Universidad de las Ciencias Informáticas
Facultad 3

Título:

Módulo para la evaluación de la composición de equipos de proyectos informáticos en la plataforma GESPRO.

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

Autor: Leiza Causilla Rojas

Tutor: MSc. Elizabeth Rodríguez Stiven

Co-tutor: MSc. Yadenis Piñero Pérez

La Habana, Cuba
Junio de 2013

“La responsabilidad nuestra es luchar porque la calidad del producto que aquí se haga sea de las mejores y la mejor posible…”

Ernesto Che Guevara

Declaración de autoría

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

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

_____
(Autor)

_____ _____

(Tutor) (Co-Tutor)

Datos de contacto

Tutores:

MSc. Elizabeth Rodríguez Stiven

Graduada de Ciencia de la Computación en la Universidad de la Habana (UH) en el 2012, Ciudad de La Habana, Cuba. Graduada de Máster en Gestión de Proyectos Informáticos año 2012.

Profesora Asistente, título otorgado por la Universidad de las Ciencias Informáticas en el 2008.

Ha asistido a diez eventos nacionales e internacionales como ponentes o autor. Tiene en su haber quince publicaciones electrónicas en eventos y revistas referenciadas. Ha cursado y aprobado más de veinte cursos de postgrado.

Imparte docencia en el pregrado en la universidad de las ciencias informáticas desde el 2006.

MSc. Yadenis Piñero Pérez

Graduada de Ingeniería Informática del Instituto Superior Politécnico José Antonio Echeverría (CUJAE), 2002, Ciudad de La Habana, Cuba con Título de Oro. Graduada de Máster en Gestión de Proyectos Informáticos año 2007.

Ha asistido a más de treinta congresos y eventos nacionales e internacionales como delegado, ponente o autor. También ha participado en varios encuentros de generalización con el sector empresarial y social.

Tiene en su haber 22 publicaciones electrónicas en eventos avaladas por la Cámara del Libro de la República de Cuba. Ha cursado y aprobado veinte cursos de postgrado.

Ha impartido docencia de pregrado y postgrado. Su actividad académico-investigativa está respaldada por el impacto socio-político y cultural y generalizada a otros centros y entidades del país.

Agradecimientos

Son muchas las personas especiales a las que me gustaría agradecer su amistad, apoyo, ánimo y compañía en las diferentes etapas de mi vida. Algunas se encuentran presentes hoy en día y otras en mis recuerdos y en el corazón. Sin importar en donde estén o si alguna vez llegan a leer estas palabras quiero darles las gracias por formar parte de mí y por todo lo que me han brindado.

***A mi madre**, esa incansable luchadora que día a día creyó en mí y me brindó las palabras de aliento cuando más las necesité. Gracias por darme la vida y una maravillosa formación, por su ternura y todo su amor y por contagiarme con sus mayores fortalezas. Porque gracias a sus consejos he llegado a realizar una de mis grandes metas, este trabajo simboliza mi gratitud por toda la responsabilidad e invaluable ayuda que siempre me ha proporcionado. Sé que este triunfo es motivo de orgullo para ti, y es porque este triunfo también es tuyo.*

***A titi** por haberme brindado todo su cariño y apoyo, por cuidar de mí como su otra hija y aconsejarme como mi padre.*

A todos mis compañeros de aula y de cuarto, que juntos pasamos hermosos momentos durante estos cinco años de la carrera y a mis amistades de estudios anteriores que han perdurado a través de los años. En especial a Yasmany,Cire ,Kirenia ,Meylin ,Josbel, Yennis por estar siempre presente.

A mi bebe por haber llegado para sembrar en mi algo extraordinario y por estar siempre ahí apoyándome cuando necesite de él. Por su amor y comprensión.
Al claustro de profesores y a todos aquellos que ayudaron de una forma u otra a mi formación profesional.

A Elita , Marlenys y Carlos por acogerme entre su familia como una nueva hija.

A mis tutores por su dedicación y labor durante todo este trabajo. Por sus inconformidades educativas en vista a obtener un trabajo con la mejor calidad posible y por verter toda su experiencia y empeño en la realización de esta tesis.

Agradecimientos

A toda mi familia desde los más jóvenes hasta los que ya no son tan jóvenes pero que guardan en ese corazón el profundo cariño para brindárselo a los que estamos a sus alrededores.

A la Revolución cubana por darnos esta magnífica oportunidad de estudiar en tan grandiosa universidad.

Las palabras son fáciles de escribir pero el agradecimiento de mi corazón jamás podrá ser expresado en tan cortas letras. Pasará el tiempo, llegará la experiencia y también los dolores, por qué no, pero aún permanecerá el placer de haberlos tenido conmigo. Gracias a todos por su amor y compañía.

Dedicatoria

Esta dedicatoria va dirigida especialmente a quienes jamás encontraré la forma de agradecer el cariño, comprensión y apoyo brindado en los momentos buenos y malos de mi vida:
Mi mama y mi padrastro.
Mis amigos incondicionales.
Mi familia.
A la memoria de mi padre Mario.
A todos ellos, dedico este trabajo con mucho amor y cariño.

Leiza.

Resumen

Actualmente, en la Universidad de las Ciencias Informáticas se utiliza el GESPRO como gestor de proyectos, el mismo es desplegado por cada centro de producción de forma independiente. A pesar que esta herramienta cuenta con gran número de funcionalidades aún presenta algunas limitaciones, debido a que no evalúa equipos de proyectos informáticos, siendo esto un aspecto importante que ayudaría a saber si las estrategias de los equipos de desarrollo surten efecto luego de su conformación según elementos de su composición.

La presente investigación pretende diseñar e implementar un módulo que permita evaluar la composición de equipos de proyectos informáticos en la plataforma de gestión de proyectos GESPRO.

Para llevar a cabo el trabajo se realiza un diseño que cumple con los parámetros establecidos por la arquitectura y se emplean las herramientas y tecnologías seleccionadas para la implementación. Además, se realiza un período de pruebas que validan el funcionamiento del módulo asegurándose que el mismo cumpla con las normas vigentes para el desarrollo y lograr el objetivo propuesto.

Con la implementación del mismo se brinda la posibilidad a los jefes de proyectos de poder contar con una útil y rápida herramienta que ayuda a evaluar los equipos de proyectos de acuerdo a su composición.

Palabras clave: composición de equipos, evaluación, equipos de proyecto, módulo.

Tabla de contenidos

Índice de Ilustraciones

Índice de Tablas

Introducción

La informática sin duda alguna es uno de los más grandes logros del hombre, desde su surgimiento la sociedad ha experimentado numerosos cambios. Gracias a los grandes avances tecnológicos y el auge de la misma hoy existen empresas y compañías dedicadas enteramente a la producción de software, las cuales han utilizado la informática para el perfeccionamiento de la producción y su desarrollo, convirtiéndose la producción de software en uno de los principales renglones de la economía de muchos países.

A nivel mundial las empresas tienen la imperiosa necesidad de organizar y gestionar de manera eficiente las acciones relacionadas con la información de sus recursos humanos, ya que constituye un factor estratégico cuando se busca incrementar los niveles de productividad, calidad, seguridad, alcance de las metas establecidas y alto grado de competitividad.

La evaluación de equipos de proyectos es una actividad clave en la gestión de recursos humanos dentro del desarrollo de un proyecto informático. Una tarea de primer orden de un líder de proyectos es asegurar que los equipos de proyectos estén bien integrados y sean eficientes[1, 2]. La motivación para dedicarle el tiempo requerido a esta tarea y efectuarla con lujo de detalle y sumo cuidado se reduce al hecho de que los equipos mal integrados ocasionan proyectos interrumpidos fuera de presupuesto y de mala calidad. Por otro lado los equipos bien integrados sobreviven a los proyectos para seguir produciendo con eficiencia nuevos proyectos y venciendo nuevos retos. Sin embargo, la evaluación de la composición de los equipos es un aspecto pobremente tratado en el ámbito del software, resultando escasos los trabajos donde se modele la evaluación de la composición de los equipos desde la perspectiva del equipo [3, 4].

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 [5]. Entre las principales entidades productoras de software en Cuba se encuentra la Universidad de Ciencias Informáticas (UCI), la cual ha organizado su infraestructura productiva en centros de desarrollo de software donde se lleva a cabo todo un proceso encaminado en el desarrollo de proyectos de soluciones informáticas.

GESPRO es la herramienta web que sirve de apoyo a la Gestión de Proyectos Informáticos en todos los centros de desarrollo de la universidad, entre sus funcionalidades incluye un módulo de Gestión de Recursos Humanos. Este módulo respecto a la evaluación de equipo está muy limitado, solo brinda funcionalidades para

analizar el índice de productividad individual de los miembros de un proyecto, no evalúa el equipo como un todo, ni tiene en cuenta indicadores tales como las competencias del equipo y la tendencia personal de cada individuo de comportarse en este de acuerdo a los roles de Belbin, los cuales son indicadores recogidos por el Centro de Innovación de la Calidad de la Enseñanza (CICE) que no se aprovechan.

A pesar del GESPRO ser una herramienta que ha contribuido a mejorar la gerencia de proyectos, el desarrollo de los proyectos cuenta todavía con diversas limitaciones como la ausencia de un módulo dónde los líderes de proyecto puedan evaluar la composición de sus equipos de trabajo y saber el avance que están teniendo los mismos cada cierto período de tiempo. Tampoco permite visualizar el avance de los equipos en cuanto a su composición, para saber si fueron efectivas las estrategias de desarrollo realizadas.

Actualmente, en la universidad existe un modelo para evaluar la composición de equipos de proyectos informáticos, el cual todavía no está informatizado, resultando incómodo y demorado llevar a cabo el proceso de evaluación manual de equipos a través del mismo. Debido a las limitaciones expuestas anteriormente se hace necesaria una herramienta incorporada al sistema de gestión de la universidad que evalúe equipos de proyectos de manera que se obtengan visiones concretas del desempeño de los equipos con los que se cuenta en estos momentos.

A manera de resumen los principales problemas identificados están dado porque:

- No existe una herramienta que informatice el modelo de (Stiven, 2012) existente en la universidad para evaluar la composición de equipos de proyectos informáticos. Implicando esto un aumento significativo del esfuerzo realizado por los especialistas para evaluar los equipos, lo cual se ve reflejado en el siguiente resultado:
    - Con el objetivo de estimar el esfuerzo requerido para evaluar equipos manualmente siguiendo el modelo de Stiven, se realizó un caso de estudio a seis equipos de tamaño siete obteniendo como resultado que el esfuerzo oscila entre 0.75 y 1 hora por hombre para evaluar cada equipo.
- La plataforma GESPRO no cuenta con un módulo que evalúe al equipo como un todo desde su composición.
- En el GESPRO no es posible conocer si las estrategias de los equipos de desarrollo surten efecto, dada la ausencia de una herramienta en el módulo de Capital Humano que brinde la posibilidad de evaluar la composición de los

equipos de proyectos de forma tal que se facilite el trabajo de los especialistas y se minimicen sus esfuerzos.

- En la plataforma GESPRO no existe una funcionalidad que ayude a determinar luego de la conformación de los equipos si la agrupación hecha es la más conveniente según elementos de su composición.

Este trabajo surge como necesidad de dar solución a las situaciones antes expuestas; por lo que se plantea el siguiente **problema a resolver:** ¿Cómo informatizar la evaluación de la composición de equipos de proyectos informáticos integrada a la plataforma de gestión de proyectos GESPRO que facilite el trabajo de los especialistas respecto a la evaluación de la composición de los equipos de proyectos informáticos?

El diseño de la investigación permite al autor especificar como **objeto de estudio:** La evaluación de la composición de equipos de proyectos informáticos, en el que se enmarca el **campo de acción:** La evaluación de la composición de equipo de proyectos informáticos en la plataforma GESPRO.

Con el propósito de resolver el problema planteado se precisa como **objetivo general:** Desarrollar un módulo para la evaluación de la composición de equipos de proyectos informáticos integrado a la plataforma GESPRO que facilite el trabajo de los especialistas respecto a la evaluación de los equipos.

Para dar cumplimiento al objetivo general se desglosan los siguientes **objetivos específicos:**

1. Realizar el marco teórico de la investigación que comprende el estudio de:
    a. La evaluación de la composición de equipos de proyectos informáticos.
    b. La integración de un módulo para la evaluación de equipos en la plataforma GESPRO.
2. Desarrollar el módulo para la evaluación de equipos de proyectos informáticos integrado a la plataforma GESPRO.
3. Validar el módulo mediante las pruebas de funcionalidad y de aceptación.

La **idea a defender** que guiará la investigación es la siguiente:

“El desarrollo de un módulo integrado a la plataforma GESPRO que permita evaluar la composición de equipos de proyectos informáticos, facilitará el trabajo de los especialistas respecto a la evaluación de los equipos de proyectos informáticos.”

Para el cumplimiento de la presente investigación se combinan los siguientes **Métodos y Técnicas en la búsqueda y procesamiento de la información**:

Métodos Teóricos:

Método Analítico-sintético: el empleo de este método permitió realizar un estudio y análisis de toda la bibliografía consultada determinando la importancia de la

composición de los equipos de proyectos informáticos integrado a la plataforma GESPRO.

Permitió además, analizar las partes de la evaluación de equipos de proyectos informáticos y llegar a conclusiones sobre las mismas, resaltando la necesidad de desarrollar el módulo para evaluar la composición de equipos de proyectos que facilite la evaluación.

Método histórico-lógico: se utilizó este método para conocer los antecedentes y las tendencias actuales referidas a la evaluación de equipos en la actividad de composición de equipos.

Métodos Empíricos:

La observación: a través de la misma se recopilaron datos sobre los mecanismos de evaluación de la composición de los equipos en el centro y su tratamiento en la plataforma de gestión de la universidad.

La entrevista: se utiliza para obtener información aislada sobre el funcionamiento del GESPRO a través de sus desarrolladores, así como saber cuál ha sido el aprovechamiento por parte de los proyectos del modelo que se pretende informatizar.

También, se utilizó para llevar a cabo el levantamiento de requisitos de la aplicación y para validar la efectividad de la misma mediante las pruebas de aceptación.

La encuesta: se emplea como una técnica de adquisición de información a través de los cuestionarios que se realizan a la hora de validar el nivel de aceptación de la herramienta, donde se podrá conocer la opinión o valoración de los sujetos seleccionados a realizar dicha encuesta.

El presente trabajo se encuentra estructurado en tres capítulos, como se detalla a continuación:

Capítulo 1: Fundamentación Teórica:

Este primer capítulo tiene como objetivo realizar un estudio valorativo de los fundamentos teóricos generales que sirven como punto de partida a la solución del problema. Hacer alusión a diferentes herramientas y técnicas existentes que permitan implementar el módulo planteado. También contiene conceptos básicos relacionados con el tema de evaluación de equipos de proyectos que ayudan a comprender mejor los modelos investigados y se realiza un estudio de los sistemas de gestión de proyectos empleados por la universidad.

Capítulo 2: Propuesta de solución:

Se expone una visión del sistema, se explica el alcance de la herramienta que se va a desarrollar. Se realiza el proceso de captura de requisitos, se describe y muestra el

diagrama de clases y el de despliegue, finalmente se lleva a cabo la implementación del módulo que posteriormente se pondrá en funcionamiento.

Capítulo 3: Validación de la propuesta:

Se realiza todo lo relacionado con las pruebas, se plantea la estrategia de validación a utilizar que responda el problema planteado. Además, se elaboran los casos de pruebas para saber la efectividad de la herramienta y lograr que el mismo sea un software con calidad.

