Universidad de las Ciencias Informáticas

Facultad 5

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Análisis y diseño del módulo de gestión de los recursos humanos para el sistema GESPRO

Autor: Arlan Hernández Alfonso

Tutor: MSc. Surayne Torres López

La Habana, Junio 2012

"Invertir en conocimientos produce siempre los mejores beneficios."

Benjamin Franklin

DECLARACIÓN DE AUTORÍA

Declaro ser el único autor del trabajo “Análisis y diseño del módulo de gestión de los recursos humanos para el sistema GESPRO” y autorizo a la Facultad 5 y a la Universidad de Ciencias Informáticas a hacer uso del mismo, con carácter exclusivo.

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

_____

Arlan Hernández Alfonso

Firma del Autor

_____

MSc. Surayne Torres López

Firma de la Tutora:

DATOS DE CONTACTO

MSc. Surayne Torres López

Graduada de Ingeniera en Ciencias Informáticas en el 2008. Profesora Instructora con 4 años de experiencia como docente y especialista en el área de Gestión de Proyectos. Máster en Gestión de Proyectos Informáticos.

Correo: storres@uci.cu

AGRADECIMIENTOS

A toda mi familia por haberme apoyado siempre en todas mis decisiones.

A Lazarita, Manuel, Manuelito, Maikel, Dayana, Fermina, Isa, Mariana y familia por haberme recibido como un hijo más.

A todos mis compañeros que me han acompañado y apoyado a lo largo de estos años de carrera: Yani, Eileen, Danae, Sulemy, Yari, Hanoi, Belo, Oscar, Enrique, Gabriel, Yandy, Osmar, Enier, Miguel, Frank, Orestes, Consuegra, Osmany.

A todos los que me acompañaron y apoyaron en la travesía que fue la misión: Suleika, Diannys, Nailiuvis, Yuneisi, Yanet, Zamaharys, Yanelis, Odelkys, Yurixay, Alvaro, Leitniz, Tejeda.

A mi tutora por su entrega, dedicación y sus consejos.

A los profes del laboratorio 308 de IP por todo el apoyo que me dieron, en especial al profe Félix por toda la ayuda que me brindó.

A las compañeras de tesis del laboratorio Rosalia, Lourdes, Yissel y Silvia por la ayuda que me ofrecieron.

A mi tribunal y oponente por sus sugerencias y sus críticas constructivas, las cuales me hicieron crecer como profesional.

A todos los profesores que durante todos los años de mi vida contribuyeron a mi formación.

A todas aquellas personas que de una forma u otra me han apoyado y ayudado.

A Obatalá y todos los Orishas.

A TODOS GRACIAS

DEDICATORIA

A mi mamá por ser mi razón de ser.

A mis abuelos Israel y Sergio que aunque no están aquí conmigo, se que están orgullosos de mi.

A mi papá y mi madrasta Aymet por todo el apoyo que me han dado.

A mis hermanas Arleny y Aris M., que son mis estrellas.

A mi sobrino Reymond Anthony por llegar a alegrarnos la vida.

A mis abuelas Humbe y Zela por estar siempre apoyándome en todo momento.

A Tata, a mis tíos (Chino, Rolando y Osniel), a Ale por apoyarme siempre.

A Nena por todo su apoyo, en sus 106 años.

A toda mi familia.

A todas aquellas personas que siempre me han apoyado y han tenido fé en mí.

A Obatalá y todos los Orishas.

RESUMEN

El sistema GESPRO es una suite de herramientas desarrollada en la UCI con el objetivo de mejorar el proceso de desarrollo de software. Dicha herramienta posee un módulo de Gestión de Recursos Humanos el cual almacena datos personales de los usuarios que pertenecen a los proyectos y los roles de los mismos en cada proyecto, pero dicho módulo no guarda información relacionada con las competencias laborales, ni con los planes de formación a seguir para que un trabajador alcance el nivel requerido para desempeñar un determinado rol.

Este trabajo tiene como objetivo realizar el análisis y diseño de un módulo de Gestión de los Recursos Humanos para el sistema GESPRO basados en competencias y regido por lo descrito en las Normas Cubanas 3000, 3001 y 3002 del 2007 que permita guardar información relacionada con las competencias requeridas por los trabajadores, para desempeñar de manera eficiente su trabajo, y de los planes de formación necesarios para aumentar el nivel de competencia del equipo de trabajo, para realizar de manera más eficiente las tareas asignadas.

Palabras clave: Competencias, GESPRO, Recursos Humanos.

INDICE

INTRODUCCIÓN

El mundo de hoy está regido sin dudas por un gran avance tecnológico y de los medios de comunicación. Cada día que pasa se da un paso de avance en dichas materias.

Desde la segunda mitad del siglo XX los adelantos científico-técnicos en las ramas de las comunicaciones, la electrónica y la computación se han hecho inminentes. Las Tecnologías de la Información y las Comunicaciones (TIC) han alcanzado tal avance que ocupan hoy uno de los más importantes renglones en la vida diaria de las empresas y de los países.

El desarrollo de software es uno de los eslabones más importantes que han permitido alcanzar dicho avance. Hoy sin duda la producción y comercialización de software constituye una importante fuente de ingresos para cualquier país, por lo que constituye un factor fundamental para el desarrollo económico (Montero Fang-Tac, 2011). En el mundo de hoy existe una gran competencia en el desarrollo y comercialización de estos productos por lo que cada día que pasa debemos crear aplicaciones con mayor calidad que satisfagan la necesidad de los clientes.

La gestión de proyectos es la disciplina que se encarga de organizar y administrar recursos de manera tal que se pueda culminar todo el trabajo requerido en el proyecto dentro del alcance, el tiempo, y coste definidos (Pons Achell, 2009). El uso de herramientas para la gestión de proyectos se ha ido generalizando al paso del tiempo con el objetivo de facilitar el trabajo de los especialistas (Laboratorio de Gestión de Proyectos, 2012).

Los constantes avances tecnológicos, la globalización de los mercados y el aumento de la competitividad ponen de manifiesto como consecuencia, que la diferencia en el éxito de las organizaciones laborales dependa de su capital humano. Esto justifica la necesidad de un nuevo enfoque de los Recursos Humanos (RH o RRHH), y por tanto, una nueva manera de gestionar el personal de la organización, que posibilita y contribuye a un mejor alcance de los objetivos estratégicos de la misma (Guach Castillo, 2004).

La creciente importancia de los recursos humanos se debe al nuevo papel que se le asigna dentro de la organización para dar respuesta a los cambios experimentados en la sociedad en general y del mundo laboral en particular. A esto se añade el reconocimiento de que a través de la gestión de los recursos humanos se puede influir de manera determinante en los objetivos de la organización.

Algunos de los aspectos que influyen en la importancia de la gestión de recursos humanos son:

- Aumento de la competencia y por lo tanto de la necesidad de ser competitivo.
- Los costos y ventajas relacionadas con el uso de los recursos humanos.
- La crisis de productividad.
- El aumento del ritmo y complejidad de los cambios sociales, culturales, normativos, demográficos y educacionales.
- Los síntomas de las alteraciones en el funcionamiento de los lugares de trabajo.
- Las tendencias del siglo XXI (Dolan, 2007).

INTRODUCCIÓN

Para conseguir la creación de aplicaciones y herramientas de excelencia es necesario alcanzar una mayor preparación de los trabajadores que se encargarán de la elaboración de dichos productos. Los recursos humanos constituyen, el factor principal para lograr un producto de calidad y que satisfaga la necesidad de los clientes, es por ello que se hace necesario ubicarlos en los puestos adecuados de acuerdo con el nivel de competencia y conocimiento de los mismos para propiciar así un uso adecuado de dichos recursos.

La Universidad de Ciencias Informáticas (UCI), institución de altos estudios creada para promover el desarrollo de software en Cuba, utilizando el modelo de aprendizaje desde la producción, tiene como objetivo lograr productos de calidad para satisfacer a los clientes y así ayudar al desarrollo económico del país además de fomentar el desarrollo y formación de recursos humanos aptos y competentes.

El sistema GESPRO (Piñero Pérez, y otros, 2011) para la gestión de proyectos de la UCI es una suite de herramientas con diversas funcionalidades que permiten la gestión de áreas de la gestión de proyectos como el alcance, el tiempo, entre otros, así como el control y seguimiento de los proyectos de los centros de desarrollo.

El paquete de herramientas GESPRO soporta como uno de sus módulos la gestión de los recursos humanos. Como parte de sus funcionalidades se almacenan datos personales de los usuarios que pertenecen a los proyectos y los roles de los mismos en cada proyecto. Sin embargo no se guarda información relacionada con las competencias requeridas por los usuarios, para que dado el rol que le haya sido asignado puedan desarrollar eficientemente las actividades del proyecto, lo que provoca que no se conozca realmente el nivel de competencia necesario para que un usuario pueda o no desempeñar un determinado rol o hasta qué punto dicho usuario está capacitado para llevar a cabo las tareas de dicho rol.

Además no se tienen funcionalidades que permitan facilitar la evaluación de las competencias de acuerdo al desempeño demostrado en el desarrollo de las tareas. En consecuencia no se identifican las necesidades de formación de los miembros del equipo, por lo que no se sabe con exactitud los planes de formación que se deben seguir para que un usuario presente un desempeño adecuado o superior en el desarrollo de las tareas que le son asignadas, así como mejorar la formación del resto del equipo de trabajo.

Ante esta situación surge como **problema científico** Las insuficiencias en las funcionalidades del módulo de gestión de los recursos humanos en el sistema GESPRO, está afectando la oportuna toma de decisiones asociadas a esta área de gestión de proyectos.

Para darle solución al problema antes planteado se definió como **objeto de estudio**: Módulo para la gestión de los recursos humanos del sistema GESPRO. Para resolver el problema anterior se propone como **objetivo general** de la investigación realizar el análisis y diseño del módulo para la gestión de los recursos humanos del sistema GESPRO, teniendo en cuenta lo establecido por las normas cubanas y que ayude a la toma de decisiones en cuanto a la elaboración de planes de formación; teniendo como **campo de acción** el análisis y diseño del módulo de gestión de los recursos humanos del sistema GESPRO.

Como **idea a defender** se define:

Si se realiza el análisis y diseño de un módulo para la gestión de recursos humanos del sistema GESPRO, se logrará el diseño de un módulo que permitirá almacenar información relacionada con las competencias requeridas por los usuarios, y de la evaluación de las competencias de acuerdo al desempeño alcanzado en las tareas realizadas, así como ayudar a la toma de decisiones en cuanto a la elaboración de planes de formación.

Para darle cumplimiento al objetivo planteado se proponen las siguientes **Tareas de Investigación:**

- Definición de la estrategia de trabajo (cronograma e hitos de la investigación).
- Actualización en los conceptos fundamentales y las principales tendencias en las herramientas de gestión de proyectos y gestión de recursos humanos.
- Evaluación del contenido de la información obtenida sobre las herramientas de gestión de los recursos humanos, establecer un diagnóstico de las tendencias actuales y definir posición al respecto.
- Definición de los requerimientos del módulo de gestión de los recursos humanos.
- Realización de los casos de prueba basados en requisitos.
- Validación de los requerimientos a través de las listas de chequeo.
- Validación del diseño del módulo a través de métodos de expertos.

Marco Metodológico

Métodos

Métodos teóricos

Analítico-Sintético: Con el objetivo de consultar la documentación existente acerca de las herramientas y módulos gestores de recursos humanos y llegar a conclusiones sobre cuáles tendencias seguir.

Análisis Histórico-Lógico: Con el objetivo de conocer información sobre la evolución y las tendencias actuales de las herramientas y módulos de gestión de recursos humanos.

Métodos empíricos

Observación: Observar el funcionamiento de herramientas y módulos de gestión de recursos humanos existentes para de esta manera obtener las funcionalidades básicas para el nuevo componente a desarrollar así como ver las posibilidades para mejorarlas.

Encuesta: Con motivo de adquirir información sobre el nivel de satisfacción existente con el módulo actual de gestión de recursos humanos del sistema GESPRO.

El presente trabajo de investigación está estructurado en tres capítulos, además contiene varios anexos así como varios artefactos adjuntos generados durante el desarrollo. A continuación se describe el objetivo principal de cada uno de los capítulos:

Capítulo 1: Fundamentación Teórica.

En este capítulo se realiza el estudio de las diferentes herramientas informáticas existentes en el sector de la gestión de proyectos y gestión de recursos humanos, se estudia además conceptos relacionados con la gestión de competencias, así como un estudio de metodologías de desarrollo de software, herramientas, tecnologías, métodos de captura, especificación y validación de requisitos que se utilizarán en el presente trabajo de investigación.

Capítulo 2: Solución Propuesta.

En este capítulo se ejecutan algunos pasos relacionados con la Ingeniería de Requisitos, haciendo énfasis en las técnicas de obtención de requisitos relacionadas con cada actividad y con la especificación de requisitos, la cual detalla las funcionalidades del sistema, así como las restricciones que este debe cumplir; los casos de prueba basados en requisitos que permiten comprobar la correcta implementación de dichos requisitos, además se realiza la descripción de los patrones arquitectónicos, de diseño y de acceso a datos que se utilizaran para el diseño del módulo; y se explica el diseño de la base de datos que se propone para la implementación de la solución.

Capítulo 3: Validación de la solución.

En este capítulo se realiza la validación de la propuesta elaborada, con el objetivo de verificar su calidad y para determinar si los requisitos identificados realmente definen el sistema que se debe construir. Se realizan revisiones formales utilizando técnicas como las listas de chequeo; se analizan los resultados de la aplicación de los casos de prueba, así como la comparación entre diferentes herramientas de gestión de proyectos y del módulo de gestión de recursos humanos anterior con el que se pretende obtener luego de la investigación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se describen los principales conceptos y aspectos relacionados con el análisis, diseño y competencias que respaldan la realización de esta investigación, así como las metodologías, tecnologías y lenguajes de programación a utilizar; se expone además el estado del arte referente a las tendencias y herramientas de hoy que se utilizan en los procesos de gestión de proyectos.

1.1 Análisis

El ***análisis*** de ***los requisitos*** es una tarea de ingeniería del software que cubre el hueco entre la definición del software a nivel sistema y el diseño del software. El análisis de requisitos permite al ingeniero de sistemas especificar las características operacionales del software (función, datos y rendimientos), indica la interfaz del software con otros elementos del sistema y establece las restricciones que debe cumplir el software (Pressman, 2002).

El análisis de requisitos del software puede dividirse en cinco Áreas de esfuerzo: (1) reconocimiento del problema, (2) evaluación y síntesis, (3) modelado, (4) especificación y (5) revisión (Pressman, 2002).

El modelo de análisis debe lograr tres objetivos primarios: (1) describir lo que requiere el cliente. (2) establecer una base para la creación de un diseño de software, y (3) definir un conjunto de requisitos que se pueda validar una vez que se construye el software (Pressman, 2002).

Dentro del modelo del análisis se han desarrollado dos enfoques o visiones principales:

Análisis estructurado, considera que los datos y el proceso que transforman los datos son entidades separadas. Los procesos que manipulan los objetos de los datos se modelan de tal manera que muestran cómo transforman los datos, mientras los objetos de datos fluyen por el sistema (Pressman, 2005). Sus principales pasos son: Modelado de datos, Modelado funcional y flujo de información y Modelado del comportamiento.

Este método al momento de enfrentarse a empresas pequeñas, puede ser muy adaptable a las mismas, pero en empresas grandes, el método estructurado se convierte en una vista no objetiva de la situación real. Presenta como inconveniente que ignora los datos en favor de las operaciones y no da prioridad entre las cosas importantes y los detalles. Los cambios en los requisitos afectan notablemente a la funcionalidad de un sistema, por lo que afectan mucho al software desarrollado con métodos estructurados.

Análisis orientado a objetos (AOO), se centra en la definición de clases y en la manera en que éstas colaboran entre ellas para efectuar los requisitos del cliente. El UML (Lenguaje Unificado de Modelado) y el proceso unificado están orientados a objetos en forma predominante (Pressman, 2005).

Análisis del dominio: es la identificación, análisis y especificación de requisitos comunes de un dominio de aplicación específico, normalmente para su reutilización en múltiples proyectos dentro del mismo dominio de aplicación (Pressman, 2005).

Este tiene como inconveniente una alta curva de aprendizaje, es costoso, requiere de conocimientos adicionales y no es recomendable para proyectos pequeños.

Posee grandes ventajas como: pone especial énfasis en la reutilización, y proporciona mecanismos efectivos que permiten reutilizar lo que es común, sin impedir por ello describir las diferencia; posibilita el desarrollo incremental y la programación por diferencia, al poder ir añadiendo funcionalidad vía herencia; permite la utilización masiva de librerías de clases que garantizan la fiabilidad, ya que los componentes sólo se añaden a la librería cuando se ha verificado la corrección de su funcionamiento, además los cambios afectan en mucha menor medida a los objetos que componen o manejan el sistema, que son mucho más estables.

1.2 Ingeniería de Requisitos

Se define como un conjunto de actividades en las cuales, utilizando técnicas y herramientas, se analiza un problema y se concluye con la especificación de una solución. Se adapta a las necesidades del proceso, el proyecto, el producto y las personas que realizan. 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 conforme estos se transforman en un sistema operacional (Pressman, 2005).

Etapas de la Ingeniería de Requisitos

La Ingeniería de Requisitos (IR) propone actividades que rigen todo el proceso a partir de la identificación de los requerimientos hasta su exitosa validación (Fornaris Licea, y otros, 2011):

- **Elicitación:** Aquí, los analistas de requerimientos deben trabajar junto al cliente para descubrir el problema que el sistema debe resolver, los diferentes servicios que el sistema debe prestar, las restricciones que se pueden presentar, etc. (Fornaris Licea, y otros, 2011).
- **Análisis:** sobre la base de la extracción realizada previamente, comienza esta fase, la cual se enfoca en descubrir problemas con los requerimientos del sistema identificados hasta el momento. Usualmente se hace un análisis después de haber producido un bosquejo inicial del documento de requerimientos; en esta etapa se leen los requerimientos, se investigan, se intercambian ideas con el resto del equipo, se resaltan los problemas, se buscan alternativas y soluciones y luego se fijan reuniones con el cliente para discutir los requerimientos (Fornaris Licea, y otros, 2011).
- **Especificación:** en esta fase se documentan los requerimientos acordados con el cliente en un nivel apropiado de detalle. (Fornaris Licea, y otros, 2011).
- **Gestión:** es un conjunto de actividades que ayudan al equipo de trabajo a identificar, controlar y seguir los requisitos y los cambios a estos en cualquier momento mientras se desarrolla el proyecto. La gestión formal de requisitos se inicia sólo para proyectos grandes,

los cuales tienen cientos de requisitos identificables. En los proyectos pequeños esta función es menos formal (Fornaris Licea, y otros, 2011).

1.3 Técnicas de Captura de Requisitos

Es el proceso en el cual los datos son extraídos de las personas pudiendo variar, dependiendo de la persona consultada (González Moreno, 2008-2009).

Introspección: Esta técnica recomienda que el ingeniero en requisitos se ponga en el lugar del cliente y trate de imaginar cómo desearía el Sistema.

Entrevistas: Existen diferentes tipos de entrevistas dentro de las cuales están:

- Entrevistas de Cuestionarios, se genera un cuestionario de preguntas, el cuál será aplicado al cliente para comenzar la captura de requisitos.
- Entrevistas de Final Abierto, el ingeniero en requisitos permite que el cliente le vaya narrando su problemática y el ingeniero de software lo guíe a través de la narración para ir determinando los requisitos del sistema.
- Entrevistas en grupo de desarrollo, se forman grupos específicos con el personal del cliente. Estos grupos tendrán en común algún área de trabajo o especialidad. El objetivo es poder contar con los expertos en cierta área de la empresa para poder llegar en conjunto a la especificación de requisitos.
- Discusiones, pretende que el ingeniero sostenga una *discusión con el cliente* sobre su problemática para tratar de determinar en conjunto los requisitos del sistema. (González Moreno, 2008-2009).

Lluvia de ideas (Brainstorm): Es una técnica de reuniones en grupo cuyo objetivo es que los participantes muestren sus ideas de forma libre. Consiste en la mera acumulación de ideas y/o información sin evaluar las mismas. El grupo de personas que participa en estas reuniones no debe ser muy numeroso (máximo 10 personas), una de ellas debe asumir el rol de moderador de la sesión, pero sin carácter de controlador.

Las reglas básicas a seguir son:

- Los participantes deben pertenecer a distintas disciplinas y, preferentemente, deben tener mucha experiencia. Esto trae aparejado la obtención de una mayor cantidad de ideas creativas.
- Conviene suspender el juicio crítico y se debe permitir la evolución de cada una de las ideas, porque si no se crea un ambiente hostil que no alienta la generación de ideas.
- Por más locas o salvajes que parezcan algunas ideas, no se las debe descartar, porque luego de maduradas probablemente se tornen en un requerimiento sumamente útil.
- A veces ocurre que una idea resulta en otra idea, y otras veces se puede relacionar varias ideas para generar una nueva.
- Escribir las ideas sin censura (Fornaris Licea, y otros, 2011).

Arqueología de Documentos: permite determinar posibles requisitos sobre la base de inspeccionar la documentación utilizada y generada en la empresa. Sirve fundamentalmente

como complemento de las demás técnicas y mediante la misma se obtiene información que de otra forma sería imposible, como son manuales de procedimientos, reglamentos, facturas, entre otras (Guillén Suárez, 2011).

