Universidad de las Ciencias Informáticas

Facultad 3

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

Título: Extensión del Sistema de Gestión de Certificación de Roles 2.0

Autor

Arian Cruz Castillo

Tutores

Dr. C. Yoan Martínez Márquez

Ing. Haydee Fernández Díaz

Co-Tutores:

Ing. Felinda Rosabel León Mendoza

Ing. Eliodanis Maceo Rosales

La Habana, junio de 2016

"Año 58 de la Revolución"

DECLARACIÓN DE AUTORÍA

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

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

_____ _____

Arian Cruz Castillo Dr.C. Yoan Martínez Márquez

A mí mamá,

por ser la persona más especial que tengo en mi vida, ojalá siempre estés ahí para poder ponerte la mano en el hombro y decirte, tenías razón. Gracias por creer en mí, gracias por cuidar de mí, por ser la luz que guía mi camino, a ti te debo todo lo que soy.

A mí papá,

Que a pesar de ser de pocas palabras, me ha dedicado su vida, gracias por luchar para que pudiera cumplir mi sueño, nunca te vayas de mi lado porque te necesito, ¨digo que digo que te amo, de lo orgulloso que vivo, y de que no hay en el mundo, otro padre como el mío¨.

¨Ejemplos palpables los dos, de tenacidad, vivan mis padres, vivan por la eternidad¨.

A mí novia,

Por ser mi psicóloga del alma, y mi luz en la oscuridad, porque si naciera de nuevo me volviera a enamorar de ti. Por besarme cada vez como si fuera el primer beso, lo tuyo y lo mío es distinto, mientras el tiempo pasa, mejor se pone como el vino tinto, un amor como el nuestro, imposible que exista.

A mis padres y mi novia, porque este logro es tanto de ustedes como mío. Los Amo.

A mis tíos, Laura y Rafael, por ser mis segundos padres, y guiarme por el buen camino siempre.

A mis hermanos Yosbel y Ronier, por estar siempre a mi lado y brindarme su apoyo.

A mis abuelos, este logro es de ustedes también.

A mi familia, por apoyarme en todo paso que di durante mi formación profesional.

A los socios futbolistas del barrio, Dariel, René, Leo, Leandro, Yoandri ¡VIZÇA BARÇA!

A los "nuevos amigos", los que están: Alejandro, Paul, Osbel, Manolo, el Flaco, al secretario, Alian y los que no están: Isael, Arley y Eddy. Nunca voy a olvidar las bromas y las malas noches.

A Juan Miguel, por ser amigo y hermano, y darme fuerzas siempre que hacían falta. ¡Negrón, tu sabes, cualquier lío que tengas, después que lo resuelvas, me ves!

A mis tutores, Yoan, Haydee, Felinda y Eldionais, por haberme ¡soportado todo este tiempo!, los quiero.

A mi grupo 3501 por hacer de las clases algo ameno.

A todos los profesores que durante los años de estudio me han apoyado.

A todos ellos, quiero agradecer por estar presentes en mi camino.

RESUMEN

La certificación de roles en la Universidad de las Ciencias Informáticas, se basa en el procesamiento de las evidencias que aportan los estudiantes, en forma de artefactos, como parte de su actividad en las asignaturas Producción Investigación y Desarrollo. Estos artefactos se van generando de acuerdo a las tareas que contempla el plan de formación a lo largo del semestre, quedando evidenciadas en el Sistema de Gestión de Certificación de Roles 2.0. Este sistema centra su trabajo en el almacenamiento de los artefactos y las evaluaciones de las tareas, sin embargo, no se está explotando esta información con el objetivo de reducir y agilizar el trabajo de los tutores en el proceso de certificación de roles que deben realizar los estudiantes. Esto trae como consecuencia que se dificulte la toma de decisiones de los profesores, sobre cuáles escenarios o competencias se debe enfocar la preparación para lograr un incremento en el resultado de los educandos. Por la importancia que reviste este proceso se extendió el Sistema de Gestión de Certificación de Roles 2.0 con un paquete de funcionalidades que permiten evaluar los planes de formación, e incorporan un sistema de información de soporte a las decisiones. Este último visualiza mediante gráficos el desempeño real del estudiante, en correspondencia con las evidencias obtenidas y se contrasta con el desempeño ideal de acuerdo con la evaluación previa del plan de formación.

Palabras Clave: certificar, evaluar, extender, rol, visualizar.

INTRODUCCIÓN

El avance continuo y acelerado de las Tecnologías de la Información y las Comunicaciones (TIC), propicia la actualización y renovación constante de las aplicaciones y sistemas informáticos, desplegados en las más variadas esferas de la actividad humana. Son muchas las ventajas que ofrecen, que van desde la mejora del rendimiento y calidad de los procesos a los que son aplicadas, hasta la obtención de resultados relevantes en materia de ciencia (Aguilar, 2013). En el marco de este proceso de avance surgen los Sistemas de Información (SI), que tienen como objetivo mejorar la gestión de los datos y el desempeño dentro de las organizaciones.

Según (Peña, 2006) “un SI es un conjunto de elementos interrelacionados, con el propósito de prestar atención a las demandas de información de una organización, para elevar el nivel de conocimientos, que permitan un mejor apoyo a la toma de decisiones y desarrollo de acciones”. La creciente relevancia adquirida por estos en las organizaciones, hace que frecuentemente sean catalogados como piezas fundamentales en los nuevos paradigmas de gestión, en las principales teorías de la gestión del conocimiento y en las mejores prácticas organizacionales. El uso de los SI ha demostrado que son herramientas competitivas y generadoras de valor.

Los SI se han aplicado a diferentes áreas del quehacer científico en Cuba, entre las que se encuentran la salud, la economía y la educación. Ejemplo de ello son los sistemas: Sistema de Información Estadística Territorial y el Sistema de Información Estadístico Complementario. Estos sistemas se encuentran integrados en el Sistema Estadístico Nacional el cual ayuda a la toma de decisiones de políticas públicas sustentadas en la evidencia empírica. Todo ello forma parte del proceso de informatización que se está llevando a cabo en el país, donde la Universidad de las Ciencias Informáticas (UCI) juega un importante papel.

Esta universidad, fundada por el Comandante en Jefe Fidel Castro en el año 2002, posee una infraestructura productiva formada por diferentes centros de desarrollo de software que brindan soluciones informáticas a diferentes problemas existentes en disímiles esferas de la sociedad. En estos centros los estudiantes desempeñan diferentes roles productivos desde la asignatura de Producción, Investigación y Desarrollo (PID) correspondiente a la disciplina de Práctica Profesional (PP). A lo largo de la carrera los estudiantes deben desarrollar un sistema de habilidades y competencias establecidas dentro del plan de formación elaborado por el profesor de la asignatura, el cual debe estar diseñado para los niveles: básico, intermedio y avanzado.

Este plan está compuesto por macro-tareas que se subdividen en tareas, a las que deben dar solución mediante la generación de artefactos, que servirán de evidencia en caso de que opten por certificar algún rol (Madrigal, 2014).

La certificación de roles, se basa en el procesamiento de evidencias, consistentes en artefactos generados por los estudiantes en la asignatura PID de la carrera Ingeniería en Ciencias Informáticas (León, 2015). Estos artefactos se van generando de acuerdo a las tareas que contempla el plan de formación a lo largo del semestre, quedando evidenciadas en el Sistema de Gestión de Certificación de Roles 2.0.

Este sistema centra su trabajo en el almacenamiento de las evaluaciones de las tareas de los estudiantes, las cuales tributan a la evaluación de los escenarios, evaluación de las competencias y la evaluación final del rol. Cada dato aporta a la evaluación del nivel inmediato superior. Esta información no se está explotando y trae como consecuencia que se dificulte la toma de decisiones por parte de los profesores, sobre cuáles escenarios o competencias se debe enfocar la preparación para lograr un incremento en el resultado de los educandos.

Este resultado se ve afectado frecuentemente, pues el sistema anterior no permite al profesor visualizar el proceso de certificación de roles en el que se encuentran los estudiantes que él asesora. Esto trae consigo que los educandos no obtengan en ciertas ocasiones, el nivel de rol que desean, pues no se visualizan las distintas variables implícitas en la capacitación de un rol, como son: tareas, escenarios y competencias. El sistema no posibilita el análisis, comparación y valoración de estas variables. Además, no ofrece informes dinámicos e interactivos con los cuales el profesor pueda tomar decisiones más acertadas a la hora de asignar tareas a los estudiantes, que aporten al rol que estos últimos estén certificando.

Por la situación antes mencionada, se decide hacer un análisis de los elementos que inciden en el desempeño de los estudiantes durante el cumplimiento de las tareas del plan de formación en las asignaturas de PID. El objetivo de esta acción es brindar información al profesor para contribuir a la toma de decisiones importantes y estratégicas en el proceso de certificación de roles.

Teniendo en cuenta la problemática explicada anteriormente se identifica como **problema** a resolver: ¿Cómo extender el Sistema de Gestión de Certificación de Roles 2.0, de forma que permita evaluar los planes de formación, e incorpore un sistema de información de soporte a las decisiones en la certificación del rol?

Se plantea como **objeto de estudio:** el mantenimiento de software; siendo el **campo de acción:** el mantenimiento perfectivo de software.

Para dar solución al problema propuesto se traza como **objetivo general:** extender el Sistema de Gestión de Certificación de Roles 2.0, de manera que permita evaluar los planes de formación, e incorpore un sistema de información de soporte a las decisiones en la certificación de rol.

Del objetivo general se desglosan los siguientes **objetivos específicos:**

1. Elaborar el marco teórico de la investigación, mediante el estudio de los referentes teóricos de los sistemas de información, metodología de desarrollo, técnicas y herramientas de software.

2. Realizar el análisis y diseño de la extensión del Sistema de Gestión de Certificación de Roles 2.0, mediante la metodología de desarrollo de software seleccionada.

3. Extender el Sistema de Gestión de Certificación de Roles 2.0, mediante las herramientas y tecnologías seleccionadas.

4. Valorar la efectividad de la solución propuesta mediante pruebas de software para dar cumplimiento al objetivo de la investigación.

Para el cumplimiento de los objetivos se tienen en cuenta las siguientes **tareas de investigación**:

1. Revisión de los referentes teóricos de los sistemas de información y los sistemas de apoyo a la toma de decisiones.

2. Selección de la metodología y herramientas a utilizar para el desarrollo de la extensión del Sistema de Certificación de Roles 2.0.

3. Análisis del sistema para determinar los requisitos funcionales y no funcionales.

4. Confección del modelo de implementación del sistema de información de soporte a las decisiones.

5. Ejecución de pruebas de caja blanca al código fuente obtenido de la extensión del Sistema de Certificación de Roles 2.0 en la Disciplina de Práctica Profesional.

6. Valoración de la efectividad de la extensión del Sistema de Gestión de Certificación de Roles 2.0 en cada rol a certificar, mediante las pruebas de caja negra para comprobar su correcto funcionamiento.

7. Valorar el nivel de aceptación del cliente final de la aplicación mediante las pruebas de aceptación.

Idea a defender:

Desarrollando el paquete de funcionalidades en el Sistema de Gestión de Certificación de Roles 2.0, que incorpore un sistema de información de soporte a las decisiones, se podrá evaluar el plan de formación y visualizar el estado del estudiante en el rol a certificar.

Como métodos científicos de la investigación se emplearon los que a continuación se describen:

Métodos Teóricos:

- ✓ **Histórico-lógico:** Este método se utilizó durante la realización del estudio de la revisión bibliográfica, permitiendo evaluar el desarrollo de los marcos de trabajo existentes, la evolución de sistemas y de las herramientas que pudieron ser usadas para la elaboración de la solución propuesta.

- ✓ **Analítico-sintético:** permite el procesamiento de la información y arribar a conclusiones prácticas y teóricas de la investigación. Mediante este método se realizó un análisis de los documentos, las publicaciones, bibliografías y en general de toda la información relacionada con la certificación de roles y el empleo de sistemas de información de soporte a las decisiones. Se realizó un estudio sobre las tendencias actuales, posibilitando hacer una síntesis de estas principales fuentes y llegar a conclusiones parciales sobre el tema en estudio.

Métodos Empíricos:

- ✓ **Técnica de Entrevista:** permite obtener información valiosa sobre las necesidades del cliente para identificar los requisitos en el desarrollo del software. Se utilizó en encuentros sostenidos con los autores del Sistema de Gestión de Certificación de Roles 2.0. El objetivo fundamental de estos encuentros fue recopilar información acerca del sistema que contribuyera a entender el funcionamiento del negocio.

La investigación se compone de tres capítulos, los cuales se presentan a continuación:

Capítulo I. Fundamentación teórica: en este capítulo se abordan los aspectos teóricos relacionados con la investigación tales como: conceptos fundamentales abordados en el documento, el proceso de certificación de roles, sistemas de apoyo a la toma de decisiones, metodología de desarrollo de software, herramientas a emplear en la construcción de la solución, lenguajes, marco de trabajo y tecnologías.

Capítulo II. Propuesta de solución: en este capítulo se especifican los requisitos funcionales y no funcionales, teniendo en cuenta las restricciones o políticas a cumplir por el sistema. Se realiza la descripción del comportamiento del sistema a través de las Historias de usuario (HU). Como parte del diseño se define el patrón de arquitectura y se identifican los patrones de diseño que se van a utilizar. Se confeccionan las Tarjetas Clase-Responsabilidad-Colaborador (CRC) que contienen el nombre de las clases, las responsabilidades y sus colaboradores. Se especifican los patrones de diseño que se utilizan en la implementación de la propuesta. Finalmente, se implementa la propuesta de solución con el desarrollo de las tareas de ingeniería para las cuales se usaron varios estándares de codificación.

Capítulo lll. Validación de la propuesta de solución: en este último capítulo, se lleva a cabo la verificación de las funcionalidades que se tuvieron que implementar en el sistema para asegurar que corresponden y satisfacen los requisitos del cliente. Finalmente, se valida el sistema mediante las pruebas de aceptación para asegurar que el sistema cumple con las expectativas del cliente.

CAPÍTULO I. Fundamentación teórica

1.1. Introducción

En el presente capítulo se dejan plasmados los conceptos fundamentales abordados en la investigación y la descripción del proceso de certificación de roles que se realiza en la UCI. Se refleja el estudio realizado sobre los sistemas de información, se especifican un conjunto de enfoques teóricos e investigativos sobre la metodología, lenguajes y herramientas a utilizar, para el desarrollo de la extensión del Sistema de Gestión de Certificación de Roles 2.0. Se describen los patrones de diseños y arquitectónico que se emplearán en la solución. Finalmente, se describen las pruebas que se le realizarán al sistema.

Estudiantes pertenecientes a la UCI realizaron el Sistema de Gestión de Certificación de Roles 2.0 (León, 2015), el cual contribuye a perfeccionar la ponderación establecida por competencia en cada rol a certificar, así como la conformación y funcionamiento del tribunal. Dicho sistema requiere de un grupo de funcionalidades para visualizar los estados del proceso de certificación de roles, para permitir evaluar los planes de formación, e incorporar un sistema de información de soporte a las decisiones.

A continuación, se relacionan las principales definiciones que han sido estudiadas, con el objetivo de comprender las actividades que se realizan en el proceso de certificación de roles.

1.2. Conceptos fundamentales

En el presente epígrafe se abordan un conjunto de conceptos que a juicio del autor contribuirán a una mejor comprensión y toma de posición para el desarrollo de la investigación. En correspondencia con la definición del mantenimiento de software como objeto de estudio y el mantenimiento perfectivo de software como campo de acción se parte de asumir estos conceptos a los efectos de la presente investigación.

- ✓ **Mantenimiento de software:** proceso general de cambiar un sistema después de que este ha sido entregado. El término se aplica normalmente a software a medida en donde grupos de desarrollo distintos están implicados antes y después de la entrega. Los cambios realizados al software pueden ser cambios sencillos para corregir errores de código, cambios más extensos para corregir errores de diseño o mejoras significativas para corregir errores de especificación o acomodar nuevos requerimientos. Los cambios se implementan modificando

los componentes del sistema existente y añadiendo nuevos componentes al sistema donde sea necesario (Sommerville, 2005).

- ✓ **Mantenimiento perfectivo:** perfeccionar el software implementando nuevos requisitos; en otros casos significa mantener la funcionalidad del sistema pero mejorando su estructura y su rendimiento (Sommerville, 2005).

El Sistema de Gestión de Certificación de Roles 2.0 es un software desarrollado a la medida que tiene como cliente al Jefe de Departamento de Práctica Profesional del Centro de Gobierno Electrónico. Esta solución ha sido desarrollada en varias etapas, comenzando por una tesis en el curso 2013-2014 (Madrigal, 2014) y otra tesis en el curso 2014-2015 (León, 2015). La solución actual persigue añadir nuevas funcionalidades requeridas por el cliente.

Se considera que el mantenimiento que se realiza es perfectivo teniendo en cuenta que se incorporan nuevas funcionalidades relacionadas con la evaluación del plan de formación y la visualización del desempeño de los estudiantes en el rol al que aspiran certificar. El cliente solicita se determine el nivel máximo de certificación de rol al que puede certificar un estudiante, asumiendo que todas las tareas sean evaluadas con la máxima calificación en su plan de formación. Esta solicitud constituye una mejora al sistema en tanto contribuye a que se redefina el plan de formación del estudiante desde los momentos iniciales del semestre, para que tenga la oportunidad de aspirar al máximo nivel de certificación de rol. Posteriormente, con la visualización del desempeño del estudiante se realiza otra mejora que favorece identificar acciones correctivas durante el proceso.

