfactorias.

CONCLUSIONES

En el presente trabajo se describe la solución propuesta para desarrollar una herramienta capaz de ayudar a la clasificación de los internos a los distintos centros penitenciarios; con el nombre “Sistema de Clasificación Penitenciaria”. La investigación desarrollada y los resultados obtenidos permiten plantear las siguientes conclusiones:

1. La búsqueda y análisis de sistemas semejantes demostró, que las herramientas internacionales poseen características no requeridas por el cliente y son propietarias, mientras que a nivel nacional no se encontraron soluciones que automaticen el proceso de clasificación penitenciaria, evidenciándose la necesidad de crear una nueva solución de este tipo para los distintos Centros penitenciarios de Cuba.
2. Se logró una efectiva ingeniería de software y gestión del proyecto mediante la aplicación de la metodología ágil XP.
3. Se formalizó un modelo computacional donde se aplican técnicas de inteligencia artificial en la consideración de la Clasificación Penitenciaria como entidad principal, empleando el Razonamiento Basado en Casos.
4. Se realizó la implementación computacional del modelo propuesto utilizando Grails 2.0.1 como framework, Groovy 2.0 como lenguaje de programación y para el modelado se utilizó Visual Paradigm 5.0.
5. Se realizaron pruebas al sistema desarrollado de acuerdo con una estrategia definida para ello, según la metodología XP dando como resultado pruebas satisfactorias.

RECOMENDACIONES

Este trabajo da solución al problema planteado y satisface los objetivos trazados al inicio de la investigación, pero existen elementos que se le pueden incorporar en un futuro para una mayor eficiencia. Debido a esto, al concluir este trabajo y en aras de darle continuidad al mismo, con el objetivo de obtener una solución más completa se presentan un conjunto recomendaciones que se deberían tener en cuenta para una posterior versión:

1. Incluir en el sistema, específicamente en la programación inteligente el trabajo con valores de incertidumbre.

2. Investigar y tener en cuenta otras estrategias de aprendizaje.

3. Investigar y tener en cuenta otras funciones de comparación además de otra función de evaluación para cada rasgo predictor y objetivo respectivamente.

REFERENCIA BIBLIOGRAFICA

1. España. Ministerio de Interior. [En línea] [Citado el: 20 de Noviembre de 2012.] http://www.institucionpenitenciaria.es/web/portal/laVidaEnPrision/clasificacion/.

2. AUSTIN, y otros, y otros. *Objective Prison Classification: A Guide for Correctional Agencies. U.S. Department of Justice.* EEUU : National Institute of Corrections, 2004.

3. Gálvez Lio, Daniel. *Sistemas Basados en el Conocimiento.Universidad Central de las Villas (UCLV).* 1988.

4. GARCÍA, MARILUZ ROMERO y RODRÍGUEZ, JORGE ENRIQUE . *SISTEMAS BASADOS EN EL CONOCIMIENTO.* Colombia : Fondo De Publicaciones De La Universidad Distrital, 2004. Vol. I. 1794-211X..

5. Colombia, Universidad Nacional de. Sistemas Basados en Conocimiento. [En línea] [Citado el: 11 de octubre de 2012.] http://disi.unal.edu.co/~lctorress/iartificial/IA0011l.pdf..

6. Sitio del Ministerio de Relaciones Exteriores.Cuba. [En línea] [Citado el: 12 de Noviembre de 2012.] http://www.cubaminrex.cu/CDH/61cdh/Derechos Humanos en Cuba/Sistema Penitenciario.htm..

7. Pérez Sánchez, Juan José . *Derecho Penitenciario.*

8. Álvarez Munárriz, Luis. Fundamentos de Inteligencia Artificial. Murcia : Secretariado de Publicaciones. 1994.

9. Law Enforcement Automated Data Repository. [En línea] http://opensourceforamerica.org/papers/case-studies/law-enforcement-automated-data-repository-leadr/.

10. Observatorio Nacional de Software de Fuentes Abiertas. [En línea] http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=730:dosiier-nu..&limitstart=4.

11. Pressman, Roger. *Ingeniería del Software. Un enfoque práctico.*

12. Cockbun, A. *Agile Software Development.* s.l. : Addison-Wesley, 2001.

13. Villafuerte, V. Extreme Programing. [En línea] 2009. [Citado el: 10 de Noviembre de 2012.] http://extremeprogramming.com/.

14. Dickinson, J. *Grails 1.1 Web Application Development.* s.l. : Packt Publishing, 2009.

15. Abdul-Jawad, B. *Groovy and Grails Recipes.* New York, : NY Apress, 2009.

16. Gutiérrez., Javier J. [En línea] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

17. Mesa, Juan. Ilustre Colegio de Abogados de Jaén. [En línea] [Citado el: 22 de Noviembre de 2012.] http://www.icajaen.es/contenido/documentos/2006/cursos/16-1-juanmesa.pdf..

18. Corrections, U.S. Department of Justice. National Institute. [En línea] [Citado el: 7 de Noviembre de 2012.] http://www.jfa-associates.com/publications/pcras/06_ObjClass2004.pdf..

19. Instituto Interamericano de Derechos Humanos. [En línea] [Citado el: 7 de Noviembre de 2012.] http://www.iidh.ed.cr/comunidades/ombudsnet/docs/docsomb_pac/Nicaragua-Publicaciones/PHD-NIC Personas privadas de libertad.pdf..