1.4 Especificación de Requisitos

La escritura de los requisitos se refiere a la tarea de reunir una descripción del producto desde el punto de vista de negocio. Para la descripción de los requisitos funcionales se utiliza la plantilla ***0113_Especificación De Requisitos De Software***.

Requisitos Funcionales: escriben lo que el sistema o el software deben hacer. La funcionalidad es la capacidad útil proporcionada por uno o más componentes de un sistema. En algunos casos, los requisitos funcionales también pueden declarar explícitamente lo que el sistema no debe hacer (Socas Alvez, y otros, 2007).

Requisitos no funcionales: Son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Los requisitos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto. Algunas de sus clasificaciones son:

- **Requisitos de Software:** debe mencionarse el software del que se debe disponer, después de implementado el sistema.
- **Requisitos de Hardware:** se deben enunciar los elementos de hardware que se necesitan para que el software cumpla sus funcionalidades.
- **Restricciones en el diseño y la implementación:** especifica o restringe la codificación o construcción de un sistema, son restricciones que han sido ordenadas y deben ser cumplidas estrictamente.
- **Requisitos de apariencia o interfaz externa:** describe la apariencia del producto. Es importante destacar que no se trata del diseño de la interfaz en detalle sino que especifican cómo se pretende que sea la interfaz externa del producto. También pueden ser necesidades de cumplir con normas estándares, o con los estándares de la empresa para la cual se esté desarrollando el software.
- **Requisitos de Seguridad:** este es el tipo de requisito más difícil, que provocará los mayores riesgos si no se maneja correctamente.
- **Requisitos de Usabilidad:** describen los niveles apropiados de usabilidad, dados los usuarios finales del producto, para ello deben revisarse las especificaciones de los perfiles de usuarios y las clasificaciones de sus niveles de experiencia.
- **Requisitos de Soporte:** abarcan todas las acciones a tomar una vez que se ha terminado el desarrollo del software con motivos de asistir a los clientes de este, así como lograr su mejoramiento progresivo y evolución en el tiempo (Socas Alvez, y otros, 2007).

1.5 Técnicas de Validación de Requisitos

Esta etapa tiene como misión demostrar que la definición de los requisitos define realmente el sistema que el usuario necesita o el cliente desea, demostrar la calidad de sus especificaciones y la satisfacción por parte del cliente. No es más que un mecanismo que examina la especificación para asegurar que todos los requisitos de software se han sido establecido de manera precisa; que se han detectado las inconsistencias, omisiones y errores y que éstos han sido corregidos, y que los productos de trabajo cumplen con los estándares establecidos para el proceso, proyecto y producto (Pressman, 2005).

Dentro de las diferentes técnicas existentes están:

- **Auditorías:** La revisión de la documentación con esta técnica consiste en un chequeo de los resultados contra una lista de chequeo predefinida o definida a comienzos del proceso, de manera que sólo una muestra es revisada (Pérez Olmos, 2009).
- **Reviews o Walk-throughs:** esta técnica consiste en la lectura y corrección de la completa documentación o modelado de la definición de requisitos. Con ello solamente se puede validar la correcta interpretación de la información transmitida. Más difícil es verificar consistencia de la documentación o información faltante (Fornaris Licea, y otros, 2011).
- **Matrices de trazabilidad:** esta técnica consiste en marcar los objetivos del sistema y chequearlos contra los requisitos del mismo. Es necesario determinar qué objetivos cubre cada requisito, de esta forma se podrán detectar inconsistencias u objetivos no cubiertos (Fornaris Licea, y otros, 2011).
- **Listas de chequeo (Checklist):** Las listas de chequeo constituyen una estrategia de fácil manejo para validar requerimientos. Básicamente consiste en un conjunto de interrogantes que se usan para evaluar cada requerimiento verificando los elementos de la lista mientras se chequea el documento de especificación de los requisitos. Cuando se descubren problemas potenciales deben ser anotados para su posterior corrección. Las listas brindan un recordatorio de lo que se debe buscar y reducen la posibilidad de obviar alguna verificación importante. (Pérez Olmos, 2009).
- **Prototipos**: algunas propuestas se basan en obtener de la definición de requisitos prototipos que, sin tener la totalidad de la funcionalidad del sistema, permitan al usuario hacerse una idea de la estructura de la interfaz del sistema. Esta técnica tiene el problema de que el usuario debe entender que lo que ve es un prototipo y no el sistema final (Fornaris Licea, y otros, 2011).
- **Casos de Pruebas basados en requisitos:** Las pruebas basadas en requerimientos son una aproximación sistemática al diseño de casos de prueba en donde el usuario considera cada requerimiento y deriva un conjunto de pruebas para cada uno de ellos. Las pruebas basadas en requerimientos son pruebas de validación en lugar de pruebas de defectos, el usuario intenta demostrar que el sistema ha implementado sus requerimientos de forma adecuada. (Wikidot, 2007).

  Como resultado, esta prueba completa ofrece las siguientes ventajas:

- La vinculación de los riesgos y los requisitos revelará requisitos vagos o desaparecidos. Es especialmente interesante para los riesgos con una alta prioridad.
- Las pruebas se pueden concentrar en las partes más importantes, ejecutando primero las que cubran los riesgos de mayor prioridad.
- La comunicación en el mismo idioma que el cliente y las partes interesadas. Esto hace que sea más fácil informar sobre el estado del proyecto y tomar una decisión sobre si se debe invertir más en las pruebas o tomar el riesgo.
- Para hacer más fácil la negociación en momentos de presión sobre los riesgos y su prioridad, el director de pruebas comienza con las pruebas con los riesgos de mayor prioridad. (QAustral, 2010).

1.6 Diseño

Es una representación significativa de ingeniería de algo que se va a construir. Se puede hacer el seguimiento basándose en los requisitos del cliente, y al mismo tiempo la calidad, se puede evaluar y cotejar con el conjunto de criterios predefinidos para obtener un diseño "bueno". En el contexto de la ingeniería de software, el diseño se centra en cuatro áreas importantes de interés:

- **Datos**: el diseño de datos transforma el modelo del dominio de información que se crea durante el análisis en las estructuras de datos que se necesitarán para implementar el software. Los objetos de datos y las relaciones definidas en el diagrama relación entidad y el contenido de datos detallado que se representa en el diccionario de datos proporcionan la base de la actividad del diseño de datos. A medida que se van diseñando cada uno de los componentes del software, van apareciendo más detalles de diseño (Pressman, 2002).
- **Arquitectura**: el diseño arquitectónico define la relación entre los elementos estructurales principales del software, los patrones de diseño que se pueden utilizar para lograr los requisitos que se han definido para el sistema, y las restricciones que afectan a la manera en que se pueden aplicar los patrones de diseño arquitectónicos. La representación del diseño arquitectónico puede derivarse de la especificación del sistema, del modelo de análisis y de la interacción del subsistema definido dentro del modelo de análisis (Pressman, 2002).
- **Interfaces**: el diseño de la interfaz describe la manera de comunicarse el software dentro de sí mismo, con sistemas que inter-operan dentro de él y con las personas que lo utilizan. Una interfaz implica un flujo de información (por ejemplo, datos y/o control) y un tipo específico de comportamiento. Por tanto, los diagramas de flujo de control y de datos proporcionan gran parte de la información que se requiere para el diseño de la interfaz (Pressman, 2002).
- **Componentes**: el diseño a nivel de componentes transforma los elementos estructurales de la arquitectura del software en una descripción procedimental de los componentes del software (Pressman, 2002).

La importancia del diseño del software se puede describir con una sola palabra ***calidad***. El diseño es el lugar en donde se fomentará la calidad en la ingeniería del software, proporciona las representaciones del software que se pueden evaluar en cuanto a calidad, es la única forma de convertir exactamente los requisitos de un cliente en un producto o sistema de software finalizado, sirve como fundamento para todos los pasos siguientes del soporte del software y de

la ingeniería del software. Sin un diseño, corremos el riesgo de construir un sistema inestable, un sistema que fallará cuando se lleven a cabo cambios; un sistema que puede resultar difícil de comprobar; y un sistema cuya calidad no puede evaluarse hasta muy avanzado el proceso, sin tiempo suficiente y con mucho dinero gastado en él (Pressman, 2002).

A continuación se analizarán los principales conceptos relacionados con la gestión de recursos humanos.

1.7 Conceptos relacionados con los Recursos Humanos

Recursos humanos: En (Fernández, 1999) se establece que los RH los constituyen las personas y sus energías, el motor fundamental para el aprovechamiento de esas energías lo constituye la motivación. Chiavenato los entiende como las personas que ingresan, permanecen y participan en la organización, en cualquier nivel jerárquico o tarea (Chiavenato, 2007). En esta investigación se toma la definición ofrecida por (Cuesta, 2010): las personas insertadas en una organización laboral. Ya que coincide en el elemento fundamental de las otras analizadas.

Capital humano: Conjunto de conocimientos, experiencias, habilidades, sentimientos, actitudes, motivaciones, valores y capacidad para hacer, portados por los trabajadores para crear más riquezas con eficiencia. Es, además, conciencia, ética, solidaridad, espíritu de sacrificio y heroísmo (ONN, 3000:2007).

Competencias laborales:

Capacidad de articular y movilizar condiciones intelectuales y emocionales en términos de conocimientos, habilidades, actitudes y prácticas, necesarias para el desempeño de una determinada función o actividad, de manera eficiente, eficaz y creativa, conforme a la naturaleza del trabajo. Capacidad productiva de un individuo que se define y mide en términos de desempeño real y demostrando en determinado contexto de trabajo y que no resulta solo de la instrucción, sino que, de la experiencia en situaciones concretas de ejercicio ocupacional. Se definen competencias básicas, específicas y generales o genéricas (OIT, 2002) .

Conjunto integrado de conocimientos, saberes, habilidades, destrezas, actitudes y comportamientos que las personas ponen en juego para desempeñarse eficazmente en distintas organizaciones y contextos laborales. Existen competencias básicas, transversales o genéricas y específicas o técnicas, siendo una combinación de ellas, las requeridas en los perfiles ocupacionales del mercado de trabajo (Schkolnik, Araos, & Machado, 2005).

En esta investigación se toma la ofrecida en (ONN, 3000:2007):Conjunto sinérgico de conocimientos, habilidades, experiencias, sentimientos, actitudes, motivaciones, características personales y valores, basado en la idoneidad demostrada, asociado a un desempeño superior del trabajador y de la organización, en correspondencias con las exigencias técnicas, productivas y de servicios. Es requerimiento esencial que esas competencias sean observables, medibles y que contribuyan al logro de los objetivos de la organización (ONN, 3000:2007).

Formación por competencias: Proceso de enseñanza-aprendizaje basado en las competencias laborales, que facilita la transmisión de conocimientos, valores y la generación de habilidades, acorde a las actividades del trabajo que se realiza, el cual desarrolla en el participante las capacidades para aplicarlos y movilizarlos, en diferentes contextos y en la solución de situaciones emergentes (ONN, 3000:2007).

Gestión por competencias: Actividades coordinadas para dirigir y controlar una organización con un enfoque basado en las competencias laborales y la capacidad de aprendizaje de los trabajadores. Su objetivo es una organización de calidad y la disposición del colectivo integrado para el logro de los objetivos de la organización (ONN, 3000:2007).

Tipos de competencias

Se puede encontrar en la literatura sobre el tema diferentes tipos de competencias que son clasificadas de distintas maneras por diversos autores. Las que generan un mayor grado de consenso son: Competencias Genéricas y las Competencias Específicas.

- **Competencias genéricas:** Adquiridas en el período escolar y en la práctica del trabajo. Sirven para cualquier actividad profesional. Son apoyadas en bases científicas y tecnológicas y en atributos humanos, tales como creatividad, condiciones intelectuales y capacidad de transferir conocimientos a nuevas situaciones. Son competencias genéricas para la toma de decisión, iniciativa, la empatía y la simpatía, la habilidad numérica y computacional, la habilidad verbal y de conversación (OIT, 2002).
- **Competencias Específicas:** Son aquellas adquiridas en la especialización profesional. No pueden ser transferibles, a no ser indirectamente, por las habilidades adquiridas que puedan ser readaptadas. Los contenidos, mientras, son ligados estrictamente a una especialidad definida (OIT, 2002).

Dimensiones de competencias: Son las pautas de conducta correspondientes a cada competencia propuesta que las hacen observables (Metodología de Gestión por Competencias Asumiendo la Norma Cubana sobre Gestión de Capital Humano, 2011).

GAP o Brecha, revela el déficit o faltante entre lo existente o actual, y lo necesitado, requerido o deseado. El mismo es base para la formación, y en particular para asumir el fundamental concepto de "organización de aprendizaje permanente" (condición que se alcanza por la organización, cuando demuestra que realiza una actividad formativa continua y se caracteriza por la eficacia en la capacitación y desarrollo del capital humano, con alto impacto en la eficiencia y la calidad (ONN, 3000:2007)).

La solución del problema de la medición del GAP o Brecha se alcanza mediante escalas porcentuales, un ejemplo es la *Figura 1* donde muestra cómo se comporta, en una escala porcentual, la brecha entre la "competencia existente" en cada una de las personas evaluadas, y la "competencia requerida" de liderazgo por determinado cargo o puesto de trabajo.

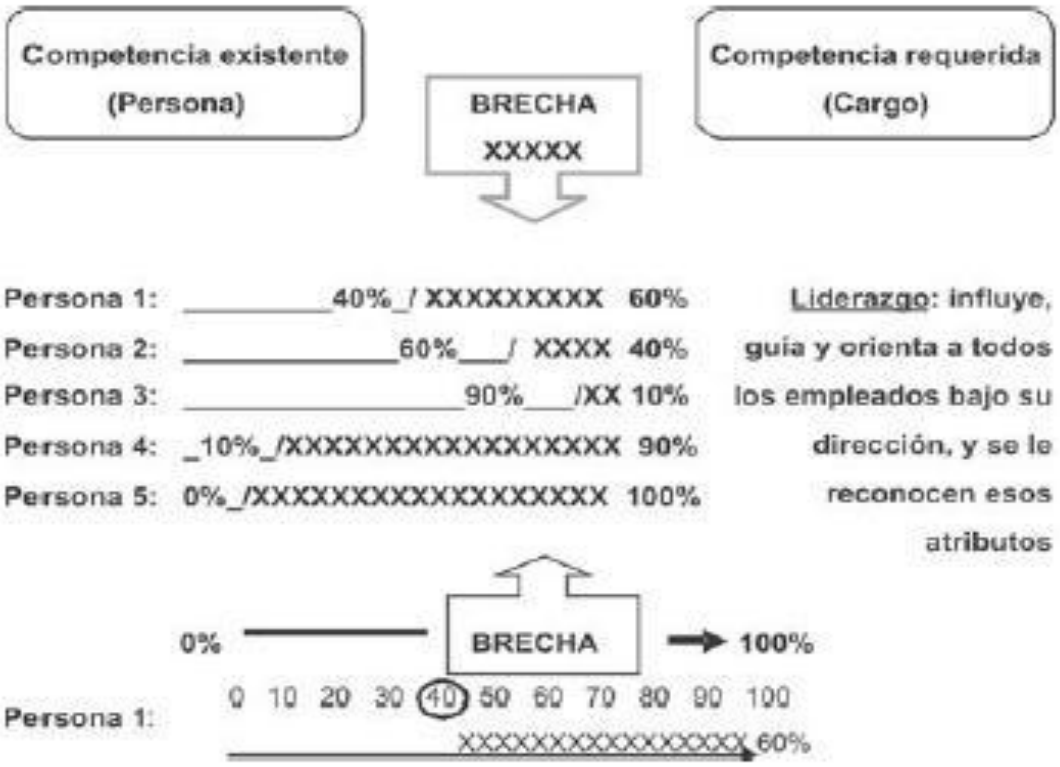


Figura 1 Brechas de competencias laborales en escalas porcentuales. (Metodología de Gestión por Competencias Asumiendo la Norma Cubana sobre Gestión de Capital Humano, 2011).

A continuación se analizaran las normas cubanas para la gestión integrada del capital humano, ya que las mismas deben ser tomadas como referencia en cualquier trabajo de este tema que se desarrolle en Cuba.

1.8 Normas Cubanas

La Oficina Nacional de Normalización (NC), es el Organismo Nacional de Normalización de la República de Cuba y representa al país ante las Organizaciones Internacionales y Regionales de Normalización.

Norma Cubana 3000: 2007

La Norma Cubana (NC) 3000: 2007 o norma de *Sistema de Gestión Integrada de Capital Humano-Vocabulario* es aquella que define los términos más utilizados en la implementación y aplicación de un Sistema de Gestión Integrada de Capital Humano. La importancia de esta norma consiste en que está dirigida a lograr una gestión integrada de capital humano y unificar la terminología utilizada en esta materia.

Norma Cubana 3001: 2007

La Norma Cubana (NC) 3001: 2007 o norma de *Sistema de Gestión Integrada de Capital Humano-Requisitos* establece el conjunto de requisitos a cumplir por las organizaciones para lograr la implementación de un Sistema de Gestión Integrada de Capital Humano, que tiene un impacto en la calidad de todos los procesos, en su eficiencia y eficacia, en el incremento de la productividad, en las relaciones laborales satisfactorias, así como en la respuesta de las

necesidades de las personas que reciben los servicios o adquieren los bienes materiales producidos.

Norma Cubana 3002: 2007

La Norma Cubana (NC) 3002: 2007 o norma de *Sistema de Gestión Integrada de Capital Humano-Implementación* establece un conjunto de precisiones y referencias, que le permiten a las organizaciones conocer cómo implementar el cumplimiento de los requisitos establecidos para el diseño y aplicación de su Sistema de Gestión Integrada de Capital Humano.

A continuación se analizaran como un grupo importante de herramientas de gestión de proyectos automatizan las actividades relacionadas con los recursos humanos.

1.9 Herramientas de gestión de proyectos

Redmine

Redmine es una herramienta de gestión de proyectos de aplicaciones web. Escrito utilizando el *framework* Ruby on Rails, es multiplataforma y como base de datos soporta tanto MySQL como PostgreSQL o SQLite. Es de código abierto y liberado bajo los términos de la GNU *General Public License v2* (GPL) (Redmine.org, 2011).

Características:

- Soporta distintos tipos de proyectos.
- Los usuarios que acceden tienen distintas funciones según el rol asignado, ya sea como usuario, jefe de proyecto, administrador, etc. todo ello en función de un sistema de permisos.
- Permite asignar tareas a los miembros de los equipos.
- Posee un sistema de notificaciones para los usuarios mediante correo electrónico ya sea porque le han asignado una tarea o porque una parte del proyecto ha cambiado o se ha actualizado.
- Tiene la posibilidad de reflejar el desarrollo del proyecto a través de diagramas de Gantt, una herramienta visual que nos ayudará a controlar los progresos del mismo (Tecnología PYME, 2011).

OpenProj

Es multiplataforma, está desarrollado en Java y funciona bajo Windows, GNU/Linux, Unix y Mac OS. Además es completamente gratuito y está traducido a ocho idiomas incluyendo el español.

Presenta distintas funcionalidades de gestión, como pueden ser:

- Gestión de calendarios de trabajo.
- Existencia de diferentes vistas del proyecto (Gantt, diagramas de red, de recursos, histogramas, etcétera).
- Gestión de tareas, con niveles de jerarquía y todas las posibilidades de dependencia necesarias.

- Gestión de recursos (tantos humanos como materiales) asignados a un proyecto.
- Funciones de seguimiento de proyectos, ya sea por dedicación de recursos o simplemente por avance por porcentaje (introducción manual).

Tiene una interfaz muy intuitiva y ligera, sin embargo, tiene algunas carencias, como errores en la traducción o que la estética de las ventanas es la de Java por defecto y no se integra con la del sistema (Tito Catá, 2009).

Microsoft Project

Microsoft Project es una herramienta que permite organizar sus tareas, subtareas, personas, horas de trabajo y recursos necesarios, con la opción de revisar la información mediante distintas vistas, entre las que figuran el diagrama de Gantt, el calendario y el gráfico de recursos.

Además, hace posible trabajar con proyectos nuevos o asesorarse con plantillas. Incluso hace posible que se utilicen los informes gráficos para fijar los resúmenes de los resultados.

(Orrillo, 2010).

Desventajas de MS Project

- Sólo funciona sobre el sistema operativo Windows, no es libre.
- No se puede medir la productividad de las máquinas y las personas, tampoco su rendimiento productivo. Alto costo e inversión (Tito Catá, 2009).

Dotproject

Es una completa herramienta que permite gestionar las distintas fases y tareas que componen un proyecto. A menudo, esta gestión implica un control en recursos humanos, materiales, que hacen que esta labor se torne compleja y prácticamente inabordable sin la ayuda de determinadas herramientas que den soporte a esta tarea de planificación y gestión de proyectos.

Dotproject se perfila como una interesante herramienta para trabajar en entornos colaborativos, permitiendo a los integrantes del equipo trabajar compartiendo información relativa a los proyectos (Izquierdo, 2008).

Dentro de sus características más importantes están:

- Basado en plataforma web permite la participación online de los miembros de un proyecto.
- Permite la asignación de recursos (equipamientos, mobiliario…) a un proyecto o varios, así como la descomposición en tareas
- Permite visualizar eventos y tareas en el calendario.
- Permite a su vez la generación de gran cantidad de informes (Tito Catá, 2009).