- ✓ **Datos:** son la mínima unidad semántica, y se corresponden con elementos primarios de información que por sí solos son irrelevantes como apoyo a la toma de decisiones (Davenport, 2016).

- ✓ **Información:** conjunto de datos procesados y que tienen un significado (relevancia, propósito y contexto), y que por lo tanto son de utilidad para quién debe tomar decisiones, al disminuir su incertidumbre (Davenport, 2016).

- ✓ **Agilizar:** Hacer ágil, dar rapidez y facilidad al desarrollo de un proceso o a la realización de algo (RAE, 2014).

- ✓ **Sistema transaccional:** Es un tipo de sistema de información diseñado para recolectar, almacenar, modificar y recuperar todo tipo de información que es generada por las transacciones en una organización (Arellano, 2008).

- ✓ **Sistema de información:** conjunto de elementos que interactúan entre sí, con el fin de apoyar las actividades de una empresa o negocio. Teniendo muy en cuenta el equipo computacional necesario para que el sistema de información pueda operar y el recurso humano que interactúa con el Sistema de Información, el cual está formado por las personas que utilizan el sistema (Peralta, 2011).

- ✓ **Sistema de apoyo a las decisiones:** conjunto de programas y herramientas que permiten obtener oportunamente la información requerida durante el proceso de la toma de decisiones, en un ambiente de incertidumbre (Mos, 2015).

El Sistema de Gestión de Certificación de Roles 2.0 es el sistema transaccional que almacena los datos generados por los estudiantes en la ejecución de las tareas en su plan de formación. Estos datos correspondientes a los artefactos obtenidos son procesados en términos de evaluación, criticidad y complejidad mediante el método de factores ponderados (León, 2015). La visualización del desempeño del estudiante con la utilización de gráficas y la generación de notificaciones al tutor aportan información que contribuye a la toma de decisiones en el proceso de certificación de roles.

Una vez relacionados los conceptos fundamentales relativos al negocio de la extensión del Sistema de Gestión de Certificación de Roles 2.0, es preciso conocer cómo se realiza el proceso de certificación de roles en la UCI para comprender la necesidad de contribuir a su mejora, en el siguiente epígrafe se resume la situación actual de dicho proceso.

1.3. El proceso de certificación de rol

La certificación de roles es una forma de evidenciar la calidad en el desempeño del estudiante en un rol determinado y obtener por ello, un reconocimiento que lo avale. Este proceso permite reconocer las competencias adquiridas como resultado de la experiencia docencia-producción, fortaleciendo las oportunidades de permanencia en un determinado empleo/ocupación y construir una carrera profesional dentro de una organización, convirtiéndose de esta forma en una competencia muy valiosa.

El proceso de certificación de roles es opcional. El estudiante tiene derecho a solicitar su certificación en los roles que no haya certificado con anterioridad. Dicho proceso de certificación se inicia desde que comienza el sexto semestre, con el desarrollo de

diferentes actividades y tareas, cuyos resultados deben ser registrados como evidencia. Al finalizar el semestre, el estudiante debe solicitar al Jefe de Departamento de PP del centro al que se encuentra vinculado, la posibilidad de certificar los roles para los cuales logró acumular evidencias, presentando un expediente con las mismas. Una vez que el Jefe de Departamento de PP del centro tenga el listado de solicitudes, debe analizar y decidir si el estudiante puede presentarse ante el tribunal de certificación, teniendo en cuenta los requisitos necesarios para solicitar la certificación.

Los Jefes de Departamento de PP del centro deben entregar al Jefe de Departamento de Ingeniería y Gestión de Software (IGSW) de la facultad, el listado de estudiantes a iniciar el proceso de certificación y los expedientes de cada uno. El jefe del Dpto. de IGSW de la facultad es el responsable de entregar las cifras de estudiantes a certificar, para en función de ello, organizar los tribunales de certificación.

Los estudiantes deben presentar ante el tribunal de certificación, las evidencias del trabajo en el rol, además de realizar una presentación, donde se expongan las actividades y tareas desarrolladas asociadas el desempeño del rol que desea certificar, así como los elementos teóricos, herramientas, métodos, procedimientos llevados a cabo, resultados y novedad de las soluciones obtenidas. Durante la presentación, el tribunal comprobará el conocimiento teórico del estudiante en el rol que desea certificar. En este ejercicio debe participar el tutor del estudiante u otro profesor o especialista del proyecto que dé fe del desempeño del estudiante en el rol.

Una vez culminado el ejercicio, el tribunal decidirá si otorga o no la certificación al estudiante. En caso de otorgarla, los niveles de certificación que puede alcanzar el estudiante son: básico, intermedio y avanzado. A los estudiantes que certifiquen un rol se les entregará un certificado que refleje la certificación del rol correspondiente, el cual se adicionará a su currículum estudiantil y tributará para optar por el Mérito Científico en el quinto año de la carrera.

Durante el proceso de certificación de rol, el profesor PID orienta el desarrollo del estudiante. Este proceso se torna engorroso a la hora de tomar decisiones sobre los escenarios o competencias en que se debe enfocar la preparación del estudiante y lograr así un mayor desempeño del mismo. La causa de este problema es que no se cuenta con un sistema de información que respalde esas decisiones, el cual centre su atención en detectar áreas de mejoras en la preparación del estudiante, que le permitan al profesor optimizar el desempeño de estos, con la finalidad de que alcance el nivel de certificación deseado.

Una vez identificado en qué consiste el proceso de certificación de roles, se hace necesario conocer qué son los sistemas de información y el aporte que brindan los mismos a la extensión del sistema.

1.4. Sistemas de información

Según (Whitten, 2004) un sistema de información es un conjunto de personas, datos, procesos y tecnología de la información que interactúan para recoger, procesar, almacenar y proveer la información necesaria para el correcto funcionamiento de la organización.

Según (Peña, 2006) un sistema de información es un conjunto de elementos interrelacionados, con el propósito de prestar atención a las demandas de información de una organización, para elevar el nivel de conocimientos, que permitan un mejor apoyo a la toma de decisiones y desarrollo de acciones.

Según (Peralta, 2011) un sistema de información es un conjunto de elementos que interactúan entre sí, con el fin de apoyar las actividades de una empresa o negocio. Se debe tener en cuenta el equipo computacional necesario para que el sistema de información pueda operar y el recurso humano que interactúa con el sistema de información, el cual está formado por las personas que utilizan el sistema.

Luego de consultadas las fuentes citadas anteriormente, el autor de la presente investigación, coincide en que los sistemas de información son un conjunto de elementos que interactúan entre sí para apoyar la necesidad de información de una organización y elevar con ello el nivel de conocimiento, lo cual permite apoyar la toma de decisiones. Estos suelen introducirse después de haber implantado los sistemas transaccionales más relevantes de la empresa, los cuales constituyen su plataforma de información.

Entonces, podemos extraer de dicho concepto los elementos fundamentales de un sistema de información:

- ✓ Información: conjunto de datos procesados y que tienen un significado (relevancia, propósito y contexto), y que por lo tanto son de utilidad para quién debe tomar decisiones, al disminuir su incertidumbre (Davenport, 2016).
- ✓ Decisión: proceso de análisis y selección entre diversas alternativas disponibles (Díaz, 2005).

Siguiendo esta línea, y de acuerdo a su función se distinguen tres tipos de sistemas de información: sistemas de información gerencial, sistemas de información ejecutiva y los sistemas de soporte a decisiones. A los efectos de la presente investigación se asume la última clasificación dada, ya que se basan en el estudio y la comparación entre un conjunto de variables con el objetivo de contribuir a la toma de decisiones dentro de una empresa. El apoyo dado por el sistema involucra la estimación, valoración y balance entre alternativas (Pérez, 2011). El mantenimiento del sistema persigue desarrollar la evaluación del plan de formación, la visualización del desempeño de los estudiantes y las notificaciones al tutor. Se considera que de esta manera se contribuirá a la toma de decisiones correctivas durante la ejecución de las tareas para la certificación de roles.

En el próximo epígrafe se mencionan las características de estos sistemas de soporte a decisiones, y cómo son usados en la actualidad.

1.4.1 Sistemas de soporte a decisiones

Los sistemas de apoyo a la toma de decisiones o *Decision Support System (*DSS), son un conjunto de programas y herramientas que permiten obtener oportunamente la información requerida durante el proceso de la toma de decisiones, en un ambiente de incertidumbre (Mos, 2015).

Un DSS puede adoptar muchas formas diferentes. En general, se puede decir que un DSS es un sistema informático utilizado para servir de apoyo más que automatizar el proceso de toma de decisiones. La decisión es una elección entre alternativas basadas en estimaciones de los valores de esas alternativas. El apoyo a una decisión significa ayudar a las personas que trabajan solas o en grupo a reunir inteligencia, generar alternativas y tomar decisiones. Apoyar el proceso de toma de decisión implica el apoyo a la estimación, la evaluación y/o la comparación de alternativas (Sinergia e Inteligencia de Negocio S.L., 2016).

El DSS es una de las herramientas más emblemáticas del *Business Intelligence* ya que, entre otras propiedades, permiten resolver gran parte de las limitaciones de los programas de gestión. Estas son algunas de sus características principales (Sinergia e Inteligencia de Negocio S.L., 2016):

- ✓ **Informes dinámicos, flexibles e interactivos**: el usuario no tiene que ceñirse a los listados predefinidos que se configuraron en el momento de la implantación, y que no siempre responden a sus dudas reales.

- ✓ **No requiere conocimientos técnicos**: un usuario no técnico puede crear nuevos gráficos e informes y navegar entre ellos sin necesidad de auxilio.

- ✓ **Cada usuario dispone de información adecuada a su perfil**: no se trata de que todo el mundo tenga acceso a toda la información, sino de que tenga acceso a la información que necesita para que su trabajo sea lo más eficiente posible.

Una vez analizados los DSS y sus principales características se puede decir que estos son útiles en los problemas en los cuales hay suficiente estructura como para construir un modelo matemático o estadístico que permita su resolución por medio de la computadora, pero que finalmente requiere del juicio del ejecutivo. El Sistema de Gestión de Certificación de Roles 2.0 tiene la estructura creada para el procesamiento de los datos. Con ella, el modelo matemático que permite procesar los datos y convertirlos en información para elevar el nivel de conocimiento, permitiéndole a los profesores tomar decisiones basadas en la información que generará el sistema luego de desarrollada la extensión del sistema.

1.5. Metodología, herramientas, y tecnologías utilizadas en la extensión del Sistema de Certificación de Roles 2.0 en la Disciplina de Práctica Profesional

1.5.1 Metodología de desarrollo

La metodología para el desarrollo de software es un modo sistemático de realizar, gestionar y administrar un proyecto para llevarlo a cabo con altas posibilidades de éxito. Esta sistematización indica cómo dividir un gran proyecto en partes más pequeñas llamadas etapas, y las acciones que corresponden en cada una de ellas. Ayuda a definir entradas y salidas para cada una de las etapas y, sobre todo, normaliza el modo de administrar un proyecto. Entonces, una metodología para el desarrollo de software son los procesos a seguir sistemáticamente para idear, implementar y mantener un producto de software desde que surge la necesidad del producto hasta que se cumpla el objetivo por el cual fue creado (Sommerville, 2005).

."La metodología de desarrollo de software se encarga de elaborar estrategias; centradas en las personas o los equipos, orientadas hacia la funcionalidad y la entrega. Su principal objetivo es elevar la calidad del software a través de un mayor control sobre el proceso" (Sommerville, 2005). Una metodología de desarrollo de software es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software (Sommerville, 2005).

En la extensión del Sistema de Certificación de Roles 2.0 en la Disciplina de Práctica Profesional se propone el uso de una metodología ágil, la cual ha sido pensada para proyectos donde el equipo de desarrollo es pequeño, con plazos reducidos y requisitos cambiantes durante el proceso. Además, la comunicación con el cliente debe ser muy frecuente, haciendo a este parte del equipo de desarrollo.

Mencionadas anteriormente las ventajas que ofrecen las metodologías ágiles, se propone el uso de Extreme Programming (XP). Esta metodología se centra 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. 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. Se define adecuada para proyectos con requisitos imprecisos y muy cambiantes (Letelier, 2006).

La metodología XP presenta características esenciales, las cuales se dividen en dos apartados fundamentales: historias de usuario (HU) y procesos o fases. Las HU son la técnica utilizada para especificar los requisitos del software. Estas describen brevemente las características que el sistema debe poseer, reflejadas en los requisitos funcionales. Para el desarrollo del mantenimiento perfectivo del Sistema de Gestión de Certificación de Roles 2.0 se siguen cinco fases de las seis que presenta XP, estas son:

1. **Exploración:** en esta fase el cliente plantea a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto (Letelier, 2006).

2. **Planificación:** en esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente (Letelier, 2006).

3. **Iteraciones:** esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. Al final de la última iteración el sistema estará listo para entrar en producción (Letelier, 2006).

4. **Producción:** esta fase requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase (Letelier, 2006).

5. **Muerte del proyecto:** es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura (Letelier, 2006).

En correspondencia con la fase de exploración de la metodología seleccionada, se presenta en los siguientes epígrafes el estudio sobre las herramientas y tecnologías.

1.5.2 Marco de trabajo

Uno de los pasos más importantes en el desarrollo de un software es la elección del marco de trabajo. Un marco de trabajo (*framework*[1] en inglés) simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, proporciona una estructura al código fuente, forzando al desarrollador a crear código más legible. Facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas (Potencier, 2008).

Symfony 2.1.7: es un marco de trabajo completo, diseñado para optimizar el desarrollo de aplicaciones web. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases, encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Permite la ejecución automatizada de tareas, brinda buena documentación y logra la implementación robusta del patrón Modelo Vista Controlador (MVC) con la integración de *Doctrine*[2]. Es compatible con la mayoría de los

[1] **Framework**: Traducido al español se refiere a marco de trabajo. Es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concreto, con base a la cual otro proyecto de software puede ser más organizado y desarrollado.

[2] **Doctrine**: Es una biblioteca para PHP que permite trabajar con un esquema de base de datos como si fuese un conjunto de objetos y no de tablas y registros.

gestores de bases de datos y puede ser ejecutado en múltiples plataformas (Potencier, 2010).

Se selecciona como marco de trabajo Symfony 2.1.7, porque es el marco de trabajo utilizado en el desarrollo del Sistema de Gestión de Certificación de Roles 2.0. El mismo ha sido diseñado para desarrollar y optimizar aplicaciones web, así como obtener el máximo rendimiento de PHP 5 y aprovechar sus características.

1.5.3 Lenguajes de programación

Los lenguajes de programación permiten crear programas mediante un conjunto de instrucciones, reglas de sintaxis y operadores que un equipo debe ejecutar (Bonanata, 2013). En el contexto de la presente investigación, donde se realiza un mantenimiento perfectivo del Sistema de Gestión de Certificación de Roles 2.0 se asume lo definido por el equipo de desarrollo de las tesis (Madrigal, 2014) y (León, 2015). A continuación, se enuncian los lenguajes de programación que se emplearán para el desarrollo de la solución.

Lenguajes del lado del Servidor:

- ✓ **PHP 5.4:** es un lenguaje de programación multiplataforma orientado a objetos y con una gran cantidad de bibliotecas de funciones. Permite la conexión con la mayoría de los gestores de base de datos, y posee capacidad para expandir su potencial utilizando módulos. No requiere definición de tipos de variables, ni manejo detallado de bajo nivel (Achour, 2014).

  Este es el lenguaje empleado en la construcción de las páginas web de contenido dinámico. Permite el manejo eficiente de excepciones y el empleo de técnicas de programación orientada a objetos. Proporciona estabilidad a la aplicación ya que utiliza su propio sistema de administración de recursos y dispone de un método de manejo de variables, conformando así un sistema robusto. Posibilita configurar diferentes niveles de seguridad para evitar ataques al código. Existe una comunidad considerable de desarrolladores formalizada, con basta documentación sobre la tecnología.

- ✓ **Twig 2.1:** es un motor de plantillas, que posee un ambiente amigable para el diseñador y el desarrollador, permitiendo añadir funcionalidades a los entornos de plantillas. Posee una sintaxis corta y concisa, similar a la de otros lenguajes

de programación. Implementa un mecanismo de herencia de plantillas, para acelerar el rendimiento del sistema que se desarrolla (Pacheco, 2013).

Este lenguaje permite compilar las plantillas hasta código PHP regular optimizado, lo que proporciona rapidez durante la implementación. Garantiza la seguridad del código de las plantillas ya que posee un modo de recinto de seguridad para evaluar el código de plantilla que no es confiable. Permite al desarrollador definir sus propias etiquetas y filtros personalizados, garantizando flexibilidad a la aplicación. Posibilita la depuración de las plantillas creadas, mediante mensajes de error con el nombre del archivo y el número de línea donde se produjo el problema (Pacheco, 2013).

Lenguajes del lado del Cliente:

- ✓ **HTML 5.0:** es un lenguaje para escritura de hipertexto, en otras palabras, documentos de texto estructurados, incluye enlaces (links, según su denominación en inglés) que conducen a otros documentos o a otras fuentes de información y permite la inclusión de información por formularios, entre otros. HTML5 se caracteriza por ofrecer una mejor estructura eliminando el uso excesivo de las etiquetas <*div*>, esta se emplea para definir un bloque de contenido o sección de la página; con el objetivo de que la web sea más coherente y comprensible (Castillo, 2012).