Capítulo 1. Fundamentación teórica

1.1 Introducción

En el presente capítulo se abordan los conceptos básicos de equipo de proyecto informático y la evaluación de equipos en la gestión de proyectos. Se describen y analizan modelos que realizan evaluación de equipos utilizando características genéricas de composición de equipos. Además, se realiza un estudio de las herramientas y tecnologías utilizadas para dar cumplimiento a la investigación, y así posteriormente realizar el desarrollo de la aplicación.

1.2 Conceptos básicos

1.2.1 Equipo de proyecto informático

Aunque se suele denominar “equipos” a muchas variedades de esfuerzos grupales, grandes o pequeños, pocas de estas iniciativas merecen ese nombre si se mide al verdadero equipo por el desarrollo superior obtenido [6]. En la amplia literatura relacionada con el tema cada autor ofrece su propia definición de equipo de trabajo.

Según Katzenbach, un equipo es un número pequeño de personas, con habilidades complementarias, que están comprometidas con un propósito, objetivos de rendimiento y enfoque común, en el que todos son responsables de los resultados obtenidos [6]; Humphrey agrega que cada uno tiene asignado roles o funciones específicas que debe desempeñar, donde completar la misión requiere alguna forma de dependencia entre los miembros [7].

A partir de las definiciones revisadas hasta el momento se puede concluir que un equipo de un proyecto informático es un conjunto pequeño de personas, enfocado en tareas específicas, con habilidades, actitudes y conocimientos complementarios, comprometidos con un propósito y objetivo común; consciente que la única forma de completar la misión es trabajar en conjunto.

1.2.2 Composición de un equipo

La composición de un equipo está referida a los atributos de los miembros del equipo y cómo estos se combinan para formar equipos interdependientes efectivos [8]. Tres elementos de la composición del equipo han sido objeto de numerosos estudios por su alta significación respecto al desempeño de los mismos, ellos son: el tamaño (número de personas que forman el equipo), las características de la personalidad de sus miembros y las competencias genéricas [4].

1.2.3 Evaluación de equipos en la gestión de proyectos

La evaluación es una actividad fundamental en los procesos claves de la gestión de recursos humanos. Su eficacia radica en la correcta selección de los indicadores, las técnicas que se utilicen para su medición y la interpretación de esta medida en un contexto determinado, estando siempre en concordancia con los objetivos de la organización [9].Según PMBOOK(Guía de los Fundamentos de la Dirección de Proyectos) la evaluación eficaz del equipo puede incluir indicadores como: mejoras en las competencias y los sentimientos que ayudan al equipo a mejorar su rendimiento como grupo, punto en el que convergen la evaluación de equipos y el desarrollo de equipos de proyectos informáticos.

En la actualidad la evaluación es considerada la mejor vía para retroalimentar el proceso de desarrollo de los equipos, para señalar con criterio de la práctica qué nivel de incompatibilidades existen entre los miembros de un equipo, qué competencias se manifestaron, en qué proporción o porcentaje y cuáles no. Siendo esta la herramienta con que cuentan los procesos de gestión de recursos humanos para determinar el estado actual de los equipos de proyectos y planificar futuras acciones para mejorar su desempeño. La evaluación de un equipo de trabajo es considerado un problema complejo debido a la cantidad de elementos que impactan sobre el desempeño eficaz de los mismos [4].

La realización de un modelo para evaluar la composición de equipos de proyectos informáticos representó una vía que permitió facilitar el análisis de la composición de estos equipos, el trabajo de los especialistas para realizar la evaluación, viabilizar el trabajo para quienes realizan este proceso de forma directa como los jefes de proyecto y posibilitar controlar las evaluaciones de los equipos y saber si de un período de tiempo a otro están mejorando en sus evaluaciones.

1.3 Modelos para la evaluación de equipos de proyecto informáticos basados en su composición

A continuación se muestra un estudio sobre modelos que abordan la evaluación de equipos teniendo en cuenta algunos elementos genéricos de composición como las competencias genéricas, las características de la personalidad y el tamaño del equipo. Sin embargo, en estos trabajos solo uno propone un modelo de evaluación de equipos centrados en su composición que contiene la mayor cantidad de los elementos antes mencionados, aunque es importante destacarlos todos porque aportan de forma general aspectos fundamentales que ayudan al proceso de evaluación de equipos.

1.3.1 Método cuantitativo para la integración y comparación de grupos de trabajo de instalación y desarrollo de software

En el siguiente modelo (Prieto,2010) presenta un método cuantitativo de comparación de equipos, a partir de la definición de un modelo donde se establecen las características básicas para que un equipo de software sea funcional, refiriéndose por funcional a lograr una integración adecuada en el equipo que potencie su desempeño como grupo. Estas características están basadas en los perfiles de Belbin y el tamaño del equipo [2].

Este modelo evalúa la personalidad de sus miembros y permite probar que equipos con perfiles similares pueden realizar el mismo trabajo de forma eficaz con tecnología diferente. Sin embargo, a pesar de ser un método bastante atractivo para comparar grupos de trabajos, no incluye entre las características básicas las competencias genéricas, además de enfocarse en la integración de los miembros del equipo.

El modelo tiene como objetivo establecer las características mínimas básicas suficientes para que un equipo de software sea funcional. En el mismo los equipos son representados en una estructura matricial. La intención del estudio es considerar el equipo como un ente que puede ser comparado en forma integral con otro ente similar. Este modelo no cuenta con un sistema que lo informatice ni incluye entre las características básicas las competencias genéricas que son un elemento importante en la composición de un equipo.

1.3.2 Modelo de evaluación de equipos de proyectos de Software

En el modelo propuesto por (Picacho & Amengual, 2009) se propone realizar una evaluación de los equipos de trabajo basándose en cuatro elementos que influyen en el rendimiento del equipo: La motivación, la comunicación, la composición y la gestión. Centra sus bases en la mejora de rendimiento del equipo a partir de la realización de buenas prácticas para cumplir con el propósito de cada uno de los factores estudiados[10].

La evaluación transita desde el nivel conceptual a un nivel operacional por medio de dos cuestionarios: uno dirigido a los gestores del proyecto y otro dirigido a todos los miembros del equipo. El objetivo de este modelo es obtener una evaluación del rendimiento del equipo y ayudar a mejorar sus prácticas en función de obtener una mejor calidad en los procesos de desarrollo. Para obtener una medida final de rendimiento se utiliza una escala nominal. Este modelo no incorpora el tamaño del equipo ni la personalidad de sus miembros en los aspectos descritos en la composición del equipo, siendo estos puntos, características fundamentales para llevar a cabo la evaluación de equipos basados en su composición, por lo que no sería

un modelo adecuado a informatizar ya que obvia aspectos fundamentales para la composición de un equipo.

1.3.3 Modelo para la asignación de recursos humanos a equipos de proyectos de software

En (André, 2009) se presenta un modelo de asignación de recursos humanos a equipos de proyectos que incorpora factores que contribuyen a la asignación individual a los roles del proyecto (competencias, carga de trabajo, costo por lejanía y características psicológicas) y a la formación del equipo (balance entre roles e incompatibilidades entre los miembros y roles del equipo). Como elementos originales, el modelo incorpora las incompatibilidades del equipo y patrones para la formación de equipos identificados a partir del uso de test psicológicos [3].

Este modelo como su nombre lo indica no está diseñado para evaluar equipos, sin embargo sí es considerado en esta investigación como un modelo importante por el aporte que brinda respecto a los patrones para la formación de equipos, el análisis en las competencias genéricas del equipo, además de tratar aspectos de composición como la personalidad de los miembros, pero obviando el tamaño del equipo como un punto importante dentro de la evaluación de la composición de equipos de proyectos informáticos.

1.3.4 Modelo para la evaluación de la composición de equipos de proyectos informáticos

En (Stiven,2012) se presenta un modelo de evaluación de equipos centrado en su composición que relaciona el tamaño del equipo, las características de la personalidad de sus miembros y las competencias genéricas con el fin de mejorar la actuación de los equipos y aumentar sus posibilidades de éxito independientemente del entorno donde se desarrolle [4].

Este modelo está constituido por tres perfiles teóricos que representan la composición ideal de los equipos y que sirven como pautas en el proceso de evaluación de la composición de equipos de proyectos informáticos. Contiene además elementos fundamentales de los modelos realizados por Prieto, Picacho & Amengual y André mencionados anteriormente. Define un modelo matemático de evaluación que facilita el análisis de la composición de los equipos de proyectos informáticos. Presenta una guía de uso del modelo para la evaluación de la composición de equipos de proyectos informáticos que facilita la interpretación de los resultados de la evaluación.

Del estudio realizado sobre los modelos en la siguiente tabla que abordan la evaluación de equipos como un todo, se concluye que solo uno de los modelos

revisados trata la evaluación de equipos relacionando tamaño, las competencias genéricas y los roles de equipos, con el objetivo de determinar cuan aceptada es la composición del equipo. Por lo que se decide que el modelo de Stiven es el más completo y el que más se ajusta a este tipo de evaluación, además de ser diseñado para suplir necesidades específicas en la estructura productiva de la universidad.

Tabla 1: Resumen de los modelos de evaluación de equipos.

Modelos	**Características de interés a tomar para el módulo a implementar.**
Método cuantitativo para la integración y comparación de grupos de trabajo de instalación y desarrollo de software (Prieto, 2010).	De las características presentes en este modelo serán de interés para la solución, tomar elementos fundamentales de composición como el tamaño del equipo y la personalidad de los miembros de un equipo utilizando los perfiles de Belbin que Prieto propone.
Modelo de evaluación de equipos de proyectos de Software (Picacho & Amengual, 2009).	Sobre el estudio realizado al modelo de (Picacho & Amengual) se encontró como elemento fundamental para tratar durante el desarrollo del módulo para evaluar equipos las competencias genéricas.
Modelo para la asignación de recursos humanos a equipos de proyectos de software (André, 2009).	En este modelo se puede tomar de interesante para responder el problema planteado la personalidad y las competencias genéricas que plantea el autor.
Modelo para la evaluación de la composición de equipos de proyectos informáticos(Stiven,2012)	Este modelo es el más completo evaluando equipos de acuerdo a su composición. Además de apropiarse de elementos fundamentales de los modelos anteriores. Son de interés para la solución del trabajo el tamaño del equipo, la personalidad de sus miembros y las competencias genéricas.

1.4 Indicadores que contribuyen a realizar evaluación de equipos.

A continuación se muestra un estudio de indicadores que contribuyen a realizar evaluación de equipos de proyectos informáticos.

1.4.1 TSP (Team Software Process)

El llamado **Team Software Process** (TSP) en combinación con el Personal Software Process (PSP) proporciona un marco de trabajo de procesos definidos que está diseñado para ayudarle a equipos de ingenieros a organizar y producir proyectos de software de gran escala, que tengan tamaños mayores a varios miles de líneas de código. Su objetivo es mejorar los niveles de calidad y productividad de un proyecto de desarrollo de software de un equipo, con el fin de ayudarlos a alcanzar los acuerdos de costos y tiempos en dicho desarrollo. Establecer estándares para medir la calidad y el comportamiento, proporcionar métricas para equipos y evaluar roles y equipos[11].

Se enfoca en el proceso de la construcción de un equipo productor de software, estableciendo objetivos del equipo, distribuyendo los roles, y otras actividades de trabajo en equipo. Este método no contempla las características personales de los miembros de un equipo y su influencia en el trabajo del mismo durante el desarrollo del software. Se basa en general en evaluar los equipos desde un punto puramente técnico, o sea mediante indicadores de productividad. No tiene en cuenta la evaluación del equipo como un todo, ni elementos de su composición, por lo que no resuelve todos los aspectos que dan lugar a la situación problémica planteada.

En la solución del problema resulta apropiado tomar de este indicador su enfoque en las competencias genéricas de los miembros de un equipo. Sin embargo no se aplica el tamaño de hasta 20 integrantes en que puede oscilar el equipo porque resulta ser muy grande para el objetivo buscado de equipos pequeños.

1.4.2 Sistema de soporte a la decisión para la asignación de recursos humanos a equipos de proyectos de software

El sistema de soporte a la decisión diseñado por Margarita André Ampuero y María Gulnara Baldoquín de la Peñacon tiene el propósito de apoyar el proceso de asignación de personal en las organizaciones de software. La herramienta sustenta un modelo formal de asignación que integra factores que contribuyen a la asignación individual a los roles del proyecto y a la formación del equipo como un todo, así como algunos de los métodos y algoritmos de solución del modelo posibles a implementar.
La herramienta nombrada TEAMSOFT considera como objetivos seleccionados por defecto para enfrentar el proceso de asignación de personal: maximizar las competencias del personal, balancear la carga de trabajo y minimizar las incompatibilidades del equipo [12].

TEAMSOFT, constituye un sistema de soporte a la decisión, en tanto apoya a los directivos de las organizaciones de software a enfrentar la gestión de recursos humanos basado en el enfoque por competencias. El hecho de sustentar el modelo formal para la asignación de recursos humanos a equipos de proyectos de software lo convierte en un instrumento especialmente útil a estudiar en la investigación de herramientas que contribuyan a realizar la evaluación de equipos de proyectos informáticos, aunque cabe señalar que incorpora un conjunto de funcionalidades que apoyan los procesos de capacitación, desarrollo de planes de carrera y evaluación del desempeño.

Además, constituye un instrumento útil para la gestión de recursos humanos basado en el enfoque por competencias, en especial como soporte al proceso de asignación de personal en organizaciones de software medianas y grandes [12].

Esta herramienta en comparación con la anterior si comprende algunos elementos respecto a la composición del equipo como son las competencias genéricas, por lo que aportaría ideas para la realización del módulo a desarrollar. Le falta tratar el avance del equipo como un todo y el tamaño del equipo que no debe ser grande.

1.4.3 Sistema de modelo de referencia de trabajo en equipos de proyectos

El sistema de modelo de referencia detalla los factores que deben tenerse en cuenta para valorar el trabajo en equipo. Cada factor de este modelo de referencia proporciona información en forma de un factor de identificación, un factor de nombre, una descripción que detalla los aspectos diferentes para el factor, y un conjunto de mejores prácticas de trabajo en equipo de identificación de la tarea necesaria para lograr el propósito del factor.

El modelo evalúa el trabajo para equipos de software, se compone de un Modelo de Trabajo en Equipo referencia (TRM) y un Marco de Medición. En este sistema de modelo el proceso de evaluación a seguir para realizar una evaluación de trabajo en equipo y la experiencia de su aplicación a los equipos de software se describen a través de cuestionarios que evalúan factores claves en el trabajo en equipos como gestión, comunicación, composición y motivación, de acuerdo a las buenas prácticas de trabajo planteadas [13].

Este modelo no resuelve el problema de evaluación de equipos pequeños teniendo en cuenta elementos genéricos de composición, porque el mismo sirve para equipos grandes o pequeños, y las relaciones que se establecen entre los miembros de un equipo grande no son las mismas que se establecen entre los miembros de un equipo pequeño de hasta 7 integrantes. Además, de no enfocarse en temas como las características de las personas que integran los equipos de trabajos de un proyecto,

los cuales son elementos importantes para llevar a cabo la herramienta a realizar, la cual tiene como objetivo evaluar equipos de proyecto de acuerdo a su composición.

En la tabla 2 se presenta un resumen de los principales indicadores que contribuyen a realizar evaluación de equipos, enfocándose en los objetivos específicos de cada uno de ellos y los elementos que serán apropiado tomar para la realización del trabajo. Del resumen se puede concluir que todos los indicadores pueden aportar funcionalidades al módulo a realizar, pero ninguno de ellos está completo para evaluar los equipos de proyectos de acuerdo a su composición teniendo en cuenta 3 elementos específicos como el tamaño del equipo, las competencias genéricas y la personalidad de sus miembros.

Tabla 2: Resumen de los indicadores para realizar evaluación de equipos.