20. España, Gobierno de. Ministerio del Interior. [En línea] [Citado el: 22 de Noviembre de 2012.] http://www.interior.gob.es/file/53/53022/53022.pdf..

21. Organización de Estados Americanos. Comisión Interamericana de Derechos Humanos. [En línea] [Citado el: 2012 de Noviembre de 8.] http://www.cidh.oas.org/PRIVADAS/Seminario/Seminario.III.n.pdf..

22. De Arquer, Isabel. Fiabilidad Humana: métodos de cuantificación, juicio de expertos. [En línea]

23. Fernández, Aedo, Raúl. *Los métodos de evaluación de expertos para valorar resultados de las investigaciones.* UCI. La Habana. : s.n.

24. Martínez Sánchez, MSc. Natalia, Ferreira Lorenzo, Dra. Gheisa y García Lorenzo, Dra. María M. *El Razonamiento Basado en Casos en el ámbito de la Enseñanza/Aprendizaje.* s.l. : Revista de Informática Educativa y Medios Audiovisuales, 2008. V. 1667-8338..

25. Díaz Gómez, Fernando. *Razonamiento Basado en Casos (CBR). Introducción.* Universidad de Valladolid : E. U. : s.n.

26. Bregón, Anibal, y otros. *Un sistema de razonamiento basado en casos para la clasificación de fallos en sistemas dinámicos.* Valladolid : s.n. 84-9732-449-8.

27. *Memoria Organizacional Basada en Casos.* Perez, Alonso. Recife, Brasil. : s.n., 2002.

28. Lozano, Laura y Fernández, Javier. *Razonamiento Basado en Casos: .* 2005.

29. García Martínez, Ramón y López Nocera, Marcelo Uvaldo. *Descubrimiento de conocimiento basado en la integración de algoritmos de explotación de la información.* Buenos Aires : s.n., 2009.

30. Xu, Rui y Wunsch, Donald C. *Clustering.* s.l. : IEEE, 2009, 2009. 978-0-470-27680-8..

31. *Sistema Basado en Casos para la toma de decisiones en condiciones de incertidumbre.* Gutiérrez, Iliana, Bello, Rafael e. y Tellería, Andrés. Santa Clara, Cuba : s.n., 2002, Vol. 23.

32. *Hottest New Approach to Knowledge-Based Systems Development.* Harmon, P. Vol. 7., s.l. : Intelligent Software Strategies, 1991.

33. Martínez Sánchez, Dra. Natalia, García Lorenzo, Dra. Maria Matilde y García Valdivia, Dra. Zenaida. *Modelo para diseñar sistemas de enseñanza-aprendizaje .* 2009.

BIBLIOGRAFÍA

1. España. Ministerio de Interior. [En línea] [Citado el: 20 de Noviembre de 2012.] http://www.institucionpenitenciaria.es/web/portal/laVidaEnPrision/clasificacion/.

2. AUSTIN, y otros, y otros. *Objective Prison Classification: A Guide for Correctional Agencies. U.S. Department of Justice.* EEUU : National Institute of Corrections, 2004.

3. Gálvez Lio, Daniel. *Sistemas Basados en el Conocimiento.Universidad Central de las Villas (UCLV).* 1988.

4. GARCÍA, MARILUZ ROMERO y RODRÍGUEZ, JORGE ENRIQUE . *SISTEMAS BASADOS EN EL CONOCIMIENTO.* Colombia : Fondo De Publicaciones De La Universidad Distrital, 2004. Vol. I. 1794-211X..

5. Colombia, Universidad Nacional de. Sistemas Basados en Conocimiento. [En línea] [Citado el: 11 de octubre de 2012.] http://disi.unal.edu.co/~lctorress/iartificial/IA0011l.pdf..

6. Sitio del Ministerio de Relaciones Exteriores.Cuba. [En línea] [Citado el: 12 de Noviembre de 2012.] http://www.cubaminrex.cu/CDH/61cdh/Derechos Humanos en Cuba/Sistema Penitenciario.htm..

7. Pérez Sánchez, Juan José . *Derecho Penitenciario.*

8. Álvarez Munárriz, Luis. Fundamentos de Inteligencia Artificial. Murcia : Secretariado de Publicaciones. 1994.

9. Law Enforcement Automated Data Repository. [En línea] http://opensourceforamerica.org/papers/case-studies/law-enforcement-automated-data-repository-leadr/.

10. Observatorio Nacional de Software de Fuentes Abiertas. [En línea] http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=730:dosiier-nu..&limitstart=4.

11. Pressman, Roger. *Ingeniería del Software. Un enfoque práctico.*

12. Cockbun, A. *Agile Software Development.* s.l. : Addison-Wesley, 2001.

13. Villafuerte, V. Extreme Programing. [En línea] 2009. [Citado el: 10 de Noviembre de 2012.] http://extremeprogramming.com/.

14. Dickinson, J. *Grails 1.1 Web Application Development.* s.l. : Packt Publishing, 2009.

15. Abdul-Jawad, B. *Groovy and Grails Recipes.* New York, : NY Apress, 2009.

16. Gutiérrez., Javier J. [En línea] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

17. Mesa, Juan. Ilustre Colegio de Abogados de Jaén. [En línea] [Citado el: 22 de Noviembre de 2012.] http://www.icajaen.es/contenido/documentos/2006/cursos/16-1-juanmesa.pdf..