- ✓ **CSS 3.0:** es un lenguaje creado para controlar el aspecto o la presentación de los documentos electrónicos definidos con HTML[3]. Es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. Es empleado para separar los contenidos y su presentación y es imprescindible para crear páginas web complejas ya que separa la definición de los contenidos y la descripción de su aspecto, presentando numerosas ventajas. Mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes (Jeremy, 2014).

- ✓ **Bootstrap 3.0.1:** es una de las bibliotecas más utilizadas y conocidas en la actualidad, desarrollada por Twiter, pensado y diseñado para crear interfaces

[3] **HTML**: Lenguaje de Marcado de Hipertexto

web. Los diseños creados con Bootstrap son simples, limpios e intuitivos, esto ofrece agilidad a la hora de cargar y al adaptarse a otros dispositivos. Una de las principales características que posee, es que permite crear interfaces web con CSS[4] y JavaScript que adaptan la interfaz dependiendo del tamaño del dispositivo en el que se visualice de forma nativa, en otras palabras, automáticamente se adapta al tamaño de un ordenador o de una tablet sin que el usuario tenga que hacer nada. Esto se denomina diseño adaptativo o *Responsive Designy*. Es una de las corrientes que marcan esta nueva era digital (Otto, 2014).

- ✓ **JavaScript 1.6:** es un lenguaje de programación utilizado para crear pequeños programas encargados de realizar acciones dentro del ámbito de una página web. Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Su uso se basa fundamentalmente en la creación de efectos especiales en las páginas y la definición de interactividades con el usuario (Polo, 2008).

1.5.4 Herramienta de desarrollo

Un Entorno de Desarrollo Integrado (IDE, según sus siglas en inglés), es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI) (Netbeans.org, 2016).

Los IDE proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, PHP, Python, Java, C#, Delphi, Visual Basic, entre otros. En algunos lenguajes, un IDE puede funcionar como un sistema en tiempo de ejecución, donde se permite utilizar el lenguaje de programación de forma interactiva, sin necesidad de trabajo orientado a archivos de texto.

Se selecciona como IDE Netbeans 8.0 porque posee una plataforma de aplicaciones que permite el desarrollo rápido y fácil de sistemas web, con soporte para los lenguajes sobre los cuales está implementado el Sistema de Gestión de Certificación de Roles 2.0. Este IDE permite la integración con el marco de trabajo Symfony también usado en este sistema. El IDE Netbeans es un software libre que cuenta con una comunidad que tiene un número significativo de usuarios y desarrolladores en todo el mundo. Estas

[4] **CCS**: Hojas de estilo en cascada

características favorecen el acceso a la documentación necesaria para su estudio y aprovechamiento en el desarrollo de la solución informática (León, 2015).

1.5.5 Herramientas para base de datos

Sistema Gestor de Bases de Datos: los Sistemas de Gestión de Base de Datos (SGBD) se usan como interfaz entre las aplicaciones, los usuarios y las bases de datos. Tienen como propósito principal gestionar convenientemente la información a almacenar o recuperar, de manera sencilla y fiable para el usuario (Rodríguez, 2003).

PostgreSQL 9.4: es un gestor de base de datos orientado a objetos (SGBDOO) muy conocido y usado en entornos de software libre. Está considerado como un SGBD de código abierto robusto. Posee uso de transacciones avanzadas, lenguaje procedimental, estabilidad y escalabilidad (Pérez, 2014).

Se selecciona el gestor de base de datos PostgreSQL porque la base de datos del Sistema de Gestión de Certificación de Roles 2.0 está montada sobre el mismo. Además, presenta gran capacidad de almacenamiento y buena escalabilidad.

1.5.6 Servidor web

Un servidor web o servidor HTTP[5], es un programa informático que procesa una aplicación del lado del servidor, realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente, generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente. Para la transmisión de todos estos datos generalmente se utiliza el protocolo HTTP o el protocolo HTTPS[6] (la versión cifrada y autenticada), para estas comunicaciones pertenecientes a la capa de aplicación del modelo OSI[7].

El servidor web seleccionado para procesar la aplicación del lado del servidor es Apache 2.4, la cual se fundamenta a continuación.

Apache 2.4: Apache está diseñado para ser un servidor web potente y flexible, continuamente actualizado y adaptado a los nuevos protocolos (HTTP) y puede funcionar en la más amplia variedad de plataformas y entornos. Apache se ha adaptado

[5] **HTTP**: Protocolo de transferencia de hipertexto

[6] **HTTPS**: Protocolo seguro de transferencia de hipertexto

[7] **OSI**: Modelo de interconexión de sistemas abiertos

siempre a una gran variedad de entornos a través de su diseño modular, el cual permite elegir las características del servidor seleccionando qué módulos se van a cargar, ya sea al compilar o al ejecutar el servidor. Tiene capacidad para servir páginas tanto de contenido estático, como de contenido dinámico a través de otras herramientas soportadas que facilitan la actualización de los contenidos mediante bases de datos, ficheros u otras fuentes de información (Kabir, 2003).

Es un software de libre distribución, que publica su código fuente, lo que permite que pueda ser modificado para colaborar en su desarrollo. Tiene interfaz con todos los sistemas de autenticación. Facilita la integración de complementos de los lenguajes de programación de páginas web dinámicas más comunes. Tiene integración en estándar del protocolo de seguridad SSL y provee interfaz a todas las bases de datos (Kabir, 2003).

Apache es seleccionado por la necesidad de un servidor web compatible con la plataforma de desarrollo, que permita rapidez en la navegación y la seguridad e integridad de los datos con los que se trabajen. Este servidor web permite configurar los módulos a utilizar en el servidor, logrando desechar elementos innecesarios para lograr mayor rapidez en la renderización[8] de las páginas.

1.6. Patrones

Los patrones ayudan a desarrollar aplicaciones robustas y fáciles de mantener cumpliendo ciertas reglas. Algunos de sus beneficios son la programación sólida, control de cohesión y acoplamiento, y reutilización de código (Campo, 2009).

1.6.1 Patrón arquitectónico

Un **Patrón arquitectónico** está orientado a representar los diferentes elementos que componen una solución de software y las relaciones entre ellos. Los patrones arquitectónicos forman parte de la llamada Arquitectura de Software (arquitectura lógica de un sistema) que define, de manera abstracta, los componentes que llevan a cabo alguna tarea de proceso de información, sus interfaces y la comunicación entre ellos (Rodríguez, 2003). En la presente investigación se realiza un mantenimiento perfectivo

[8] **Renderización:** cadena de acciones que realiza un navegador para presentar una página web, estas son, analizar, procesar y visualizar una web (Canga, 2015)**.**

del Sistema de Gestión de Certificación de Roles 2.0. Este sistema está basado en el patrón modelo-vista-controlador, el cual se fundamenta a continuación:

- ✓ **Modelo-Vista-Controlador (MVC):** Este 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 MVC 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 la capa de abstracción que se sitúa sobre el Sistema de Gestión de Base de Datos en este caso PostgreSQL, y la lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista. La vista representa el modelo en un formato adecuado para interactuar con el usuario. Transforma el modelo en una página web que permite al usuario interactuar con ella. Se encarga de la presentación visual de los datos captados por el modelo para después mostrarlos al usuario. Interactúa con el modelo y muestra los datos al usuario. Los controladores reciben la entrada o sea los eventos producidos por el usuario mediante el uso del teclado o el mouse a través de las vistas. Los eventos son traducidos para servir las demandas del modelo o las vistas. Cuando se realiza algún cambio, entra en acción bien puede ser por cambios en el modelo o en la vista (Rodríguez, 2003).

  La aplicación de este patrón de arquitectura presenta varias ventajas:

  - ✓ Reduce el esfuerzo de programación necesario en la implementación de sistemas.
  - ✓ El modelo, la vista y los controladores se tratan como entidades separadas lo cual hace que un cambio producido en el modelo se refleje automáticamente en las vistas.
  - ✓ Se pueden construir varias vistas sin necesidad de modificar el modelo.

La extensión del Sistema de Gestión de Certificación de Roles 2.0 hará uso del patrón arquitectónico argumentado anteriormente, sobre el cual está diseñado el sistema actual. Este patrón facilita el desarrollo de las funcionalidades en el sistema, ya que presenta distintos diseños de presentación sin que se altere la lógica del negocio. Es importante destacar que proporciona reutilización de los componentes y simplicidad en el mantenimiento del sistema.

1.6.2 Patrón de diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí, adaptada para resolver un problema de diseño general en un contexto particular (Gamma, 2002). La utilización de patrones de diseño, permite ahorrar grandes cantidades de tiempo en la construcción de software. El producto obtenido es más fácil de comprender, mantener y extender. Existen versiones ya implementadas de estas funcionalidades comunes (marcos de trabajo) que permiten centrarse en desarrollar sólo la funcionalidad específica requerida por cada aplicación y que además, dan mejor imagen de profesionalidad y calidad.

Los patrones de diseño se agrupan en dos grandes categorías: GRASP[9] y GoF[10]. Los primeros describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. Dentro de este grupo de patrones se encuentran los siguientes: Experto, Creador, Bajo Acoplamiento, Controlador y Alta Cohesión. Los patrones de diseño GoF son 23 y se clasifican según su propósito en Creacionales, Estructurales, Comportamiento y según su ámbito en Objeto y Clase.

En el Capítulo 2 se explican los patrones de diseño que fueron empleados en la implementación de la solución.

1.7. Pruebas

Según (Scalone, 2006) las pruebas de software son “las actividades en la que un sistema o componente es ejecutado bajo condiciones especificadas. El objetivo es diseñar una serie de casos de pruebas que tengan una alta probabilidad de encontrar errores”. Para ello se aplican las técnicas de pruebas del software, las cuales facilitan una guía sistemática para diseñar pruebas que comprueben la lógica interna de los componentes de software y verifiquen los dominios de entrada y salida del programa para descubrir errores en la funcionalidad, el comportamiento y rendimiento (Pressman, 2000).

En la presente investigación se realizarán pruebas a nivel de unidad, correspondientes a los métodos de prueba de caja blanca y caja negra. Para el primer método se aplica

[9] **GRASP:** Acrónimo de “General Responsibility Assignment Software Patterns” en español Patrones generales de software para asignación de responsabilidades.

[10] **Gang of Four**, en español **Banda de Cuatro**, formada por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides.

la técnica de camino básico y para el segundo se emplea la técnica de partición de equivalencia. Además, se realizarán pruebas de nivel de aceptación, correspondientes a las pruebas de aceptación.

1.7.1 Nivel de unidad

Las pruebas del nivel de unidad se aplican para verificar que el software cumple los requisitos funcionales y asegurar la calidad del código entregado. Además, son la mejor forma de detectar fallas tempranamente en el desarrollo y está demostrado que mientras más pronto se encuentren los errores, menos costará corregirlos. A este nivel se realizan las pruebas de funcionalidad, utilizando el método de prueba de caja blanca y caja negra para comprobar que tanto el código como la interfaz no contienen errores y se ejecutan adecuadamente.

- ✓ **Método de prueba de caja blanca**

Este método, denominado a veces prueba de caja de cristal es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba. Mediante los métodos de prueba de caja blanca, el ingeniero de software puede obtener casos de prueba que: (1) garanticen que se ejercita por lo menos una vez todos los caminos independientes de cada módulo; (2) ejerciten todas las decisiones lógicas en sus vertientes verdadera y falsa; (3) ejecuten todos los ciclos en sus límites y con sus límites operacionales; y (4) ejerciten las estructuras internas de datos para asegurar su validez (Pressman, 2000).

Para ejecutar este tipo de pruebas según (Pressman, 2000) se utiliza la **técnica de camino básico:** esta técnica permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución.

- ✓ **Método de prueba de caja negra**

Las pruebas de caja negra son aquellas que se llevan a cabo sobre la interfaz de software, por lo que los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta, así como que la integridad de la información externa se mantiene. Esta prueba examina algunos aspectos del funcionamiento del sistema sin tener mucho en cuenta la estructura interna del software (Fernández, 2011).

Estas pruebas permiten encontrar:

1. Funciones incorrectas o ausentes
2. Errores de interfaz
3. Errores en estructuras de datos o en accesos a las bases de datos externas
4. Errores de rendimiento
5. Errores de inicialización y terminación

Según (Pressman, 2000), para desarrollar la técnica de caja negra existen varias técnicas, entre las que se encuentran:

1. **Técnica de la Partición de Equivalencia:** divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
2. **Técnica del Análisis de Valores Críticos:** prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
3. **Técnica de Grafos de Causa-Efecto:** permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

En este caso se escoge dentro del método, la técnica de la partición de equivalencia que es una de las más efectivas pues permite examinar los valores válidos e inválidos de las entradas existentes en el software. La partición equivalente se dirige a la definición de casos de pruebas que descubran clases de errores, reduciendo así el número de casos de prueba que hay que desarrollar.

1.7.2 Nivel de aceptación

Dentro de los tipos de pruebas que hay para validar un software, una de las más importantes son las del nivel de aceptación (user´s test). A este nivel se aplican las pruebas de aceptación que generalmente son diseñadas por el equipo de desarrollo y ejecutadas por el cliente o un especialista de la aplicación y se consideran pruebas del tipo de caja negra. Son conducidas a determinar cómo el sistema satisface sus criterios de aceptación, validando los requisitos que han sido levantados para el desarrollo, incluyendo la documentación y procesos de negocio. Están consideradas como la fase final del proceso, para crear un producto confiable y apropiado para su uso (Pressman, 2000).

1.8. Conclusiones parciales

Culminado el estudio de los sistemas de información, asumidas la metodología, herramientas, y tecnologías que serán utilizadas en el desarrollo de la extensión del Sistema de Gestión de Certificación de Roles 2.0 en la Disciplina de Práctica Profesional, se arriba a las siguientes conclusiones:

1. Los profesores que se encargan de asesorar el proceso de certificación de roles de los estudiantes en la UCI carecen de información necesaria para mejorar el desempeño de sus estudiantes.

2. El desarrollo de la solución se enmarca en el mantenimiento perfectivo de software. Las principales funcionalidades que se incorporarán al sistema consisten en evaluar el plan de formación y visualizar el desempeño del estudiante.

3. Los sistemas de información de soporte a las decisiones constituyen uno de los aspectos estratégicos claves para el buen hacer de la organización. Para ello es necesario que la totalidad de la organización esté concienciada de su utilidad.

4. Se elige un enfoque de desarrollo ágil, asumiendo XP como metodología de desarrollo. Los lenguajes y herramientas de desarrollo seleccionados están en correspondencia con las características del equipo de desarrollo y de la solución que se persigue obtener.

CAPÍTULO II. Propuesta de solución

2.1. Introducción

En el desarrollo del siguiente capítulo se describe la propuesta de solución a desarrollar, teniendo en cuenta las fases de Planificación, Diseño y Codificación del ciclo de vida que propone la metodología XP. Se realiza la identificación de los requisitos funcionales y no funcionales resultantes de la aplicación de las técnicas que para ello se utilizan. Posteriormente se confeccionan las HU, donde se estima el esfuerzo de cada una de ellas y se realiza el plan de iteraciones definido para el desarrollo del proyecto. Se describen la arquitectura del sistema, las tarjetas CRC identificadas y los patrones de diseño, para establecer un diseño de la solución a implementar. Finalmente se realiza la codificación de la solución, donde se describen las tareas de ingeniería que fueron necesarias para la implementación de cada una de las historias de usuario.

2.2. Propuesta de solución

En la extensión del Sistema de Gestión de Certificación de Roles 2.0, el profesor de la asignatura PID debe elaborar un registro con los estudiantes que supervisará en el semestre y asignarle a cada uno de ellos un plan de formación. Este plan de formación carece de la descripción de los escenarios a los cuales aportan las tareas, lo cual es necesario para poder evaluar ese plan. Es por ello que se deben realizar cambios en la funcionalidad de importar plan, para que tenga presente la descripción antes mencionada.

Una vez importado el plan de formación del estudiante, este puede acceder a las tareas a través del sistema, y subir para cada una de ellas los artefactos que las mismas generan. De esta forma se va conformando el expediente, asociado a un rol que será candidato a certificar. En la medida que el profesor evalúe las tareas del estudiante para las cuales tiene artefactos generados, se crea el portafolio del estudiante. En esta etapa del proceso, se debe evaluar el plan de formación del estudiante. Se determina el nivel máximo al que podrá aspirar en el rol que está desempeñando, en correspondencia con la complejidad, criticidad y asumiendo que todas las tareas sean evaluadas con la máxima calificación. Esta primera evaluación es de gran importancia para que se asigne al estudiante un plan de formación que le permita contar con las tareas suficientes para optar por el nivel avanzado en la certificación de rol. Este proceso debe ser notificado al profesor mediante correo, en el que se muestre el registro de evaluaciones del

estudiante y la evaluación del plan de formación resultante de lo explicado anteriormente.

En la medida que el profesor vaya revisando las tareas asignadas, estas van aportando al nivel inmediato superior, evaluación del escenario, la cual tendrá un peso dentro de la evaluación de la competencia y estas a su vez tendrán un peso dentro del nivel alcanzado en la certificación de rol. En la **Imagen 1** se muestra el modelo de esta granularidad.