Indicadores	Objetivos	Interés a tomar de ese indicador.
Team Software Process (TSP)	Se basa en evaluar los equipos partiendo solamente de indicadores de productividad. No tiene en cuenta la evaluación de los equipos como un todo, ni elementos de su composición como las características personales de los miembros del equipo.	Es importante destacar el enfoque en las competencias genéricas que brinda este indicador.
Sistema de soporte a la decisión para la asignación de recursos humanos a equipos de proyectos de software	Su objetivo es la asignación de personas a equipos, por lo que no contempla la evaluación del equipo como un todo, ni trata a este como la unidad mínima de producción, de manera que permita así saber en qué estado está el equipo en distintos momentos según su composición.	Del modelo mencionado es de interés tomar para el desarrollo de la herramienta la asignación del personal del proyecto a los equipos.
Sistema de modelo	Este modelo guía el trabajo en	Los elementos importantes

de referencia de trabajo en equipos de proyectos.	equipo de acuerdo a un conjunto de buenas prácticas. Incluyendo factores como la composición, pero no teniendo en cuenta dentro de la composición elementos como el tamaño del equipo.	a tomar de este sistema son las competencias genéricas como un elemento fundamental para evaluar equipos.

1.5 Herramientas de gestión de proyectos

Son disímiles las herramientas que contribuyen a llevar a cabo la planificación y el control del desarrollo de un proyecto de software. A continuación se realiza un análisis de los principales sistemas de gestión utilizados por la universidad desde su surgimiento haciendo énfasis en el que se emplea en la actualidad, el cual con el paso de los años ha incrementado el número de funcionalidades con el objetivo de ir eliminando limitaciones encontradas durante su utilización.

1.5.1 OpenProj

OpenProj es un software libre (de licencia GNU) de gestión de proyectos que fue creado como alternativa a Microsoft Project (software comercial, por el que hay que pagar por su licencia). Disponible para diversos sistemas operativos como Linux, Mac Unix o Windows. Permite visualizar el estado de los proyectos mediante diagramas de Gantt. No tiene seguimiento de errores, ni notificaciones automáticas.

Controla todos los aspectos referentes a la gestión de proyectos, como la planificación y programación, la gestión y asignación de recursos, la simulación de alternativas en procesos críticos. Asimismo, proporciona la funcionalidad necesaria para trabajar con entornos multiproyectos, incluso relacionarlos[14].

En la Gestión de Recursos OpenProj distingue entre recursos humanos y materiales permitiendo así no asignar una tarea a un técnico que ya esté ocupado con otra. Tener a mano la información básica y de contacto de cada trabajador. Asignar las tareas a cada uno de los miembros del equipo. Balancear/distribuir de forma uniforme el trabajo. Calcular costes de ejecución y no utilizar un recurso, máquina, o herramienta en una tarea si ya está reservado para otra.

Este sistema el cual no es de los más utilizados en la universidad por los proyectos de desarrollo, en cuanto a gestión de recursos humanos se refiere no provee ninguna herramienta que ayude a realizar la evaluación de equipos, permitiendo solamente asignar tareas y realizar las actividades mencionadas anteriormente.

1.5.2 Redmine

El Redmine es un sistema multiplataforma e independiente de la base de datos, programado con el marco de trabajo Ruby on Rails, código abierto con licencia GPL, con una interfaz limpia y funcionalidades asombrosas para gestión de proyectos [15]. Entre sus características principales se encuentran:

La Gestión de múltiples proyectos, personalización de proyectos, sistema flexible de seguimiento de tareas, integración en repositorios de código, uso de calendario y diagrama de Gantt, exportación a distintos formatos. Evaluación minuciosa de las tareas de cada miembro en los proyectos. Verificación del esfuerzo realizado por cada miembro. Calificación de la eficiencia del equipo de trabajo de forma sencilla.

Además, es una herramienta donde generalmente cualquier elemento puede configurarse o contiene opciones. Algunas otras funcionalidades que habría que destacar son la página personal de cada usuario, que ofrece una vista personalizable con información de todos los proyectos donde está participando, como un calendario global, o peticiones asignadas. Cabe mencionar que admite como bases de datos MySQL, SQLite y PostgreSQL.

Una de las principales deficiencias de Redmine queda en la instalación, que puede resultar muy compleja, no solo por los paquetes requeridos, sino por los pasos que se debe seguir para la realización de esta.

El Redmine en el módulo de gestión de recursos humanos no contempla una herramienta para evaluar equipos de proyectos informáticos como un todo, pudiéndose entre otras pocas cosas planificar tareas a los recursos humanos individualmente de manera que se reduzca el número de tiempos muertos y cada persona se utilice lo mejor posible. Este sistema ha sido muy utilizado en la universidad, y fue tomado como punto de partida para realizar el GESPRO que es la herramienta que se emplea actualmente para llevar a cabo la gestión de los proyectos.

1.5.3 GESPRO

El GESPRO es un Paquete para la Gestión de Proyectos desarrollado por la Universidad de las Ciencias Informáticas, que debido a la gran cantidad de funcionalidades que facilita es tomado como propuesta de herramienta para la gestión de los proyectos de la universidad teniendo en la actualidad un entorno para cada centro, y estando constantemente en proceso de desarrollo de mejoras que incluye integración con otras herramientas para la gestión de diferentes aspectos del proyecto y la inclusión de otros módulos realizados en cursos anteriores. La interacción con la herramienta es a través de la Web, por lo que es necesario tener instalado algún navegador web (Mozilla, Opera, IExplorer).

GESPRO desarrollada con el marco de trabajo Ruby on Rails. Es una aplicación web que tiene la ventaja de ser software libre bajo licencia GPL (GNU General Public License v2). Además, cuenta con una serie de módulos que amplían sus funcionalidades [16].Entre las principales características se tiene el soporte de múltiples proyectos, foros, seguimiento al tiempo, publicación de noticias, control de tareas y proyectos, integración con manejadores de configuración de código, gestión de riesgos , además del control y seguimiento de los recursos humanos.

El módulo de Gestión de Recursos Humanos incluye los procesos que organizan y dirigen el equipo del proyecto el cual está compuesto por las personas a quienes se les han asignado roles y responsabilidades para concluir el proyecto. Si bien, es común hablar de asignación de roles y responsabilidades, los miembros del equipo deberían participar en gran parte de la planificación y toma de decisiones del proyecto. La participación temprana de los miembros del equipo aporta experiencia durante el proceso de planificación y fortalece el compromiso con el proyecto. El tipo y la cantidad de miembros del equipo del proyecto a menudo pueden cambiar, a medida que avanza el proyecto. Los miembros del equipo del proyecto pueden denominarse personal del proyecto.

El GESPRO en este módulo no cuenta con una herramienta que permita evaluar la composición de equipos de proyectos informáticos, pudiéndose solamente asignar tareas a las personas e ir llevando un control sobre las mismas, además de permitir conocer el índice de productividad individual de los miembros de un equipo, no realiza análisis a nivel de equipo. Sin embargo, es superior en temas de Gestión de Recursos Humanos respecto al resto de las herramientas, y es donde se insertará el módulo a desarrollar.

Tabla 3: Comparación entre sistemas de Gestión de Proyectos.

Características	OpenProj	Redmine	GESPRO
Gratuito	x	X	x
Software libre	x	X	x
Multiplataforma	x	X	x
Seguimiento de errores		X	x
Disponible para diversos sistemas operativos.	x	X	x
Asignación de	x	X	x

recursos humanos y materiales.			
Gestión de miembros del proyecto		X	x
Licencia	GNU	GPL(v2)	GNU
Lenguaje de programación	Java	Ruby	Ruby
Gestión de foro, creación de wiki.		X	x

De los sistemas anteriormente estudiados se percibe que el OpenProj es un sistema de gestión de proyectos que en el módulo de recursos humanos no realiza ninguna actividad referida a la evaluación de los equipos de proyectos. Redmine además cuenta con mayor número de funcionalidades que el OpenProj y en el módulo antes mencionado vela por el avance del equipo de trabajo en su conjunto, tomando al equipo de trabajo como todos los integrantes del proyecto, también observa el aporte de cada miembro individual para el desarrollo del proyecto, pero no tiene en cuenta la evaluación de equipos pequeños de trabajo de acuerdo a elementos de su composición. Y el GESPRO a pesar de haber avanzado en la gestión de los recursos humanos, todavía en la actividad de composición de equipos de proyectos tampoco tiene implementado una herramienta que evalúe si estos equipos están integrados de forma correcta teniendo en cuenta aspectos de su composición.

1.6 Metodologías de desarrollo de software

Dado el enorme auge que ha adquirido la producción de software en el mundo se ha creado una gran cantidad de documentación formal referida a los procesos, políticas y procedimientos para asegurar la eficiencia del producto. Cumplir con los requisitos iniciales y minimizar las pérdidas de tiempo en el proceso de desarrollo, son los principales objetivos de las metodologías de desarrollo a las cuales se va hacer referencia.

Es de suma importancia establecer una metodología de desarrollo de software adecuada para así minimizar riesgos e incrementar las posibilidades de éxito en el desarrollo de los productos.

1.6.1 SCRUM

Es una metodología para la gestión y desarrollo de software basada en un proceso iterativo e incremental utilizado comúnmente en entornos basados en el desarrollo ágil de software. Desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle [17]. La

misma primeramente surgió como metodología para el desarrollo de productos tecnológicos, pero en la actualidad se emplea también en ambientes que trabajan con requisitos inestables y que requieren rapidez y flexibilidad; situaciones frecuentes en el desarrollo de determinados sistemas de software. Es sencillo, requiere de poco esfuerzo para comenzar a trabajar, permite el desarrollo, prueba y correcciones de manera rápida.

Aunque solo funciona bien en equipos pequeños y ágiles, se necesita que los miembros del equipo de desarrollo sean experimentados y si un miembro del equipo abandona su puesto puede conllevar grandes problemas.

De esta metodología se hace necesario tomar para el contexto de trabajo que es basada sobre un proceso iterativo e incremental.

1.6.2 Rational Unified Process (RUP)

El Proceso Unificado de Rational (RUP) describe cómo aplicar efectivamente el desarrollo de software[18].

RUP provee a cada miembro del equipo de las guías de proceso, plantillas y mentores de herramientas necesarios para que estos tomen ventaja de, entre otras, las siguientes mejores prácticas:

- Administración de los requisitos.
- Arquitectura basada en componentes.
- Modelar visualmente el software.
- Verificar la calidad.
- Controlar los cambios de software

Constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Enfocado al desarrollo de proyectos grandes, centrado en la arquitectura y guiado por casos de uso.

Esta metodología no satisface las necesidades del trabajo a realizar, pues su proceso de desarrollo es demasiado largo y genera gran números de artefactos innecesarios por lo que no se podrá terminar en tiempo el módulo propuesto.

1.6.3 Programación Extrema (XP)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. Se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios [19].

XP es una metodología basada en la simplicidad, la comunicación e interacción permanente con el cliente (comprobación de requisitos constante) y en el ”pair programming”, que es la técnica de programación por parejas donde uno de los programadores escribe código y el otro lo prueba y después se cambian los papeles.
Esta metodología se caracteriza por:

- Permitir introducir nuevos requisitos o cambiar los anteriores ágilmente.
- Adecuada para proyectos pequeños y medianos.
- Adecuada para proyectos con alto riesgo.
- Su ciclo de vida es iterativo e incremental.
- Cada iteración dura entre una y tres semanas.

XP se basa fundamentalmente en ser ligero, cercano al desarrollo, basado en Historias de Usuario, fuerte comunicación con el cliente, el código pertenece a todos, programación por parejas y pruebas como base de la funcionalidad [20].
Esta metodología es de interés a utilizar pues su ciclo de vida se enfatiza en el carácter interactivo e incremental del desarrollo, una iteración de desarrollo es un período de tiempo en el que se realiza un conjunto de funcionalidades determinadas que en el caso de XP corresponden a un conjunto de historias de usuarios. Por lo que se generan artefactos como la descripción de los requisitos funcionales, las historias de usuario y el plan de iteraciones.
Teniendo en cuenta las características y peculiaridades de las metodologías analizadas anteriormente y basándose en que el proyecto es pequeño y lo realiza una sola persona se decidió utilizar XP-Programación Extrema, para el proceso de desarrollo y solución del problema a resolver ya que cuenta con diferentes características que son similares a las condiciones de trabajo y desarrollo, además de ser la metodología utilizada en el GESPRO donde se vinculará el módulo a implementar. Las demás metodologías, aunque cabe destacar que son muy buenas y ampliamente utilizadas en el mundo, no se ajustan a las necesidades del trabajo. No siendo estas entonces las más aconsejables a manejar ya que traerían consecuencias negativas en las posteriores fases de desarrollo del producto.

1.7 Herramientas y tecnologías a utilizar para el desarrollo:

1.7.1 Ruby

Ruby es un lenguaje de programación interpretado, reflexivo y orientado a objetos. Creado por el programador japonés Yukihiro "Matz" Matsumoto, quien comenzó a trabajar en Ruby en 1993, y lo presentó públicamente en 1995. Combina una sintaxis inspirada en Python y Perl con características de programación orientada a objetos

similares a Smalltalk. Su implementación oficial es distribuida bajo una licencia de software libre [21]. Es la base del marco de trabajo Ruby on Rails.

Características generales del lenguaje:

- Orientado a objetos.
- Cuatro niveles de ámbito de variable: global, clase, instancia y local.
- Manejo de excepciones.
- Expresiones regulares nativas similares a las de Perl a nivel del lenguaje.
- Posibilidad de redefinir los operadores (sobrecarga de operadores).
- Altamente portable.
- Hilos de ejecución simultáneos en todas las plataformas.
- Carga dinámica de DLL/bibliotecas compartidas en la mayoría de las plataformas.
- Amplia librería estándar.
- Soporta inyección de dependencias.
- Soporta alteración de objetos en tiempo de ejecución.
- Continuaciones y generadores.

1.7.2 Ruby on Rails 3.2

El marco de trabajo (framework) Ruby on Rails también conocido como RoR o Rails es sobre el cual está desarrollado el GESPRO. Ruby on Rails es de código abierto y está escrito en el lenguaje de programación Ruby, siguiendo el paradigma de la arquitectura Modelo Vista Controlador (MVC), trata de combinar la simplicidad con la posibilidad de desarrollar aplicaciones del mundo real escribiendo menos código que otros marcos de trabajos y con un mínimo de configuración [22].

Sigue dos principios básicos:

Convención sobre la configuración: Con Rails, en lugar de archivos de configuración se utilizan una serie de convenciones. Es decir que no es necesario configurar la aplicación en un archivo XML (por ejemplo), basta con utilizar una convención que sustituya la configuración.

Menos software: Se refiere a que se escribe menos líneas de código, lo que trae consigo más rapidez y menos errores en la implementación.

Se selecciona para la creación de la aplicación este lenguaje porque además de las ventajas que se comentaron anteriormente, el GESPRO fue implementado en Ruby y todo nuevo componente que se le desee agregar deberá estar desarrollado en el mismo lenguaje.

1.7.3 Herramientas CASE (Computer Aided Software Engineering)

Las Herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) son las aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas [23].

Estas herramientas contribuyen de manera directa en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores.

1.7.3.1 Visual Paradigm for UML

Visual Paradigm For UML (VP-UML) es una Herramienta Case Cruzado de Ciclo de Vida pues soporta el ciclo de vida completo del desarrollo de software. Esta herramienta provee el modelado de procesos de negocios, además de un generador de mapeo de objetos-relacionales para los lenguajes de programación Java, .NET y PHP. Ayuda a graficar todo tipo de diagramas de clases, generar código desde diagramas y documentación. Proporciona abundantes tutoriales, demostraciones interactivas y proyectos UML. Permite la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso. También ofrece la posibilidad de generar bases de datos pues transforma diagramas de Entidad-Relación en tablas de base de datos [24].