Trac

Trac es una herramienta de uso libre con interfaz web, simple y minimalista que integra herramientas para comunicación, gestión, seguimiento de proyecto; y gestión de la configuración (control de versiones) (Camus, 2009).

Trac permite:

- Llevar una gestión basada en Wiki del proyecto, en el que pueden colaborar todos los participantes.
- Sistema de tickets (peticiones de cambios o mejoras de nuestros desarrollos) pudiendo asociarlo a nuestro control de versiones.
- Timeline (eventos de nuestro proyecto) del proyecto (Ignacio, 2007).

Jira

Es una herramienta de gestión de proyectos que ayuda a los equipos a construir mejor software. Se sitúa en el centro de su proceso de desarrollo, conectando su equipo con el trabajo.

Dentro de sus funcionalidades están:

- Gestiona todo y a todos, permite la gestión de tareas, historias, funcionalidades y, por supuesto, bugs. Permite definir sus propios tipos de tarea para monitorizar la información que le importa a su equipo.
- Configura su propio workflow, modifique los workflows de Jira al instante, proporcionando a su equipo estructura y adaptabilidad al mismo tiempo.
- Búsqueda e informes Avanzados, Jira Query Language (JQL) es especialmente útil para encontrar tareas escurridizas y construir cuadros de mando personalizados, desde la interfaz de usuario de Jira hasta los informes personalizados (Atlassian.com, 2011).

TeamworkPM

TeamworkPM es una solución en línea para el trabajo en equipo y gestión de proyectos que ayuda a los gerentes, empleados y clientes trabajar juntos de manera más productiva en línea. Las características incluyen listas de tareas, hitos, cuadernos, mensajes, archivos, registro de riesgos y un calendario mundial.

Características

- Rápida visión general de la actividad reciente.
- Generar informes de tareas personalizado.
- Ver todo el tiempo dedicado registra en un proyecto o en varios proyectos.

Oracle Human Capital Managenent

Oracle tiene la única solución de gestión de capital humano HCM (*Human Capital Managenent*) galardonada y de categoría mundial para organizaciones de cualquier tamaño región y sector.

Sólo Oracle posee una solución HCM global basada en la web que en un solo sistema cubre cada faceta del ámbito HCM: desde las funciones transaccionales básicas de recursos humanos (RRHH) hasta la automatización y prestación de servicios, o soluciones completas de gestión del talento empresarial (Oracle, 2011).

Los beneficios que Oracle muestran son:

- Construir los cimientos básicos de datos y procesos de RRHH que crecerán al ritmo de la empresa.
- Implantar interfaces de usuario flexibles basadas en funciones para generar informes periódicos.
- Supervisar las horas realmente trabajadas respecto a las previstas y gestionar los cambios de calendario.
- Aplicar una solución global flexible basada en reglas para calcular las ausencias acumuladas, la elegibilidad y el salario bruto.
- Implantar datos y transacciones de RRHH para todos los miembros de la empresa.
- Capacitar a directivos y empleados para reducir costes administrativos.
- Aumentar la satisfacción de los empleados con un modelo de servicios compartidos.
- Analizar y modelar la capacitación del personal laboral para planificar con precisión su futuro y liderazgo.
- Atraer y retener a los empleados ideales para incorporarlos a su plan de personal laboral.
- Optimizar la contribución de los empleados suministrándoles la formación adecuada, de la manera más eficaz y más económica.
- Compaginar la contribución de los empleados con las necesidades de la empresa mediante las compensaciones, los planes de rendimiento y de ascenso profesional apropiados.

SAP ERP Human Capital Management

Es un software para la gestión del capital humano, un programa que maximiza el valor de los empleados y del mismo modo los integra con procesos y estrategias, a fin de alcanzar los objetivos empresariales.

SAP ERP HCM proporciona funcionalidades integradas y para toda la empresa que:

- Optimiza los procesos de HCM y los integra a la perfección en todas las operaciones globales.
- Proporciona acceso en tiempo real a la información que acelera la toma de decisiones por parte del personal.
- Le permite asignar las personas correctas a los proyectos adecuados y en el momento oportuno.
- Da soporte tanto a los empleados como a los directivos a lo largo de todo el ciclo de vida del empleado.

La solución coordina e integra datos maestros que se utilizan en los procesos de gestión del capital humano (HCM) y permite que los departamentos de Recursos Humanos realicen las actividades siguientes:

- Analizar las aptitudes y las cualificaciones de los empleados.
- Evaluar los procesos de contratación y los programas de aprendizaje.
- Evaluar la preparación del programa de sucesión de los empleados para el progreso de la carrera profesional.
- Supervisar el progreso hacia los objetivos de la empresa.
- Analizar la rentabilidad de los programas de compensación (SAP AG., 2008).

Cornerstone OnDemand

Cornerstone OnDemand es el proveedor global líder de una completa solución de gestión del talento y el aprendizaje. Su solución consiste en cinco plataformas integradas para la gestión de la formación, las redes sociales empresariales, la gestión del desempeño, la planificación de la sucesión y la empresa extendida. Sus clientes usan su solución para permitir a sus empleados evolucionar a lo largo de su trayectoria profesional, retenerles de forma eficaz, mejorar la explotación del negocio, crear futuros líderes e integrarse en redes externas de clientes, proveedores y distribuidores. (Haworth, 2011).

Desde su creación ha trabajado conjuntamente con los recursos humanos, el talento, y los ejecutivos de aprendizaje para construir su plataforma de gestión del talento para el aprendizaje y el desarrollo, la gestión del rendimiento, planificación de la sucesión, el cumplimiento y gestión de la compensación.

Cornerstone OnDemand ofrece el aprendizaje y el software de gestión del talento y de servicios de apoyo a los recursos humanos estratégicos (recursos humanos) y las iniciativas de la compañía, tales como la planificación de la preparación talento, desarrollo de liderazgo, planificación de carreras, la incorporación de los empleados, pagar por los programas de desempeño, el cumplimiento de la empresa y la gestión de la certificación, socio de canal formación, capacitación de los clientes, y los programas de compromiso de los empleados. Cornerstone soluciones se basan en un multi-tenant, multi-usuario de software como servicio (SaaS) de la arquitectura (OnDemand, 2011).

1.10 Metodologías de desarrollo del software

El proceso de desarrollo de un software es riesgoso y difícil de controlar por lo que es necesario llevar a cabo una metodología que vaya guiando el proceso a fin de obtener un producto con calidad. (Otero Cutiño, 2007).

Las metodologías de desarrollo de software surgen con el objetivo de minimizar el tiempo, hacen muy eficiente y reproducible el camino para obtener resultados confiables, contienen procedimientos de gestión que coordinan y guían técnicas, que determinan las herramientas necesarias para garantizar un eficaz soporte a automatizar (Silveira Orozco, y otros, 2010). Estas son un conjunto de procedimientos, técnicas y ayudas a la documentación para el

desarrollo de productos software (García Martínez, y otros, 2007). Están clasificadas en metodologías tradicionales y metodologías ágiles:

Metodologías tradicionales o Pesadas: son aquellas con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado. Estas metodologías imponen una disciplina de trabajo sobre el proceso de desarrollo del software, con el fin de conseguir un software más eficiente. Para ello, se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto software. Se centran especialmente en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada.

Entre las metodologías tradicionales o pesadas podemos citar:

- RUP (Rational Unified Procces)
- MSF (Microsoft Solution Framework)
- Win-Win Spiral Model
- Iconix

Metodologías ágiles: estas surgen de la necesidad de proveer respuestas rápidas y que sean adaptables al cambio. El desarrollo de manera ágil en estos tiempos resulta indispensable ya que las necesidades de un cliente pueden sufrir cambios importantes desde el momento de contratación de un software al momento de su entrega; y es mucho más importante satisfacer las necesidades del cliente. Esto requiere procesos de software que en lugar de rechazar los cambios sean capaces de incorporarlos.

Los procesos ágiles son una buena elección cuando se trabaja con requisitos desconocidos o variables. Por otra parte, los procesos de desarrollo adaptativos también facilitan la generación rápida de prototipos y de versiones previos a la entrega final, lo cual agradará al cliente.

Entre las metodologías ágiles más destacadas hasta el momento se pueden nombrar:

- XP (Extreme Programming)
- Scrum
- Crystal Clear
- DSDM (Dynamic Systems Developmemt Method)
- FDD (Feature Driven Development)
- ASD (Adaptive Software Development)
- XBreed
- Extreme Modeling

La *Tabla 1* recoge esquemáticamente las principales diferencias de las metodologías ágiles con respecto a las tradicionales. Estas diferencias que afectan no sólo al proceso en sí, sino también al contexto del equipo así como a su organización.

Metodologías Ágiles	Metodologías Tradicionales

Basadas en heurísticas provenientes de prácticas de producción de código.	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo.
Especialmente preparadas para cambios durante el proyecto.	Cierta resistencia a los cambios.
Impuestas internamente (por el equipo).	Impuestas externamente.
Proceso menos controlado, con pocos principios.	Proceso mucho más controlado, con numerosas políticas/normas.
No existe contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Más artefactos.
Pocos roles	Más roles.
Menos énfasis en la arquitectura del software.	La arquitectura del software es esencial y se expresa mediante modelos.

Tabla 1 Diferencias entre Metodologías Ágiles y no Ágiles.

Tomando las ideas de la *Tabla 1* podemos decir que las metodologías tradicionales presentan los siguientes problemas a la hora de abordar proyectos:

- Existen unas costosas fases previas de especificación de requisitos, análisis y diseño. La corrección durante el desarrollo de errores introducidos en estas fases será costosa, es decir, se pierde flexibilidad ante los cambios.
- El proceso de desarrollo está encorsetado por documentos firmados.
- El desarrollo es más lento. Es difícil para los desarrolladores entender un sistema complejo en su globalidad.

Las metodologías ágiles presentan diversas ventajas, entre las que podemos destacar:

- Capacidad de respuesta a cambios de requisitos a lo largo del desarrollo.
- Entrega continua y en plazos breves de software funcional.
- Trabajo conjunto entre el cliente y el equipo de desarrollo.
- Importancia de la simplicidad, eliminado el trabajo innecesario.
- Atención continua a la excelencia técnica y al buen diseño.
- Mejora continua de los procesos y el equipo de desarrollo (Amaro Calderón, 2007).

La *Tabla 2* muestra una comparación de diferentes metodologías en cuanto a las características del Proyecto.

Modelo de Proceso	Tamaño del Proceso	Tamaño del Equipo	Complejidad del Problema
RUP	Medio / Extenso	Medio / Extenso	Medio / Alto
ICONIX	Pequeño / Medio	Pequeño / Medio	Pequeño / Medio
XP	Pequeño / Medio	Pequeño	Medio / Alto
SCRUM	Pequeño / Medio	Pequeño	Medio / Alto

Tabla 2 Comparación de diferentes metodologías en cuanto a las Características del Proyecto (Roberth G. Figueroa, 2011).

En este cuadro se presenta una comparativa de las modelos de proceso en cuanto a las características del proyecto, analizamos el tamaño del proceso, del equipo y la complejidad del problema para cada uno de los modelos. Podemos resaltar que: con un pequeño equipo de desarrollo se puede realizar grandes proyectos, de alta complejidad; es el caso de XP y SCRUM (Roberth G. Figueroa, 2011).

Las metodologías ágiles están especialmente indicadas para proyectos con requisitos poco definidos o cambiantes. Estas se aplican bien en equipos pequeños que resuelven problemas concretos, lo que no está reñido con su aplicación en el desarrollo de grandes sistemas, ya que una correcta modularización de los mismos es fundamental para su exitosa implantación. Dividir el trabajo en módulos abordables minimiza los fallos y el coste. Es por ello que se selecciona las metodologías ágiles para el desarrollo de esta investigación, además de que el sistema GESPRO se desarrolló bajo este tipo de metodologías.

Scrum

Scrum desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días si bien pueden contemplarse casos de hasta 60. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo proyecto, entre ellas destaca una reunión para determinar el trabajo que se va a realizar, otra al final para evaluar el resultado, y revisiones diarias que realiza el equipo en su auto-gestión (ver *Figura 2*) (José H. Canós, 2003).

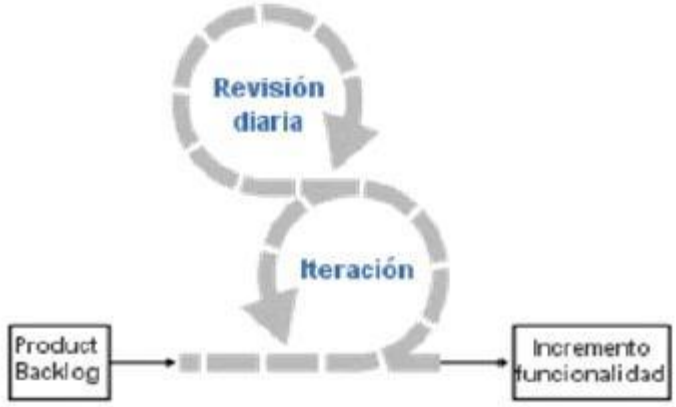


Figura 2 Representación del proceso de desarrollo de la metodología Scrum (Palacio, 2006).

Scum se basa en:

- El desarrollo incremental de los requisitos del proyecto en bloques temporales cortos y fijos (iteraciones de un mes natural y hasta de dos semanas, si así se necesita).
- La priorización de los requisitos por valor para el cliente y coste de desarrollo en cada iteración.
- El control empírico del proyecto. Por un lado, al final de cada iteración se demuestra al cliente el resultado real obtenido, de manera que pueda tomar las decisiones necesarias en función de lo que observa y del contexto del proyecto en ese momento. Por otro lado, el equipo se sincroniza diariamente y realiza las adaptaciones necesarias.
- La potenciación del equipo, que se compromete a entregar unos requisitos y para ello se le otorga la autoridad necesaria para organizar su trabajo.
- La sistematización de la colaboración y la comunicación tanto entre el equipo y como con el cliente.
- El timeboxing de las actividades del proyecto, para ayudar a la toma de decisiones y conseguir resultados.

1.11 Herramienta de modelado

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque Orientado a Objetos (Visual Paradigm International Ltd., 2007).

Permite aumentar la calidad del software, a través de la mejora de la productividad en el desarrollo y mantenimiento del software. Aumenta el conocimiento informático ayudando así a

la búsqueda de soluciones para los requisitos. También permite la reutilización del software, portabilidad y estandarización de la documentación, además del uso de las distintas metodologías propias de la Ingeniería del Software (EcuRed, 2010).

Algunas de sus características son:

- Entorno de creación de diagramas para UML 2.1.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad de integrarse en los principales IDEs.
- Disponibilidad en múltiples plataformas (Sierra, 2008).

1.12 Tendencias, tecnologías y herramientas

Ubuntu

Es una distribución de Linux basada en Debian GNU/Linux, y predominantemente enfocado en la facilidad de uso e instalación, la libertad de los usuarios, y los lanzamientos. El nombre proviene del concepto africano ubuntu, que significa "humanidad hacia otros" o "yo soy porque nosotros somos". Al igual que otras distribuciones se compone de múltiples paquetes de aplicaciones normalmente distribuidos bajo una licencia libre o de código abierto.

La filosofía de Ubuntu se basa en los siguientes principios:

- El usuario debe tener la libertad de descargar, ejecutar, copiar, distribuir, estudiar, compartir, cambiar y mejorar su software para cualquier propósito, sin tener que pagar derechos de licencia.
- Debe ser capaz de utilizar su software en el idioma de su elección.
- Debe ser capaz de utilizar todo el software independientemente de su discapacidad.

Esta distribución ha sido y está siendo traducida a más de 130 idiomas, y cada usuario es capaz de colaborar voluntariamente a esta causa, a través de Internet. Los desarrolladores de Ubuntu se basan en gran medida en el trabajo de otros proyectos de aplicaciones de código abierto, pero en especial en el de la comunidad de Debian.

Ruby

Es un lenguaje de programación dinámico y de código abierto enfocado en la simplicidad y productividad. Su elegante sintaxis se siente natural al leerla y fácil al escribirla. Es un lenguaje con un balance cuidado. Ruby es totalmente libre. No sólo gratis, sino también libre para usarlo, copiarlo, modificarlo y distribuirlo.

Dentro de sus funcionalidades se encuentran:

- manejo de excepciones, como Java y Python, para facilitar el manejo de errores.
- un verdadero mark-and-sweep garbage collector para todos los objetos de Ruby. No es necesario mantener contadores de referencias en bibliotecas externas.
- escribir extensiones en C para Ruby es más fácil que hacer lo mismo para Perl o Python, con una API muy elegante para utilizar Ruby desde C. Esto incluye llamadas para embeber Ruby en otros programas, y así usarlo como lenguaje de scripting. También está disponible una interfaz SWIG.
- puede cargar bibliotecas de extensión dinámicamente si lo permite el sistema operativo.
- tiene manejo de hilos (threading) independiente del sistema operativo. De esta forma, tienes soporte multi-hilo en todas las plataformas en las que corre Ruby, sin importar si el sistema operativo lo soporta o no, ¡incluso en MS-DOS!
- Ruby es fácilmente portable: se desarrolla mayoritariamente en GNU/Linux, pero corre en varios tipos de UNIX, Mac OS X, Windows 95/98/Me/NT/2000/XP, DOS, BeOS, OS/2, etc. (Comunidad de Ruby, 2011).

Sistema Gestor de BD

Un Sistema Gestor de Bases de Datos (SGBD) es un sistema de software cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones, además que permiten a los usuarios definir, crear y mantener la base de datos y proporciona un acceso controlado a la misma (Fidel Gil, 2005). Los SGBD relacionales son una herramienta efectiva que permite a varios usuarios acceder a los datos al mismo tiempo. Brindan facilidades eficientes y un grupo de funciones con el objetivo de garantizar la confidencialidad, la calidad, la seguridad y la integridad de los datos que contienen, así como un acceso fácil y eficiente a los mismos (EcuRed, 2011).

PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando (Martínez, 2010).

Brinda un control de concurrencia multi-versión (MVCC por sus siglas en inglés) que permite trabajar con grandes volúmenes de datos; soporta gran parte de la sintaxis SQL y cuenta con un extenso grupo de enlaces con lenguajes de programación.

Debido a la liberación de la licencia, PostgreSQL se puede usar, modificar y distribuir de forma gratuita para cualquier fin, ya sea privado, comercial o académico (EcuRed, 2011).

Netbeans

NetBeans es una plataforma para el desarrollo de aplicaciones de escritorio usando el lenguaje Java y un entorno de desarrollo integrado (IDE) para desarrollar bajo esta plataforma, pero

también admite otros lenguajes de programación como C y C++ mediante los cuales se pueden crear aplicaciones gráficas, por ejemplo usando librerías como wxWidgets.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software (Ubuntu.org, 2012).

Conclusiones Capítulo 1

Después de haber analizado las características y funcionalidades de las diferentes herramientas de gestión de proyectos y gestión de recursos humanos se ha concluido la necesidad de desarrollar un nuevo módulo para la gestión de los recursos humanos del sistema GESPRO debido a que las actuales no cumplen con las restricciones del presente problema.

Muchas de las herramientas de gestión de proyectos presentan módulos para la gestión de los recursos humanos con funciones básicas como guardar los datos personales de los usuarios, asignar una tarea o rol, entre otras, pero muy pocas con orientación a la gestión del talento o las competencias, además aquellas con dichos módulos son sistemas privativos, por lo que no cumplen con los paradigmas de software libre.

El estudio de las diferentes metodologías de desarrollo de software, herramientas CASE y tecnologías de desarrollo permitió seleccionar las más adecuadas que se ajustan para el desarrollo de dicho módulo, en este caso las seleccionadas fueron:

- Scrum como metodología de desarrollo debido a que es una metodología ágil especialmente indicada para proyectos con un rápido cambio de requisitos, además de ser la utilizada en el desarrollo del sistema GESPRO.
- Visual Paradigm como herramienta de modelado porque es multiplataforma, puede integrarse con varios IDEs, permite capacidades de ingeniería directa e inversa y es una herramienta profesional que soporta ciclo completo de desarrollo de software.
- Ruby como lenguaje de programación ya que es dinámico y de código abierto enfocado en la simplicidad y productividad, además de que el Redmine, base del sistema GESPRO, está desarrollado sobre este lenguaje de programación.
- PostgreSQL como gestor de base de datos ya que es el sistema de gestión de bases de datos de código abierto más potente del mercado y posee integridad con una diversidad de lenguajes de programación, como Ruby.
- NetBeans como IDE de desarrollo ya que admite diversos lenguajes de programación como Java, C, C++ y Ruby, y permite que las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos.

CAPÍTULO 2: SOLUCIÓN PROPUESTA

Introducción

En el presente capítulo se realiza la descripción de la propuesta de solución de este trabajo, para lo que se desarrollan las actividades de elicitación, análisis y especificación de requisitos, obteniendo la Especificación de Requisitos Software, que detalla las funcionalidades del sistema y las restricciones que este debe cumplir; se seleccionan las técnicas para validar los requisitos, y se proponen los patrones de arquitectura, diseño y acceso a datos a utilizar, así como las nuevas clases a agregar a la base de datos.

La figura que se muestra a continuación es una representación de propuesta de solución.