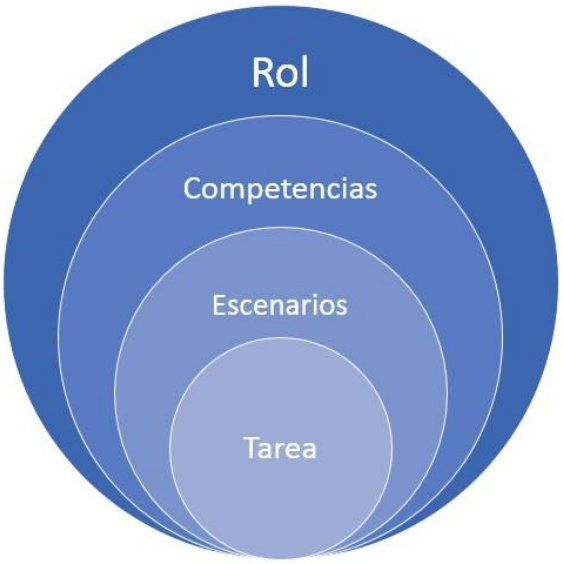

Imagen 1 *Granularidad en la certificación de rol (León, 2015)*

Una vez el estudiante tenga evaluaciones asignadas en alguna de las tareas, se debe crear un gráfico que muestre su desempeño en la certificación de rol. Este gráfico mostrará dos líneas, una de ellas será el desempeño ideal, sobre la cual se ubican los pesos de cada competencia. Estos pesos son el aporte de cada competencia a la evaluación final del rol. La otra línea de este gráfico muestra el desempeño real del estudiante para cada una de las competencias que contiene el rol (ver **Imagen 2**).

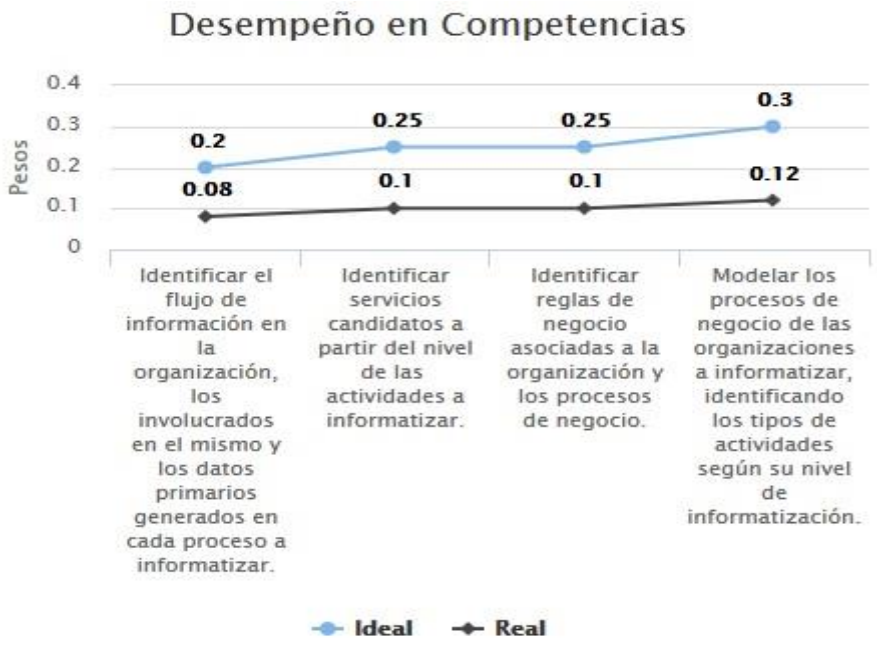

Imagen 2 *Gráfico competencias por rol (fuente: elaboración propia)*

Una vez que el profesor accede al gráfico de las competencias puede seleccionar una de ellas, y obtendrá un gráfico con el desempeño ideal y el desempeño real en los distintos escenarios de la competencia seleccionada (ver **Imagen 3**).

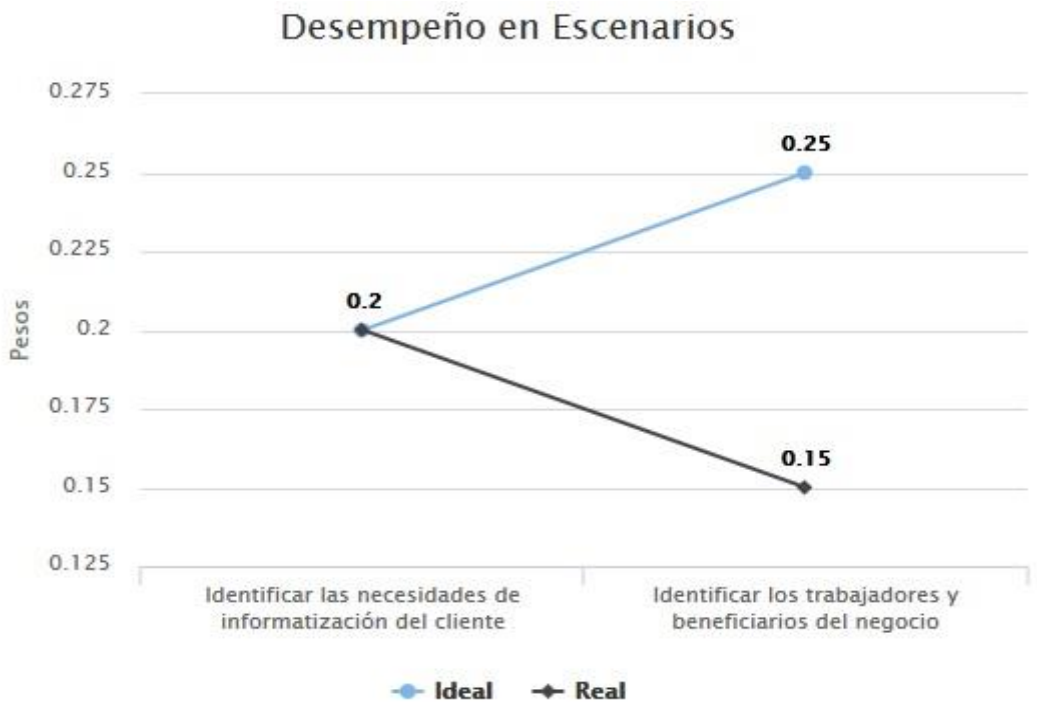

Imagen 3 *Desempeño de los escenarios de una competencia (fuente: elaboración propia)*

El profesor debe seleccionar un escenario para poder ver las tareas que están modificando su evaluación. Esta acción se realiza para identificar los resultados que están afectando el desempeño del estudiante. Se muestra a continuación el gráfico de tareas que tiene asociado su criticidad, complejidad y evaluación con el fin de identificar concretamente el indicador con mayores niveles de afectación (ver **Imagen 4**).

Imagen 4 *Desempeño en las tareas de un escenario seleccionado (fuente: elaboración propia)*

Para obtener la evaluación de un rol, de los siguientes niveles de certificación (Básico, Intermedio o Avanzado), se deben tener en cuenta la evaluación de la tarea, los escenarios y las competencias. Cada tarea tiene un aporte según su evaluación, el mismo se muestra en la siguiente tabla:

Tabla 1 *Aporte por evaluación (León, 2015)*

Evaluación de la tarea	Aporte
2	1
3	1.5
4	2
5	2.5

Otro de los valores presentes en la evaluación de una tarea es la complejidad, la cual se describe en la siguiente tabla:

Tabla 2 *Tabla de complejidad (León, 2015)*

Complejidad (descripción)	Aporte
Alta	1
Media	0.8
Baja	0.4

El otro valor que contribuye a la evaluación de la tarea es la criticidad, en la siguiente tabla se describe este indicador.

Tabla 3 *Tabla de criticidad (León, 2015)*

Criticidad (descripción)	Aporte
Alta	1.5
Media	1.2
Baja	0.6

Teniendo estos valores del aporte de la evaluación, la complejidad y la criticidad, se deben sumar y luego se dividen en 5, como se muestra en la siguiente ecuación (León, 2015).

$$Evaluación\ Escenario = \frac{Aporte + Criticidad + Complejidad}{5}$$

Este cálculo arroja un valor entre 0 y 1, el cual dirá qué evaluación tiene el escenario. En la siguiente tabla se especifican esos intervalos.

Tabla 4 *Evaluación de las tareas según los intervalos (León, 2015)*

Evaluación de la tarea	Intervalo
2	[0 , 0.59]
3	[0.6 , 0.69]
4	[0.7 , 0.84]
5	[0.85 , 1]

La evaluación de las competencias se calcula sumando las multiplicaciones de la evaluación del escenario por su peso, dentro de la competencia, ese valor se divide entre 5, ubicando el mismo en la escala del 0 al 1 como se explicó anteriormente. La siguiente ecuación muestra este proceso de forma más clara (León, 2015).

$$Evaluación\ Competencia = \frac{Evaluación\ Escenario_1 * Peso\ Escenario_1 + \ Evaluación\ Escenario_n * Peso\ Escenario_n}{5}$$

La evaluación final del rol se calcula sumando las multiplicaciones de la evaluación de las competencias por su peso dentro del rol, de forma similar a la evaluación de las competencias, como se muestra en la siguiente ecuación (León, 2015).

$$Evaluación\ Rol = \frac{Evaluación\ Competencia_1 * Peso\ Competencia_1 + \ Evaluación\ Competencia_n * Peso\ Competencia_n}{5}$$

El resultado de esta ecuación se ubica en los intervalos mencionados, lo cual dará la evaluación del rol en cuestión. En la siguiente imagen se evidencia cómo el sistema muestra al estudiante, el posible nivel que puede alcanzar en el rol, según las tareas que tenga evaluadas (León, 2015).

Imagen 5 *Barra de progreso para el nivel avanzado del rol (León, 2015)*

2.2.1 Actores del sistema

Se define como actores del sistema, a los estudiantes que se incorporan a la asignatura PID en el segundo semestre de tercer año en adelante y a los profesores y especialistas de la universidad. En la siguiente tabla se definen los actores del sistema.

Tabla 5 *Actores relacionados con el sistema (fuente: elaboración propia)*

Actores relacionados con el sistema	Descripción
Estudiante	Se encarga de incorporar evidencias relacionadas con las tareas asignadas en el sistema, para su posterior procesamiento. Es el encargado de realizar una solicitud para certificar un rol.
Profesor PID	Se encarga de subir el plan de formación correspondiente a los estudiantes que atiende. Es el encargado de evaluar cada una de las tareas asignadas a los estudiantes y evalúa cada tarea junto con la evidencia que le corresponde.
Administrador del sistema	Se encarga de administrar el sistema, con privilegios y credenciales para realizar cualquier cambio en la aplicación

2.3. Identificación de requisitos

Los requisitos para un sistema son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos se clasifican en dos grupos: requisitos funcionales y requisitos no funcionales. El proceso de descubrir, analizar, documentar y verificar estos servicios y restricciones se denomina ingeniería de requisitos (Sommerville, 2005), la cual proporciona el mecanismo apropiado para entender lo que el cliente quiere, analizar las necesidades, evaluar la factibilidad, negociar una solución razonable, especificar la solución sin ambigüedades, validar la especificación y administrar los requisitos (Gómez, 2011).

2.3.1 Requisitos funcionales

Los requerimientos funcionales de un sistema describen las funcionalidades con que el sistema debe contar, describen la función de este, sus entradas, salidas y excepciones. A continuación, se especifican los requisitos con que debe contar el sistema.

Tabla 6 *Requisitos funcionales (fuente: elaboración propia)*

No.	Historia de usuario	Prioridad para el cliente
1	Importar plan de formación	Alta
2	Evaluar tarea	Alta
3	Asignar tarea	Media
4	Generar alerta	Alta
5	Evaluar escenario	Media
6	Evaluar competencia	Media
7	Evaluar rol	Media
8	Visualizar desempeño del estudiante	Alta
9	Mostrar gráfico para tareas	Alta
10	Mostrar gráfico de escenarios por competencia	Alta
11	Mostrar gráfico de competencias por rol	Alta

Para identificar los requisitos funcionales se empleó la técnica de entrevista (Anexo 1), la cual permitió hacer el procedimiento de forma más adecuada. Los requisitos no funcionales con los que debe cumplir el sistema son los definidos por el equipo de desarrollo en el curso 2014-2015 en la tesis de diploma titulada Sistema de Gestión de Certificación de Roles 2.0.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales, como su nombre indica, son requisitos que no se refieren directamente a las funciones específicas que proporciona el sistema (Sommerville, 2005). Estos requisitos especifican el comportamiento del producto. A continuación, se mencionan los requisitos determinados por la fase anterior de desarrollo del sistema:

- ✓ **RNF1 Confiabilidad**

  El sistema debe ser preciso con la información que maneja y le proporciona al usuario, haciendo énfasis en los resultados de los análisis que ejecutará, para evitar errores que puedan incidir negativamente en la toma de decisiones.

- ✓ **RNF2 Seguridad**

  La seguridad del sistema será gestionada mediante la asignación de credenciales para mantener la integridad de los datos en función del nivel de acceso asignado.

- ✓ **RNF3 Software(Cliente)**

  Las computadoras (PC) clientes deben tener instalado un navegador web Mozilla Firefox 20.0 o superior o Google Chrome 20.0 o superior.

- ✓ **RNF4 Software(Servidor)**

  - ✓ **RNF4.1** El servidor debe poseer el Sistema Operativo GNU/Linux o Windows Seven o superior.
  - ✓ **RNF4.2** Se debe contar con un servidor web Apache versión 2.0 o superior.

- ✓ **RNF5 Hardware(cliente)**

  - ✓ **RNF5.1** Las PC clientes deben poseer requisitos mínimos como: Procesador 1ghz, con 512 de RAM.
  - ✓ **RNF5.2** Cada PC cliente debe tener conexión de red.

- ✓ **RNF6 Hardware(servidor)**

  - ✓ **RNF6.1** Se debe disponer de un servidor con requisitos mínimos como:

Pentium Core-2-Duo o AMD Athlon 64 x2, con 2GB de RAM

- ✓ **RNF6.2** Se debe disponer de un disco duro de 380 GB o superior.
- ✓ **RNF6.3** El servidor debe tener conexión de red.

2.4. Planificación

En esta fase el cliente establece la prioridad de cada historia de usuario (HU), y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas (Letelier, 2006). Al finalizar el equipo de desarrollo cuenta con suficiente información para hacer una primera entrega. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas y tecnologías que serán utilizadas en el proyecto.

Uno de los artefactos que se generan con el uso de la metodología de desarrollo XP para la especificación de requisitos del software y las características del sistema son las HU.

2.4.1 Historias de usuario

Las HU son una manera simple de describir una tarea concisa que aporta valor al usuario o al negocio. No se detalla más hasta el momento que la HU se vaya a desarrollar. Las HU pueden ser creadas durante las conversaciones con los usuarios interesados (stakeholders) sobre nuevas funcionalidades o mejoras del proyecto (Letelier, 2006).

Estas HU siguen los principios básicos de las metodologías ágiles, potencian la participación del equipo en la toma de decisiones, se crean y evolucionan a medida que el proyecto avanza, son peticiones pequeñas y concretas, contienen la información imprescindible y apoyan la cooperación, colaboración y conversación entre los miembros del equipo, lo cual es fundamental (Sánchez, 2016).

En el Anexo 2 se reflejan todas las HU referentes al desarrollo del negocio. Las siguientes tablas muestran dos ejemplos de HU diseñadas para el desarrollo de los requisitos que debe cumplir el sistema.

Tabla 7 *HU Generar alerta (fuente: elaboración propia)*

Historia de usuario	
Número: 4	**Nombre historia de usuario:** Generar alerta
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana
Descripción: • Mediante correo se le envía al profesor un registro con las evaluaciones del estudiante y la clasificación del plan de formación según los escenarios que tenga cubierto el plan de formación para el rol que se desea.	
Observaciones: Debe haberse importado el plan de formación y tener al menos una evaluación para que el sistema envíe el registro de evaluaciones y la clasificación de dicho plan.	

Tabla 8 *HU Visualizar desempeño del estudiante (fuente: elaboración propia)*

Historia de usuario	
Número: 8	**Nombre historia de usuario:** Visualizar desempeño del estudiante
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana

Descripción:
• El sistema muestra gráficos de desempeño para los distintos niveles de granularidad en la certificación de rol (tareas, escenarios, competencias).
Observaciones: Si el estudiante no tiene evaluaciones asignadas, el gráfico de competencias por rol cogerá los valores mínimos para cada una de las competencias.

En esta fase el cliente establece la prioridad que tendrá cada HU según sus necesidades más inmediatas, luego los programadores realizan una estimación del esfuerzo que se necesita para cada una de ellas. La planificación es una fase corta, en la que el cliente y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las HU y, asociadas a estas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entregas (release[11]). El cronograma fijado en la etapa de planeamiento se realiza en un número de iteraciones, cada una de ellas tarda de una a cuatro semanas de ejecución (Letelier, 2006).

La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar. Al planificar según alcance del sistema, se divide la suma de puntos de las HU seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación (Letelier, 2006).

2.4.2 Desarrollo de iteraciones

Esta es la fase principal en el ciclo de desarrollo de XP. Las funcionalidades son desarrolladas en esta etapa, generando al final de cada una, un entregable funcional que implementa las HU asignadas a la iteración. Como las HU no tienen suficiente detalle como para permitir su análisis y desarrollo, al principio de cada iteración se

[11] **Release:** Es un ciclo desde la entrevista con el usuario hasta la obtención de una solución (Beck, 2001).

realizan las tareas necesarias de análisis, recabando con el cliente todos los datos que sean necesarios. El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo. Las iteraciones son también utilizadas para medir el progreso del proyecto.