Soporta estándares claves de la industria, tales como Lenguaje de Modelado Unificado (UML), SysM, BPMN, XMI. Ofrece un completo conjunto de herramientas para la captura de requisitos, la planificación de programas, la planificación de controles, modelado de clase y modelado de datos [25].

Genera la documentación del proyecto automáticamente en varios formatos como son Web o Pdf y permite el control de versiones.

VP-UML es multiplataforma, lo cual le permite al usuario utilizar esta herramienta en varios sistemas operativos como Windows, Linux, Unix y otros; además se encuentra disponible en distintas versiones: Enterprise, Professional, Standard, Modeler, Personal y Community. De las versiones anteriores se emplea la versión libre Visual Paradimg for UML 8.0 Enterprise Edition.

1.7.4 IDE NetBeans 6.9

Plataforma libre de código abierto, entorno de desarrollo integrado para desarrolladores de software. Brinda todas las herramientas necesarias para crear

aplicaciones profesionales de escritorio, empresariales, web y aplicaciones móviles con la plataforma Java, así como C / C + +, PHP, JavaScript, Groovy y Ruby [26].
Este IDE es el más recomendable ya que contempla el lenguaje a utilizar para la implementación del módulo permitiendo el completamiento de código y la integración con el entorno de desarrollo. Además es muy fácil de usar para la programación web, por lo que se va a estar utilizando durante todo el ciclo de implementación del módulo.
Se ejecuta en varias plataformas, incluyendo Windows, Linux, Mac OS X y Solaris. NetBeans IDE 6.9 soporta el lenguaje Ruby 1.9 con el frameworks Rails 3.2.8 y ha sido probado con distintos tipos de servidores de aplicación [27].

1.7.5 Sistema Gestor de Base de Datos (SGBD): PostgreSQL.

Es un sistema de gestión de base de datos relacional orientada a objetos de software libre que lleva más de una década de desarrollo y cuenta con una arquitectura probada; ganándose en este tiempo una sólida reputación de confiabilidad e integridad de datos. Funciona en los principales sistemas operativos, incluyendo Linux, UNIX y Windows [28]. También es compatible con el almacenamiento de objetos binarios incluyendo imágenes, sonidos o videos; y utiliza el Lenguaje de Consulta Estructurado (SQL), lenguaje utilizado por todas las Base de Datos (BD) relacionales.
SGBD de código abierto , siendo más eficiente cuando el tamaño de la base de datos es grande. Además, siendo la eficiencia un requisito fundamental en el desarrollo de una Aplicación Web, es fácil entender su selección como SGBD para la aplicación a desarrollar. Como muchos otros proyectos de software libre, el desarrollo de PostgreSQL es manejado por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo, siendo beneficiosas debido a que ayuda a buscar soluciones más rápidamente a los problemas que se presenten durante el desarrollo del trabajo. Algunas de sus principales características [29] son: la alta concurrencia permitiendo que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos y amplia variedad de tipos nativos brindando soporte para:

- Números de precisión arbitraria.
- Texto de largo ilimitado.
- Figuras geométricas (con una variedad de funciones asociadas).
- Direcciones IP (IPv4 e IPv6).
- Bloques de direcciones estilo CIDR.
- Direcciones MAC.
- Arrays.

1.7.6 Pgadmin III

Pgadmin III es una aplicación gráfica para el gestor de bases de datos PostgreSQL, con licencia de Código Abierto, escrita en C++ usando la librería gráfica multiplataforma wxWidgets. Esto permite que se pueda usar tanto en Linux, Mac OS X y Windows.

Es capaz de gestionar versiones a partir de la PostgreSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de PostgreSQL como Pervasive Postgres, EnterpriseDB Mammoth Replicator y SRA PowerGres. Pgadmin III está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados, soporte para el motor de replicación Slony-I entre otras características. La conexión al servidor puede hacerse mediante conexión TCP/IP o Unix Domain Sockets (en plataformas *nix), y puede encriptarse mediante SSL para mayor seguridad.

1.8 Conclusiones del capítulo

En este capítulo se efectuó un análisis general de la evaluación de la composición de equipos de proyectos informáticos, abordando sus características y conceptos, a través de diversos modelos que realizan la evaluación de equipos de alguna manera.

- De los modelos revisados se concluye que el modelo de (Stiven,2012) es el que aborda de forma más completa la evaluación de la composición de equipos de proyectos informáticos, ya que relaciona el tamaño ,las competencias y los roles de equipos con el objetivo de determinar cuan aceptada es la composición del equipo.
- Del estudio realizado respecto a los sistemas de gestión se determina que ninguno presenta una herramienta que permita evaluar la composición de equipos de proyectos informáticos, demostrándose así la necesidad de un módulo para evaluar la composición de equipos.
- Las herramientas y tecnologías a utilizar para la realización de la aplicación, serán aquellas que se tienen planificadas por políticas de restricción del proyecto GESPRO, de manera que se logre insertar el módulo sin problemas. Finalmente, debido a lo antes mencionado se utiliza a Ruby como lenguaje de programación con el framework Ruby on Rails, el Netbeans es la herramienta para llevar a cabo el entorno de desarrollo de la aplicación. Para la realización

del diseño de diagramas y el modelado de las clases, así como la captura de los requisitos se emplea el Visual Paradigm y el gestor de base de datos es el PostgreSQL.

Incluir el modelo en la plataforma GESPRO permitirá facilitar el trabajo de evaluación de la composición de equipos, generalizarlo en los centros productivos para que se haga lo más semejante posible el proceso de evaluación en todos los proyectos. Además como GESPRO es la herramienta que se utiliza en la universidad para llevar la gestión de sus proyectos y el control de la producción, donde se realiza un seguimiento de las distintas actividades durante el ciclo de vida del proyecto, estará disponible directamente para que lo utilicen de forma generalizada todas las organizaciones productivas.

Capítulo 2. Propuesta de solución

2.1 Introducción

En el presente capítulo se realiza un análisis de la propuesta de solución. Se describen las características y funcionalidades del sistema, se mencionan los requisitos funcionales y no funcionales que debe cumplir la aplicación. Además, se explican los patrones utilizados, se presenta el diseño y la arquitectura del sistema a desarrollar y por último se realiza la implementación del módulo.

2.2 Objetivos de la informatización del modelo

La informatización del “Modelo para evaluar la composición de equipos de proyectos informáticos” tiene como objetivos:

Automatizar los procesos centrales que componen el modelo propuesto así como los subprocesos, actividades, artefactos y otros procesos que puedan surgir de la ejecución de este modelo. Tener una adecuada gestión de los equipos de proyectos en la universidad, además de poseer la información referente a estos equipos de manera organizada y centralizada, disponible para utilizarla en diversos momentos.

2.3 Técnicas utilizadas para apoyar la informatización del modelo

Se utilizó la entrevista como técnica para comprender el proceso de informatización del modelo, así como para identificar las necesidades reales existentes en los proyectos. El resultado de las entrevistas es decisivo para el posterior desarrollo de la informatización, puesto que con ellas se obtiene toda la información, las cuales sirven de base para cualquier tipo de desarrollo o adaptación. Existen diferentes tipos de entrevistas, entre las que se pueden mencionar: cuestionarios, entrevistas de final abierto, entrevistas en grupos de desarrollo y las discusiones. De las antes mencionadas se hizo énfasis en los cuestionarios y las entrevistas de final abierto las que contribuyeron para saber cuáles son los requisitos específicos a implementar, además de emplearse durante el transcurso de validación del módulo.

2.4 Estudio del modelo para la evaluación de la composición de equipos de proyectos informáticos.

Del estudio realizado en el capítulo anterior sobre los modelos de evaluación de equipos de proyectos informáticos se llegó a la conclusión que el modelo más completo que satisface las necesidades propuestas y cumple con la evaluación de equipos basados en su composición es el modelo de (Stiven ,2012), el cual se toma como guía para informatizar e incorporarlo al sistema de gestión de proyectos de la universidad.

2.4.1 Descripción general del modelo.

El modelo está constituido básicamente por dos componentes. El “Modelo Conceptual de Equipo” donde se define teóricamente la composición ideal de los equipos de proyectos informáticos en función del tamaño, características de la personalidad de sus miembros (roles de equipo) y las competencias genéricas. Estrechamente relacionado con el modelo conceptual de equipo se encuentra el “Modelo Matemático de Evaluación”. Componente este encargado de dictaminar cuantitativamente el estado actual de la composición de los equipos de proyectos informáticos tomando como referencia el modelo conceptual de equipo[4].

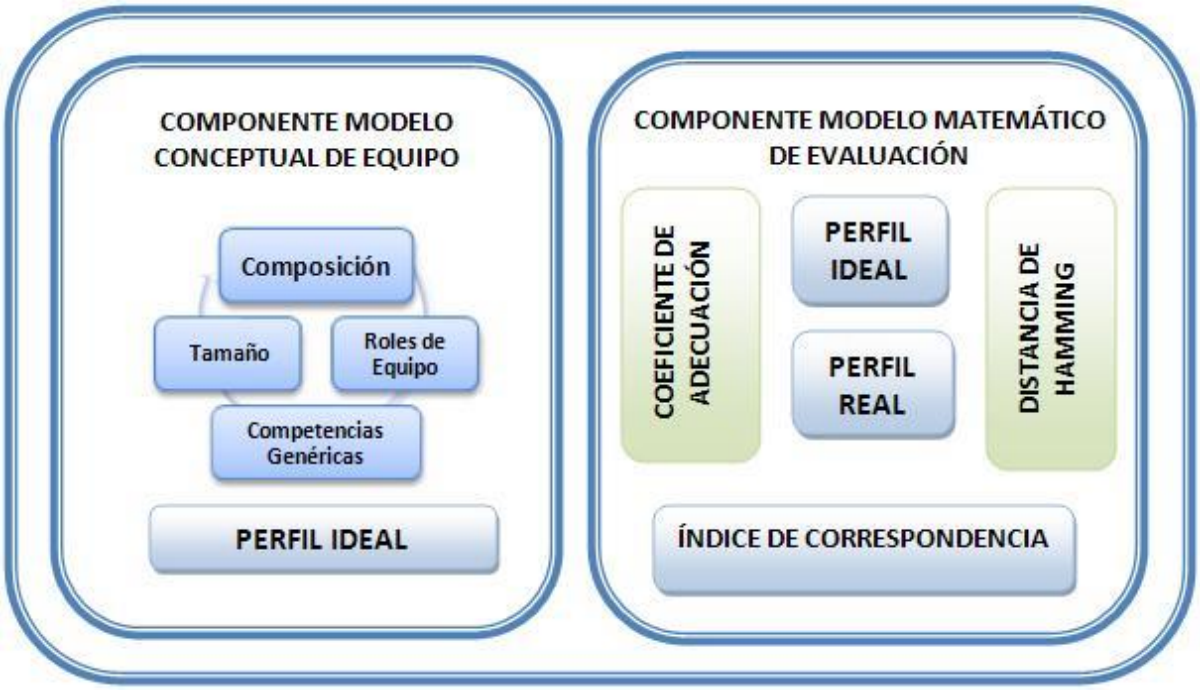


Ilustración 1: Modelo para la evaluación de la composición de equipos de proyectos informáticos.

2.4.2 Modelo conceptual de equipo

El modelo conceptual de equipo describe teóricamente la composición ideal de los equipos de proyectos informáticos a partir del tamaño del equipo, los roles de equipo (características de la personalidad) y las competencias genéricas. Está compuesto por tres perfiles ideales que describen la composición del equipo en función del tamaño de los mismos [4].

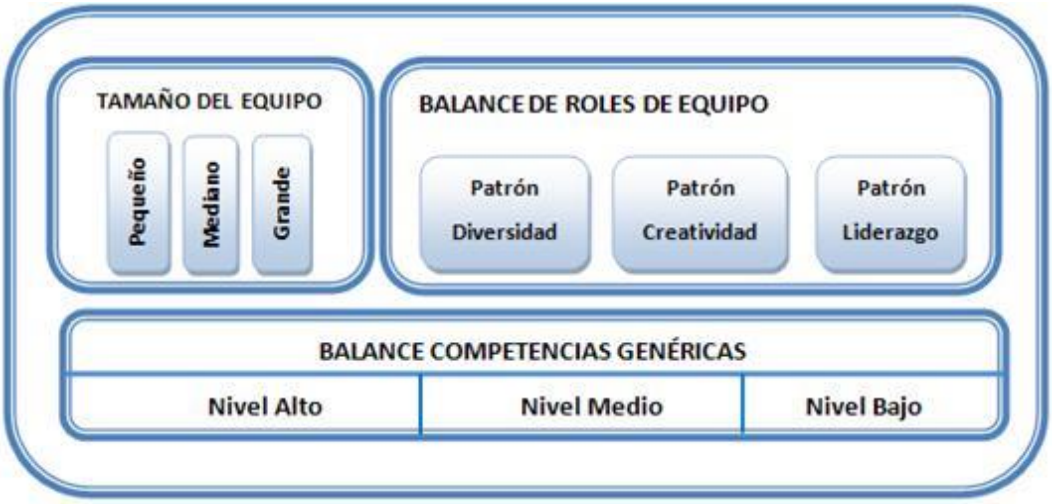


Ilustración 2: Descripción general de los perfiles ideales.

En el modelo conceptual de equipos se realiza un análisis del tamaño del equipo, los roles de equipo y las competencias genéricas con el fin de obtener un equilibrio entre estos elementos que describan la composición ideal de los equipos de proyectos informáticos.

2.4.3 Modelo matemático de evaluación

El modelo matemático de evaluación se basa en un proceso de comparación de las exigencias impuestas en el modelo conceptual de equipo (los perfiles ideales) y las características de los equipos reales, estableciendo que de esta comparación se obtienen criterios para la evaluación final de los equipos reales. Su objetivo principal es obtener una medida de la correspondencia que existe entre el perfil ideal y el perfil real de un equipo. Usa como métodos matemáticos de comparación el Coeficiente de Adecuación y la Distancia de Hamming obteniendo como salida en ambos casos el índice de correspondencia que posee el equipo real respecto a su ideal [4].

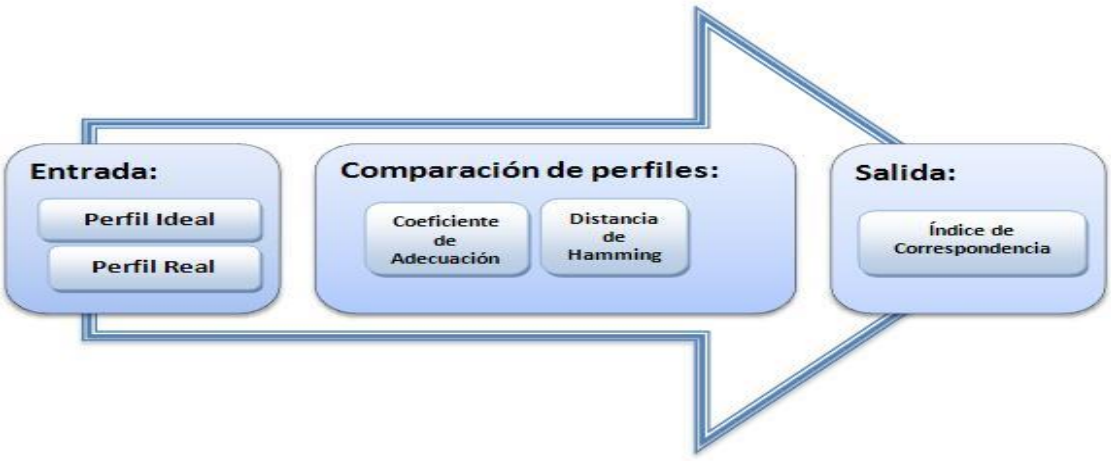


Ilustración 3: Modelo matemático de evaluación.