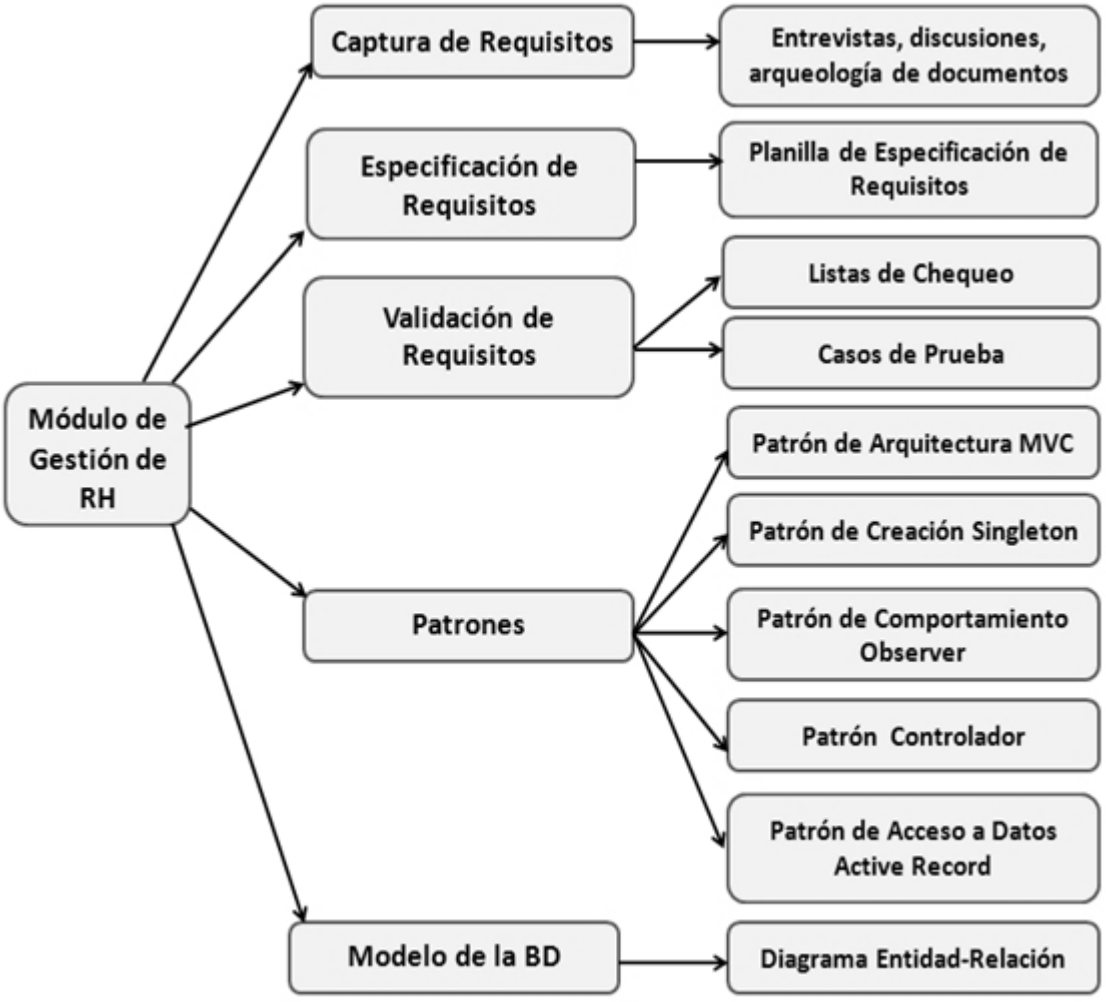


Figura 3 Representación de la propuesta de solución.

Primeramente se realiza la captura de requisitos utilizando métodos como las entrevistas, discusiones, arqueología de documentos; luego se realizará la especificación de los requisitos utilizando la planilla de especificación de requisitos; después se validará dichos requisitos utilizando las listas de chequeo y los casos de prueba; posteriormente se definirán los distintos patrones a utilizar en el diseño del sistema como son el patrón de arquitectura Modelo Vista

Controlador (MVC), patrón Singleton, Observer, Controlador, patrón de acceso a datos *Active Record*; y luego se propone el modelo de la BD mediante el diagrama de entidad-relación.

2.1 Identificación de los requisitos

Atendiendo a las características del sistema para realizar el levantamiento de requisitos se definen las diferentes técnicas de captura de requerimientos que se utilizarán, cuyo objetivo principal es lograr capturar la mayor cantidad de requisitos, de acorde a los intereses del cliente y que aumenten y mejoren las funcionalidades del sistema.

Dentro de las técnicas de captura de requisitos definidas en el epígrafe 1.15 se utilizaron varias como el método de Entrevista, específicamente la variante Discusión debido a que se sostuvo encuentros de discusión con el cliente sobre su problemática para determinar en conjunto los requisitos del sistema y de esta manera se logró una mayor claridad de lo que realmente desea el cliente.

Se realizaron varias "discusiones" para chequear que los requisitos extraídos eran los correctos y que no faltara ningún otro requisito por anotar. De esta manera se logra obtener los requisitos deseados por el cliente para proveer de nuevas funcionalidades al módulo de gestión de recursos humanos. Algunas de las mismas se realizaron con expertos como el Dr. Armando Cuesta. Además se aplicaron otras técnicas como arqueología de documentos donde se revisaron diversos tipos de documentación existente, y entrevistas en grupo de desarrollo.

2.2 Especificación de los requisitos

Los requisitos obtenidos fueron especificados de manera correcta en la plantilla ***0113_Especificación De Requisitos De Software*** con la cual se dejan especificados y descritos de manera correcta los requisitos identificados para la creación del módulo de gestión de recursos humanos del sistema GESPRO. En dicha plantilla se describen y especifican los requisitos funcionales y no funcionales (para obtener mayor detalle dirigirse a al documento adjunto a la investigación).

Requisitos Funcionales: Luego del levantamiento de requisitos se obtuvieron un total de 29 requisitos funcionales, agrupados en 16 grupos. Estos requisitos se describen de manera más específica mediante una tabla, que se encuentra en el documento ***0113_Especificación De Requisitos De Software*** (adjunto a la investigación). Los requisitos funcionales detectados fueron:

El sistema debe permitir:

RF1 Gestionar una competencia.

RF 1.1 Registrar una nueva competencia

RF 1.2 Editar las competencias con sus niveles.

RF 1.3 Eliminar las competencias.

RF 1.4 Definir el nivel deseado o esperado de la competencia para ese rol en ese proyecto.

RF 1.5 Definir la importancia de la competencia para ese rol en ese proyecto.

RF 2 Gestionar una dimensión.
- RF 2.1 Agregar una nueva dimensión.
- RF 2.2 Editar las dimensiones.
- RF 2.3 Eliminar las dimensiones asociadas a una competencia.

RF 3 Asociar una competencia a un rol en un proyecto.

RF 4 Permitir heredar competencias de un proyecto padre.

RF 5 Permitir copiar la configuración de competencias de otro proyecto.

RF 6 Realizar Filtros.
- RF 6.1 Filtrar las dimensiones por competencias.
- RF 6.2 Filtrar las competencias por rol.

RF 7 Gestionar nivel real de la competencia.
- RF 7.1 Insertar el nivel real de la competencia.
- RF 7.2 Modificar el nivel real en la competencia.

RF 8 Mostrar Brechas.

RF 8.1 Mostrar la diferencia entre el nivel real y el nivel deseado (Brecha) de cada competencia por individuo.

RF 8.2 Mostrar las diferencias entre la brecha anterior y la actual en el nivel de las competencias.

RF 9 Registrar una nueva actividad de formación.

RF 10 Gestionar impacto de la actividad de formación.
- RF 10.1 Insertar el impacto de la actividad en la competencia.
- RF 10.2 Modificar el impacto de la actividad.
- RF 10.3 Eliminar el impacto de la actividad.
- RF 10.4 Mostrar el impacto de la actividad en la competencia.

RF 11 Identificar la relación de las tareas de una persona con las dimensiones de las competencias asociadas al rol que desempeñe en el proyecto.

RF 12 Caracterizar las dimensiones de las competencias de acuerdo a los indicadores de eficacia y eficiencia.

RF 13 Calcular el nivel de las competencias de una persona en un proyecto a través de la caracterización que tenga de las dimensiones.

RF 14 Mostrar el costo por actividad.

RF 15 Gestionar programas de formación.
- RF 15.1 Proponer un programa de formación que permita disminuir la brecha de los niveles de las competencias para una persona.
- RF 15.2 Mostrar el programa de formación de una persona.

RF 16 Mostrar por usuario las competencias asociadas a él, agrupadas según los roles que desempeñan.

Para mayores detalles ver *Anexo 1*.

Requisitos no funcionales: Conjuntamente con los requisitos funcionales se extrajeron también los requisitos no funcionales, los cuales también son de gran interés para los clientes. Se

obtuvieron un total de veintiséis requisitos no funcionales agrupados según su clasificación, dichos requisitos se muestran a continuación:

Requisitos de Software:

Software (Servidor de Aplicación)	
Sistema operativo	Ubuntu server 10.04, Debian 4 GNU/Linux o superior
Servidor de aplicación	Apache2
Módulos básicos para el despliegue de Redmine	Ruby1.8, ri1.8, libpgsql-ruby1.8, ruby1.8-dev, librmagick-ruby1.8, libopenssl-ruby1.8, build-essential, apache2-threaded-dev, rake, libxslt-dev, libxml2-dev
Aplicación base	Redmine 1.2
Framework de desarrollo	Ruby on Rails
Paquete de PHP	php5, libapache2-mod-php5, php5-cli,php5-gd, php5-mysql, php5-pgsql, php5-sqlite, php5-xsl.
Componentes de PATDSI-Generador de Reportes	RServer, Chart Server, librería rjson
Control de versiones	Subversion 2.6
Gestor documental	eXcriba 2.0/Alfresco
Software (Servidor de almacenamiento de información)	
Gestor de Base de Datos	PostgreSQL versión 9.0

Requisitos de Hardware:

Hardware (Servidor de aplicación)	
Cantidad	**3**
CPU	**4 x 2.33 GHz** (Intel Xeon 5140 Core2 2.33 GHz)

RAM	**8 Gb**
HDD	**250 Gb** RAID 5
LAN	**2** x NIC (1 Gbit)
SAN	**-**
Fuentes de Alimentación	**2** x 800W
Hardware (Servidor de almacenamiento de información)	
Cantidad	**2**
CPU	**4 x 2.33 GHz** (Intel Xeon 5140 Core2 2.33 GHz)
RAM	**8 Gb**
HDD	**250 Gb** RAID 5
LAN	**2** x NIC (1 Gbit)
SAN	**1.5 Tb**
Fuentes de Alimentación	**2** x 800W

Restricciones en el diseño y la implementación:

- Se utilizará como sistema operativo la distribución de GNU/Linux UBUNTU.
- La herramienta IDE de desarrollo utilizada es el NetBeans 6.9.1.
- La herramienta gestor de base de datos es PostgreSQL 9.0.
- El lenguaje de programación es Ruby on Rails, XML.

Requisitos de apariencia o interfaz externa:

- Interfaz Web:
  - ✓ la interfaz es sencilla con colores suaves a la vista, amigable y sin cúmulo de imágenes u objetos que distraigan al cliente del objetivo.
  - ✓ Debe utilizar como idioma principal el español.
  - ✓ Los botones expresarán su función ya sea mediante su texto o la imagen que acompaña a este.

- La comunicación entre el servidor de aplicaciones y la base de datos se lleva a través del protocolo de conexión TCP/IP.
- La comunicación entre el cliente y el servidor de aplicaciones se lleva a través del protocolo HTTP.
- Portabilidad: el sistema deberá permitir su uso en los diferentes sistemas operativos, como Linux y Windows.

Requisitos de Seguridad:

- Confidencialidad: La información manejada por el sistema deberá estar protegida de acceso no autorizado y divulgación.
- Integridad: el sistema deberá reconocer al usuario a través de su autenticación en el sistema y le permitirá realizar cualquier acción solo si la autenticación es satisfactoria.
- Disponibilidad: el sistema debe estar disponible las 24 horas del día para garantizar la ejecución de las tareas en el momento requerido.

Requisitos de Usabilidad:

- Facilidad de uso por parte de los usuarios: El sistema debe presentar una interfaz amigable que permita la fácil interacción con el mismo y llegar de manera rápida y efectiva a la información buscada. Debe ser una interfaz de manejo cómodo que posibilite a los usuarios sin experiencia una rápida adaptación.
- Especificación de la terminología utilizada: El sistema debe adaptarse al lenguaje y términos utilizados por los usuarios en la rama abordada con vista a una mayor comprensión por parte del cliente de la herramienta de trabajo.
- Potencialidades de capacitación orientadas a interfaces intuitivas, lo que enaltece la posibilidad de que el usuario aprenda mediante el uso y explotación de la herramienta.
- Menús: El sistema debe presentar una serie de menús tanto laterales como en barra de iconos flotantes que permitan el acceso rápido a la información por parte de los usuarios, lo que aprovecha las potencialidades de estas estructuras.
- La aplicación deberá visualizarse con calidad en diferentes navegadores: Mozilla Firefox 3.6 o superior, Microsoft Internet Explorer, Opera, Google Chrome, etc.

Requisitos de Soporte:

- El sistema debe ser de fácil instalación y puesta en ejecución.

Confiabilidad

- El sistema debe ser capaz de mostrar información a las PC Cliente en caso de excepciones como fallas en el sistema o en las comunicaciones.
- El sistema debe permitir regirse por la hora del servidor.

Eficiencia

- El sistema debe soportar un tiempo de respuesta menor o igual a 5 segundos.
- El sistema debe soportar una conexión simultánea de más de 3000 usuarios.

Requisitos para la documentación de usuarios en línea y ayuda del sistema

- Manual de usuario: el sistema debe tener un manual de usuario, permitiendo con ello un correcto uso de sus funcionalidades y brindarle al usuario una mayor experiencia del trabajo con el mismo.
- Documentación actualizada del grupo de desarrollo: se precisa que la documentación del sistema esté actualizada en todos los aspectos, fases de trabajo y ciclos de desarrollo del mismo, permitiendo con ello un respaldo tanto ingenieril como legal del desarrollo de dicho sistema.

Para mayores detalles ver *Anexo 2*.

2.3 Validación de los requisitos

Para lograr la validación de los requerimientos extraídos, se le aplicó al documento de especificación de requisitos, una lista de chequeo propuesta por la universidad como técnica para validar dichos requerimientos.

Listas de Chequeo: estas listas están compuestas por diversas preguntas cuyo propósito es la validación de los requisitos y comprobar la existencia de errores y dificultades existentes. Cuentan con diferentes campos a llenar los cuales se muestran en el documento adjunto a la investigación.

Casos de Prueba basados en requisitos: Otra de las técnicas seleccionadas para validar los requisitos son los Casos de Prueba basados en requisitos los cuales permitirán saber si el sistema está construido según las especificaciones. Por cada requisito se diseña uno o más casos de prueba concentrándose en las partes más importantes ejecutando primero las que cubran los riesgos de mayor prioridad.

Para mayores detalles ver *Anexo 3*.

2.4 Patrones

Para lograr un buen software es necesario lograr un diseño de excelencia para lo cual utilizaremos a los patrones los cuales definen una posible solución correcta para un problema de diseño dentro de un contexto dado, describiendo las cualidades invariantes de todas las soluciones.

Según la escala o nivel de abstracción:

- **Patrones de arquitectura**: Aquellos que expresan un esquema organizativo estructural fundamental para sistemas de software.
- **Patrones de diseño**: Aquellos que expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software.

- **Dialectos**: Patrones de bajo nivel específicos para un lenguaje de programación o entorno concreto.

Patrón de Arquitectura: Modelo-Vista-Controlador.

Es un patrón de diseño de arquitectura que está asociado a la idea de tres capas, se usa principalmente en aplicaciones que procesan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de conceptos para que el desarrollo este estructurado de manera eficiente, facilitando la programación en diferentes capas y de manera paralela e independiente.

Las partes que lo componen son:

- **Modelo**: es responsable de mantener el estado de la aplicación. A veces, este estado es transitorio, que dura sólo un par de interacciones con el usuario. A veces, el estado es permanente y se almacena fuera de la aplicación, a menudo en una base de datos.
- **Vista**: Es responsable de generar una interfaz de usuario, normalmente basada en los datos del modelo. En el caso de una aplicación Web, la Vista es una página HTML con contenido dinámico sobre el cual el usuario puede realizar distintas operaciones.
- **Controlador**: Se encarga de manejar y responder las solicitudes del usuario, procesando así la información necesaria y modificando el modelo en caso de existir cambios, es decir, de organizar la aplicación recibiendo los eventos del exterior (por lo general la entrada del usuario), de interactuar con el modelo, y de mostrar una visión apropiada para el usuario.

La *Figura 4* muestra la arquitectura general del patrón MVC (Modelo-Vista-Controlador o *Model-View-Controller*), mientras que la *Figura 5* muestra una representación de dicho patrón llevado al módulo de gestión de Recursos Humanos donde se muestran un ejemplo de las clases separadas en sus diferentes capas, en ambas se explican los siguientes pasos:

1. El navegador envía una solicitud.
2. El controlador interactúa con el modelo.
3. El contralor invoca a la vista.
4. La vista hace que el navegador muestre la próxima pantalla.

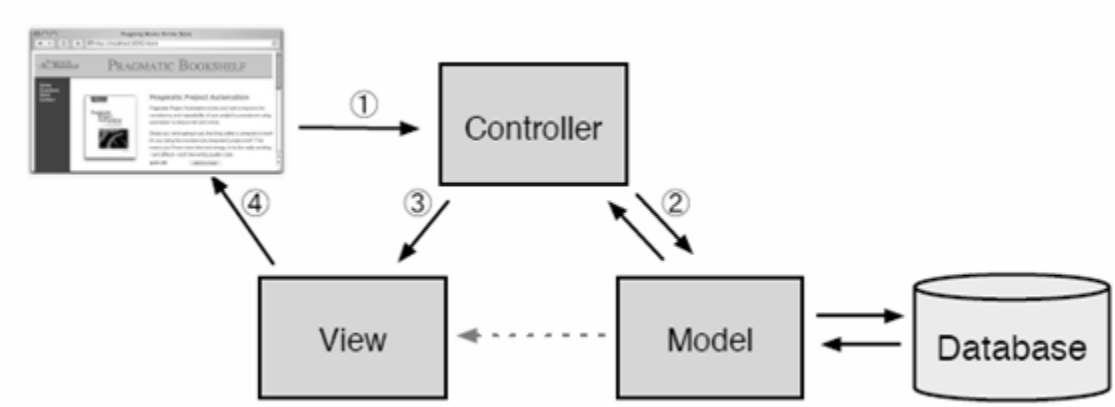


Figura 4 Representación de la arquitectura del patrón Modelo-Vista-Controlador (MVC) (Ruby, y otros, 2010).

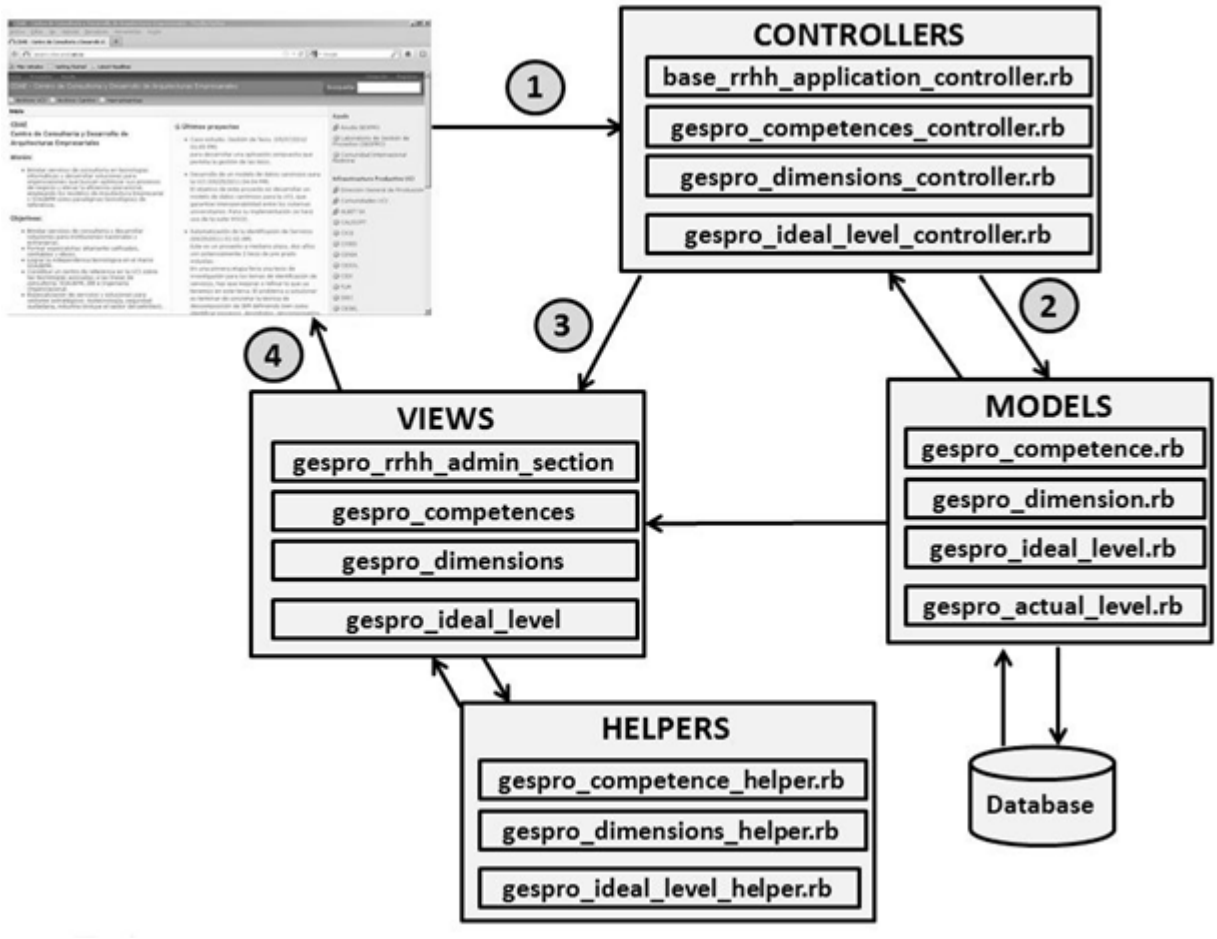


Figura 5 Representación de la arquitectura del patrón Modelo-Vista-Controlador (MVC) para el módulo de Gestión de Recursos Humanos.