Una vez definidas las HU y estimado el esfuerzo propuesto para la realización de cada una de ellas, se distribuyó la realización del sistema en dos iteraciones, las cuales se describen a continuación de manera más detallada:

Iteración I: esta iteración tiene como objetivo realizar las HU de prioridad alta referentes a importar plan de formación, evaluar tarea y generar alerta, y la HU de prioridad media referente a asignar tarea.

Iteración II: esta iteración tiene como objetivo realizar las HU de prioridad media referentes a evaluar escenarios, evaluar competencias y evaluar rol, y las HU de prioridad alta referentes a visualizar desempeño del estudiante, mostrar gráfico para tareas, mostrar gráfico escenarios por competencias, mostrar gráfico competencias por rol.

Después de realizada la estimación del esfuerzo y el plan de iteraciones, y continuando los pasos que propone XP, se crea el plan de duración de las iteraciones, así como el orden en que serán implementadas cada una de las HU, según la prioridad asignada por el cliente.

Tabla 9 *Plan de duración de las iteraciones (fuente: elaboración propia)*

Iteración	Historias de usuario	Duración total de las iteraciones (semana).
Iteración I	HU (1-4)	2
Iteración II	HU (5-11)	2
Total	11	4

Luego del plan de duración de las iteraciones, se presenta el plan de entregas para la fase de implementación. Atendiendo al mismo se harán entregas del sistema al finalizar cada iteración en la fecha aproximada que se indica en la siguiente tabla:

Tabla 10 *Duración del plan de iteraciones (fuente: elaboración propia)*

Iteración	Fecha de entrega
Iteración I	5 de mayo 2016
Iteración II	28 de mayo de 2016

2.5. Diseño de la solución

La metodología XP sugiere que hay que conseguir diseños simples y sencillos. Hay que realizarlo lo menos complicado posible, para conseguir un diseño entendible y de fácil implementación, que a la larga costará menos tiempo y esfuerzo desarrollar. Como parte de esta fase, se define el patrón arquitectónico a utilizar para solucionar problemas de arquitectura, y se precisan los patrones de diseño que se van a emplear para remediar problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño.

2.5.1 Arquitectura del sistema

La arquitectura de la extensión del Sistema de Gestión de Certificación de Roles 2.0, está basada en el patrón arquitectónico MVC, el cual separa la lógica de negocio (el modelo), del controlador y la presentación (la vista), por lo que se consigue una mayor organización y un mantenimiento más sencillo de la aplicación. En la **Imagen 6** se muestra la estructura de este patrón arquitectónico.

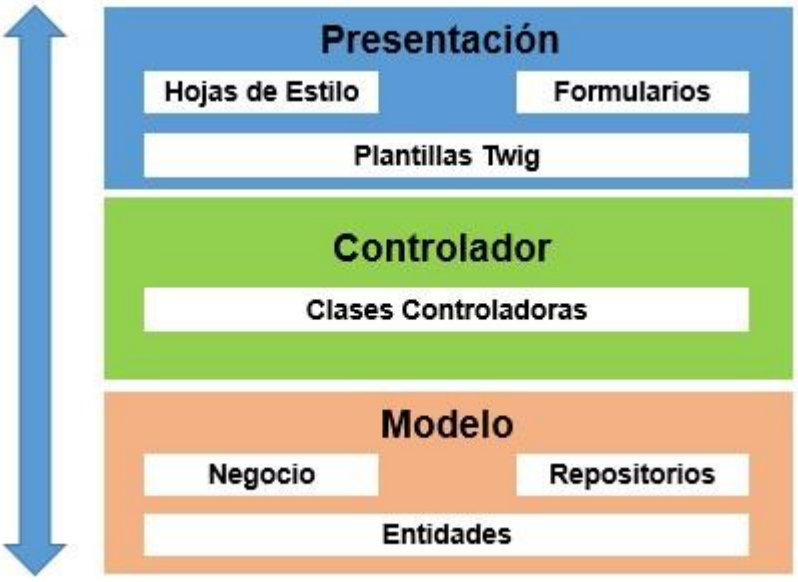

Imagen 6 *Arquitectura del sistema (fuente: elaboración propia)*

La arquitectura del sistema en el nivel superior, o sea la capa de presentación, encapsula las interfaces de usuario representadas por las clases twig, hojas de estilo, funciones JavaScript y formularios necesarios para la interacción con el cliente. La capa controlador contiene las clases controladoras que se encargan de dar respuesta a las peticiones realizadas por el usuario. La capa de negocio contiene las clases de negocio, que tienen como principal función separar la lógica de negocio del resto de la aplicación y obtener una mayor reutilización del código. Esta capa se encarga de recibir las peticiones, procesar la información, hacer pedidos a las clases Repositorio y devolver respuestas a las clases Controladoras, las que a su vez las envían a las clases enmarcadas en la capa vista.

Inicialmente el usuario interactúa con la interfaz (Vista). El controlador recibe la petición de la acción solicitada y gestiona el evento. El controlador accede al modelo actualizándolo o buscando la información requerida. El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario y mostrar los datos del modelo para generar la interfaz apropiada. (Ver **Imagen 7**)

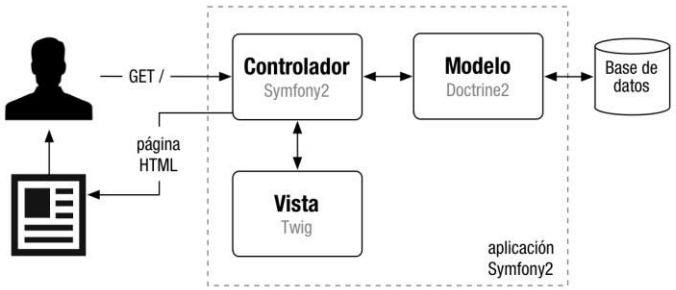

Imagen 7 *Patrón MVC (Eguiluz, 2012)*

A continuación, se muestra la estructura de carpetas del proyecto, siguiendo el patrón arquitectónico MVC obtenido con la ayuda del marco de trabajo Symfony2.

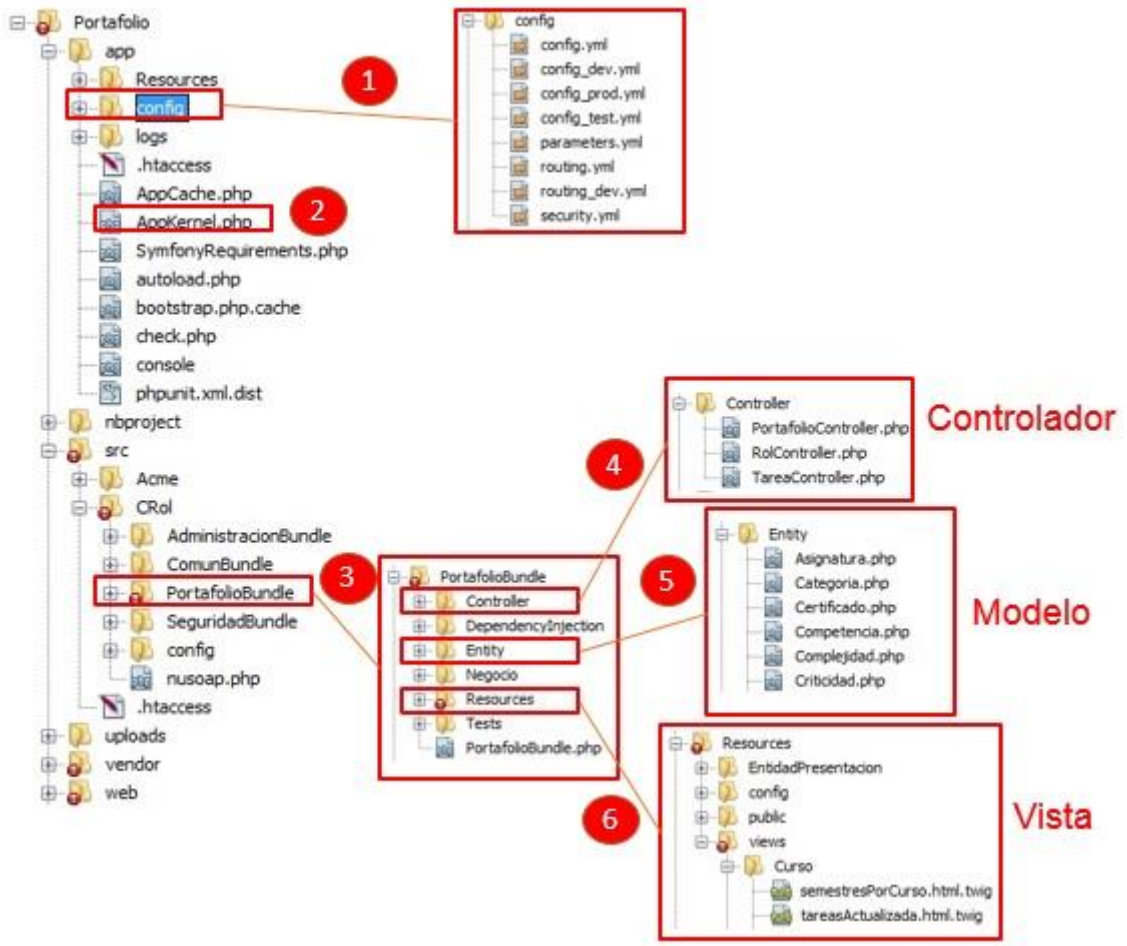

Imagen 8 *Estructura de clases (fuente: elaboración propia)*

Symfony 2 posee un controlador frontal el cual es el encargado de crear el kernel de la aplicación mediante una instancia de la clase **(2)** y es la responsable de toda la configuración. Ofrece las rutas de los módulos o Bundles que el usuario necesita y que se encuentran en la carpeta config **(1).** Es el núcleo de Symfony 2 y por tanto uno de los componentes fundamentales para su correcto funcionamiento. En la carpeta src se localiza el proyecto llamado Crol, nombre que se deriva de Certificar un Rol y que identifica la apliación. El sistema está organizado en cuatro bundles (módulos, según su denominación en inglés): Administración, Común, Portafolio y Seguridad, los cuales se pueden observar dentro de la estructura de clases mostrada anteriormente. Los módulos poseen una organización común, la cual se muestra en **(3)**, dentro de la cual se localizan las carpetas con los componentes específicos de la arquitectura:

- ✓ Controller **(4)**: posee los ficheros con los códigos de las clases Controladoras.

- ✓ Entity **(5)**: en la carpeta Entity se encuentran las Entidades del Modelo y en la

carpeta Negocio se encuentran las clases que contienen los métodos que acceden a los datos de la base de datos.

- ✓ Resources **(6):** dentro de esta se encuentran las carpetas con los CSS, los JS y las TWIG que conforman la Vista.

2.5.2 Tarjetas Clases, Responsabilidad y Colaborador (CRC)

Las tarjetas CRC son la base para la obtención del modelo entidad relación. Cada tarjeta se convirtió en objeto, sus responsabilidades en métodos públicos y sus colaboradores en llamados a otras clases. Las tarjetas CRC constituyen una primera aproximación a los objetos que luego se van a utilizar en el desarrollo del sistema. En XP el proceso de diseño es iterativo por lo que la creación de las tarjetas no es en un mismo tiempo, se crean según las iteraciones y se le añaden responsabilidades y colaboradores según se haga necesario (Casas, 2009). Estas tarjetas se dividen en tres secciones que contienen la información del nombre de la clase, sus responsabilidades y sus colaboradores. A continuación, se muestran las tarjetas CRC confeccionadas durante la fase de diseño (ver **Tablas** de la **11** a la **14**).

Tabla 11 *Tarjeta CRC número 1 (fuente: elaboración propia)*

PortafolioController	
Responsabilidad	**Colaboradores**
Evaluar Desempeño	PortafolioGtr
Gráfico Rol	PersonaGtr
Gráfico Escenarios	SeguridadGtr
Gráfico Tareas	RolGtr

Tabla 12 *Tarjeta CRC número 2 (fuente: elaboración propia)*

PortafolioGtr	
Responsabilidad	**Colaboradores**
Listar Competencias	Competencia
Obtener Portafolio	Portafolio
Obtener Tareas por Usuario	Tarea

Tabla 13 *Tarjeta CRC número 3 (fuente: elaboración propia)*

TareaController	
Responsabilidad	**Colaboradores**
Evaluar Tarea	Persona
	Estudiante
	Usuario
	Portafolio
	Expediente
	TareaGtr
	RolGtr

Tabla 14 *Tarjeta CRC número 4 (fuente: elaboración propia)*

RolGtr

Responsabilidad	**Colaboradores**
Buscar Roles por Usuario	Rol
Obtener Tipo Competencia	Tipo Competencia
Obtener Competencia por Tipo de competencia	Competencia

2.5.3 Patrones de diseño

Los patrones de diseño son herramientas que proveen facilidades para crear un software reutilizable de buena calidad. Cada patrón describe un problema que ocurre repetidamente en nuestro entorno, y describe el núcleo de la solución a ese problema, de tal forma que ésta pueda ser usada un millón de veces, sin hacer el mismo trabajo dos veces (González, 2003). A continuación, se muestran los patrones utilizados en la solución:

Patrones de diseño GRASP

- ✓ **Experto:** El uso de este patrón se evidencia en las clases del negocio y del modelo, que poseen funciones concretas de acuerdo con la información que gestionan.
- ✓ **Creador:** Este patrón se evidencia en la clase Tarea, la cual es la única que puede crear objetos de tipo Evidencia.
- ✓ **Controlador:** Se evidencia en las clases de negocio, entre ellas la clase TareaGtr que maneja todas las peticiones relacionadas con las tareas.

Patrones de diseño GoF

Patrones Estructurales: Los patrones estructurales se ocupan, de cómo las clases y objetos se combinan para formar grandes estructuras y proporcionar nuevas funcionalidades.

- ✓ **Fachada**: provee de una única interfaz para acceder a un sistema completo, que actúa como único punto de acceso al mismo, y hace que éste sea más fácil de utilizar. Ejemplo de esto se evidencia en la clase PortafolioGtr que es la encargada del acceso a los datos de la entidad Tarea en la base de datos.

- ✓ **Decorador:** es aplicado a la generación de vistas, la solución que ofrece dicho patrón es añadir funcionalidades adicionales a las plantillas. Por ejemplo, añadir el menú y el pie de página a las plantillas que lo requieran. Se trata de decorar las plantillas con elementos adicionales reutilizables.

2.6 Codificación de la solución

En la fase actual de desarrollo de la extensión del sistema, se genera todo el código fuente necesario para satisfacer las HU definidas para la solución. Iniciando cada iteración se realiza una revisión del plan de iteraciones y se modifica de ser necesario.

Para llevar a cabo la correcta implementación de las HU se deben definir por parte del equipo de desarrollo las Tareas de Ingeniería (TI) que se realizan en cada una de las iteraciones. Las TI también conocidas como tareas de implementación permiten a los desarrolladores obtener un nivel de detalle más avanzado que el que propicia las HU. En las tablas siguientes se describen las TI más importantes pertenecientes a la primera y segunda iteración respectivamente (ver **Tablas 15** y **16**). En el Anexo 3 quedan plasmadas las demás TI que se debieron realizar para el desarrollo del proyecto.

Tabla 15 *TI Generar alerta (fuente: elaboración propia)*

Tarea de ingeniería	
Número:4	**Nombre historia de usuario:** Generar alerta
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Media	**Puntos estimados:** 1 día
Riesgo en desarrollo: Bajo	**Puntos reales:** 1 día
Descripción: se debe implementar la funcionalidad generar alerta, la cual después de evaluada una tarea, se le notifica al profesor mediante un correo con el registro de evaluaciones y el nivel para el cual está diseñado el plan de formación del estudiante	

evaluado.

Tabla 16 *TI Mostrar gráfico competencias por rol (fuente: elaboración propia)*

Tarea de ingeniería	
Número:11	**Nombre historia de usuario:** Mostrar gráfico competencias por rol.
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: se debe implementar la funcionalidad mostrar gráfico competencias por rol, la cual muestra el desempeño de un estudiante en el rol asignado.	

Para una correcta comprensión y ejecución de la codificación resulta imprescindible el uso de estándares de codificación.

2.6.1 Estándares de codificación

Los estándares de codificación son aquellos que permiten entender de manera rápida y sencilla el código empleado en el desarrollo de un software. Garantizan el mantenimiento óptimo de dicho código por parte del programador (Arias, 2009). A continuación, se muestran algunas pautas del estándar definido por el equipo de desarrollo, así como ejemplos de su uso.

1. Todas las nomenclaturas a utilizar se definen en idioma español.

2. Los identificadores para las variables y los parámetros se escriben con letras en minúsculas y en caso de ser un nombre compuesto se divide cada palabra con

un guión bajo.

```
$evaluacion_competencias=[];
$evaluacion_escenario=[];
```

3. En caso de que los métodos se nombren con una sola palabra, esta se escribe en minúsculas. En caso de ser un nombre compuesto, las palabras que lo conforman se escriben juntas, de la segunda en adelante se escriben con letra inicial mayúscula. Se emplea la notación variante LowerCamelCase.

```
public function graficoEscenariosAction(Request $request)
```

4. Los nombres de las clases se escriben con la primera letra de cada palabra que lo compone en mayúscula, haciendo uso de la notación variante UpperCamelCase.

```
class PortafolioController extends BaseController
```

5. Se hizo uso de llaves para ganar en comprensión del código

```
foreach ($roles as $rol) {
    $temp['id'] = $rol->getId_rol();
    $temp['nivel'] = $rol->getNivel();
    $temp['tipo'] = $rol->getTipo_rol()->getDescripcion();
    $temp['usuario'] = $rol->getUsuario()->getIdUsuario();
    $result[] = $temp;
}
```

Los estándares de codificación permitieron establecer un estilo de programación homogéneo permitiendo que cualquier persona que consulte el código lo pueda entender en menos tiempo.