2.5 Propuesta del sistema

Se propone como solución módulo que se integre a la herramienta GESPRO con el cual se pretende satisfacer la necesidad de contar con un sistema para la gestión de la evaluación de los equipos de proyectos informáticos.

Se trata de un complemento web completamente exportable para la gestión de la evaluación de equipos de proyectos que contribuya a evaluar los mismos y refleje las evaluaciones más frecuentes de estos grupos dentro del proyecto. Facilitando el proceso de control y evaluación de los equipos y además logrando un manejo eficiente de los recursos humanos en un proyecto, teniendo a los miembros de un equipo en el lugar que puedan resultar más útiles.

La arquitectura que se propone para el desarrollo de este plugin es: Modelo-Vista-Controlador (MVC), el sistema cuenta como lenguaje básico a Ruby utilizando el marco de trabajo (framework) Ruby on Rails, a implementarse en el IDE de desarrollo NetBeans que cuenta con el soporte necesario para este. Además, se utiliza como servidor de base de datos PostgreSQL.

2.6 Ventajas de implementar un módulo que informatice el modelo de evaluación de la composición de equipos de proyectos informáticos.

La implementación de un módulo que informatice el modelo de evaluación de composición de equipos de proyectos informáticos debe recoger información sobre características personales y colectivas de los miembros de un equipo, conocer las potencialidades que posee un equipo de funcionar lo mejor posible teniendo en cuenta elementos de su composición que son independientes del entorno de trabajo donde se desarrolle. Poder analizar el comportamiento de los equipos en la historia y facilitar el proceso de evaluación mejorando con eso la calidad de los proyectos. Contribuir además a realizar organizadamente la composición de los mismos y permitir tener un control de todos los equipos con los que se cuenta en los proyectos. Así como saber los elementos que están impactando sobre su desempeño eficaz.

2.7 Importancia de la integración del modelo para la evaluación de la composición de equipo de proyecto informático a la herramienta GESPRO.

La integración del modelo para evaluar la composición de equipo de proyecto en la herramienta GESPRO, debe suplir la necesidad de que los proyectos puedan aumentar las posibilidades para los equipos de obtener altos niveles de rendimiento en su entorno productivo, a partir de la adecuada identificación de las potencialidades de sus integrantes y áreas de mejoras en relación a su composición, integrada al sistema de gestión de proyectos que hoy existe en la universidad, sobre el cual todos los

proyectos llevan registradas las actividades realizadas durante su ciclo de vida, permitiéndole el control y seguimiento del mismo. Pudiéndose además a partir de ahí utilizar las experiencias obtenidas de sus resultados en los futuros planes y estrategias de desarrollo a fin de aumentar las posibilidades de éxito de los equipos. Permitirá formar equipos competentes capaces de responder a diversas necesidades en un determinado proyecto. Además, debe permitir la informatización del modelo como una estrategia fundamental para la utilización en todos los proyectos.

La solución a implementar debe poder evaluar equipos de proyectos informáticos desde la perspectiva de la composición de los mismos, mediante características como los roles, las competencias genéricas y el tamaño del equipo. Se necesita tener una herramienta que ayude a comprobar el cumplimiento de los trabajos de los equipos en un proyecto productivo, evaluando el nivel de desempeño de las tareas y sabiendo cuanto están rindiendo en trabajo de equipo como un todo, de modo que se aprovechen al máximo las posibilidades de los integrantes de los mismos de trabajar en conjunto.

2.8 Requisitos de la herramienta

Los requisitos representan las condiciones o capacidades que debe tener un sistema o componente de un sistema, para satisfacer un contrato, estándar o documento impuesto formalmente. Tiene como propósito, establecer un entendimiento común entre el usuario y el grupo de desarrollo del software sobre los requisitos que solicita el usuario.

Se dividen en dos grandes categorías:

- Requisitos funcionales.
- Requisitos no funcionales.

A continuación se exponen los requisitos funcionales y no funcionales, definidos para la realización y desarrollo del sistema que se propone. Los mismos se han especificado a partir de las características y funcionalidades que debe tener el sistema que se va a desarrollar.

2.8.1 Requisitos funcionales:

1. Gestionar equipos del proyecto.

   1.1 Adicionar equipos al proyecto.

   1.2 Mostrar equipos del proyecto.

   1.3 Eliminar equipos de proyecto.

1.4 Actualizar los equipos de proyectos.

2. Gestionar integrantes de los equipos de proyectos.

   2.1 Adicionar los integrantes del equipo de proyecto.

   2.2 Mostrar los integrantes del equipo de proyecto.

   2.3 Eliminar integrantes del equipo de proyecto.

   2.4 Actualizar los integrantes del equipo de proyectos.

3. Evaluar equipo de proyecto.

   3.1 Capturar información primaria de las competencias genéricas de los miembros del equipo.

   3.2 Capturar información primaria sobre los roles de Belbin de los miembros del equipo.

   3.3 Mostrar la verificación de las columnas de las evaluaciones cargadas para comprobar la coincidencia con el formato establecido en la plantilla.

   3.4 Evaluar el equipo de proyecto a partir del modelo matemático de evaluación.

   3.5 Mostrar los resultados de la evaluación del equipo de proyecto.

4. Realizar análisis de los resultados de las evaluaciones.

   4.1 Mostrar matriz de evaluación grupal de equipo.

   4.2 Mostrar matriz de competencias genéricas de los miembros del equipo.

   4.3 Listar historial de las evaluaciones de todos los equipos.

   4.4 Filtrar por identificador del equipo los resultados de sus evaluaciones anteriores.

2.8.2 Requisitos no funcionales

Describen aspectos del sistema que son visibles por el usuario y no incluyen una relación directa con el comportamiento funcional del sistema. Son propiedades o cualidades que el producto debe tener y cumplir.

Software:

- A nivel del cliente o usuario el módulo puede ser accedido desde cualquier plataforma o sistema operativo, solo es necesario contar con una conexión a la red apropiada y un navegador web.
- Se recomienda el uso del navegador Mozilla Firefox para su funcionamiento óptimo.

SO en los servidores:

- GNU/Linux Ubuntu 12.04 LTS.

SO en los clientes:

- Cualquier SO que decida la entidad cliente.
- PostgreSQL debe estar correctamente configurado para aceptar conexiones vía TCP/IP utilizando el método de autenticación por md5.

Hardware

- Como requerimiento mínimo el sistema debe contar con procesador Pentium II o versiones superiores con no menos de 256 de RAM.
- Se recomienda para su funcionamiento óptimo máquinas con procesadores Pentium IV y 512 de RAM.

Seguridad

- El módulo debe mantener la integridad y confiabilidad ante la información manejada, ya que la ausencia de estos aspectos puede ocasionar grandes riesgos, viéndose afectado entonces el manejo de los datos de las evaluaciones de los equipos.

Disponibilidad

- Disponibilidad del sistema global del 90%. Está configurado para que permanezca trabajando las 24 horas los 365 días del año.
- El tiempo medio de reparación debe ser menor de 1 día.

Confidencialidad

- Permite el acceso al sistema a través de una cuenta de usuario única que puede ser local o por autenticación con el LDAP.

Confiabilidad

- Se garantiza la llegada de los datos de forma íntegra y segura al destino. El acceso por las cuentas de administración a los servidores se realiza a través de una contraseña generada con algoritmo MD5. El listado de las claves de cada servidor se almacena en un .rar cifrado.

Accesibilidad

- Los usuarios creados en el sistema GESPRO tienen acceso (o permiso) limitado, podrán acceder a un conjunto de funcionalidades según el rol o perfil donde fue clasificado el usuario, que a su vez este rol es especificado en un nivel determinado para la toma de decisiones.

Apariencia o interfaz

- La aplicación cuenta con varios elementos para personalizar el color del entorno de la aplicación de los posibles clientes en correspondencia con el color

característico para su identidad. Además, la interfaz deberá ser sencilla de usar, amigable y donde cualquier usuario podría hacer uso de esta sin dificultad.

Rendimiento

- El sistema debe permitir el acceso concurrente de un gran número de usuarios. Además de ser lo más eficiente posible para poder lograr un tiempo de respuesta de 0.1 a 0.7 segundos ante la concurrencia de peticiones.

Ejemplo de escenarios soportados por el GESPRO:

- Cantidad de proyectos: más de 200.
- Cantidad de usuarios conectados: 6000

Usabilidad

- El sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente web en sentido general. También debe ser lo más interactivo posible, brindará una interfaz simple y amigable para que el usuario no tenga dificultad al utilizarlo.

Requerimientos del diseño e implementación

- Se debe implementar en el lenguaje Ruby utilizando el marco de trabajo Ruby on Rails
- Para el tratamiento de la base de dato se utiliza PostgreSQL.
- Para el análisis y el diseño del módulo se emplea la metodología XP, usando el lenguaje de modelación UML y como herramienta para llevar a cabo el modelado Visual Paradigm.
- Para la implementación de la aplicación se hace uso del IDE NetBeans.

2.9 Historias de usuarios:

Las historias de usuario (HU) constituyen una manera simple de describir una tarea concisa que aporta valor al usuario o al negocio. Representan una breve descripción del comportamiento del sistema, emplea terminología del cliente sin lenguaje técnico, se realiza una por cada característica principal del sistema, se emplean para hacer estimaciones de tiempo y para el plan de lanzamientos, reemplazan un gran documento de requisitos y presiden la creación de las pruebas de aceptación [30].

Las Historias de Usuario deben cumplir las siguientes características para realizar su función de manera correcta:

- *Independientes*. Deben ser atómicas en su definición. Es decir, se debe intentar que no dependa de otras historias para poder completarla.
- *Negociables*. Deben ser ambiguas en su enunciado para poder debatirlas, dejando su concreción a los criterios de aceptación.

- *Valoradas*. Deben ser valoradas por el cliente. Para poder saber cuánto aporta al valor de la aplicación y junto con la estimación convertirse en un criterio de prioridad.
- *Estimables*. Aunque sea siempre un poco como leer de una bola de cristal, deben poder ser estimadas. Tener su alcance lo suficientemente definido como para poder suponer una medida de trabajo en la que pueda ser completada.
- *Pequeñas*. Para poder realizar una estimación con cierta validez y no perder la visión de la Historia de Usuario, se recomienda que sean mayores de dos días y menores de dos semanas.
- *Verificables*. Este es el gran avance de las Historias de Usuario. Que, junto con el cliente, se acuerdan unos Criterios de Aceptación que verifican si se ha cumplido con las funcionalidades descritas y esperadas.

Clasificación de las historias de usuarios:

La prioridad en el negocio:

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

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

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

El riesgo en su desarrollo:

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

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

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

A continuación de muestran algunas historias de usuarios y el resto las puedes encontrar en el anexo 4.

Tabla 4: Historia de usuario. Gestionar equipos.

Historia de Usuario	
Código: HU5	**Nombre Historia de Usuario:** Gestionar equipos del proyecto.

Modificación de Historia de Usuario: Primera	
Referencia: RF1	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Alta	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 1 semana
Descripción: La funcionalidad permitirá realizar operaciones como adicionar, mostrar, eliminar o actualizar los diferentes equipos existentes en el proyecto.	
Observaciones:	

Tabla 5: Historia de usuario. Gestionar integrantes.

Historia de Usuario	
Código: HU6	**Nombre Historia de Usuario:** Gestionar integrantes de los equipos de proyectos.
Modificación de Historia de Usuario: Primera	
Referencia: RF2	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Alta	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 1 semana
Descripción: La funcionalidad permitirá realizar operaciones como adicionar, mostrar, eliminar o actualizar a los integrantes de los equipos existentes en el proyecto.	
Observaciones: Para realizar las operaciones pertinentes a este requisito funcional se debe haber creado con anterioridad el equipo de proyecto.	

Tabla 6: Historia de usuario. Importar datos.

Historia de Usuario	
Código: HU1	**Nombre Historia de Usuario**: Captura de información primaria de las competencias genéricas de los miembros del equipo.
Modificación de Historia de Usuario: Primera	

Referencia: RF3.1	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Alta	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 1 semana
Descripción: La funcionalidad permite cargar los datos de los miembros del equipo referido a sus competencias genéricas, para luego incorporarlos en la base de datos de persona en el GESPRO.	
Observaciones:	

Tabla 7: Historia de usuario. Captura de información sobre los roles de Belbin.

Historia de Usuario	
Código: HU2	**Nombre Historia de Usuario**: Captura de información primaria sobre los roles de Belbin de los miembros del equipo.
Modificación de Historia de Usuario: Primera	
Referencia: RF3.2	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Alta	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 1 semana
Descripción: La funcionalidad permite cargar los datos de los miembros del equipo sobre los roles de Belbin, para luego incorporarlos en la base de datos de persona en el GESPRO.	
Observaciones:	

Tabla 8: Historia de usuario. Verificación de columnas.

Historia de Usuario	
Código: HU3	**Nombre Historia de Usuario:** Mostrar la verificación de las columnas de las evaluaciones cargadas para comprobar la coincidencia con el formato establecido en la plantilla.
Modificación de Historia de Usuario: Primera	

Referencia: RF3.3	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Alta	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 1 semana
Descripción: La funcionalidad permite mostrar la verificación de las columnas de las evaluaciones cargadas para comprobar la coincidencia con el formato establecido en la plantilla, además de saber si los miembros pertenecen a ese equipo.	
Observaciones:	

Tabla 9: Historia de usuario. Evaluar equipo.

Historia de Usuario	
Código: HU7	**Nombre Historia de Usuario:** Evaluar el equipo de proyecto a partir del modelo matemático de evaluación.
Modificación de Historia de Usuario: Primera	
Referencia: RF3.4	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Alta	**Puntos Estimados:** 2 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 2 semana
Descripción: La funcionalidad permite evaluar el equipo de proyecto a partir de lo planteado en el modelo matemático de evaluación.	
Observaciones:	

2.10 Plan de iteraciones

Todo proyecto que utiliza la Metodología XP, como parte de su ciclo de vida, crea un plan de duración de cada una de las iteraciones, en este caso se hace para el único equipo de desarrollo con el cual se cuenta. Este plan tiene como objetivo mostrar la duración de cada iteración, así como el orden en que serán implementadas las historias de usuario en cada una de las mismas.

Tabla 10: Plan de duración de las iteraciones.

Iteraciones	Historias de Usuario	Duración total

1ra iteración	1. Captura de información primaria de las competencias genéricas de los miembros del equipo. 2. Captura de información primaria sobre los roles de Belbin de los miembros del equipo. 3. Mostrar la verificación de las columnas de las evaluaciones cargadas. 4. Definir los equipos donde trabajan las personas en los proyectos.	**4 semanas**
2da iteración	5. Gestionar equipos del proyecto. 6. Gestionar integrantes de los equipos de proyectos. 7. Evaluar el equipo de proyecto a partir del modelo matemático de evaluación. 8. Mostrar los resultados de la evaluación del equipo de proyecto.	**5 semanas**
3ra iteración	9. Listar historial de las evaluaciones de todos los equipos. 10. Filtrar por identificador del equipo los resultados de sus evaluaciones anteriores. 11. Realizar un análisis de los resultados de las evaluaciones.	**3 semanas**

2.11 Plan de entrega de las iteraciones

En el plan de entrega que se plantea a continuación se hace una propuesta de la fecha aproximada en que se harán versiones (releases) del sistema al finalizar cada iteración en la fase de implementación. El objetivo de este plan es producir rápidamente versiones de la aplicación que sean operativas, aunque estas no cuenten con toda la funcionalidad pretendida.