Patrón de Creación Singleton (instancia única)

Este patrón restringe la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto, es decir provee una única instancia global lo que posibilitará tener solamente un objeto de la clase.

El patrón *singleton* se implementa creando en nuestra clase un método que crea una instancia del objeto sólo si todavía no existe alguna. Para asegurar que la clase no puede ser instanciada nuevamente se regula el alcance del constructor (con atributos como protegido o privado).

Las situaciones más habituales de aplicación de este patrón son aquellas en las que dicha clase controla el acceso a un recurso físico único o cuando cierto tipo de datos debe estar disponible para todos los demás objetos de la aplicación.

El patrón *singleton* provee una única instancia global gracias a que:

- La propia clase es responsable de crear la única instancia.
- Permite el acceso global a dicha instancia mediante un método de clase.
- Declara el constructor de clase como privado para que no sea instanciable directamente.

Este patrón se encuentra implementado en la base del sistema GESPRO en el archivo *singleton.rb*, es implementado en dicho archivo en el *module Singleton,* y se hace uso de él desde el archivo *observer.rb*.

Patrón de Comportamiento Observer

Permite a los objetos captar dinámicamente las dependencias entre objetos, de tal forma que un objeto notificará a los objetos dependientes de él cuando cambia su estado, siendo actualizados automáticamente.

El objetivo de este patrón es desacoplar la clase de los objetos clientes del objeto, aumentando la modularidad del lenguaje, así como evitar bucles de actualización (espera activa o polling).

Este patrón suele observarse en los marcos de interfaces gráficas orientados a objetos, en los que la forma de capturar los eventos es suscribir 'listeners' a los objetos que pueden disparar eventos.

Este patrón se encuentra implementado en el sistema GESPRO, su implementación principal se encuentra en el archivo *observer.rb* donde se encuentra la clase observer de la cual heredan un conjunto de clases como *CommentObserver* y *JournalObserver*, entre otras que heredan de *ActiveRecord::Observer*. La *Figura 6* muestra una representación del uso de este patrón.

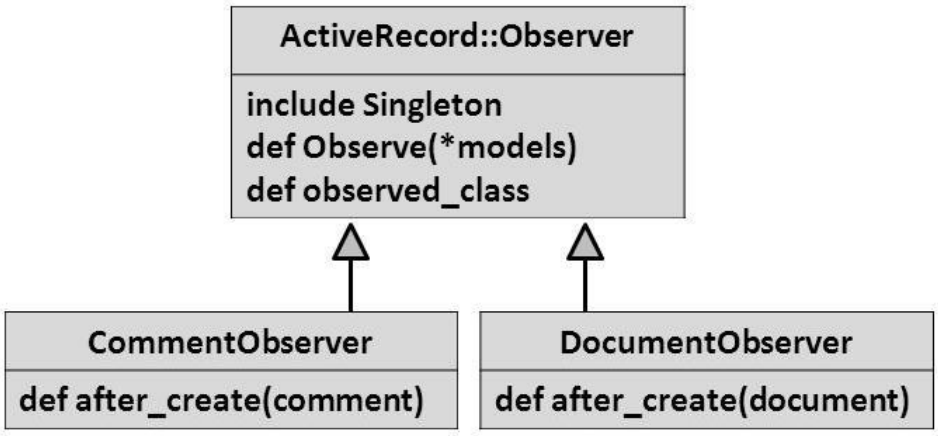


Figura 6 Representación del uso del patrón Observer.

Patrón Controlador

Este patrón se evidencia en las clases controladoras, pues estas tienen la responsabilidad de recibir o manejar los eventos del sistema. En la aplicación se dividen los eventos del sistema en varias clases controladoras para lograr disminuir el acoplamiento y aumentar la cohesión.

Este patrón se encuentra implementado en el módulo de Gestión de Recursos Humanos y en el sistema GESPRO. La clase *BaseRrhhApplicationController* que se encuentra en el archivo *base_rrhh_application_controller.rb* se encarga de controlar al resto de las clases controladoras del módulo de Gestión de Recursos Humanos y a su vez dicha clase hereda de las clases controladoras del sistema GESPRO como *ApplicationController* que hereda a su vez de *ActionController::Base*. La *Figura 7* muestra una representación del uso de este patrón.

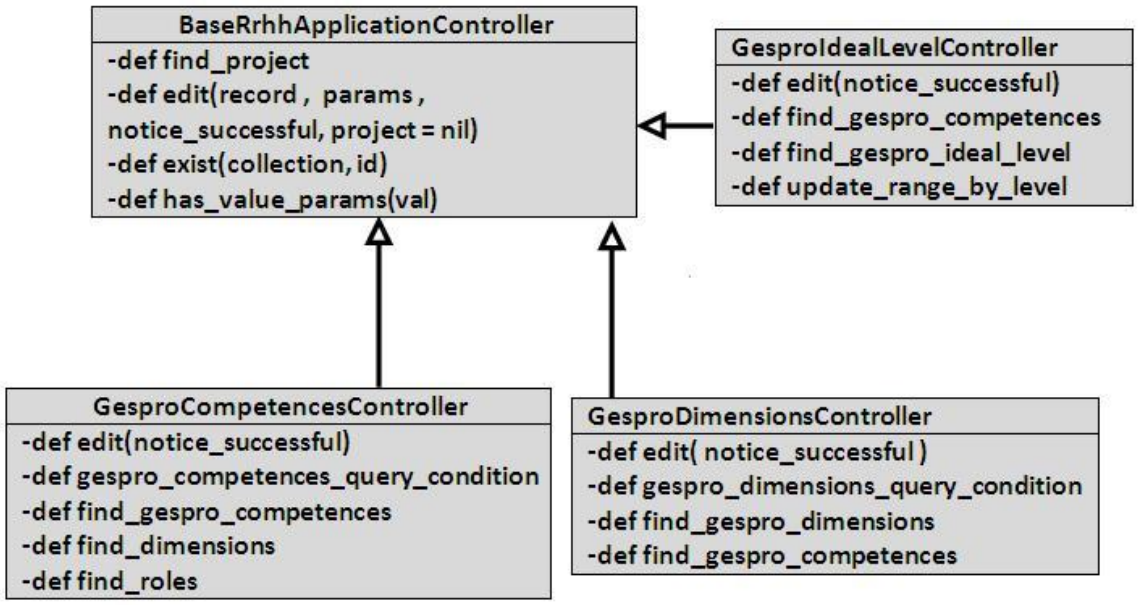


Figura 7 Representación del uso del patrón Controlador.

Patrón de Acceso a Datos *Active Record*

Entre sus características está que el objeto contenga los datos presentes en una fila de la tabla que represente. Además el objeto debe encapsular la lógica necesaria para interactuar con la BD. De esta forma el acceso a datos se realiza de una forma más sencilla y uniforme.

Sirve no solamente para acceder, sino que puede garantizar cierta lógica de dominio en ese acceso, posee métodos que responden no simplemente a la lectura/escritura sino que puede garantizar transformación de dominio sobre los datos.

Una clase Active Record consiste en el conjunto de propiedades que representa las columnas de la tabla más los típicos métodos de acceso como las operaciones CRUD, búsqueda (Find), validaciones, y métodos de negocio.

Este patrón se encuentra implementado en el módulo de Gestión de Recursos Humanos. Un ejemplo es la clase *GesproCompetence,* que se encuentra en el archivo *gespro_competence.rb,* y la misma hereda de *ActiveRecord::Base*. Los cambios que sean necesarios se realizan en la base de datos mediante los objetos de *ActiveRecord*. La *Figura 8* muestra una representación del uso de este patrón.

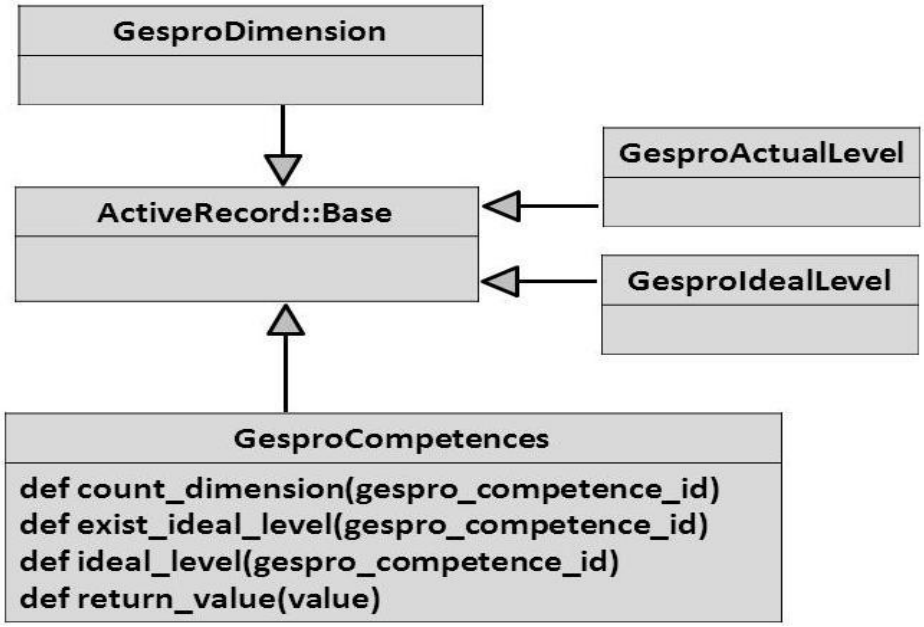


Figura 8 Representación del uso del patrón Active Record.

2.5 Descripción de la Base de Datos

Para la incorporación del Módulo de Gestión de Recursos Humanos al sistema GESPRO fue necesario agregar diferentes tablas a la base de datos para lograr la interacción con el sistema. La *Figura 9* muestra las principales tablas a agregar en la base de datos:

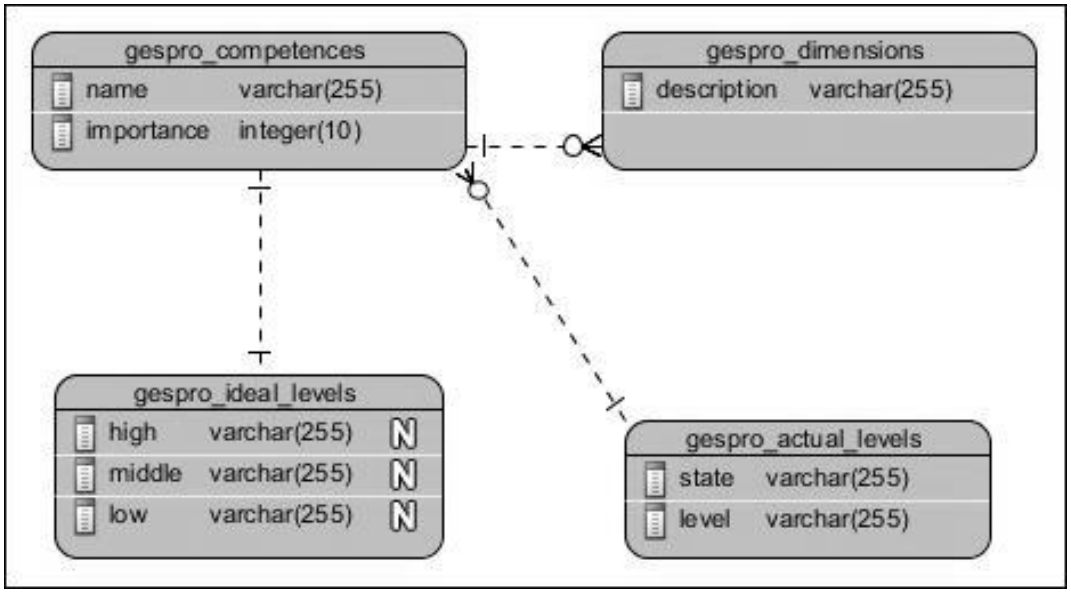


Figura 9 Diagrama Entidad-Relación de las principales tablas a agregar en la Base de Datos.

Gespro_competences: en esta tabla se almacenan las competencias las cuales poseen como atributos un nombre (*name*) y la importancia (*importance*) de la misma.

Gespro_dimensions: en esta tabla se almacenan las dimensiones asociadas a las competencias las cuales poseen como atributo la descripción (*description*) de la misma.

Gespro_ideal_levels: en esta tabla se almacena el nivel ideal de las competencias las cuales poseen como atributos los diferentes niveles: alto (*how*), medio (*middle*) y bajo (*low*).

Gespro_actual_levels: en esta tabla se almacena el nivel actual de las competencias cuyos atributos son: estado (*state*) y nivel (*level*).

Para agregar dichas tablas se utiliza el archivo ***Migrate*****,** el cual son sentencias de DDL (lenguaje de definición de datos), escritas en Ruby, que nos permiten administrar el esquema de la BD. Cuando se ejecuta una *migration* (migración) se cambia la versión de un *schema* a otra anterior o posterior.

Las migraciones son almacenadas en archivos en el directorio db/migrate, uno por cada clase. El nombre del archivo es de la forma *YYYYMMDDHHMMSS_name_file.rb*, esto es la marca de tiempo UTC (Tiempo Universal Coordinado) identificando la migración seguida por un '_' seguido del nombre de la migración. El nombre de clase de la migración debe coincidir con la última parte del nombre de archivo. Por ejemplo *20080906120000_create_products.rb* debe ser definida como *CreateProducts* y *20080906120001_add_details_to_products.rb* como *AddDetailsToProducts*. Si siente la necesidad de cambiar el nombre de archivo entonces debe actualizar el nombre de la clase adentro o simplemente ***Rails*** emitirá un error acerca de la clase faltante. Además los ***migrates*** generan un identificador por defecto.

A continuación se muestran algunos ejemplos de los ***migrates*** *hechos para agregar las tablas a la base de datos.*

20120321170322_create_gespro_competences.rb

```
class CreateGesproCompetences < ActiveRecord::Migration
  def self.up
    create_table :gespro_competences do |t|
          t.column :name, :string, :null => false
            t.belongs_to :role
          t.column :importance, :integer, :null => false
    end
  end
  def self.down
   drop_table :gespro_competences
    end
  end
```

Este *migrates* agrega una tabla llamada *gespro_competences* con una columna de tipo *string* de nombre *name* y una columna de tipo *integer* llamada *importance*. Una columna llave primaria llamada id también es creada por omisión.

La línea *belongs_to* establece una relación con otro modelo, tal que cada instancia del modelo que declara "*belongs_to*" pertenece a otra instancia de otro modelo. En este caso la aplicación incluye competencias y rol, y cada competencia puede ser asignada a un rol.

20120321170502_create_gespro_dimension.rb

```
class CreateGesproDimension< ActiveRecord::Migration
  def self.up
    create_table : gespro_dimensions do |t|
          t.column :description, :string, :null => false
      t.belongs_to :gespro_competence, :null => false
        t.column :importance, :integer, :null => false
    end
  end
  def self.down
    drop_table : gespro_dimensions
  end
end
```

Este *migrates* agrega una tabla llamada *gespro_dimensions* con una columna de tipo *string* de nombre *description* y una columna de tipo *integer* llamada *importance*. Una columna llave primaria llamada id también es creada por omisión.

La línea *belongs_to* establece una relación con otro modelo, tal que cada instancia del modelo que declara "*belongs_to*" pertenece a otra instancia de otro modelo. En este caso la aplicación incluye dimensiones y competencias, y cada dimensión puede ser asignada a una competencia.

20120321170531_create_gespro_actual_levels.rb

```
class CreateGesproActualLevels < ActiveRecord::Migration
    def self.up
        create_table :gespro_actual_levels do |t|
            t.belongs_to :gespro_competence, :null => false
            t.belongs_to :user, :null => false
          t.column :state, :string, :null => false
            t.column :level, :string, :null => false
  end
```

```
 end
    def self.down
          drop_table :gespro_actual_levels
    end
  end
```

Este *migrates* agrega una tabla llamada *gespro_actual_levels* con una columna de tipo *string* de nombre *state* y otra columna de tipo *string* llamada *level*. Una columna llave primaria llamada id también es creada por omisión.

La línea *belongs_to* establece una relación con otro modelo, tal que cada instancia del modelo que declara "*belongs_to*" pertenece a otra instancia de otro modelo. En este caso la aplicación incluye nivel actual y competencias, donde cada nivel actual puede ser asignado a una competencia; e incluye nivel actual y usuario, donde cada nivel actual puede ser asignado a un usuario.

20120321170602_create_gespro_ideal_levels.rb

```
class CreateGesproIdealLevels < ActiveRecord::Migration
    def self.up
      create_table :gespro_ideal_levels do |t|
          t.belongs_to :gespro_competence, :null => false
      t.column :high, :string, :null => false
            t.column :middle, :string, :null => false
            t.column :low, :string, :null => false
        end
      end
     def self.down
    drop_table :gespro_ideal_levels
  end
 end
```

Este *migrates* agrega una tabla llamada *gespro_ideal_levels* con tres columnas de tipo *string* de nombre *high, middle* y *low*. Una columna llave primaria llamada id también es creada por omisión.

La línea *belongs_to* establece una relación con otro modelo, tal que cada instancia del modelo que declara "*belongs_to*" pertenece a otra instancia de otro modelo. En este caso la aplicación incluye nivel ideal y competencias, y cada nivel ideal puede ser asignado a una competencia.

Conclusiones Capítulo 2

En este capítulo se identificaron los requerimientos funcionales y no funcionales, que permitió obtener una descripción en lenguaje natural de las necesidades de los clientes a partir de la aplicación de entrevistas, lo que posibilitó la obtención de la especificación de requisitos software, en la cual se detalla las funcionalidades del sistema y las restricciones que este debe cumplir; se propusieron los patrones de arquitectura, diseño y acceso a datos a utilizar y como los mismos son utilizados en el desarrollo de este módulo, así como el diseño de las nuevas tablas a agregar en la Base de Datos.

CAPÍTULO 3: VALIDACIÓN DE LA SOLUCIÓN

Introducción

En este capítulo se determina la validez y objetividad de la propuesta desarrollada con el objetivo de verificar su calidad; para determinar si los requisitos identificados realmente definen el sistema que se debe construir, así como demostrar la importancia del desarrollo de un módulo de gestión de recursos humanos basados en competencias para el sistema GESPRO. El siguiente diagrama muestra la estructura organizativa de los elementos abordados en la propuesta.

Figura 10 Estructura organizativa del capítulo.

3.1 Resultados de la aplicación de las listas de chequeo

Después de terminado el documento de Especificación de Requisitos, al mismo se le aplica una lista de chequeo, la cual fue propuesta por la Universidad de Ciencias Informáticas (UCI) como técnica de validación. Dicha lista de chequeo posee una serie de preguntas con el objetivo de

validar los requisitos y así detectar la existencia de errores y dificultades a la hora de especificar estos requisitos.