2.7 Conclusiones parciales

Finalizado el desarrollo del presente capítulo, se arriba a las siguientes conclusiones:

1. La obtención de requisitos funcionales y no funcionales, permitió definir el comportamiento y restricciones del sistema para su implementación.

2. Después de haber realizado el análisis del sistema, quedaron definidas las historias de usuario, proporcionando una comprensión detallada de las funcionalidades con que debe cumplir el sistema.

3. La utilización de estándares de codificación en la implementación de la solución propició mayor organización, comprensión y menor tiempo en el mantenimiento del sistema.

CAPÍTULO III. Validación de la propuesta de solución

3.1 Introducción

Siempre que se desarrolla un sistema informático se deben realizar las pruebas necesarias para asegurar que la aplicación cumpla con las especificidades y calidad requeridas. El objetivo de este proceso es valorar y mejorar el sistema durante su desarrollo y al finalizar el mismo.

En el presente capítulo se verifica el correcto funcionamiento de las funcionalidades desarrolladas en el sistema. La verificación se refiere al conjunto de actividades que aseguran que el software implementa correctamente una función específica. Este proceso de verificación se debe realizar por cada iteración de implementación, para asegurar el correcto funcionamiento del software. Esta fase plantea la realización de los métodos de prueba de caja negra y caja blanca, los cuales se describirán en el presente capítulo.

Finalmente se realizó la validación del sistema. En esta fase se ejecutaron las pruebas de aceptación, que aseguran que las funcionalidades desarrolladas se ajustan a los requisitos del cliente.

3.2 Verificación del sistema

Las pruebas de verificación del sistema tienen como objetivo valorar y mejorar la calidad de los productos del trabajo generado durante el desarrollo y modificación del software. Según (Pressman, 2000), verificación es el conjunto de actividades que aseguran que el software implemente correctamente una función específica.

A continuación, se aplican las técnicas para este objetivo, las cuales se clasifican en dos métodos de prueba, el de caja blanca y el de caja negra.

3.2.1 Método de prueba de caja blanca

Para aplicar el método de pruebas de caja blanca se empleó la técnica del camino básico. Esta permitió obtener una medida de la complejidad lógica para el diseño de los casos de pruebas y usar esta medida como guía para la definición de un conjunto básico de caminos de ejecución. Se tomó como ejemplo el método evaluarTarea, perteneciente a la clase TareaController. Para ello se procede a enumerar las sentencias del código y a partir del mismo se construye el grafo de flujo asociado.

```
1  $evalString='';
   $nivel='';
2  foreach($evaluacion_rol as $eval){
3      if($eval<0.6){
4          $nivel='No certifica';
           $evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion().'\\n';
5      }else if($eval>=0.6 && $eval<0.7){
6          $nivel='Básico';
           $evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion().'\\n';
7      }else if($eval>=0.7 && $eval<=0.84){
8          $nivel='Intermedio';
           $evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion().'\\n';
9      }else if($eval>=0.85){
10         $nivel='Avanzado';
           $evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion().'\\n';
       }
11 }
12 return new Response(1);
```

Imagen 9 *Código fuente del método evaluarTarea (fuente: elaboración propia)*

A partir del código que se muestra en la **Imagen 9**, se obtiene el flujo básico del mismo:

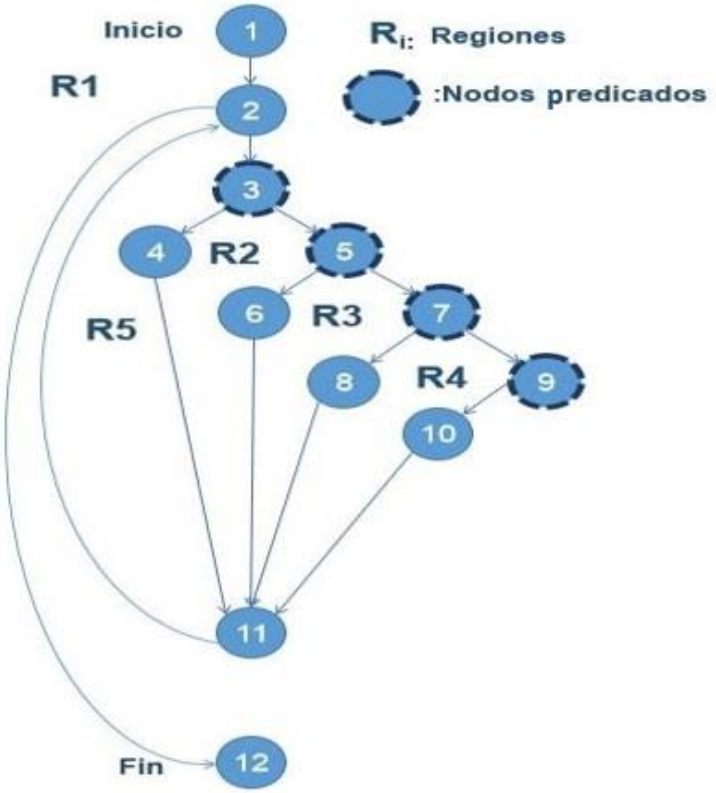

Imagen 10 *Grafo de flujo para el método evaluarTarea (fuente: elaboración propia)*

Luego de haber realizado la construcción del grafo de flujo se procede a calcular la complejidad ciclomática mediante la siguiente fórmula:

En la fórmula $\boldsymbol{V(G)}$ representa el valor del cálculo de la complejidad ciclomática.

$$\boldsymbol{V(G) = (A - N) + 2}$$

Donde $\boldsymbol{A}$ es el número de aristas y $\boldsymbol{N}$ la cantidad de nodos presentes en el grafo.

$$\boldsymbol{V(G) = (15 - 12) + 2}$$

$$\boldsymbol{V(G) = 5}$$

El número de regiones del grafo es igual a la complejidad ciclomática.

El cálculo efectuado anteriormente dio como resultado que, la complejidad ciclomática es de cinco. Este valor indica que existen cinco posibles caminos por donde el flujo puede circular. Este número de caminos básicos determina el número de casos de prueba que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez. Como resultado de la aplicación de la técnica se obtuvieron los siguientes caminos básicos por los que puede transitar el flujo:

- ✓ **Camino 1:** 1 – 2 – 12
- ✓ **Camino 2:** 1 – 2 – 3 – 4 – 11 – 2 – 12
- ✓ **Camino 3:** 1 – 2 – 3 – 5 – 6 – 11 – 2 – 12
- ✓ **Camino 4:** 1 – 2 – 3 – 5 – 7 – 8 – 11 – 2 – 12
- ✓ **Camino 5:** 1 – 2 – 3 – 5 – 7 – 9 – 10 – 11 – 2 – 12

Luego de establecidos los caminos básicos se procede a realizar los casos de prueba para cada uno de ellos, de forma que los datos introducidos provoquen que se visiten las sentencias vinculadas a cada nodo del camino. En las tablas siguientes se muestran los cinco casos de pruebas realizados a esta funcionalidad (ver **Tablas 17** y **18**). En el Anexo 4 se muestran los demás casos de prueba realizados al método evaluar tarea.

Tabla 17 *Caso prueba para el camino 4 (fuente: elaboración propia)*

Entrada	Debe existir un portafolio para el estudiante para el cual se desea saber el alcance del dicho portafolio.
Resultados esperados	Le envía una notificación al estudiante sobre el posible nivel que puede alcanzar con el portafolio actual, el cual debe estar dado por el valor Intermedio.

Condiciones	foreach($evaluacion_rol as $eval){
	
	else if($eval>=0.7 && $eval<=0.84){
	
	$nivel='Intermedio';
	
	$evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion();
	
	}
	
	}
	
	$notificacion = new Notificacion();
	
	$emitida_por = $em->getRepository('SeguridadBundle:Usuario')->findOneBy(array('id_usuario' => $usuario->getIdUsuario()));
	
	$notificacion->setDescripcion('El plan de formación para el estudiante '.$usuario->getPersona()->getNombre().' '.$usuario->getPersona()->getApellidos().' está diseñado para alcanzar el nivel '.$evalString);
	
	$notificacion->setEmitidoPor($emitida_por);
	
	$tipo_notificacion = $em->getRepository('ComunBundle:TipoNotificacion')->findOneBy(array('id_tipo_notificacion' => Util::not_inf));
	
	$notificacion->setTipoNotificacion($tipo_notificacion);

	$notificacion->setPortafolio($this->getPortafolioGtr()->obtenerPortafolio($id_portafolio)); $this->getPortafolioGtr()->salvarActualizar($notificacion);

Tabla 18 *Caso prueba para el camino 5 (fuente: elaboración propia)*

Entrada	Debe existir un portafolio para el estudiante para el cual se desea saber el alcance del dicho portafolio.
Resultados esperados	Le envía una notificación al estudiante sobre el posible nivel que puede alcanzar con el portafolio actual, el cual debe estar dado por el valor Avanzado.

Condiciones	foreach($evaluacion_rol as $eval){
	
	else if($eval>=0.85){
	
	$nivel='Avanzado';
	
	$evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion();
	
	}
	
	}
	
	$notificacion = new Notificacion();
	
	$emitida_por = $em->getRepository('SeguridadBundle:Usuario')->findOneBy(array('id_usuario' => $usuario->getIdUsuario()));
	
	$notificacion->setDescripcion('El plan de formación para el estudiante '.$usuario->getPersona()->getNombre().' '.$usuario->getPersona()->getApellidos().' está diseñado para alcanzar el nivel '.$evalString);

```
$notificacion->setEmitidoPor($emitida_por);

$tipo_notificacion = $em-
>getRepository('ComunBundle:TipoNotificacion')-
>findOneBy(array('id_tipo_notificacion' => Util::not_inf));

$notificacion->setTipoNotificacion($tipo_notificacion);

$notificacion->setPortafolio($this->getPortafolioGtr()-
>obtenerPortafolio($id_portafolio));

$this->getPortafolioGtr()->salvarActualizar($notificacion);
```

Una vez ejecutados todos los casos de pruebas obtenidos a través de la aplicación de la técnica camino básico, se concluye que los mismos fueron probados satisfactoriamente demostrando que el código generado no presenta ciclos infinitos y no existe código innecesario en el sistema desarrollado.

3.2.2 Método de prueba de caja negra

Para la realización de las pruebas de caja negra a este nivel de unidad se empleó la técnica partición de equivalencia que garantizó efectividad al examinar los valores válidos e inválidos de las entradas existentes en el software. En la **Tabla 19** se brinda un ejemplo de un caso de prueba de uno de los requisitos que componen la funcionalidad mostrar gráfico de competencias por rol.

Tabla 19 *Caso de prueba para el requisito 9 (fuente: elaboración propia)*

Escenario	Descripción	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Seleccionar competencia	Permite seleccionar una competencia	V La competencia tiene	El sistema debe mostrar el gráfico de competencias por rol	Desempeño/Seleccionar Estudiante/Seleccionar Rol

	presente en el gráfico	escenarios evaluados.		
EC 1.2 Seleccionar competencia	Permite seleccionar una competencia presente en el grafo	F La competencia no tiene escenarios evaluados	El sistema debe mostrar un mensaje de error para indicar que el estudiante seleccionado no tiene rol asignado.	Desempeño/Seleccionar Estudiante

En la **Tabla 20** se muestra un ejemplo de campo de entrada presente en las funcionalidades desarrolladas.

Tabla 20 *Descripción de variables (fuente: elaboración propia)*

No	Nombre del campo	Clasificación	Valor nulo	Descripción
1	Seleccionar rol	Campo de selección	No	El campo se muestra en caso de que el estudiante seleccionado tenga asociado un rol a certificar

Luego de aplicar las pruebas de caja negra por el Grupo de Calidad del Centro de Gobierno Electrónico (CEGEL) se obtuvieron los siguientes resultados:

Las no conformidades encontradas en la primera iteración estaban relacionadas con errores de interfaz, errores ortográficos y error de funcionalidad. Se pudo comprobar en la segunda iteración realizada que estos errores fueron corregidos y que la aplicación desarrollada funciona correctamente (ver **Imagen 11**)

Imagen 11 *Gráfico de no conformidades (fuente: elaboración propia)*

Después de concluidas las pruebas, el grupo de calidad CEGEL liberó la aplicación Extensión del Sistema de Gestión para la Certificación de Roles v2.0, entregándose al equipo de desarrollado el Acta de Liberación Interna de Productos del Software (Anexo 6) en la que consta que la aplicación está apta para ser utilizada.

3.3 Validación del sistema

Según (Pressman, 2000) la validación es un conjunto de diferentes actividades que aseguran que el software construido corresponde y satisface los requisitos del cliente. Con la validación del sistema se pretende comprobar que el software cumple las expectativas del cliente. Para llevar a cabo la validación de la aplicación se aplicaron las pruebas de aceptación por cada HU definida en la etapa de planificación.

3.3.1 Pruebas de aceptación

Las pruebas de aceptación son destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida por el cliente final. Estas pruebas aseguran el comportamiento del sistema y especifican los aspectos a probar cuando una HU ha sido implementada correctamente.

A cada una de las HU se le realizaron pruebas de aceptación. Estas pruebas son reflejadas mediante casos de pruebas de aceptación, las cuales están conformadas por

ocho parámetros, que dan información acerca de la prueba realizada. Los parámetros a medir son:

- ✓ **Código:** muestra un identificador para cada prueba realizada (normalmente se pone el nombre del componente seguido de las letras PA: pruebas de aceptación).
- ✓ **Nombre de la historia de usuario:** indica el nombre de la prueba.
- ✓ **Nombre de la persona que realiza la prueba:** indica la persona que realiza la prueba de aceptación.
- ✓ **Descripción:** se describe cuál es la funcionalidad que se va a medir del componente al que se le esté realizando la prueba.
- ✓ **Condiciones de ejecución:** indica cuáles son las condiciones que se tienen que cumplir para que el componente realice correctamente la funcionalidad que se va a medir.
- ✓ **Entrada / Pasos de ejecución:** indica los pasos a seguir para realizar la prueba.
- ✓ **Resultados esperados:** muestra cuál es el resultado que se obtendría de un correcto funcionamiento de la prueba.
- ✓ **Evaluación de la prueba:** indica el estado de la prueba si es satisfactoria o insatisfactoria.

A continuación, se muestran los casos de pruebas de aceptación que se le realizaron a las HU de prioridad alta. Las seleccionadas para ser mostradas en el documento conforman el flujo base del sistema, correspondiente a generar alerta, donde se le envía al profesor la evaluación del plan de formación, y visualizar desempeño del estudiante para las competencias del rol a certificar (ver **Tablas 21** y **22**). En el Anexo 5 se reflejan los demás casos de prueba de aceptación que fueron ejecutados para la validación del sistema.

Tabla 21 *PA Generar alerta (fuente: elaboración propia)*

Caso de prueba de aceptación

Código caso de prueba: PA4-HU4	**HU:** Generar alerta
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema envía un correo al profesor PID con el registro de evaluaciones del estudiante al que se le ha evaluado una tarea, y el posible nivel que alcanza con el plan de formación que tiene asignado.	
Condiciones de ejecución: el estudiante debe tener un portafolio asociado para el cual debe tener tareas sin evaluaciones.	
Entrada / Pasos de ejecución: ✓ El usuario selecciona la pestaña registro de estudiantes ✓ El usuario selecciona el estudiante. ✓ El usuario selecciona la tarea a evaluar de las disponibles. ✓ El usuario selecciona la nota a asignar a la tarea. ✓ El usuario selecciona la competencia a la cual asociar la tarea. ✓ El usuario selecciona el escenario al cual está asociada la tarea en el portafolio. ✓ El usuario selecciona si la tarea aporta al mérito científico o no. ✓ El usuario acepta la nota asignada.	
Resultado esperado: el sistema notifica al estudiante del posible nivel que puede alcanzar con el portafolio que tiene asignado y le envía un correo al profesor PID con el registro de evaluaciones y el posible nivel que alcanza el estudiante con el plan de formación que tiene asignado.	
Evaluación de la prueba: satisfactoria	

Tabla 22 *PA Gráfico competencias por rol (fuente: elaboración propia)*

Caso de prueba de aceptación	
Código caso de prueba: PA11-HU11	**HU:** Mostrar gráfico competencias por rol
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que se visualiza el gráfico competencias por rol.	
Condiciones de ejecución: el estudiante debe tener un plan de formación asignado.	
Entrada / Pasos de ejecución: ✓ El usuario selecciona la pestaña Desempeño. ✓ El usuario selecciona el estudiante. ✓ El usuario selecciona el rol para el que desea ver sus competencias.	
Resultado esperado: el sistema visualiza el gráfico Competencias por rol el cual refleja el desempeño del estudiante.	
Evaluación de la prueba: satisfactoria	

Después de aplicadas las pruebas de aceptación, la Extensión Sistema de Gestión de Certificación de Roles 2.0 cuenta con todas las funcionalidades que requería el cliente final del mismo, el cual emite el Acta de Aceptación (Anexo 7), en la que consta que la aplicación está apta para ser utilizada.