Tabla 11: Plan de entrega de las iteraciones.

Iteraciones	**Final de la iteración (7 de marzo de 2013)**	**Final de la iteración (14 de abril de 2013)**	**Final de la iteración (5 de mayo de 2013)**
Iteración 1	x		
Iteración 2		x	
Iteración 3			x

2.12 Diseño del sistema

El diseño constituye una parte imprescindible dentro del proceso de desarrollo de todo producto, el mismo transforma el esbozo preliminar y crea un conjunto de elementos

del modelo que serán posteriormente implementados utilizando para su modelación el UML.

2.12.1 Estilos de arquitectura

La arquitectura proporciona una visión global del sistema a construir. Describe la estructura y la organización de los componentes del software, sus propiedades y las conexiones entre ellos. Los componentes del software incluyen módulos de programas y varias representaciones de datos que son manipulados por el programa [31].

Un estilo arquitectónico es un conjunto coordinado de restricciones arquitectónicas que establece los roles/rasgos de los elementos arquitectónicos y las relaciones permitidas entre esos elementos dentro de la arquitectura que se conforma a ese estilo. Los estilos se encuentran en el centro de la arquitectura y constituyen buena parte de su sustancia.

Estilos de Llamada y Retorno

Entre los estilos de arquitectura se encuentra el estilo de llamada y retorno .Esta familia de estilos enfatiza la modificabilidad y la escalabilidad. Son los estilos más generalizados en sistemas de gran escala. Miembros de la familia son las arquitecturas de programa principal y subrutina, los sistemas basados en llamadas a procedimientos remotos, los sistemas orientados a objeto y los sistemas jerárquicos en capas[31].

El estilo arquitectónico seleccionado para el desarrollo de este plugin es el estilo de llamada y retorno que contiene el patrón arquitectónico MVC (Ver Anexo 1). Este estilo o patrón, estructura el desarrollo de la aplicación separándola en tres componentes distintos, lógica, control y presentación.

2.12.2 Patrón de arquitectura Modelo Vista Controlador:

Es una arquitectura de software que separa la lógica de negocio de la interfaz de usuario, su objetivo es organizar el flujo de datos en las aplicaciones, permitiendo construir sistemas más robustos y fáciles de mantener y extender. Se ve frecuentemente en aplicaciones web, donde la vista es la página HTML (lenguaje de marcado de hipertexto), el controlador es el código que obtiene datos dinámicamente y genera el contenido HTML y el modelo incluye la información almacenada en la base de datos junto con las reglas de negocio que transforman esa información [32].

El mismo soporta múltiples vistas, dado que la vista se halla separada del modelo y no existe una dependencia directa entre ambos. Se puede modificar uno de los componentes sin conocer cómo funcionan los otros, el programador no debe preocuparse por solicitar la actualización de las vistas, ya que este proceso es

realizado automáticamente. El objetivo de este modelo es intentar disminuir las repeticiones y tenerlo todo organizado, o sea, hacer una distribución entre la lógica de toda la aplicación y la presentación.

Se seleccionó para el desarrollo del sistema informático, el patrón arquitectónico Modelo Vista Controlador, ya que a pesar de ser un patrón de diseño de arquitectura que está asociado a la idea de 3 capas su objetivo es aún más exquisito, además el GESPRO basa su estructura en este patrón. El mismo se centra en la secuencia de ejecución, desde que se produce un evento en la capa de presentación hasta que es atendido en forma completa.

Por otra parte el sistema informático contará con métodos, que de no usar este patrón arquitectónico se invocarían desde la interfaz, convirtiendo a la misma dependiente del negocio como sucede en la arquitectura 3 capas, no siendo esto conveniente para el sistema ya que la responsabilidad de la interfaz debe ser solamente la de captar y mostrar información, no de hacer llamadas al negocio.

Ilustración 4: Patrón de arquitectura MVC.

Vista (View): Se presenta usualmente como un medio de interfaz de usuario o para mostrar información contenida en el modelo [33].Una vista está asociada a un modelo, pudiendo existir también varias vistas asociadas al mismo modelo. Una vista obtiene del modelo solamente la información que necesita para desplegar y se actualiza cada vez que el modelo del dominio cambia por medio de notificaciones generadas por el modelo de la aplicación.

Modelo (Model): El modelo es un conjunto de clases que representan la información del mundo real que el sistema debe procesar, sin tomar en cuenta ni la forma en la que esa información va a ser mostrada ni los mecanismos que hacen que esos datos estén dentro del modelo, es decir, sin tener relación con ninguna otra entidad dentro de la aplicación. Se realizan validaciones permitiendo asegurar la integridad de los datos con los que se trabajan, permite el apoyo a la controladora en la creación de funciones para la muestra de la información enviada a las vistas [33].

Controladora (Controller): Se ejecuta mediante eventos tanto en la interfaz de usuario como en la modelo, mayormente provocado por acciones de los usuarios dentro de las interfaces presentes en las vistas realizando modificaciones posteriores en las vistas y los modelos [33].

2.13 Patrones de diseño

Los patrones de diseño son descripciones de clases cuyas instancias colaboran entre sí. Cada patrón es adecuado para ser adaptado a un cierto tipo de problema. Representa un esquema o micro-arquitectura que supone una solución a problemas semejantes; una estructura común que tienen aplicaciones similares.

Los patrones de diseño brindan una solución generalmente ya probada y documentada a problemas que se dan durante el proceso de desarrollo de software [34]. Emplean un conjunto de buenas prácticas que facilitan el trabajo, definen una estructura de clases que da respuesta a uno o varios problemas en particular y presentan la ventaja de que son fáciles de comprender, además de que no dependen del lenguaje, haciéndolos genéricos. Lo complejo es cuando se tiene que decidir cuál usar, pues presentan diferentes soluciones, ya sea a través del empleo de uno u otro, o la combinación de varios. De ahí la importancia de conocer y estudiar los diferentes patrones que existen para poder determinar su uso.

A continuación se relacionan los patrones de diseño empleados y su aplicación en la realización de los diagramas, así como durante la implementación.

Patrones GRASP: Patrones que asignan responsabilidades a través de la descripción de los principios fundamentales del diseño y la solución a un problema. Entre ellos existen 5 patrones fundamentales:

- Experto: Se estableció la asignación de responsabilidades específicas por cada una de las clases del sistema. Se cuenta con las clases controladoras, modelos y entidades, que poseen funcionalidades específicas atendiendo a la actividad que realizan y los procesos que gestionan.

- Creador: Se aplica para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto sólo pueda ser creada por el objeto que contiene la información necesaria para ello. El uso de este patrón permite crear las dependencias mínimas necesarias entre las clases, lo que favorece al mantenimiento del sistema y ofrece mejores oportunidades de reutilización. Se emplea este patrón para alcanzar también el bajo acoplamiento, encapsulación y posibilitar la reutilización de código. Las clases controladoras son las encargadas de crear las modelos y estas a su vez las entidades garantizando con ello la distribución por capas de la arquitectura.

- Controlador: Las clases Controladoras que son las encargadas de gestionar el acceso a lógica de negocio y controlar todo el proceso. En el caso de que sean muy extensas estas se dividen en varias controladoras para separar la carga de procesamiento, disminuir la complejidad y responsabilidad de las clases.

- Bajo acoplamiento: Se diseñó bajo este principio, lo que permite el desarrollo por separado a partir de las especializaciones individuales de cada uno. El intercambio de información se realiza a través de servicios implementados que se encargan de distribuir la información para la realización de procesos dependientes. Además permitir la menor dependencia posible entre las clases de manera que si se realiza una modificación en alguna de ellas no afecte el funcionamiento de las otras.
- Alta cohesión: El diseño y la dependencia entre clases están elaborados a partir de las funcionalidades que realizan, presentando alta relación de afinidad en las operaciones que controlan. En el caso de las actividades de alta complejidad comparten relación con otros objetos, disminuyendo la carga de transacciones entre ellas.

2.14 Diagrama de clases

El diagrama de clases describe gráficamente las especificaciones de las clases y las interacciones entre estas. Contiene la siguiente información:
Clases, asociaciones y atributos, métodos, información sobre los tipos de atributos, navegabilidad, dependencias.

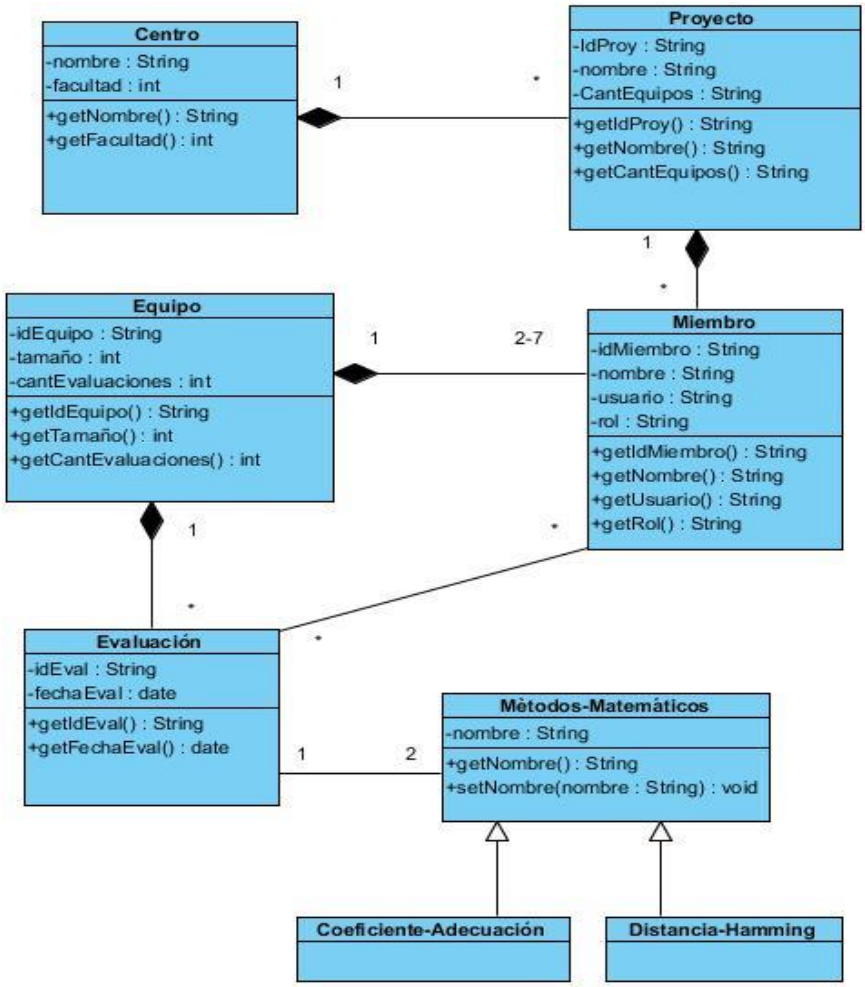


Ilustración 5: Diagrama de clases.

El diagrama de clases mostrado está compuesto por ocho entidades fundamentales, entre las que se encuentran centro, proyecto, equipo, individuo, evaluación y métodos matemáticos.

El GESPRO está dividido en centros productivos que contienen diversos proyectos, con diferentes equipos formados. Cada equipo está compuesto por hasta 7 individuos de ese proyecto a los cuales se les realizan evaluaciones frecuentes como una manera de controlar la realización de las tareas productivas. Esta evaluación se basa en dos métodos matemáticos fundamentales los cuales son Coeficiente de Adecuación y Distancia de Hamming.

2.15 Modelo de Base Datos

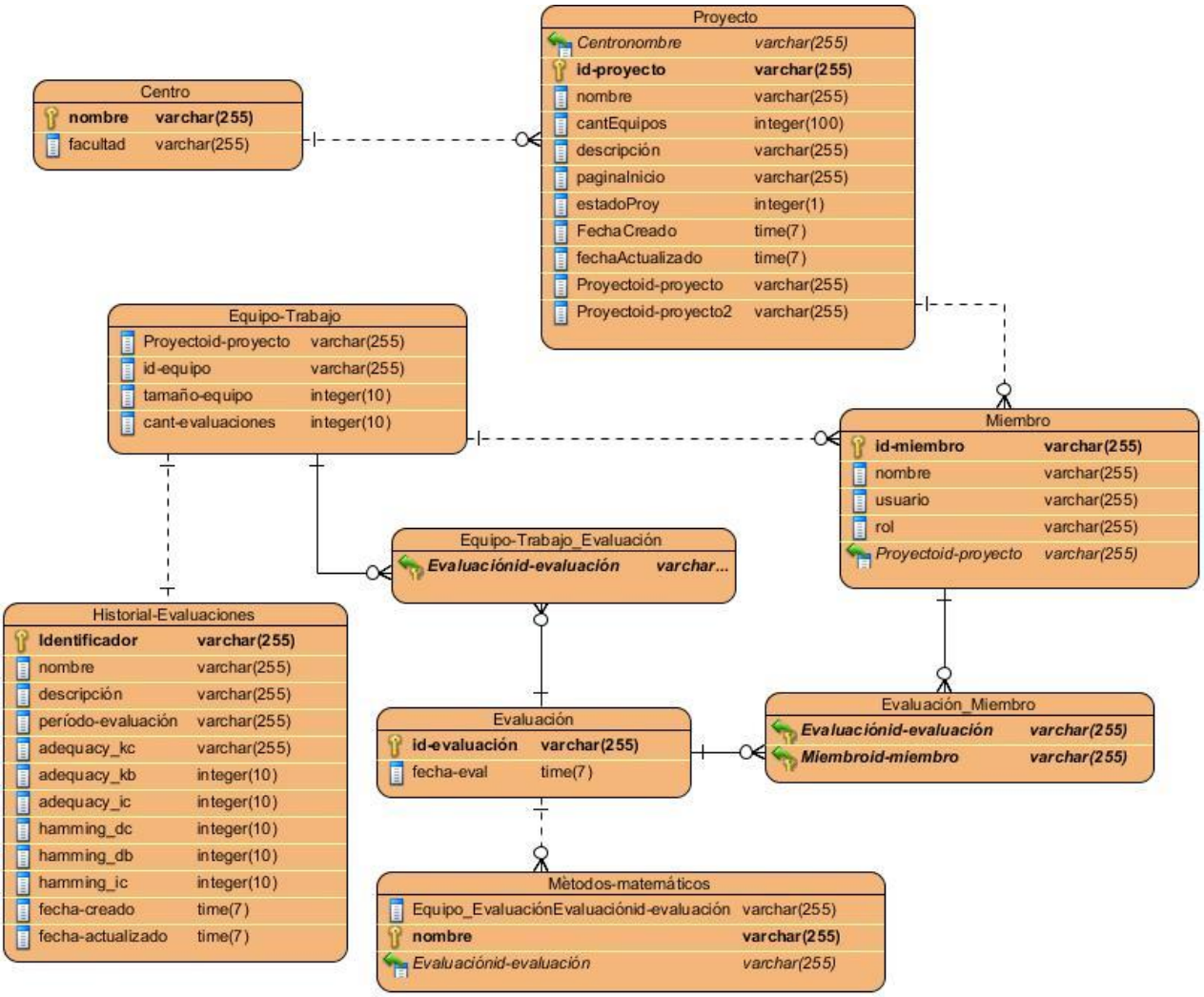


Ilustración 6: Diagrama de entidad - relación.

En el anterior modelo de datos se representa la relación entre las clases persistentes. Entre la tabla Centro y la tabla Proyecto existe una relación de “uno a muchos” mostrando que un centro tiene de uno a muchos proyectos, por tal razón su llave primaria pasa a ser llave foránea de la tabla Proyecto; esta a su vez se relaciona con la clase Miembro a través de una relación de “uno a muchos” también, aplicándose el mismo procedimiento que en las anteriores tablas.