La lista de chequeo cuenta con diferentes campos a llenar como son:

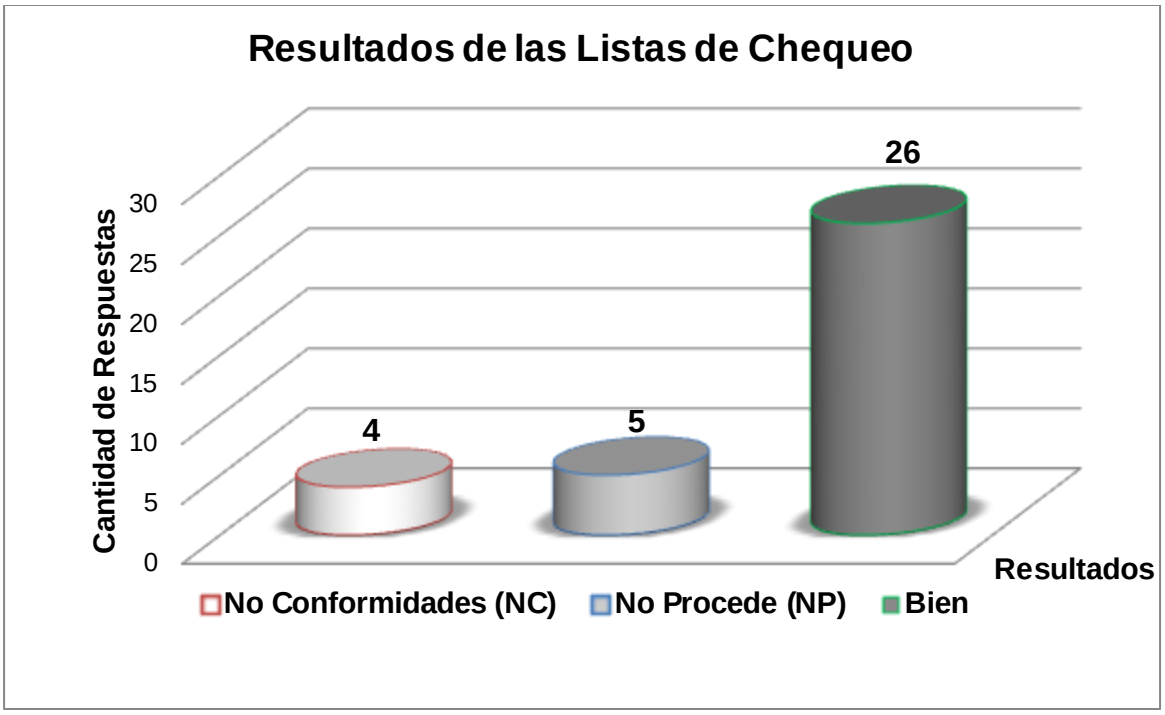
- Peso: Se define si el indicador a evaluar es crítico o no.
- Indicadores a Evaluar: Se especifican los indicadores que se le aplicaran al documento.
- Evaluación (Eval):Se define la evaluación del indicador a evaluar, de 1 en caso de mal y 0 en caso que elemento revisado no presente errores.
- NP (No Procede): Se especifica si el indicador a evaluar no se puede aplicar en ese caso.
- Cantidad de elementos afectados: Especifica la cantidad de errores encontrados sobre el mismo indicador.
- Comentarios: Se recogen los señalamientos o sugerencias que quiera incluir la persona que aplica la lista de chequeo.

En el *Anexo 4* se puede observar más detalladamente la lista de chequeo utilizada para validar los requisitos. Luego de aplicada dicha lista se obtuvieron los siguientes resultados:

De un total de 35 preguntas aplicadas,

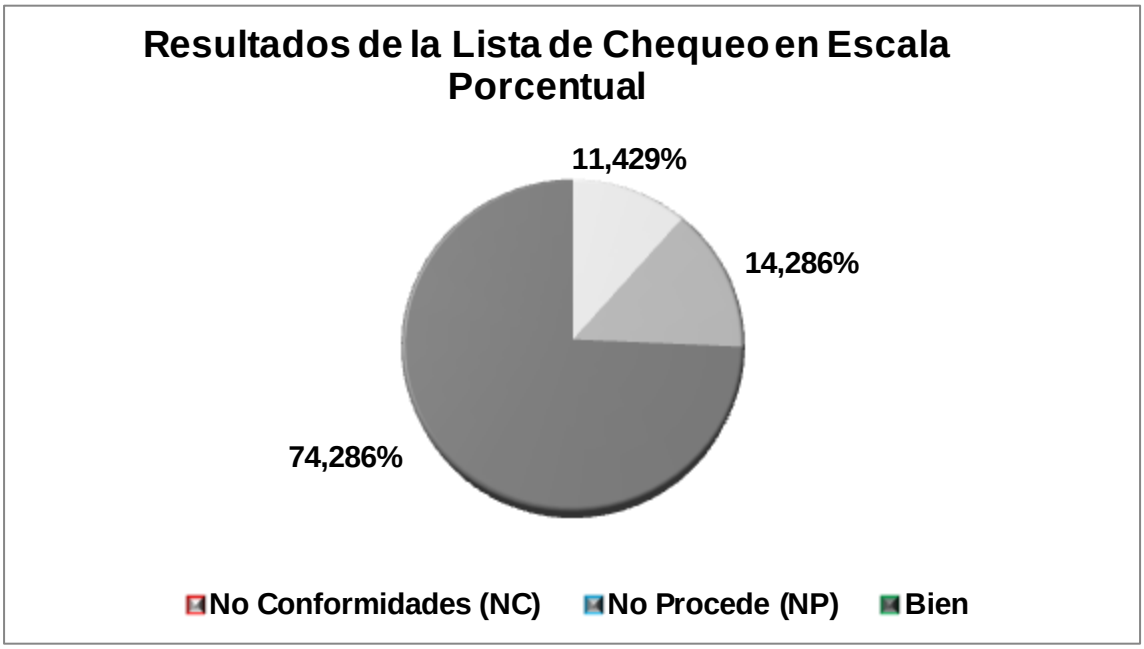
- Cuatro fueron No Conformidades (NC), es decir, cuatro de las preguntas fueron evaluadas de mal. Dichas No Conformidades son referentes a las preguntas:
  - ✓ ¿Debería especificarse algún requisito con más detalle?
  - ✓ ¿Han sido abordadas e identificadas los valores de entradas y salidas?
  - ✓ ¿Se han enumerado los requisitos incluso los que se derivan de otros requisitos?
  - ✓ ¿Se han especificados todas las definiciones y reglas requeridas?
- Cinco fueron No Proceden (NP), es decir, el indicador a evaluar no se puede evaluar en este caso ya que no son objetivos de esta investigación. Dichos indicadores fueron los referentes a las preguntas:
  - ✓ ¿Se puede trazar cada requisito al origen en el entorno del problema, (caso de uso del negocio)?
  - ✓ ¿Existe correspondencia entre el modelo de caso de uso, las Especificaciones Suplementarias y las especificaciones de requerimientos?
  - ✓ ¿Soportan los requerimientos los objetivos del negocio, sistema de software y el proyecto?
  - ✓ ¿El número de página que aparece en el índice coincide con el contenido que se refleja realmente en dicha página?
  - ✓ ¿El total de páginas que aparecen en las reglas de confidencialidad coincide con el total de páginas que tiene el documento?
- El resto de las preguntas, veintiséis, coincidieron con aspectos positivos, es decir estaban bien.

Para lograr un mayor entendimiento de la cantidad de respuestas, la *Gráfica 1* en forma de cilindro, muestra la cantidad de respuestas de cada tipo.



Gráfica 1 Resultados de las Listas de Chequeo primera iteración (Gráfico Cilíndrico).

En la *Gráfica 2* se muestran los resultados de la aplicación de la lista de chequeo, pero haciendo un análisis porcentual de los mismos.



Gráfica 2 Resultados de las Listas de Chequeo primera iteración (Gráfico Circular con Escala Porcentual).

Para evaluar el documento de especificación de requisitos con los resultados obtenidos en la aplicación de las listas de chequeo se utiliza la fórmula expuesta en dicha lista (para mayor información ver el documento adjunto a la investigación *Anexo 4*).

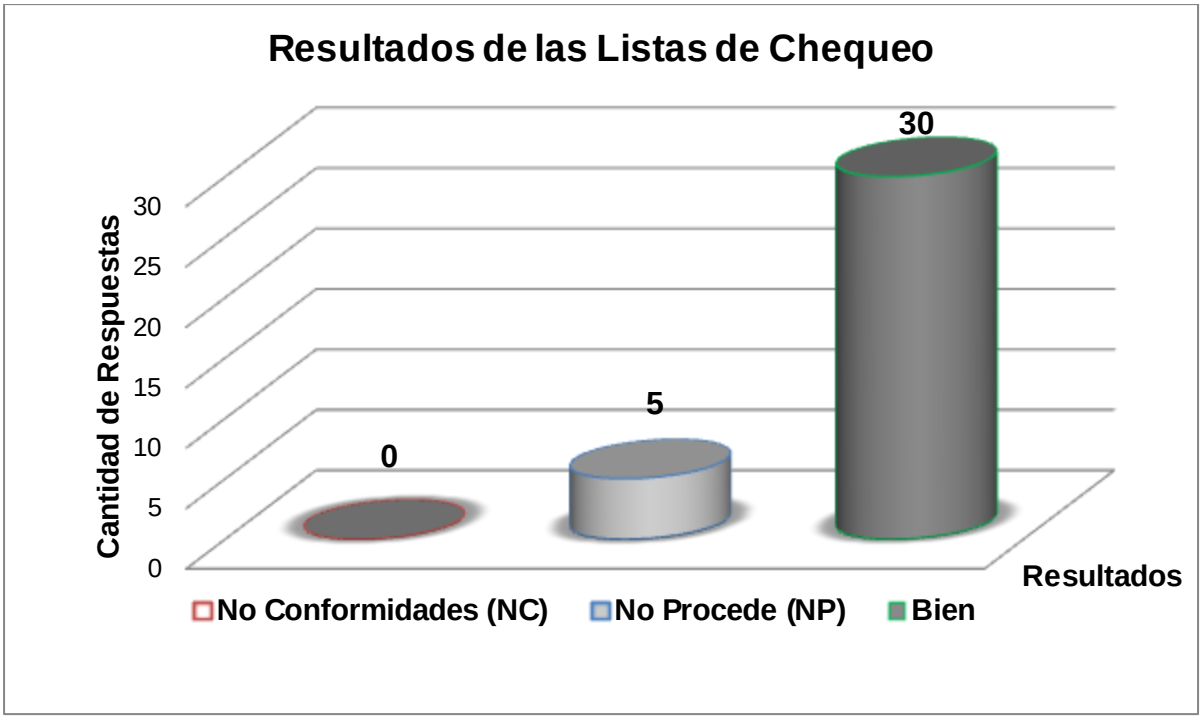
Utilizando la fórmula expuesta en la lista de chequeo, en la primera iteración con la lista de chequeo se obtuvo una evaluación de ***Regular*** debido a que una de las No Conformidades

correspondía a uno de los indicadores de *peso crítico*, este indicador es el que responde a la pregunta de: ¿Han sido abordadas e identificadas los valores de entradas y salidas?

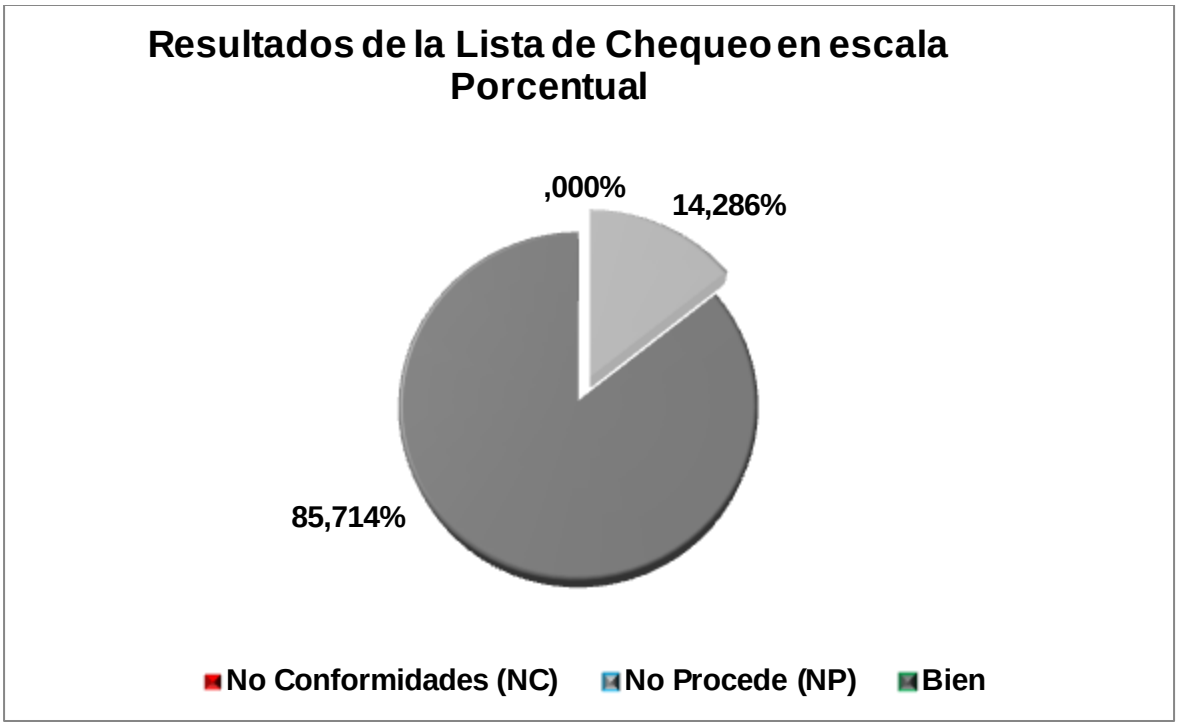
Luego de constatar que aún existían algunos problemas en la especificación de los requisitos, se pasó a realizar una segunda iteración con el objetivo de arreglar las No Conformidades encontradas en el documento, donde luego de realizada la segunda iteración de la lista de chequeo se obtuvieron los siguientes resultados:

- Se eliminaron los problemas encontrados, cero No Conformidades.
- Se mantuvieron los cinco que No Proceden ya que no son objetivos de esta investigación.
- El resto treinta, se respondieron correctamente.

La *Gráfica 3* y la *Gráfica 4* muestran el avance obtenido respecto a las dos gráficas anteriores luego de haber rectificado las deficiencias encontradas en la primera aplicación de la lista de chequeo.



Gráfica 3 Resultados de las Listas de Chequeo segunda iteración (Gráfico Cilíndrico).



Gráfica 4 Resultados de las Listas de Chequeo segunda iteración (Gráfico Circular con Escala Porcentual).

Luego de la segunda iteración con la lista de chequeo se obtuvo una evaluación de ***Bien*** debido a que se eliminaron las No Conformidades, principalmente la correspondía a uno de los indicadores de *peso crítico*.

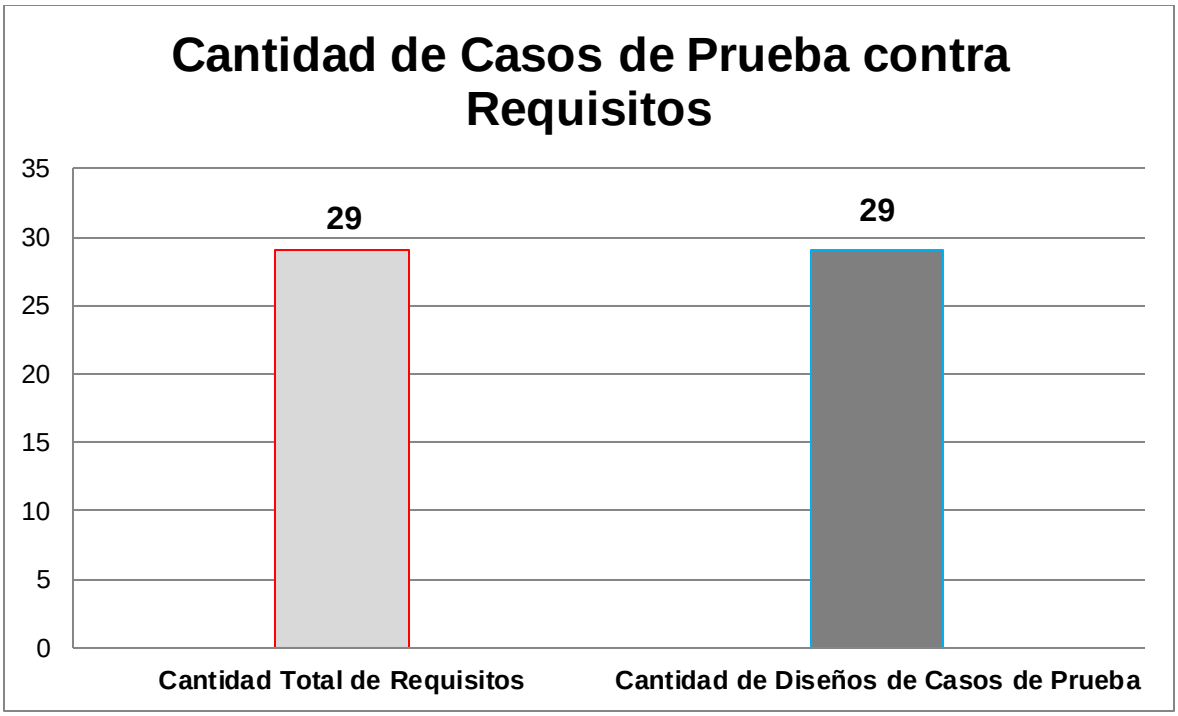
3.2 Resultados de los Casos de Prueba

Luego de la realización de los casos de prueba se obtuvo como resultado que para el módulo de Gestión de Recursos Humanos de los 29 requisitos funcionales, agrupados en 16 grupos, se realizaron 29 casos de prueba (ver *Tabla 3*). Además la *Gráfica 5* expresa la cantidad de casos de prueba que se hicieron por requisitos.

Módulo	**Gestión de Recursos Humanos**
Cantidad Total de Requisitos	**29**
Cantidad de Grupos donde se agrupan los requisitos	**16**
Cantidad de Diseños de Casos de Prueba	**29**

Tabla 3 Resultados de los Casos de Prueba.

Para ver los Casos de Prueba con mayor profundidad ver *Anexo 3*.



Gráfica 5 Cantidad de Casos de Prueba por Requisitos.

3.3 Comparación de GESPRO con otras herramientas

Se realiza una comparación del Sistema GESPRO con algunas de las principales herramientas líderes en la gestión de proyectos para observar la fortaleza y el nivel que ha alcanzado el sistema en comparación con otras herramientas de gestión de proyectos.

La primera comparación se realiza en cuanto a las principales funcionalidades que dichas herramientas presentan (ver *Tabla 4*), mientras que la segunda comparación (ver *Tabla 5*) se realiza en cuanto a los módulos de gestión de recursos humanos.

	Entorno Colaborativo	Control de Seguimiento	Tiempo (Planificación)	Gestión de Portafolio	Gestión de los Recursos	Gestión de Documental	Aplicación Web
GESPRO	Si	Si	Si	Si	Si	Si	Si
dotproject	Si	Si	No	No	No	Si	Si
JIRA	Si	Si	Si	No	No	No	Si
Microsoft Project	No	No	Si	No	Si	No	No
OpenProj	No	No	Si	No	Si	No	No

Oracle Project Portfolio Management	**Si**	**Si**	**Si**	**Si**	**Si**	**Si**	**Si**
Redmine	**Si**	**Si**	**Si**	**Si**	**No**	**Si**	**Si**
SAP RPM	**Si**	**Si**	**Si**	**Si**	**Si**	**Si**	**Si**
TeamworkPM	**Si**	**No**	**Si**	**Si**	**Si**	**Si**	**Si**
Trac	**Si**	**Si**	**No**	**No**	**No**	**No**	**Si**

Tabla 4 Comparación entre las principales funcionalidades de algunas herramientas de gestión de proyectos.

Como se observa en la *Tabla 4* el sistema GESPRO es hoy una de las herramientas más potentes, al nivel de que puede competir con cualquiera de los líderes mundiales en la fabricación de software de gestión de proyectos. Dicha tabla nos refleja cómo el sistema GESPRO es una de las más completas herramientas que existen hoy para la gestión de proyectos, en cuanto a las más importantes funcionalidades que presentan estos sistemas.

	Gestión de RH	**Gestión de las competencias de los RH**	**Licencia**	**Fabricante**
GESPRO	**Si**	**En Desarrollo**	**GPL**	**UCI**
dotproject	**Si**	**No**	**Open Source**	**Dot Marketing**
JIRA	**Si**	**No**	**Propietary**	**Atlassian**
Microsoft Project	**Si**	**No**	**Propietary**	**Microsoft**
OpenProj	**Si**	**No**	**Open Source**	**SourceForge**
Oracle Human Capital Managenent	**Si**	**Si**	**Propietary**	**Oracle**
Redmine	**Si**	**No**	**Open Source**	**Jean-Philippe Lang**
SAP ERP HCM	**Si**	**Si**	**Propietary**	**SAP AG**
TeamworkPM	**Si**	**No**	**Propietary**	**Digital Crew, Ltd.**
Trac	**Si**	**No**	**Open Source**	**Edgewall Software**

Cornerstone OnDemand	**Si**	**Si**	**Propietary**	**Cornerstone OnDemand, Inc**

Tabla 5 Comparación de herramientas de gestión de proyectos en cuanto a módulos de gestión de RH.

La *Tabla 5* nos presenta una comparación de GESPRO con algunas de las herramientas más importantes de gestión de proyectos en cuanto al módulo de gestión de Recursos Humanos, el cual es hoy uno de los aspectos de mayor importancia dentro de una empresa por el nivel de importancia que representan dichos recursos para cualquier empresa.

En dicha tabla se ilustra, como la mayoría de estas herramientas poseen módulos de gestión de recursos humanos, los cuales cumplen funciones básicas como guardar información de los usuarios, asignarles un rol, una tarea, etc.; hay algunas de las mismas que poseen módulos de gestión de recursos humanos basados en competencias o del talento (como le llaman en algunos casos), pero estas herramientas que poseen estos módulos de gestión de recursos humanos basados en competencias con software privativos por lo que no son económicamente fiables para nuestro país.

Por este motivo es de gran importancia el desarrollo de un módulo de gestión de recursos humanos basado en competencias laborales y siguiendo los requerimientos establecidos en la norma cubana 3000, 3001 y 3002 del 2007.