3.4 Conclusiones parciales

Al concluir el presente capítulo se evidencia que:

1. Después de aplicada la técnica de camino básico como método de prueba de caja blanca, se aseguró la calidad del código entregado, detectando las fallas tempranamente en el desarrollo.

2. Con la aplicación de la técnica de partición de equivalencia como método de prueba de caja negra se logró obtener un software operativo.

3. Con la aplicación de las pruebas de aceptación se comprobó que el sistema cumple con las expectativas del cliente.

CONCLUSIONES GENERALES

Con la culminación de la presente investigación, se concluye que:

1. El estudio realizado sobre los referentes teóricos para el objeto y el campo de investigación propició enmarcar la tesis en el mantenimiento perfectivo de software. Se asumió el desarrollo de un sistema de información que contribuya a la toma de decisiones sobre la base de la evaluación del plan de formación y la visualización del desempeño de los estudiantes en la ejecución de las tareas.

2. La selección de XP como metodología de desarrollo guió la realización del análisis y diseño de la solución. Se elaboraron un total de once historias de usuario, de ellas siete de prioridad alta y cuatro de prioridad media que fueron distribuidas en dos iteraciones para su ejecución exitosa.

3. La extensión del Sistema de Gestión de Certificación de Roles 2.0 mediante las herramientas y tecnologías seleccionadas contribuyó a la incorporación de un paquete de funcionalidades que le dieron solución a la problemática planteada en la presente investigación.

4. Mediante el diseño y aplicación de los casos de prueba se logró valorar los resultados y verificar que las funcionalidades cumplieran con las expectativas y necesidades del cliente.

BIBLIOGRAFÍA

1. **Achour, Mehdi y otros. 2014.** Manual de PHP. 2014.

2. **Aguilar, Lisandra. 2013.** Propuesta de optimización de la base de datos del proyecto Sistema de Informatización para la gestión de los Tribunales Populares Cubanos. La Habana : s.n., 2013.

3. **Arellano, Madelein. 2008.** Sistemas de información: ¿adecuación a los cambios tecnológicos o herramienta de gestión? Revista de Ciencias Sociales. [En línea] Marcaibo, 7 de Febrero de 2008. [Citado el: 10 de Junio de 2016.] http://www.scielo.org.ve/scielo.php?script=sci_arttext&pid=S1315-95182008000300008. ISSN 1315-9518.

4. **Arias, Manuel. 2009.** Estandares de codificación. [En línea] 7 de Mayo de 2009. [Citado el: 5 de Mayo de 2016.] http://www.cisiad.uned.es/carmen/estilo-codificacion.pdf..

5. **Bonanata, Maximiliano. 2013.** Programación y algoritmos. 2013.

6. **Campo, Gustavo Damián. 2009.** Patrones de Diseño, Refactorización y Antipatrones. Ventajas y Desventajas de su Utilización en el Software Orientado a Objetos. Buenos Aires : s.n., 2009.

7. **Canga, Manuel. 2015.** Trasweb. Optimización web. [En línea] 2015. [Citado el: 25 de mayo de 2016.] http://trasweb.net.

8. **Casas, Sandra, Reinaga, Héctor. 2009.** Aspectos tempranos: Un enfoque basado en tarjetas CRC. Argentina : s.n., 2009.

9. **Castillo, Cantón, Alejandro. 2012.** Manual de HTML5 en español. 2012.

10. **Cohen, Muñoz, Antonio, Ponjuán. 2012.** Eumed.net. Tipos de sistemas de información. [En línea] 3 de marzo de 2012. [Citado el: 11 de mayo de 2016.] http://www.eumed.net/libros-gratis/2012a/1169/tipos_de_sistema_de_informacion.html.

11. **Cristiá, Maximiliano. 2011.** Introducción a la ingeniería de requerimientos. [En línea] Agosto de 2011. [Citado el: 20 de Mayo de 2016.] http://www.fceia.unr.edu.ar/~mcristia/publicaciones/ingreq-a.pdf..

12. **Davenport, Thomas H., Prusak, Laurence. 2016.** Sinnexus Business Intelligence Informática Estratégica. [En línea] 2016. [Citado el: 5 de Mayo de 2016.] http://www.sinnexus.com/business_intelligence/piramide_negocio.aspx.

13. **Díaz, Daymara. 2005.** Toma de decisiones: el imperativo diario de la vida en la organización. Ciudad de la Habana : s.n., 2005.

14. **Eguiluz, Javier. 2012.** Desarrollo web ágil con Symfony 2. s.l. : easybook, 2012.

15. **Fernández, Juan Manuel. 2011.** Pruebas de Software. [En línea] 2011. [Citado el: 7 de Mayo de 2016.] http://www.uv.mx/personal/jfernandez/files/2010/07/8_Calidad.pdf.

16. **Fernández, María de las Mercedes, Ponjuán, Gloria. 2008.** Analisis conceptual de las principales interacciones entre la gestión de información, la gestión documental y a gestión del conocimiento. La Habana : Acimed, 2008.

17. **Fernández, Yenisleidy, Díaz, Yanette. 2012.** Patrón Modelo-Vista-Controlador. La Habana : Revista Digital de las Tecnologías de la Información y las Comunicaciones, 2012. ISSN 1729-3804.

18. **Gamma, Erich. 2002.** Patrones de Diseño: elementos de software orientado a objetos reutilizable. s.l. : Addison - Wesley, 2002.

19. **Gómez, María del Carmen. 2011.** Análisis de requerimientos. México : UNIVERSIDAD AUTONOMA METROPOLITANA, 2011. ISBN: 978-607-477-442-9.

20. **González, Diego Andrés, Ríos, Alejandro. 2003.** Patrones de Diseño. La Habana : s.n., 2003.

21. **Jeremy. 2014.** Características de Css3. [En línea] 2014. [Citado el: 3 de Mayo de 2016.] http://www.css3.com..

22. **Kabir, Mohammed J. 2003.** La Biblia del Servidor Apache 2.0. s.l. : Anaya Multimedia, 2003.

23. **León, Felinda, Rosabel,Maceo, Eliodanis. 2015.** Sistema de Gestión para la Certificación de Roles 2.0. La Habana : Tesis para optar por el título de Ingeniero en Ciencias Informáticas, 2015.

24. **Letelier, Patricio, Penadés, Mª Carmen. 2006.** Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). Valencia : s.n., 2006.

25. **Lincke, Rüdiger, Lundberg, Jonas y Löwe, Welf. 2009.** Comparing Software Metrics Tools. Suecia : s.n., 2009.

26. **Madrigal, Carlos Rafael, García, Frank Carlos. 2014.** Portafolio de evidencias para el proceso de certificación de roles. La Haban, Cuba : Tesis para optar por el título de Ingeniero en Ciencias Informáticas, 2014.

27. **Mos, Sílvia. 2015.** Per Tutatis. [En línea] 20 de 1 de 2015. [Citado el: 12 de mayo de 2016.] http://pertutatis.cat/que-son-los-sistemas-de-apoyo-a-la-toma-de-decisiones-dds/.

28. **Netbeans.org. 2016.** Netbeans.org. [En línea] 2016. [Citado el: 20 de Enero de 2016.] netbeans.org.

29. **Otto, Jacob Mark. 2014.** [En línea] 2014. [Citado el: 3 de Mayo de 2016.] http://getbootstrap.com/.

30. **Pacheco, Nacho. 2013.** Manual de Twig. 2013.

31. **Peña, Alejandro. 2006.** Tecnologías de la Información: Su alineamiento al Negocio de las Organizaciones. México : s.n., 2006. ISBN: 970-94797-5-X.

32. **Peralta, Manuel. 2011.** Ilustrados. [En línea] 2011. [Citado el: 24 de Abril de 2016.] http://www.ilustrados.com/tema/3351/Sistema-Informacion.html.

33. **Pérez, Aritz. 2011.** Los sistemas de información en las organizaciones. Bilbao : UPV-EHU, 2011.

34. **Pérez, Oscar, G. Ginestà, Marc. 2014.** Bases de datos en PostgreSQL. s.l. : UOC, 2014.

35. **Polo, Macario, Villafranca, Daniel. 2008.** Introducción a las aplicaciones web con Java. [En línea] 2008. [Citado el: 7 de Enero de 2016.] http://www.inf cr.uclm.es/www/mpolo/asig/0708/tutorJavaWeb.pdf..

36. **Potencier, Fabian. 2010.** Symfony 2.1 el libro oficial. 2010.

37. **Potencier, Fabian, Zaninotto, François. 2008.** Symfony, la guia definitiva. 2008.

38. **Pressman, Roger S. 2000.** Ingeniería de Software. Un Enfoque Práctico. 2000.

39. **RAE. 2014.** Diccionario de la lengua española. [En línea] 2014. [Citado el: 21 de junio de 2016.] http://dle.rae.es/?id=15CkAOi.

40. **Revista Vinculando. 2010.** Propuesta de una guía de métricas para evaluar el desarrollo de los Sistemas de Información Geográfica. [En línea] 4 de Enero de 2010. [Citado el: 3 de Mayo de 2016.] http://vinculando.org/articulos/sociedad_america_latina/propuesta_guia_de_medidas_para_evaluacion_sistemas_informacion.html.

41. **Rodríguez, Edgar Manuel. 2014.** MODELOS DE CALIDAD DE SOFTWARE. Santander : Universidad de Santander, 2014.

42. **Rodríguez, José Manuel, Daudero, María José. 2003.** Aspectos Técnicos y Legales. s.l. : Universidad de Sistemas de Información, 2003.

43. **Ruiz, Jenny, Aguilera, Oscar. 2006.** Sistema de apoyo a la toma de decisiones en la negociación comercial. Holguin : E-ISSN, 2006.

44. **Sánchez Fornaris, Maite y Alcantara Rabí, Dayanis. 2009.** Propuesta de una guía de métricas para evaluar el desarrollo de los Sistemas. 2009.

45. **Sánchez, Emilio A., Letelier, Patricio, Canós, José H. 2016.** Mejorando la gestión de historias de usuario en eXtrme Programming. Valencia : Universidad Politécnica de Valencia, 2016.

46. **Scalone, Fernanda. 2006.** Estudio comparativo de los modelos y estándares de calidad del software. Buenos Aires : s.n., 2006.

47. **Sinergia e Inteligencia de Negocio S.L. 2016.** Sinnexus Business Intellingence Infórmatica Estrategica. Sinnexus. [En línea] 2016. [Citado el: 28 de Abril de 2016.] http://www.sinnexus.com/business_intelligence/.

48. **Sommerville, Ian. 2005.** Ingenieria de Software. Madrid, España : PEARSON EDUCACION, S.A., 2005.

49. **Visconti, Marcello y Astudillo, Hernán. 2004.** Fundamentos de Ingeniería de Software. 2004.

50. **Whitten, Bentley, Dittman. 2004.** Eumed. [En línea] 2004. [Citado el: 27 de mayo de 2016.] http://www.eumed.net/ce/2012/ddb.html.

ANEXO

Anexo 1 Entrevista

Entrevista a especialistas del Sistema de Gestión de Certificación de Roles 2.0

Compañero se le solicita su contribución en el llenado de la presente encuesta como parte de la obtención de la información necesaria para el desarrollo de una Tesis de Diploma. En el Centro de Gobierno Electrónico se desarrolla el Sistema de Certificación de Roles 2.0 y su respuesta contribuirá notablemente a obtener una solución que se ajuste a las particularidades de este proceso.

Objetivo: identificar el mecanismo de funcionamiento del Sistema de Gestión de Certificación de Roles 2.0 para comprender la ponderación establecida por cada variable y la forma de evaluación de los distintos niveles de granularidad del sistema.

¿Cómo se establecieron los valores en la ponderación?

Los valores en la ponderación se establecieron a partir de entrevistas realizadas a los tribunales de certificación de roles.

¿Cuál es el rango de valores para establecer la evaluación del escenario, competencia y el rol?

- ✓ De 0 hasta 0,59 la evaluación es mal.
- ✓ De 0,6 hasta 0,69 la evaluación es regular
- ✓ De 0,7 hasta 0,84 la evaluación es bien
- ✓ De 0,85 hasta 1 la evaluación es excelente

¿Cómo se obtuvo el peso de los escenarios en la evaluación de las competencias?

Los valores para el peso de los escenarios en la evaluación de las competencias se obtuvieron a partir de entrevistas realizadas a los tribunales de certificación de roles.

¿Cómo se obtuvo el peso de la competencia para evaluar el rol?

Los valores para el peso de las competencias en la evaluación del rol se obtuvieron a partir de entrevistas realizadas a los tribunales de certificación de roles.

¿Cómo se evalúa el escenario a partir de las tareas almacenadas?

La evaluación de un escenario se calcula multiplicando el aporte de la tarea (2.5 si la evaluación de la tarea es 5, 2 si la evaluación de la tarea es 4, 1.5 si la evaluación de la tarea es 3 y 1 si la evaluación de la tarea es 2), la complejidad de la tarea (1 si complejidad es Alta, 0.8 si complejidad es Media, 0.4 si complejidad es Baja) y la criticidad de la tarea (1.5 si criticidad es Alta, 1.2 si criticidad es Media y 0.6 si criticidad es Baja). El resultado de la multiplicación del aporte, la complejidad y la criticidad se divide por 5, el cual dirá que evaluación va a tener el escenario, 2 si el valor es menor que 0.6, 3 si el valor esta entre 0.6 y 0.7, 4 si el valor esta entre 0.7 y 0.85, 5 si el valor es mayor que 0.85

¿Cómo se evalúa la competencia a partir de la evaluación de los escenarios?

La evaluación de las competencias se calcula sumando las multiplicaciones de la evaluación del escenario por su peso dentro de la competencia y ese valor se divide entre 5, ubicando ese valor en la escala del 0 al 1 como se explicó anteriormente.

¿Cómo se evalúa el rol a partir de la evaluación de las competencias?

Teniendo las evaluaciones asociadas a cada competencia del rol en cuestión, se multiplican por sus pesos respectivos, se suman y se dividen entre 5, haciendo el mismo proceso explicado anteriormente.

¿Cómo visualiza el sistema el desempeño del estudiante en las competencias y en los escenarios?

El sistema no visualiza el desempeño del estudiante. El sistema visualiza un gráfico que se basa en una recta que muestra el nivel por el cual se encuentra dicho estudiante en el proceso de certificación del rol en cuestión, esta recta se le muestra al estudiante en la pestaña Expediente, por lo que el profesor evalúa las tareas de ese estudiante, pero no sabe en qué nivel de certificación se encuentra el estudiante.

¿Para qué niveles de certificación puede estar diseñado un plan de formación?

El plan de formación puede estar diseñado para los niveles: básico, intermedio y avanzado. Sin embargo, si las tareas que tiene ese plan de formación no cubre la cantidad necesaria de escenarios para alcanzar el nivel básico, no se podrá alcanzar ningún nivel con ese plan.

¿Cómo sabe el estudiante qué nivel de certificación de rol puede alcanzar con el portafolio asociado?

El estudiante no sabe a qué nivel puede llegar en la certificación de rol, el sistema le muestra el nivel que va alcanzando con las evaluaciones asociadas.

Anexo 2 Historias de usuario

Historia de usuario	
Número:1	**Nombre historia de usuario:** Importar plan de formación
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID/ Tutor	**Iteración asignada**: 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 día
Riesgo en desarrollo: Alta	**Puntos reales:** 2 días
Descripción: • El sistema debe permitir a los profesores PID importar un plan de formación.	
Observaciones: El plan de formación es un documento Excel.	

Historia de usuario	
Número:2	**Nombre historia de usuario:** Evaluar Tarea
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID/ Tutor	**Iteración asignada**: 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 día
Riesgo en desarrollo: Alta	**Puntos reales:** 2 días
Descripción:	

• El sistema debe permitir a los profesores PID evaluar las tareas de los estudiantes asociados a estos.
Observaciones:

Historia de usuario	
Número:3	**Nombre Historia de Usuario:** Asignar tarea
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID/ Tutor	**Iteración asignada:** 1
Prioridad en negocio: Media	**Puntos estimados:** 1 día
Riesgo en desarrollo: Bajo	**Puntos reales:** 1 día
Descripción: • El sistema debe permitir agregar las tareas al estudiante.	
Observaciones: el profesor PID y el tutor son las personas encargadas de agregar tareas a los estudiantes.	

Historia de usuario	
Número: 4	**Nombre historia de usuario:** Generar alerta
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana
Descripción: • Mediante correo se le envía al profesor un registro con las evaluaciones del estudiante y la clasificación del plan de formación según los escenarios que tenga cubierto el plan de formación para el rol que se desea.	
Observaciones: Debe haberse importado el plan de formación y tener al menos una evaluación para que el sistema envíe el registro de evaluaciones y la clasificación de dicho plan.	

Historia de usuario	
Número: 5	**Nombre historia de usuario:** Evaluar Escenario
Modificación de historia de Usuario número: ninguna	
Usuario: Sistema	**Iteración asignada:** 2
Prioridad en negocio: Media	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: • El sistema evalúa cada escenario del plan de formación importado por el profesor con la evaluación máxima.	
Observaciones: el profesor debe haber importado un plan de formación para un estudiante.	

Historia de usuario	
Número: 6	**Nombre historia de usuario:** Evaluar competencia
Modificación de historia de usuario número: ninguna	
Usuario: Sistema	**Iteración asignada:** 1
Prioridad en negocio: Media	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: • El sistema evalúa cada competencia del plan de formación importado por el profesor teniendo en cuenta los escenarios activos que han sido evaluados en cada una de ellas.	
Observaciones: el profesor debe haber importado un plan de formación para un estudiante.	