La tabla Miembro se relaciona con las tablas Equipo y Evaluación, a través de la relación de “uno a muchos” y de “muchos a muchos” respectivamente. Originando la última relación de esta manera una nueva tabla llamada Evaluación_Miembro, relacionándose entonces la tabla Evaluación con la de Métodos_matemáticos y Equipo ya que las evaluaciones se realizan mediante diferentes métodos matemáticos a varios equipos de trabajo. Además de tenerse almacenado un historial de las

evaluaciones de esos equipos para posteriormente poder analizar la evolución de los mismos durante un determinado período de tiempo escogido.

2.15.1 Descripción de las tablas

Nombre: Centro		
Descripción: Almacena los datos principales del centro productivo.		
Atributo	**Tipo**	**Descripción**
Nombre	varchar	Nombre que identifica al centro.
Facultad	Integer	Identificador de la facultad a la que perteneces el centro.

Tabla 12: Descripción de la tabla Centro.

Nombre: Proyecto		
Descripción: Guarda los datos referidos al proyecto que puedan servir para realizar la evaluación de equipos.		
Atributo	**Tipo**	**Descripción**
Identificador	varchar	Identificador del proyecto.
Nombre	varchar	Nombre que fue asignado al proyecto.
Cantidad equipos	Integer	Cantidad de equipos presentes en el proyecto.
Descripción	varchar	Descripción del proyecto.
Página inicio	varchar	Página de inicio del proyecto.
Estado proyecto	varchar	Estado en que se encuentra el proyecto.
Fecha creado	Time	Fecha de creado el proyecto
Fecha actualizado	Time	Fecha de actualizado el proyecto.

Tabla 13: Descripción de la tabla Proyecto.

Nombre: Equipo-Trabajo		
Descripción: Guarda los datos referidos a los equipos presentes en el proyecto.		
Atributo	**Tipo**	**Descripción**
Identificador	varchar	Identificador del equipo.
Tamaño	Integer	Tamaño del equipo de proyecto.
Cantidad evaluaciones	Integer	Cantidad de evaluaciones que presenta el mismo.

Tabla 14: Descripción de la tabla Equipo_Trabajo.

Nombre: Miembro		
Descripción: Guarda los datos de los miembros de un proyecto.		
Atributo	**Tipo**	**Descripción**
Identificador	varchar	Identificador del miembro del proyecto.
Nombre	varchar	Nombre del miembro del proyecto
Usuario	varchar	Usuario que tiene el miembro.
Rol	varchar	Rol que ocupa el miembro.

Tabla 15: Descripción de la tabla Miembro.

Nombre: Evaluación		
Descripción: Guarda los datos de las evaluaciones realizadas a los equipos del proyecto.		
Atributo	**Tipo**	**Descripción**
Identificador	varchar	Identificador de la evaluación.
Fecha	Time	Fecha de realizada la evaluación.

Tabla 16: Descripción de la tabla Evaluación.

Nombre: Métodos_Matemáticos
Descripción: Guarda los datos referentes a los métodos matemáticos utilizados para realizar la evaluación.

Atributo	Tipo	Descripción
Nombre	varchar	Nombre que identifica al método utilizado.

Tabla 17: Descripción de la tabla Mètodos_Matemáticos.

Nombre: Historial_Evaluaciones		
Descripción: Guarda los datos referentes a los métodos matemáticos utilizados para realizar la evaluación.		
Atributo	**Tipo**	**Descripción**
Identificador	Time	Identificador del historial.
Nombre	varchar	Nombre del equipo al que se le realizó la evaluación que se está guardando.
Descripción	varchar	Descripción de la historia de usuario.
Período_evaluación	Time	Período en el que se realizó la evaluación.

Tabla 18: Descripción de la tabla Historial_Evaluaciones.

2.16 Diagrama de despliegue

El Modelo de Despliegue o diagrama de despliegue es un tipo de diagrama que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. En muchos casos el modelado de la vista de despliegue implica modelar la topología del hardware sobre el que se ejecuta el sistema.

El modelo de despliegue que se presenta está compuesto por tres nodos que representan la computadora del usuario del sistema, el servidor de la herramienta GESPRO interconectados por el protocolo HTTP y la base de datos con la cual interactúa el sistema.

La herramienta de gestión de proyectos GESPRO está instalada dentro de la UCI en un Servidor Apache, utilizando PostgreSQL como SGBD encargado de almacenar los valores con que trabajará la aplicación. Todas las PC clientes podrán acceder a dicho servidor mediante el protocolo de comunicación HTTP, logrando que en cada estación de trabajo los usuarios tengan acceso a la herramienta.

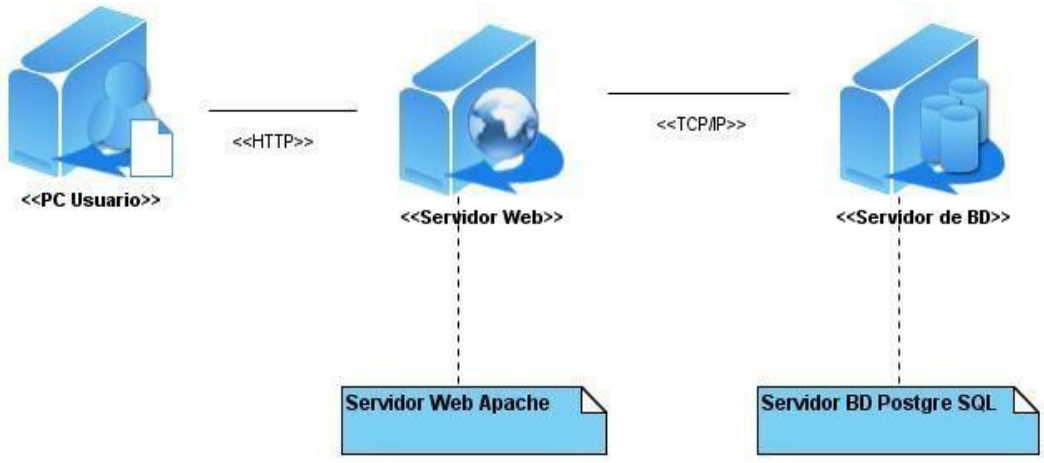


Ilustración 7: Diagrama de despliegue.

2.17 Framework de desarrollo Ruby on Rails

2.17.1 Descripción del Framework Ruby on Rails

Rails es una plataforma de aplicaciones Web de código abierto la cual fue desarrollada en el lenguaje de programación Ruby. La simplicidad en el código y en la configuración son las principales ventajas al desarrollar aplicaciones ya que se ahorra código en comparación con otros frameworks. Esto permite al programador ahorrar tiempo en la producción de código y crear una sintaxis que muchos de sus usuarios encuentran muy legible. La plataforma Rails se distribuye a través de RubyGems, que es el formato oficial de paquete y canal de distribución de librerías y aplicaciones Ruby. Ver anexo 3.

Estructura del Framework Ruby on Rails

Dentro de este framework existe una gran estructura de paquetes y carpetas de las cuales las más importantes se describen a continuación:

Los subdirectorios más importantes dentro de este lenguaje son:

app: contiene el core de la aplicación, la división entre los subdirectorios model, view, controller y helper.

Subdirectorios de app:

- controllers: aquí es donde el framework pretende encontrar las clases controladoras, el objetivo del controlador es manejar las peticiones web realizadas por los usuarios desde el navegador.
- views: sujeta las plantillas de visualización que se llenan con los datos de la aplicación, las mismas que serán convertidas en HTML y devueltas al navegador del usuario.
- models: posee las clases de modelación de datos almacenados en la base de datos de la aplicación.
- helpers: en ella se guardan las clases de ayuda que se usan para asistir a las clases tanto models como views y los controladores. Esto ayuda a que dichas clases se mantengan sin crecer de tamaño, dedicadas y ordenadas a funciones específicas.

config: contiene el archivo database.yml que proporciona los detalles de configuración y acceso a la base de datos que se usa en la aplicación.

- locales: se almacenan los archivos dedicados a los idiomas de la aplicación permitiendo la fácil migración a cualquiera de estos.
- db: contiene archivos SQL para la creación de tablas en la base de datos.

- log: contiene un registro de todas las acciones que Rails lleva a cabo, muy útil para rastrear errores.
- public: es el directorio disponible para Apache, que incluye subdirectorios de imágenes, javascripts, y de hojas de estilos.

2.18 Clases fundamentales dentro del lenguaje

ActionController: Son el núcleo de una solicitud web en Rails. Se componen de una o más acciones que se ejecutan y luego, o bien hacer una plantilla o redirigir a otra acción. Una acción se define como un método público en el controlador, lo que automáticamente se pondrá a disposición del servidor a través de las rutas de Rails. De forma predeterminada, sólo el ApplicationController en una aplicación Rails hereda de ActionController :: Base. Todos los controladores de otros a su vez heredan de ApplicationController.

```
class ProjectTeamController < BaseTeamEvaluationApplicationController
  unloadable

  before_filter :find_project , :except => [:preview]
  before_filter :find_project_team , :only => [:update, :delete, :retrieve, :project_team_evaluat
    :project_team_evaluation_index, :project_team_evaluation_delete,
    :project_team_member_new,:project_team_member_delete,:project_team_member_index]
  before_filter :require_login
  before_filter :authorize, :except => [:preview ]

  helper :project_team
  include ProjectTeamHelper
```

Ilustración 8: ActionController

ActiveRecord: La representación de los datos y las relaciones entre ellos (llamado también lógica de negocios) consiste en las clases que representan a las tablas de la base de datos y las mismas son gestionadas por ActiveRecord. Por lo general, lo único que tiene que hacer el programador es heredar de la clase ActiveRecord::Base, y el programa averigua automáticamente qué tabla usar y qué columnas tiene.

```
class ProjectTeamEvaluation < ActiveRecord::Base
  unloadable
  #relaciones del model de las evaluaciones
  belongs_to :project_team

  acts_as_event :title => Proc.new {|o| "#{o.project_team.name} ##{o.project_team.id} : #{o.projec
    :url => Proc.new {|o| {:controller => 'project_team', :action => 'retrieve', :id => o.project_
    :author => Proc.new {|o| nil }
```

Ilustración 9: ActiveRecord

Estas son las principales clases que dan funcionamiento al framework Ruby on Rails aunque claramente sin todas las otras que existen dentro de las estructura de app la aplicación no funciona.

2.19 Estructura de un proyecto

En el GESPRO el código fuente está organizado en una estructura de tipo proyecto y los archivos del proyecto se almacenan en una estructura jerárquica, de tipo árbol. En la figura siguiente se ilustra cómo queda la estructura de un proyecto generado en Rails:

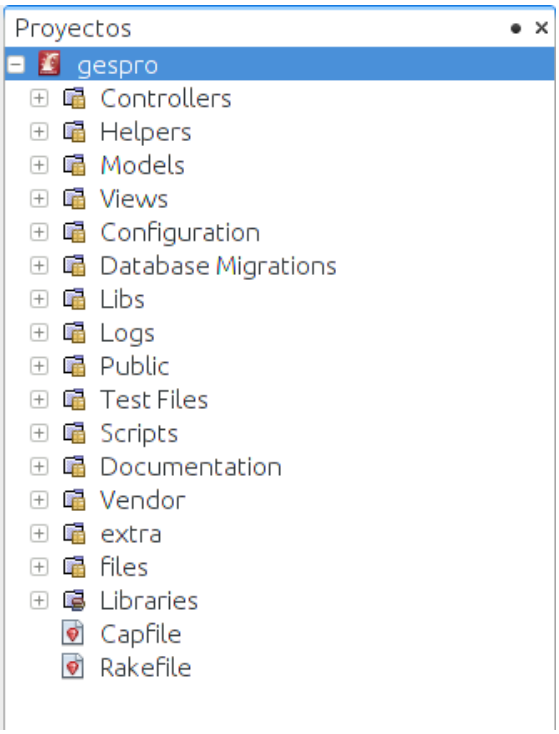


Ilustración 10: Estructura de un proyecto en el gespro.

2.20 Estructura de la aplicación

Un proyecto en el GESPRO puede estar compuesto por una o más aplicaciones. El sistema implementado cuenta con varias de ellas que se encuentran en la carpeta gespro dentro de la carpeta Vendor como se muestra en la figura:

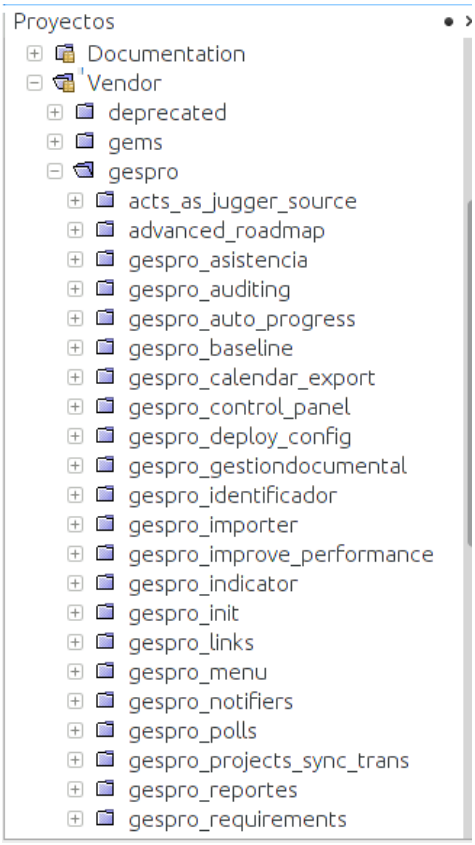


Ilustración 11: Estructura de una aplicación.

2.21 Estructura del módulo gespro_team_evaluation

El módulo implementado tiene como función gestionar todo lo relacionado con la evaluación de los equipos de proyecto pertenecientes al GESPRO. Dentro del mismo existe una gran estructura de paquetes y carpetas de las cuales las más importantes se muestran en la figura a continuación:

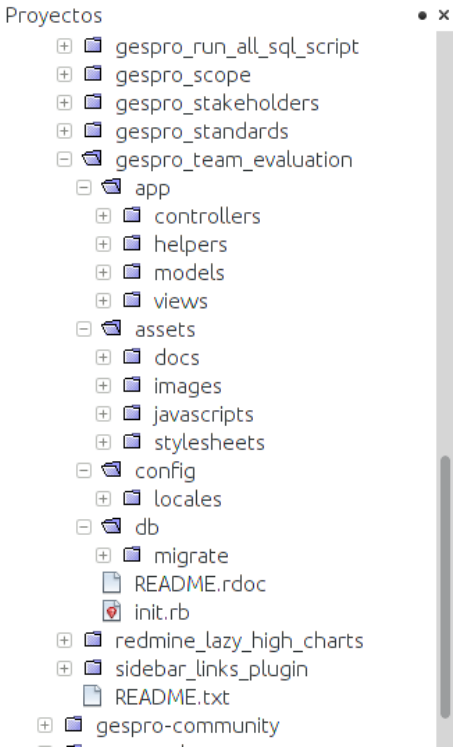


Ilustración 12: Estructura del módulo de evaluación de equipos.

2.22 Conclusiones

En este capítulo se plantearon diversos conceptos necesarios para el diseño del módulo. Se realizó un estudio a profundidad del modelo perteneciente a la autora (Stiven,2012) a informatizar partiendo del estudio del modelo conceptual y matemático planteado en el mismo.

Se describe detalladamente la arquitectura del sistema. Se explican los patrones de diseño utilizados, y se muestran los diagramas de clases, bases de datos y de despliegue generados. Partiendo de todo lo planteado y analizado en este capítulo se afirma que se ha logrado una eficaz implementación, pues el diseño desarrollado cumple con los requerimientos arquitectónicos necesarios para su correcto funcionamiento e integración en el sistema. Por tanto se concluye que durante el mismo se implementó un módulo para la evaluación de equipos de proyectos informáticos en el GESPRO en estado funcional.