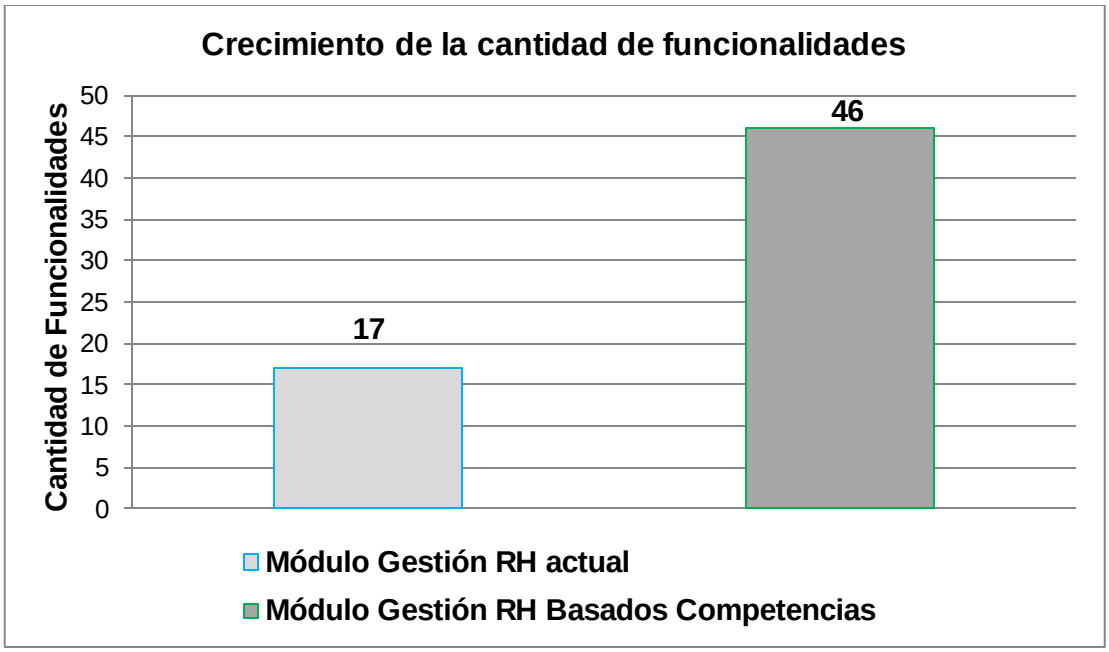
3.4 Evolución del Módulo de Gestión de Recursos Humanos.

El módulo de Gestión de Recursos Humanos juega un papel de gran importancia en el funcionamiento del sistema GESPRO, dicho módulo ha ido evolucionando conjuntamente con el propio sistema en general.

A continuación se hace un análisis de las funcionalidades que posee el módulo de recursos humanos que actualmente posee el sistema GESPRO y el módulo que se propone en esta investigación.

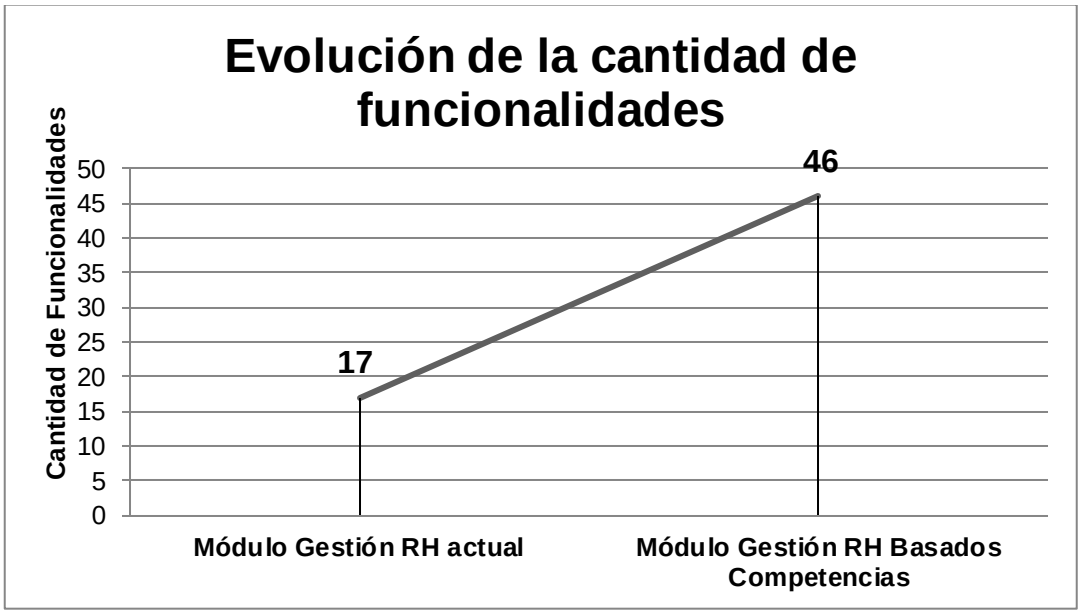
El módulo actual presente en el sistema GESPRO posee un grupo de funcionales centradas en funciones como creación y autenticación de usuarios, guardar datos de usuarios, crear grupos de usuarios, crear perfiles, ver resumen de tareas asignadas a los usuarios, etc., pero ninguna de estas funcionalidades está orientada a la formación de competencias laborales, por lo que se propone agregar un conjunto de nuevas funcionalidades, orientadas a la formación de competencias laborales.

La *Gráfica 6* muestra una representación estimada de la cantidad de funcionalidades que presenta el módulo actual y las que poseerá luego de que le sean añadidas las que se proponen en esta investigación.



Gráfica 6 Crecimiento de la cantidad de funcionalidades del módulo de Gestión de Recursos Humanos.

La *Gráfica 7* muestra una representación estimada de la evolución del módulo en cuanto a la cantidad de funcionalidades.



Gráfica 7 Evolución de la cantidad de funcionalidades.

Luego del análisis realizado entre el módulo actual y el módulo que se propone en esta investigación se puede constatar la importancia que tiene para el sistema GESPRO el desarrollo de este nuevo módulo basado en competencias que proporcionará un aumento gradual de las funcionalidades del módulo al sistema GESPRO.

CAPÍTULO III: VALIDACIÓN DE LA SOLUCIÓN

3.5 Relación con los Requisitos definidos en las Normas Cubanas

El desarrollo total del módulo de Gestión de Recursos Humanos basado en competencias está regido y diseñado siguiendo lo que plantean los estándares internacionales y lo que definen las normas cubanas 3000, 3001 y 3002 del 2007 sobre las competencias laborales y los Sistemas de Gestión Integrada de Recursos Humanos.

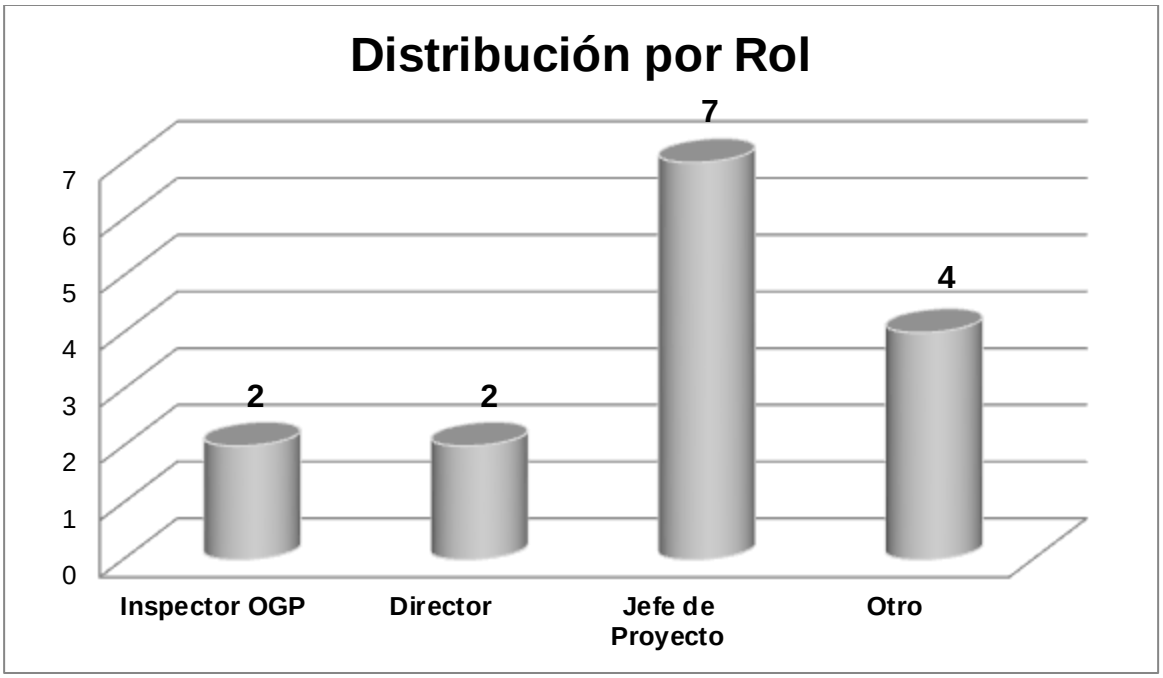
Dentro de las funcionalidades de dicho módulo se observarán aplicados diversos requisitos planteados en las Normas Cubanas 3001 y 3002 del 2007, algunos de ellos son:

- El sistema permitirá identificar a aquellos trabajadores que poseen un nivel de desempeño superior comparado con las competencias y los trabajadores cuyo desempeño es adecuado pero no es superior, para de esta manera poder diferenciar el nivel de desempeño de los trabajadores. (Requisito 4.2.4 NC 3001: 2007)
- Posibilitará utilizar las competencias laborales en los procesos de selección e integración, capacitación y desarrollo y evaluación del desempeño de los trabajadores. (Requisito 4.2.6 NC 3001: 2007)
- Permitirá determinar las necesidades de capacitación y desarrollo para los trabajadores en correspondencia con los roles que desempeñan. (Requisito 4.5.1 NC 3001: 2007)
- Se podrá identificar las brechas que presentan los trabajadores entre el nivel de competencia que necesita para el rol que desempeña y el que realmente posee. (Requisito 4.5.2 NC 3001: 2007)
- Posibilitara elaborar los planes de capacitación individuales de los trabajadores a partir de las brechas identificadas. (Requisito 4.5.4 NC 3001: 2007)
- Permitirá evaluar el impacto, eficiencia y eficacia de las dimensiones de las competencias. (Requisito 4.5.7 NC 3001: 2007)

3.6 Resultados obtenidos de la encuesta para validar la estrategia

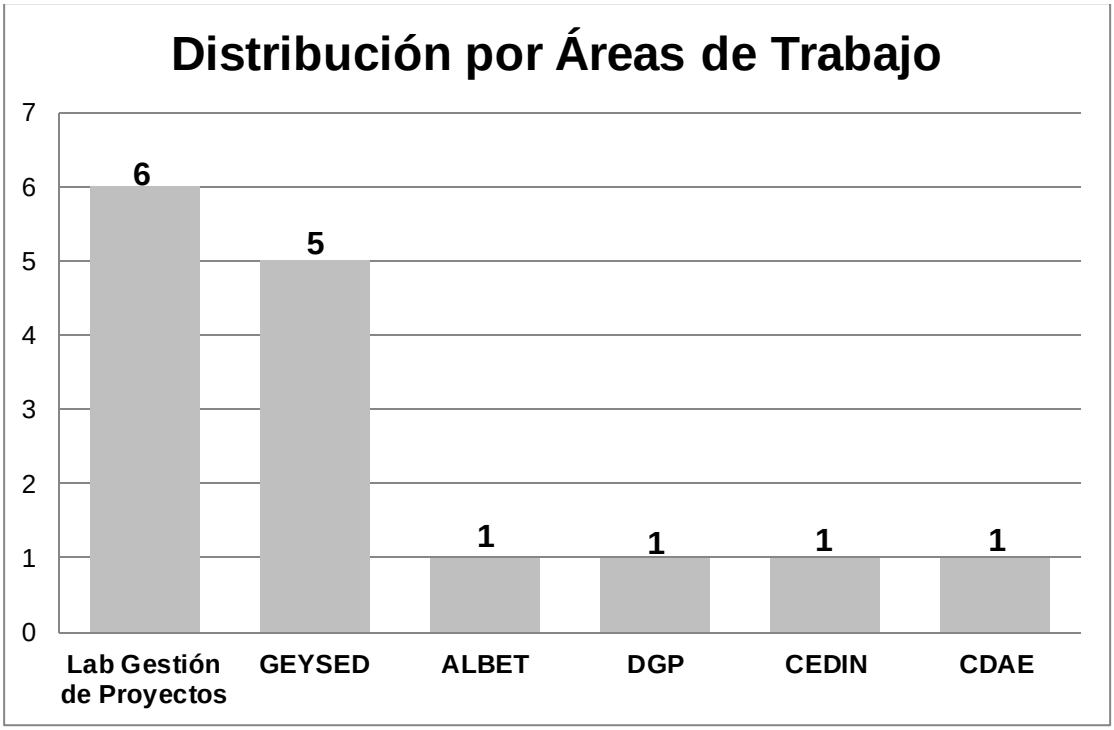
Para validar la propuesta se realiza una evaluación a partir de la valoración de varios profesionales conocedores acerca de la Gestión de Recursos Humanos. Para ello se realizó una encuesta (ver *Anexo 5*) a un total de quince personas.

Dichas personas ocupaban distintos roles (dos Directores, siete Jefe de Proyectos, dos Inspector OGP (Oficina de Gestión de Proyectos) y cuatro Otro) como se ilustra en la *Gráfica 8*.



Gráfica 8 Distribución por Rol de las personas encuestadas.

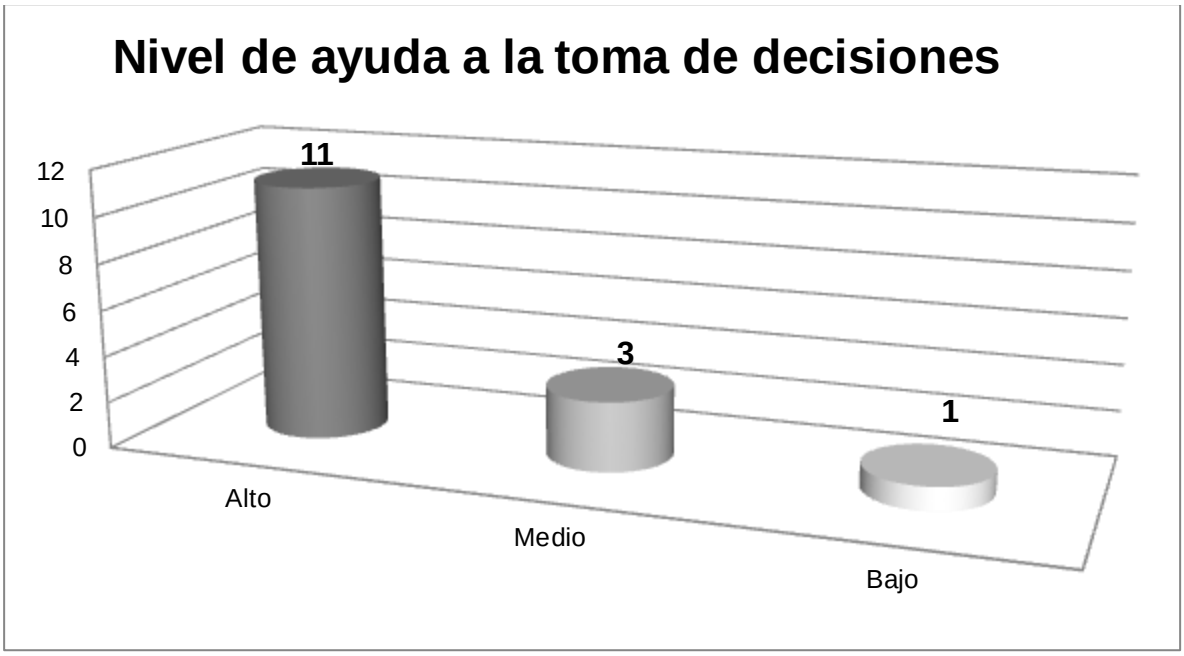
Además de pertenecer a diferentes áreas de trabajo (seis del Laboratorio de Gestión de Proyectos, cinco del Centro de Geoinformática y Señales Digitales (GEYSED), uno de ALBET, uno de Dirección General de Producción (DGP), uno Centro de Informática Industrial (CEDIN) y uno del Centro de Consultoría y Desarrollo de Arquitecturas Empresariales (CDAE)) como se ve en la *Gráfica 9*.



Gráfica 9 Distribución por Áreas de Trabajo de las personas encuestadas.

La encuesta de manera general arrojó los siguientes resultados:

Del total de quince personas encuestadas once expresaron que el sistema, luego de su implementación, permitirá un "**alto**" nivel de ayuda a la toma de decisiones, tres personas expresaron que el nivel de ayuda será "**medio**" y uno que será "**bajo**", como se expresa en la *Gráfica 10*.



Gráfica 10 Resultados de la encuesta.

Los resultados anteriores muestran que la implementación del módulo de Gestión de Recursos Humanos basado en competencias para el sistema GESPRO ayudará a mejorar la toma de decisiones en el proyecto. Los resultados también evidencian que algunos especialistas opinan que deben incluirse tenerse en cuenta ciertos aspectos para mejorar, comprender y facilitar los métodos de calificación e identificación de las competencias que poseen los usuarios; dichas recomendaciones y opiniones se tendrán en cuenta para la implementación del sistema. De manera general la propuesta de los nuevos requisitos tuvo muy buena aceptación por parte de los entrevistados.

3.7 Importancia del módulo de Gestión de Recursos Humanos

Con el desarrollo de este módulo de gestión de recursos humanos para el sistema GESPRO basado en competencias y guidado según lo planteado en las normas cubanas, se pretende:

- Lograr la implementación de un sistema, que tenga un impacto en la calidad de todos los procesos, en su eficiencia y eficacia, en el incremento de la productividad, en las relaciones laborales satisfactorias, así como en la respuesta de las necesidades de las personas que reciben los servicios o adquieren los bienes materiales producidos.
- Permitir a la universidad atraer, retener y desarrollar permanentemente sus trabajadores, así como extender el potencial de sus recursos para lograr concretar los objetivos de la institución.

- Reforzar la soberanía y la seguridad informática de los datos asociados a nuestros proyectos con un sistema basado en las competencias y siguiendo lo que dictan las normas cubanas.
- Un producto comercializado bajo licencia GPL, donde el propio producto y los activos que lo componen son dominados completamente por la Universidad de las Ciencias Informáticas potenciando la independencia tecnológica de nuestro modelo de producción.
- La ayuda a la toma de decisiones en cuanto a la creación adecuada de los planes de formación, brindará una ayuda significativa al control y seguimiento de la evolución de los recursos humanos y el nivel de competencia desarrollado por los mismos.
- El beneficio de los centros de desarrollo de la universidad y de todos a los clientes de GESPRO a la hora de elegir los planes de formación adecuados para cada trabajador, así como las competencias que poseen cada uno.

Conclusiones Capítulo 3

En este capítulo se obtuvieron resultados satisfactorios en la aplicación de las listas de chequeo ya que en el 85.714% de las preguntas se obtuvieron resultados positivos.
Con la inclusión de los nuevos requisitos el módulo de gestión de los recursos humanos de GESPRO gana en funcionalidades que lo colocan a la altura de las principales herramientas de gestión de proyectos a nivel mundial.
El módulo diseñado está en correspondencia con las normas cubanas para la gestión de los recursos humanos.
Con las nuevas funcionalidades identificadas el sistema GESPRO aumentará su ayuda a la toma de decisiones en la gestión de los recursos humanos.

CONCLUSIONES

En el presente Trabajo de Diploma se realizó un estudio sobre las herramientas de gestión de proyectos y de gestión de recursos humanos más utilizados en el mundo, incluyendo sus principales características y funcionalidades; así como de las competencias laborales descritas en la norma cubana. De esta forma se dio cumplimiento al objetivo general de la investigación y se pudo arribar además a las siguientes conclusiones:

- ✓ Se propusieron los requisitos funcionales y no funcionales para el módulo a desarrollar de forma tal que estuviera en concordancia con lo establecido en las Normas Cubanas.
- ✓ Se definió el diseño del sistema para el módulo de gestión de recursos humanos del sistema GESPRO, así como elementos de su arquitectura de software para lograr una implementación que se integrara armónicamente con el resto del sistema ya existente.
- ✓ Se planteó la importancia que tiene, para la universidad y el país, el desarrollo de un módulo de gestión de recursos humanos basado en las competencias laborales.

RECOMENDACIONES

Teniendo en cuenta los resultados y beneficios que proporcionan este Trabajo de Diploma se proponen las siguientes recomendaciones

- ✓ Llevar a cabo la implementación total del módulo de recursos humanos para el sistema GESPRO.

REFERENCIAS BIBLIOGRÁFICAS

Amaro Calderón, Sarah Dámaris, Valverde Rebaza. Jorge Carlos. 2007. *Metodologías Ágiles.* Facultad de Ciencias Físicas y Matemáticas - Escuela de Informática, Universidad Nacional de Trujillo. Trujillo – Perú : s.n., 2007.

Atlassian.com. 2011. Atlassian Jira. *Atlassian Jira.* [En línea] 14 de Noviembre de 2011. [Citado el: 7 de Mayo de 2012.] http://www.atlassian.com/es//software/jira/overview.

Camus, Ariel. 2009. 21projects. [En línea] 6 de Octubre de 2009. [Citado el: 7 de Mayo de 2012.] http://www.21projects.com/blog/gestion-de-proyectos-software-con-trac/.