Historia de usuario	
Número: 7	**Nombre historia de usuario:** Evaluar Rol
Modificación de historia de usuario número: ninguna	
Usuario: Sistema	**Iteración asignada:** 1
Prioridad en negocio: Media	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: • El sistema evalúa el rol para el cual está diseñado el plan de formación importado por el profesor teniendo en cuenta la evaluación de las	

competencias del plan.
Observaciones: el profesor debe haber importado un plan de formación para un estudiante.

Historia de usuario	
Número: 8	**Nombre historia de usuario:** Visualizar desempeño del estudiante
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana
Descripción: • El sistema muestra gráficos de desempeño para los distintos niveles de granularidad en la certificación de rol (tareas, escenarios, competencias).	
Observaciones: Si el estudiante no tiene evaluaciones asignadas, el gráfico de competencias por rol cogerá los valores mínimos para cada una de las competencias.	

Historia de usuario	
Número: 9	**Nombre historia de usuario:** Mostrar gráfico para tareas
Modificación de historia de usuario número: ninguna	
Usuario: Sistema	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana
Descripción: • Evaluada una tarea, el sistema debe mostrarle al profesor el gráfico de esa tarea.	

Observaciones: es necesario que exista tareas evaluadas para el escenario seleccionado

Historia de usuario	
Número: 10	**Nombre historia de usuario:** Mostrar gráfico de escenarios por competencias
Modificación de historia de usuario número: ninguna	
Usuario: Sistema	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana
Descripción: • Seleccionada una competencia, se debe mostrar un gráfico con los escenarios de esta competencia.	
Observaciones: la competencia debe tener escenarios evaluados para que se muestre el gráfico.	

Historia de usuario	
Número: 11	**Nombre historia de usuario:** Mostrar gráfico de competencias por rol
Modificación de historia de usuario número: ninguna	
Usuario: Sistema	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semana
Descripción: • El sistema debe mostrar un gráfico con el desempeño en las distintas competencias del rol.	
Observaciones: si la competencia no tiene escenarios evaluados, esta se muestra en el gráfico con su desempeño mínimo.	

Anexo 3 Tareas de ingeniería

Tarea de ingeniería

Número:1	**Nombre historia de usuario:** Importar plan de formación
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 2 días
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: se debe implementar la funcionalidad importar plan de formación, la cual debe permitir asignarle un plan de formación al estudiante. A este plan de formación se le debe incorporar el campo “Descripción de los escenarios”, lo cual permitirá poder evaluar ese plan de formación en los distintos niveles de la certificación de roles (Básico, Intermedio y Avanzado).	

Tarea de ingeniería	
Número: 2	**Nombre historia de usuario:** Evaluar tarea
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 día
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: se debe implementar la funcionalidad evaluar tarea, la cual debe permitir evaluar un tarea, y con ella, enviarle un correo al profesor donde se relacione	

el registro de evaluaciones del estudiante evaluado y el posible nivel que puede alcanzar en el rol que este estudiante está desempeñando.

Tarea de ingeniería	
Número: 3	**Nombre historia de usuario:** Asignar tarea
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 día
Riesgo en desarrollo: Bajo	**Puntos reales:** 2 días
Descripción: se debe implementar la funcionalidad asignar tarea, la cual debe permitir incluir una tarea en el plan de formación de un estudiante. Esta funcionalidad debe incorporar la descripción de los escenarios del rol que ha sido asignado al estudiante, lo cual permitirá poder evaluar el plan de formación cada vez que se evalúe una tarea.	

Tarea de ingeniería	
Número: 4	**Nombre historia de usuario:** Generar alerta
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 día

Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: se debe implementar la funcionalidad generar alerta, la cual emite una notificación al profesor mediante correo al profesor. En ella se muestra el registro de las evaluaciones del estudiante que ha evaluado y el nivel máximo en la certificación de rol que puede aspirar el estudiante con el plan de formación asignado.	

Tarea de ingeniería	
Número:5	**Nombre historia de usuario:** Cálculo desempeño en escenarios
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 2
Prioridad en negocio: Media	**Puntos estimados:** 1 días
Riesgo en desarrollo: Alto	**Puntos reales:** 1 días
Descripción: se debe implementar la funcionalidad desempeño en la competencia, el cual devuelve por cada competencia la evaluación de sus escenarios.	

Tarea de ingeniería	
Número:6	**Nombre historia de usuario:** Cálculo desempeño en las competencias
Modificación de historia de usuario número: ninguna	

Usuario: Profesor PID	**Iteración asignada:** 2
Prioridad en negocio: Media	**Puntos estimados:** 1 días
Riesgo en desarrollo: Alto	**Puntos reales:** 1 días
Descripción: se debe implementar la funcionalidad desempeño en competencias, la cual posibilitará mostrar en el gráfico de desempeño de competencia el desempeño real del estudiante.	

Tarea de ingeniería	
Número:7	**Nombre historia de usuario:** Obtener escenario competencia por tipo de escenario
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 día
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: se debe implementar la funcionalidad obtener escenario competencia por el tipo de escenario el cual devuelve una competencia.	

Tarea de ingeniería

Número:8	**Nombre historia de usuario:** Obtener competencias por tipo de competencia
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 día
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: se debe implementar la funcionalidad obtener competencia por tipo de competencia, la cual debe devolver la competencia por el identificador del tipo de competencia.	

Tarea de ingeniería	
Número:9	**Nombre historia de usuario:** Obtener escenarios por rol
Modificación de historia de usuario número: ninguna	
Usuario: Profesor PID	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 día
Riesgo en desarrollo: Alto	**Puntos reales:** 2 días
Descripción: se debe implementar la funcionalidad obtener escenarios por rol, la cual debe devolver todos los escenarios asociados a un rol.	

Anexo 4 Casos de prueba para el método evaluar tarea.

Entrada	El estudiante no tiene asociado un portafolio.
Resultados esperados	El sistema muestra una alerta diciendo que el estudiante no tiene rol asignado.

Condiciones	foreach($evaluacion_rol as $eval){ if($eval<0.6){
	
	$nivel='No certifica';
	
	$evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion();
	
	}
	
	}
	
	$notificacion = new Notificacion();
	
	$emitida_por = $em->getRepository('SeguridadBundle:Usuario')->findOneBy(array('id_usuario' => $usuario->getIdUsuario()));
	
	$notificacion->setDescripcion('El plan de formación para el estudiante '.$usuario->getPersona()->getNombre().' '.$usuario>getPersona()->getApellidos().' está diseñado para alcanzar el nivel '.$evalString);
	
	$notificacion->setEmitidoPor($emitida_por);
	
	$tipo_notificacion = $em->getRepository('ComunBundle:TipoNotificacion')->findOneBy(array('id_tipo_notificacion' => Util::not_inf));
	
	$notificacion->setTipoNotificacion($tipo_notificacion);
	
	$notificacion->setPortafolio($this->getPortafolioGtr()->obtenerPortafolio($id_portafolio));

	$this->getPortafolioGtr()->salvarActualizar($notificacion);

Entrada	Debe existir un portafolio para el estudiante para el cual se desea saber el alcance del dicho portafolio.
Resultados esperados	Le envía una notificación al estudiante sobre el posible nivel que puede alcanzar con el portafolio actual, el cual debe estar dado por el valor No certifica.

Condiciones	foreach($evaluacion_rol as $eval){ if($eval<0.6){
	
	$nivel='No certifica';
	
	$evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion();
	
	}
	
	}
	
	$notificacion = new Notificacion();
	
	$emitida_por = $em->getRepository('SeguridadBundle:Usuario')->findOneBy(array('id_usuario' => $usuario->getIdUsuario()));
	
	$notificacion->setDescripcion('El plan de formación para el estudiante '.$usuario->getPersona()->getNombre().' '.$usuario>getPersona()->getApellidos().' está diseñado para alcanzar el nivel '.$evalString);
	
	$notificacion->setEmitidoPor($emitida_por);

	$tipo_notificacion = $em->getRepository('ComunBundle:TipoNotificacion')->findOneBy(array('id_tipo_notificacion' => Util::not_inf)); $notificacion->setTipoNotificacion($tipo_notificacion); $notificacion->setPortafolio($this->getPortafolioGtr()->obtenerPortafolio($id_portafolio)); $this->getPortafolioGtr()->salvarActualizar($notificacion);

Entrada	Debe existir un portafolio para el estudiante para el cual se desea saber el alcance del dicho portafolio.
Resultados esperados	Le envía una notificación al estudiante sobre el posible nivel que puede alcanzar con el portafolio actual, el cual debe estar dado por el valor Básico.

Condiciones	foreach($evaluacion_rol as $eval){ else
	
	if($eval>=0.6 && $eval<0.7){
	
	$nivel='Básico';
	
	$evalString.=$nivel.' del rol '.$rol->getTipo_rol()->getDescripcion();
	
	}
	
	}
	
	$notificacion = new Notificacion();

```
$emitida_por = $em->getRepository('SeguridadBundle:Usuario')-
>findOneBy(array('id_usuario' => $usuario->getIdUsuario()));

$notificacion->setDescripcion('El plan de formación para el
estudiante '.$usuario->getPersona()->getNombre().'
'.$usuario>getPersona()->getApellidos().' está diseñado para
alcanzar el nivel '.$evalString);

$notificacion->setEmitidoPor($emitida_por);

$tipo_notificacion = $em-
>getRepository('ComunBundle:TipoNotificacion')-
>findOneBy(array('id_tipo_notificacion' => Util::not_inf));

$notificacion->setTipoNotificacion($tipo_notificacion);

$notificacion->setPortafolio($this->getPortafolioGtr()-
>obtenerPortafolio($id_portafolio));

$this->getPortafolioGtr()->salvarActualizar($notificacion);
```

Anexo 5 Casos de prueba de aceptación

Caso de prueba de aceptación	
Código caso de prueba: PA1-HU1	**HU:** Importar plan de formación
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema importa un plan de formación a un estudiante.	

Condiciones de ejecución: el estudiante debe estar asociado al profesor.
Entrada / Pasos de ejecución: ✓ El usuario selecciona un estudiante ✓ El usuario selecciona la opción “Importar plan de formación” ✓ El usuario busca el plan de formación a importar ✓ El usuario acepta la acción para importar el plan de formación al estudiante.
Resultado esperado: El sistema importa un plan de formación para el estudiante.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código caso de prueba: PA2-HU2	**HU:** Evaluar tarea
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema evalúa una tarea presente en el plan de formación del estudiante.	
Condiciones de ejecución: el estudiante debe tener tareas por evaluar.	
Entrada / Pasos de ejecución: ✓ El usuario selecciona un estudiante	

✓ El usuario selecciona una tarea disponible para evaluar. ✓ El usuario revisa los artefactos que generó la tarea seleccionada. ✓ El usuario emite una nota de la tarea. ✓ El usuario selecciona la competencia a la cual aporta esa tarea. ✓ El usuario selecciona el escenario al cual aporta la tarea en cuestión. ✓ El usuario acepta las acciones ejecutadas.
Resultado esperado: El sistema evalúa la tarea seleccionada.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código caso de prueba: PA3-HU3	**HU:** Asignar tarea
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema asigna una tarea a un estudiante.	
Condiciones de ejecución: el estudiante debe estar asociado al profesor.	
Entrada / Pasos de ejecución: ✓ El usuario selecciona un estudiante ✓ El usuario selecciona la opción “Asignar tarea”	

✓ El usuario completa los campos visibles necesarios para la tarea. ✓ El usuario acepta las acciones ejecutadas
Resultado esperado: El sistema asigna una tarea al estudiante.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código caso de prueba: PA5-HU5	**HU:** Evaluar escenario
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema evalúa los escenarios, necesario para la visualización del gráfico escenarios por competencia.	
Condiciones de ejecución: el estudiante debe tener tareas evaluadas asociadas al escenario seleccionado.	
Entrada / Pasos de ejecución: ✓ El usuario selecciona el estudiante. ✓ El usuario selecciona el rol para el cual desea ver el desempeño de sus competencias. ✓ El usuario selecciona la competencia para la cual desea ver la evaluación de sus escenarios.	
Resultado esperado: El sistema evalúa los escenarios.	

Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código caso de prueba: PA6-HU6	**HU:** Evaluar competencia
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema evalúa las competencias, necesario para la visualización del gráfico competencias por rol.	
Condiciones de ejecución: el estudiante debe tener un rol asignado.	
Entrada / Pasos de ejecución: ✓ El usuario selecciona el estudiante. ✓ El usuario selecciona el rol para el cual desea ver el desempeño de sus competencias.	
Resultado esperado: El sistema evalúa las competencias.	
Evaluación de la prueba: Satisfactoria	

Caso de prueba de aceptación	
Código caso de prueba: PA7-HU7	**HU:** Evaluar rol

Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.
Descripción de la prueba: prueba para verificar que el sistema evalúa el rol.
Condiciones de ejecución: el estudiante debe tener un rol asignado.
Entrada / Pasos de ejecución: ✓ El usuario selecciona el estudiante. ✓ El usuario selecciona el rol para el cual desea ver el desempeño de sus competencias.
Resultado esperado: El sistema muestra una información debajo del gráfico de competencias por rol sobre el nivel de certificar un rol en el que se encuentra el estudiante seleccionado.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código caso de prueba: PA8-HU9	**HU:** Mostrar gráfico de competencias por rol
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema muestra las competencias del rol seleccionado.	

Condiciones de ejecución: el estudiante debe tener un rol asignado.
Entrada / Pasos de ejecución: ✓ El usuario selecciona el estudiante. ✓ El usuario selecciona el rol para el cual desea ver el desempeño de sus competencias.
Resultado esperado: El sistema muestra el gráfico de competencia por rol.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código caso de prueba: PA5-HU5	**HU:** Mostrar gráfico de escenarios por competencia
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema muestra el gráfico de escenarios por competencias.	
Condiciones de ejecución: el estudiante debe tener tareas evaluadas asociadas a algunos de los escenarios de la competencia seleccionada.	
Entrada / Pasos de ejecución: ✓ El usuario selecciona el estudiante. ✓ El usuario selecciona el rol para el cual desea ver el desempeño de sus competencias.	

✓ El usuario selecciona la competencia para la cual desea ver la evaluación de sus escenarios.
Resultado esperado: El sistema muestra el gráfico de escenarios por competencias.
Evaluación de la prueba: Satisfactoria

Caso de prueba de aceptación	
Código caso de prueba: PA5-HU5	**HU:** Mostrar gráfico para tareas
Nombre de la persona que realiza la prueba: Carlos Rafael Rodríguez Rodríguez.	
Descripción de la prueba: prueba para verificar que el sistema muestra el gráfico de tareas.	
Condiciones de ejecución: el estudiante debe tener tareas evaluadas.	
Entrada / Pasos de ejecución: ✓ El usuario selecciona el estudiante. ✓ El usuario selecciona el rol para el cual desea ver el desempeño de sus competencias. ✓ El usuario selecciona la competencia para la cual desea ver la evaluación de sus escenarios. ✓ El usuario selecciona el escenario para el cual desea ver las tareas que están afectando su evaluación.	

Resultado esperado: El sistema muestra el gráfico de tareas por escenarios.
Evaluación de la prueba: Satisfactoria

Anexo 6 Acta de liberación

Acta de Liberación Interna de Productos Software

Fecha de emisión del acta: 7/06/2016

1. **Emitida a favor de**: Tesis Extensión del Sistema de gestión de certificación de roles 2.0

Datos del producto

Artefacto	Versión	Estado final	Cantidad Iteraciones	Tipos de pruebas realizadas
App: *Herramienta para evaluar factibilidad técnica y comercial en proyectos de software basada en lógica difusa*	*1.0*	*0*	*2*	*Evaluación dinámica* *Pruebas de Funcionalidad*

Ing. Yordanis García Leiva
Asesor de Calidad CEGEL

Arian Cruz Castillo
Autor

Responsable de la liberación

Isis Bertami Barrios

UCI Universidad de las Ciencias Informáticas
Centro de Gobierno Electrónico

1

Anexo 7 Acta de aceptación

ACTA DE ACEPTACIÓN

En cumplimiento del **Ejercicio de culminación de estudios del estudiante Arian Cruz Castillo** y en función de la ejecución del proyecto: **Trabajo de diploma "Extensión del Sistema de Gestión de Certificación de Roles 2.0"** se hace entrega de los productos que se relacionan a continuación:

- Especificación de Requisitos de Software
- Codigo fuente de la aplicación

La Parte Cliente, luego de haber revisado los productos de trabajo determina que

- Los 11 requisitos funcionales declarados en el trabajo han sido implementados de manera adecuada. Lo que se pudo comprobar al operar el sistema informático
- Las nuevas funcionalidades agregadas al sistema informático poseen una adecuada usabilidad.
- Luego de las modificaciones realizadas con este trabajo el sistema informático ofrece una mayor integralidad para la gestión del proceso de certificación de roles

Comentarios

Se realizaron dos sugerencias relacionadas con la interfaz de usuario que no constituyen No Conformidades, ni limitan o reducen la explotación del sistema informático.

Entrega	Recibe
Nombre y apellidos: Arian Cruz Castillo	**Nombre y apellidos: Lic. Rafael Rodríguez**
Cargo:	**Cargo:**
Desarrollador del proyecto	**Jefe de Dpto. Práctica Profesional CEGEL**

Centro de Gobierno Electrónico

Fecha: 07/06/2016

y apellidos Carlos R