Chiavenato, Idalberto. 2007. *Administración de Recursos Humanos. El Capital Humano de las Organizaciones.* México DF, México : McGraw-Hill, 2007. 9701061047.

Comunidad de Ruby. 2011. *Ruby a Programmer´s Best Friend.* [En línea] 2011. http://www.ruby-lang.org/es/about/.

Cuesta, Armando. 2010. *Tecnología de Gestión de los Recursos Humanos.* Tercera. La Habana : Editorial Félix Varela, 2010. Corregida y ampliada.

Dolan, Simon L. 2007. *La gestión de los recursos humanos: cómo atraer, retener y desarrollar con éxito el capital humano en tiempos de transformación.* Tercera. Madrid : McGraw-Hill, 2007. ISBN: 9788448156541.

EcuRed. 2011. *EcuRed.* [En línea] 2011. http://www.ecured.cu/index.php/Sistema_Gestor_de_Base_de_Datos.

—. 2010. *EcuRed.* [En línea] 2010. http://www.ecured.cu/index.php/Visual_Paradigm.

Fernández, José Manuel. 1999. *Recursos humanos: Fundamentos del comportamiento humano en la empresa.* Madrid, España : Ediciones Encuentro, 1999. ISBN 8474905036, 9788474905038.

Fidel Gil, Javier Albrigo, Javier Do Rosario. 2005. *SISTEMAS DE GESTIÓN DE BASE DE DATOS SGBD / DBMS.* Facultad Experimental de Ciencias y Tecnología, Departamento de Computación, Universidad de Carabobo. Valencia, Carabobo : s.n., 2005. https://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r61632.PDF.

Fornaris Licea, Yebel y Ibañez Fernández, Yoanna. 2011. *Modelamiento del Negocio e Ingeniería de Requisitos del Sistema Integral de Documentación e Información Judicial (SIDIJ).* Facultad 3, Universidad de Ciencias Informáticas. La Habana : s.n., 2011. Tesis diploma.

García Martínez, Alieny y Guzmán Sánchez, Abel. 2007. *ANÁLISIS Y DISEÑO PARA UN SISTEMA DE INFORMACIÓN DE LABORATORIOS (LIS).* Facultad 7, Universidad de Ciencias Informáticas. La Habana : s.n., 2007. pág. 17, Tesis de Diploma.

González Moreno, Juan Carlos. 2008-2009. Slideshare. [En línea] 2008-2009. [Citado el: 9 de Mayo de 2012.] http://www.slideshare.net/jcgmoreno/tema-1-ingeniera-de-requisitos.

Guach Castillo, Julia. 2004. *GESTION BASADA EN COMPETENCIAS EN LAS ORGANIZACIONES LABORALES.* 2004. pág. 7.

Guillén Suárez, Angel Alexander. 2011. *Ingeniería de Requisitos del procedimiento Diligencias Previas al proceso ejecutivo del Módulo Económico del Proyecto Tribunales Populares Cubanos (TPC).* Facultad 3, Universidad de Ciencias Informáticas. La Habana : s.n., 2011. pág. 13, Tesis de Diploma.

REFERENCIAS BIBLIOGRÁFICAS

Haworth, Michelle. 2011. *Cornerstone OnDemand, Inc.* [En línea] 23 de Mayo de 2011. http://www.cornerstoneondemand.es/cornerstone-ondemand-se-posiciona-como-líder-en-software-de-gestión-del-talento-en-un-nuevo-informe-.

Ignacio, Jose. 2007. XperimentoS. [En línea] 9 de Mayo de 2007. [Citado el: 7 de Mayo de 2012.] http://www.xperimentos.com/2007/05/09/gestion-de-proyectos-trac/.

Izquierdo, Susana. 2008. Abartia Team. [En línea] 2008. [Citado el: 7 de Mayo de 2012.] http://www.abartiateam.com/dotproject.

José H. Canós, Patricio Letelier y Mª Carmen Penadés. 2003. *Métodologías Ágiles en el Desarrollo de Software.* DSIC -Universidad Politécnica de Valencia. Camino de Vera, Valencia : s.n., 2003. 46022.

Laboratorio de Gestión de Proyectos. 2012. [En línea] GESPRO 12.05, Universidad de Ciencias Informáticas, 2012. http://gespro-help.prod.uci.cu/.

Martínez, Rafael. 2010. *PostgreSQL-es.* [En línea] 2 de Octubre de 2010. http://www.postgresql.org.es/sobre_postgresql.

Metodología de Gestión por Competencias Asumiendo la Norma Cubana sobre Gestión de Capital Humano. **Cuesta, Armando. 2011.** [ed.] Dr Joao Maurício Gama Boaventura. 13, 2011, RBGN-Revista Brasileira de Gestao de Negócios, págs. 300-311. ISSN 1806-4892.

Montero Fang-Tac, Eileen. 2011. *Propuesta de Modelos de Negocio para la Comercialización del SCADA UX.* Universidad de Ciencias Informáticas. La Habana : s.n., 2011. Tesis de Diploma.

OIT. 2002. *CERTIFICACIÓN DE COMPETENCIAS PROFESIONALES.* Brasil : Organización Internacional del Trabajo, 2002.

—. 2002. *CERTIFICACIÓN DE COMPETENCIAS PROFESIONALES: GLOSARIO DE TÉRMINOS TÉCNICOS.* Brasil : Organización Internacional del Trabajo, 2002.

OnDemand, Cornerstone. 2011. [En línea] 2011. http://www.vendor-showcase.com/software/95-22595/Human-Resources-HR/Cornerstone-OnDemand-Talent-Management-Suite.html.

ONN. 2007. *Norma Cubana NC 3000:2007.* La Habana : Oficina Nacional de Normalización, 2007.

—. 3000:2007. *SISTEMA DE GESTIÓN INTEGRADA DE CAPITAL HUMANO—VOCABULARIO.* La Habana : Oficina Nacional de Normalización, 3000:2007.

Oracle. 2011. Oracle. [En línea] 26 de Mayo de 2011. [Citado el: 7 de Mayo de 2012.] http://www.oracle.com/es/solutions/hcm/index.html.

Orrillo, Juan. 2010. Descargar Gratis. [En línea] 2010. [Citado el: 7 de Mayo de 2012.] http://www.descargargratis.com/microsoft-project.

Otero Cutiño, Aranay. 2007. *ANÁLISIS Y DISEÑO DEL SOFTWARE CON TECNOLOGÍA MULTIMEDIA EDUCACIÓN FÍSICA CUBANA.* Facultad 8, Universidad de Ciencias Informáticas. La Habana : s.n., 2007. pág. 18, Tesis de Diploma.

Palacio, Juan. 2006. Gestión de proyectos ágil: conceptos básicos. [En línea] 2006. http://www.navegapolis.net.

Pérez Olmos, Yoisy. 2009. *Técnicas y herramientas de la ingeniería de requisitos adecuadas para simuladores virtuales.* Universidad de Ciencias Informáticas. Ciudad de La Habana : s.n., 2009. pág. 62, Tesis Maestría.

REFERENCIAS BIBLIOGRÁFICAS

Piñero Pérez, Pedro Y., y otros. 2011. *GESPRO 11.05 UN SISTEMA PARA LA DIRECCIÓN INTEGRADA DE PROYECTOS PARA LA GESTIÓN DE LA PRODUCCIÓN.* Santa Clara : s.n., 2011. ISBN 978-959-250-658-9.

Pons Achell, Juan Felipe. 2009. *Análisis teórico del PMBOK® y su puesta en práctica en proyectos de Edificación.* Universidad Politécnica de Valencia. Valencia : s.n., 2009. pág. 21, Trabajo Final Master.

Pressman, Roger S. 2002. *Ingeniería de Software: Un enfoque práctico.* [trad.] Darrel Ince. Quinta Edición. s.l. : McGraw-Hill, 2002. págs. 182-236. ISBN: 8448132149.

Pressman, Roger S. 2005. *Ingeniería de Software: Un Enfoque Práctico.* Sexta Edición. s.l. : McGraw-Hill, 2005. págs. 165-180,183, 191-244. ISBN:9701054733.

QAustral. 2010. *MANUAL DE TESTLINK.* Córdoba, Argentina : s.n., 2010. Manual.

Redmine.org. 2011. Redmine. *Redmine.* [En línea] 14 de Noviembre de 2011. [Citado el: 7 de mayo de 2012.] http://www.redmine.org/.

Roberth G. Figueroa, Camilo J. Solís, Armando A. Cabrera. 2011. *METODOLOGÍAS TRADICIONALES VS. METODOLOGÍAS ÁGILES.* Universidad Técnica Particular de Loja, Escuela de Ciencias en Computación. 2011.

Ruby, Sam , Thomas, Dave y Heinemeier Hansson, David . 2010. *Agile Web Development with Rails.* Third. 2010. págs. 22-29. ISBN-10: 1-934356-16-6 / ISBN-13: 978-1-9343561-6-6.

SAP AG. 2008. *SAP.* [En línea] 2008. http://www.logrus.ru/UserFiles/Image/Multilingual/Multi_layout/SAP_ERP_HCM_20_languages/SAP ERP HCM_ES.pdf.

Schkolnik, Mariana, Araos, Consuelo y Machado, Felipe. 2005. *Certificación por competencias como parte del sistema de protección social: la experiencia de países desarrollados y lineamientos para América Latina.* Santiago de Chile : Publicación de las Naciones Unidas, 2005. pág. 10. ISBN: 92-1-322820-1.

Sierra, María. 2008. [En línea] Universidad Cantabria – Facultad de Ciencias, 18 de Octubre de 2008. http://personales.unican.es/ruizfr/is1/doc/lab/01/is1-p01-trans.pdf.

Silveira Orozco, Maritza y Pardo Duarte, Katia. 2010. *Análisis y Diseño del Sistema de Gestión Tecnológica de la Facultad Regional de Granma.* Facultad 8, Universidad de Ciencias Informáticas. La Habana : s.n., 2010. pág. 6.

Socas Alvez, Marisleydis y Rodríguez Verdecia, Lissette. 2007. *MIR-SWG: MODELO DE INGENIERÍA DE REQUISITOS PARA SOFTWARE DE GESTIÓN EN LA FACULTAD 3.* Facultad 3, Universidad de Ciencias Informáticas. La Habana, Cuba : s.n., 2007. Tesis diploma.

Tecnología PYME. 2011. Tecnología PYME. [En línea] 7 de Octubre de 2011. [Citado el: 7 de Mayo de 2012.] http://www.tecnologiapyme.com/software/redmine-gestor-de-proyectos-de-codigo-libre-para-nuestras-empresas.

Tito Catá, Karel. 2009. *Módulo de dotProject para la gestión de estadísticas del Home Compartido.GENEST.* Universidad de Ciencias Informáticas. La Habana : s.n., 2009. págs. 8-12, Tesis de Diploma.

Ubuntu.org. 2012. [En línea] 23 de Febrero de 2012. http://www.guia-ubuntu.org/index.php?title=NetBeans.

Visual Paradigm International Ltd. 2007. *Free Download Manager.* [En línea] 5 de Marzo de 2007.

http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(MÍ)_14720_p/
.

Wikidot. 2007. *Gestflor - SGI Floristerías.* [En línea] 28 de Mayo de 2007. http://gestflori.wikidot.com/pruebasrequerimientosi.

ANEXOS

Anexo 1 Descripción de los Requisitos Funcionales

*Ver la plantilla **0113_Especificación de Requisitos de Software** adjunta a esta investigación, sección referida a los Requisitos Funcionales.*

Anexo 2 Descripción de los Requisitos No Funcionales

*Ver la plantilla **0113_Especificación de Requisitos de Software** adjunta a esta investigación, sección referida a los Requisitos No Funcionales.*

Anexo 3 Casos de Prueba Basados en Requisitos

*Ver la carpeta **Casos de prueba basados en requisitos** adjunta a esta investigación, en la cual se encuentran los diferentes casos de pruebas.*

Anexo 4 Lista de chequeo

*A continuación se muestra las preguntas de la lista de chequeo aplicada. Para obtener mayores detalles ver el documento **Listas de Chequeo Especificación de Requisitos** adjunto a esta investigación, en la cual se encuentran los diferentes casos de pruebas.*

Estructura del documento					
Peso	**Indicadores a Evaluar**	**Eval**	**(NP)**	**Cantidad de elementos afectados**	**Comentarios**
crítico	1. ¿Está el documento acorde con la plantilla estándar del proyecto o del expediente de proyecto?				
crítico	2. ¿Contiene las secciones obligatorias definidas en el expediente? (Ver Expediente de Proyecto)				
Elementos definidos por la metodología					
Peso	**Indicadores a Evaluar**	**Eval**	**(NP)**	**Cantidad de elementos afectados**	**Comentarios**
crítico	1. ¿Se han identificado los requisitos funcionales que son				

	las características que el sistema debe cumplir?				
crítico	2. ¿Se han identificado los requisitos no funcionales del sistema, que son las cualidades que el sistema debe tener?				
crítico	3. ¿Están todos los requisitos redactados de forma simple y clara para aquellos que vayan a consultarlo en un futuro? Ver documento de Especificación de Requisitos.				
	4. ¿Debería especificarse algún requisito con más detalle? Ver documento de Especificación de Requisitos.				
	5. ¿Debería especificarse algún requisito con menos detalle? Ver documento de Especificación de Requisitos.				
crítico	6. ¿Todos los requisitos identificados se centran en lo que el sistema debe hacer y no como el sistema debe hacerlo? Ver documento de Especificación de Requisitos.				
crítico	7. ¿Han sido abordadas e identificadas los valores de entradas y salidas? Ver documento de Especificación de Requisitos.				
	8. ¿Han sido incluidos las				

	respuestas válidas y no válidas de los valores de entrada? Ver documento de Especificación de Requisitos. Nota(puede hacerse referencia a un documento Diccionario de Datos)				
	9. ¿Se han identificado los requerimientos de software y de hardware? Ver documento de Especificación de Requisitos.				
	10. ¿Han sido identificadas las restricciones de diseño e implementación? Ver documento de Especificación de Requisitos.				
	11. ¿Han sido identificadas las restricciones de interfaz externa? Ver documento de Especificación de Requisitos.				
	12. ¿Los requerimientos de soporte, usabilidad, fiabilidad y eficiencia han sido identificados? Ver documento de Especificación de Requisitos.				
	13. ¿Se han identificado los requerimientos de seguridad (confidencialidad, integridad, disponibilidad)? Ver documento de Especificación de Requisitos.				
crítico	14. ¿Se puede verificar				

	cada requisito? (Un requisito se dice que es verificable si existe algún proceso no excesivamente costoso por el cual una persona o una máquina pueda chequear que el software satisface dicho requerimiento, ejemplo la especificación del caso de uso). Ver documento de Especificación de Requisitos.				
	15. ¿Se han enumerado los requisitos incluso los que se derivan de otros requisitos? Ver documento de Especificación de Requisitos.				
	16. ¿Se puede trazar cada requisito al origen en el entorno del problema, (caso de uso del negocio)? Ver documento de Especificación de Requisitos.				
	17. ¿No aparece un mismo requisito en más de un lugar del documento de especificación? Ver documento de Especificación de Requisitos.				
crítico	18. ¿Son los requisitos consistentes? ¿No existe contradicción entre lo especificado por un requisito y lo especificado por otro? Ver documento de Especificación de Requisitos.				
	19. ¿Existe correspondencia entre el modelo de caso de uso, las				

	Especificaciones Suplementarias y las especificaciones de requerimientos? Ver documento de Especificación de Requisitos.				
	20. ¿Soportan los requerimientos los objetivos del negocio, sistema de software y el proyecto?				
	21. ¿Algún requerimiento no es requerido o se encuentra fuera del alcance del proyecto?				
	22. ¿Son las metas y objetivos del sistema de software clara y completamente definidos?				
	23. ¿Se han manejado todos los eventos y condiciones?				
	24. ¿Han sido especificadas todas las operaciones?				
	25. ¿Son las operaciones suficientes para reunir los objetivos del sistema de software?				
	26. ¿Se han especificados todas las definiciones y reglas requeridas?				
	27. ¿Satisface las especificaciones el nivel de detalle requerido el equipo de diseño?				
	28. ¿Los requerimientos se encuentran limpios de				

	polarización de implementación (no restringidos a una alternativa de diseño específica)?				
	29. ¿Se ha identificado algún requisito de licencia o restricción de uso que será seguidos por el software?				
Semántica del documento					
Peso	**Indicadores a Evaluar**	**Eval**	**(NP)**	**Cantidad de elementos afectados**	**Comentarios**
Crítico	1. ¿Ha identificado errores ortográficos?				
Crítico	2. ¿Se entiende claramente lo que se ha especificado en el documento?				
	3. ¿El número de página que aparece en el índice coincide con el contenido que se refleja realmente en dicha página?				
	4. ¿El total de páginas que aparecen en las reglas de confidencialidad coincide con el total de páginas que tiene el documento?				

Anexo 5 Encuesta aplicada para validar la solución

Universidad de las Ciencias Informáticas

Laboratorio de Gestión de Proyectos

Nombre de la persona: _____

Área donde labora: _____

Rol: ( ) Inspector OGP ( ) Revisor de Calidad ( ) Jefe de Proyecto ( ) Director ( ) Otro

Terminologías

Competencias laborales: Conjunto sinérgico de conocimientos, habilidades, experiencias, sentimientos, actitudes, motivaciones, características personales y valores, basado en la idoneidad demostrada, asociado a un desempeño superior del trabajador y de la organización, en correspondencias con las exigencias técnicas, productivas y de servicios. Es requerimiento esencial que esas competencias sean observables, medibles y que contribuyan al logro de los objetivos de la organización. (ONN, 2007).

Dimensiones: Son las pautas de conducta correspondientes a cada competencia propuesta que las hacen observables. (Metodología de Gestión por Competencias Asumiendo la Norma Cubana sobre Gestión de Capital Humano, 2011).

Desarrollo

El sistema GESPRO es una suite de herramientas con diversas funcionalidades que permiten la gestión de áreas de la gestión de proyectos. Dicho paquete soporta como uno de sus módulos la gestión de los recursos humanos. Como parte de sus funcionalidades se almacenan datos personales de los usuarios que pertenecen a los proyectos y los roles de los mismos en cada proyecto, así como las tareas que le son asignadas.

Sin embargo no se guarda información relacionada con las competencias requeridas por los usuarios, para que dado el rol que le haya sido asignado puedan desarrollar eficientemente las actividades del proyecto, lo que provoca que no se conozca realmente el nivel de competencia necesario para que un usuario pueda o no desempeñar un determinado rol o hasta qué punto dicho usuario está capacitado para llevar a cabo las tareas de dicho rol.

Además no se tienen funcionalidades que permitan facilitar la evaluación de las competencias de acuerdo al desempeño demostrado en el desarrollo de las tareas. En consecuencia no se identifican las necesidades de formación de los miembros del equipo, por lo que no se sabe con exactitud los planes de formación que se deben seguir para que un usuario presente un desempeño adecuado o superior en el desarrollo de las tareas que le son asignadas, así como mejorar la formación del resto del equipo de trabajo.

Para dar solución a estos problemas se ha propuesto el aumento de las funcionalidades del módulo con el objetivo de ayudar a la toma de decisiones en los proyectos sobre las competencias que necesitan los usuarios para desempeñar sus tareas, así como los planes de formación para mejorar las capacidades del equipo de trabajo.

De acuerdo a los siguientes requerimientos planteados:

Competencias	Actividad de Formación
• Registrar una nueva competencia. • Editar las competencias con sus niveles. • Eliminar las competencias.	• Registrar una nueva actividad de formación. • Registrar una nueva actividad de

• Definir el nivel deseado o esperado de la competencia para ese rol en ese proyecto. • Definir la importancia de la competencia para ese rol en ese proyecto. • Agregar una nueva dimensión. • Editar las dimensiones. • Eliminar las dimensiones asociadas a una competencia. • Asociar una competencia a un rol en un proyecto. • Permitir heredar competencias de un proyecto padre. • Permitir copiar la configuración de competencias de otro proyecto. • Filtrar las dimensiones por competencias. • Filtrar las competencias por rol. • Insertar el nivel real de la competencia. • Modificar el nivel real en la competencia. • Mostrar la diferencia entre el nivel real y el nivel deseado (Brecha) de cada competencia por individuo. • Mostrar las diferencias entre la brecha anterior y la actual en el nivel de las competencias. • Identificar la relación de las tareas de una persona con las dimensiones de las competencias asociadas al rol que desempeñe en el proyecto. • Caracterizar las dimensiones de las competencias de acuerdo a los indicadores de eficacia y eficiencia. • Calcular el nivel de las competencias de una persona en un proyecto a través de la caracterización que tenga de las dimensiones. • Mostrar por usuario las competencias	formación. • Insertar el impacto de la actividad en la competencia. • Modificar el impacto de la actividad. • Eliminar el impacto de la actividad. • Mostrar el impacto de la actividad en la competencia. • Mostrar el costo por actividad • Proponer un programa de formación que permita disminuir la brecha de los niveles de las competencias para una persona. • Mostrar el programa de formación de una persona.

asociadas a él, agrupadas según los roles que desempeñan.	

Responda, según su criterio, cuál será el nivel de ayuda a la toma de decisiones que proporcionará el módulo luego de su implementación.

( ) Alta **( ) Media** **( ) Baja**

( ) Otro criterio (Argumente): _____

_____

_____

_____