Capítulo 3. Validación de la aplicación

3.1 Introducción

En el siguiente capítulo se exponen las pruebas realizadas al software desarrollado, con el objetivo de comprobar las funcionalidades del plugin en los diferentes escenarios, verificando en todos los casos que los resultados de las pruebas sean los esperados.

3.2 Validación

Las aplicaciones informáticas no están exentas a errores, por lo que a todas ellas es necesario aplicarles un conjunto de pruebas para certificar que cuentan con la calidad requerida, este es el factor fundamental para lograr entregar un software que cumpla o supere las expectativas del cliente.

El proceso de validación se dirige fundamentalmente a componentes del software o al sistema de software en general, con el objetivo de medir hasta cuando el software cumple las funcionalidades solicitadas por el cliente. Las pruebas del software verifican y revelan la calidad de un producto. Son utilizadas para identificar posibles errores en la implementación, calidad, o usabilidad de un programa de software.

3.2.1 Estrategia de validación

La estrategia de validación que se utiliza cuenta con dos acciones, evaluar la calidad del sistema y la facilidad de uso del mismo con respecto a los usuarios que lo utilizan. Para evaluar la calidad del sistema se realizan pruebas de caja negra las que se llevan a cabo mediante los casos de pruebas y las pruebas de caja blanca que se realizan al código de la aplicación verificando el correcto funcionamiento de determinados algoritmos.

Para evaluar la facilidad de uso se llevaron a cabo encuestas a usuarios ya sea especialistas del centro o jefes de proyectos interesados en gestionar en algún momento los equipos de proyectos y manejar la evaluación.

La ilustración 13 muestra la estrategia de validación que se llevó a cabo:

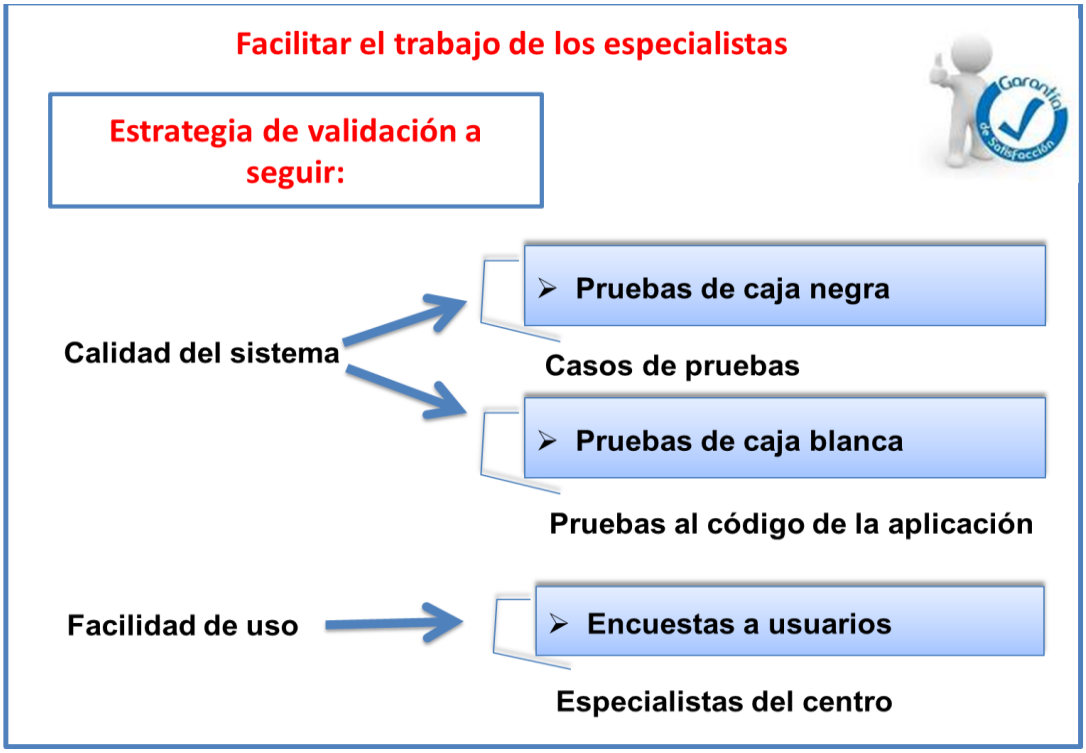


Ilustración 13: Estrategia de validación.

3.3 Pruebas

Las pruebas tienen gran importancia en el desarrollo de un software, ya que mediante éstas se pueden detectar y corregir errores tempranamente. Antes de entregar el producto al cliente final se debe garantizar que el software cumpla con todos los requerimientos y que se le hayan corregido todos los errores, pues cada vez que el programa se ejecuta, el cliente lo está probando. Con el objetivo de encontrar el mayor número posible de errores, las pruebas deben conducirse sistemáticamente y los casos de prueba deben diseñarse utilizando técnicas definidas.

3.3.1 Niveles de prueba aplicados

Propio de la metodología XP, se lleva a cabo esta Fase de Prueba. Durante el desarrollo de software, XP establece probar constantemente tanto como sea posible, esto permite un aumento de la calidad del sistema desarrollado reduciendo el número de errores no detectados.

XP divide las pruebas del sistema en dos grupos:

Pruebas de caja blanca: son diseñadas por los programadores encargadas de verificar el código. Cada uno de los desarrolladores tiene que ir probando constantemente lo que va obteniendo en el transcurso de la implementación de un

sistema, para lograr que las funcionalidades exigidas por el cliente estén siendo implementadas correctamente.

Estas pruebas que se realizan sobre las funciones internas de un módulo se llevan a cabo con el objetivo de intentar garantizar que:

- Se ejecuten al menos una vez todos los caminos independientes de cada módulo.
- Se utilizan las decisiones en su parte verdadera y en su parte falsa.
- Se ejecuten todos los bucles en sus límites.
- Se utilizan todas las estructuras de datos internas.

Para realizar la prueba es necesario efectuar primeramente el análisis de complejidad del algoritmo sobre el que se va a realizar la prueba, con el propósito de calcular los valores de la complejidad ciclomática. En la tabla 19 se observa un ejemplo de uno de los métodos al cual se le aplicó esta prueba.

Tabla 19: Complejidad ciclomática

```
#construye el hash del equipo para el test de belbin

      def build_test_belbin_hash(t)
       i = 0
 //1   size = t.project_team_evaluations.last(t.members.size).size
      evaluations = {}
     while i < size          //2
       team_hash = {}
       team_hash.store('is', 0)
  //3   evaluations.store(i+1, team_hash)
       eval = t.project_team_evaluations[i]
       izq1 = split_value(eval.is)[0].to_i
      der1 = split_value(eval.is)[1].to_i
      if der1 >= 4 || izq1< 9                  //4
        evaluations[i+1]['is'] = 0          //5
      elsif der1 == 1 && izq1 >= 18         //6
       evaluations[i+1]['is'] = 1          //7
      elsif (der1 == 1 || der1 == 2) && (izq1 >= 15 && izq1 < 18)    //8
       evaluations[i+1]['is'] = 1                        //9
     elsif (der1 == 1 || der1 == 2 || der1 == 3) && (izq1 >= 9 && izq1 < 15)    //10
       evaluations[i+1]['is'] = 0.5                              //11
      end
      i = i + 1                     //12
    end
   evaluations                      //13
end                              //14
```

Luego de este paso, es necesario representar el grafo de flujo asociado (Ilustración 14), en el cual se representan distintos componentes como es el caso de:

Nodo: Son los círculos representados en el grafo de flujo, el cual representa una o más secuencias del procedimiento, donde un nodo corresponde a una secuencia de

procesos o a una sentencia de decisión. Los nodos que no están asociados se utilizan al inicio y final del grafo.

Nodo predicado: Son los nodos que contienen una condición y se caracterizan porque de ellos salen dos o más aristas.

Aristas: Son constituidas por las flechas del grafo, son iguales a las representadas en un diagrama de flujo y constituyen el flujo de control del procedimiento. Las aristas terminan en un nodo, inclusive cuando el nodo no representa la sentencia de un procedimiento.

Regiones: Son las áreas delimitadas por las aristas y nodos donde se incluye el área exterior del grafo, como una región más. Las regiones se enumeran, siendo la cantidad de regiones equivalente a la cantidad de caminos independientes del conjunto básico de un procedimiento.

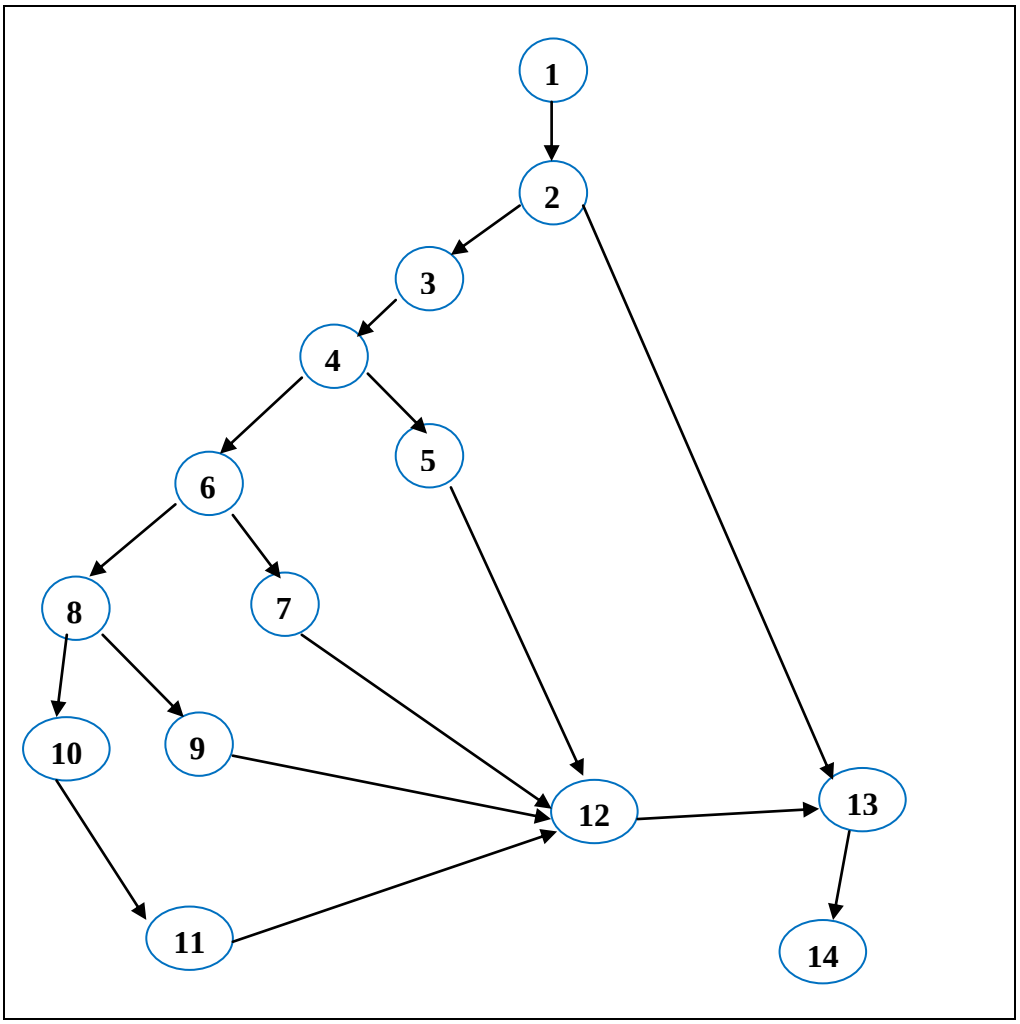


Ilustración 14: Grafo de flujo.

La complejidad ciclomática coincide con el número de regiones del grafo de flujo.

La complejidad ciclomática, V (G), de un grafo de flujo G, se define como

V (G) = Aristas - Nodos + 2

La complejidad ciclomática, V (G), de un grafo de flujo G, también se define como

V (G) = Nodos de predicado + 1

A partir del grafo de flujo de la figura 26, la complejidad ciclomática sería:

Como el grafo tiene cinco regiones, **V (G) = 5**

Como el grafo tiene 17 aristas y 14 nodos, **V (G) = 17 – 14 + 2 = 5**

Como el grafo tiene 4 nodos predicado, **V (G) = 4 + 1 = 5**

Teniendo la complejidad ciclomática se obtiene el número de caminos independientes, que dan un valor límite para el número de pruebas a diseñar.

En el ejemplo, el número de caminos independientes es 5, y estos son:

- 1-2-13-14
- 1-2-3-4-5-12-13-14
- 1-2-3-4-6-7-12-13-14
- 1-2-3-4-6-8-9-12-13-14
- 1-2-3-4-6-8-10-11-12-13-14

Para cada camino se realiza un caso de prueba, a continuación se muestran dos de los casos de prueba que corresponden al grafo.

Caso de prueba para el Camino básico #2:

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se inicializa una variable i en cero, se guardan las evaluaciones del equipo dependiendo la cantidad de miembros que este tenga en size y se crea un vector vacío llamado evaluations.

Condición de ejecución:

La variable i=2, size es 4, eval.is= (10:4)

Si i es menor que 4, creo un vector hash nuevo y almaceno en el la variable ‘is’ con valor cero, luego guardo en el hash creado inicialmente el hash anterior en la posición i+1. Divido el valor del campo is en dos partes der1 e izq1, para posteriormente realizar las comprobaciones.

Entrada: i = “2”, size = “4” y evaluations=” ”.

Resultados esperados: Se espera que el sistema guarde en la posición i+1 del hash evaluations otro hash con valor cero en la clave ‘is’. Devolviendo luego el hash evaluations.

Caso de prueba para el Camino básico #3:

Descripción: Los datos de entrada cumplirán con los siguientes requisitos:

Se inicializa una variable i en cero, se guarda las evaluaciones del equipo dependiendo la cantidad de miembros que este tenga en size y se crea un vector vacío.

Condición de ejecución:

La variable i=2, size es 4, eval.is= (19:1)

Si i es menor que 4, creo un vector hash nuevo y almaceno en el la variable ‘is’ con valor cero, luego guardo en el hash creado inicialmente el hash anterior en la posición i+1. Divido el valor del campo is en dos partes der1 e izq1, para posteriormente realizar las comprobaciones.

Entrada: i = “2”, size = “4” y evaluation=” ”.

Resultados esperados: Se espera que el sistema guarde en la posición i+1 del hash evaluations otro hash con valor uno en la clave ‘is’.Devolviendo luego el hash evaluations.

Pruebas de aceptación:

El uso de cualquier producto de software tiene que estar justificado por las ventajas que ofrece. Sin embargo, antes de empezar a usarlo es muy difícil determinar si sus ventajas realmente justifican su uso. El mejor instrumento para esta determinación es la llamada 'prueba de aceptación'. En esta prueba se evalúa el grado de calidad del software con relación a todos los aspectos relevantes para que el uso del producto se justifique. Las pruebas de aceptación también llamadas pruebas del cliente las especifica el cliente y se enfocan en las características generales y las funcionalidades del sistema. En estas serán probadas las funcionalidades exigidas por el cliente, descritas en las HU que se han implementado. Estas pruebas se llevan a cabo redactando los casos de pruebas.

Para realizar las pruebas de aceptación se utilizan técnicas de caja negra las cuales sin lugar a dudas cumplen un papel fundamental, de las mismas será de interés su forma de interactuar con el plugin, entendiendo qué es lo que hace, pero sin dar importancia a cómo lo hace. En una prueba de caja negra deben estar muy bien definidas sus entradas y salidas, es decir, su interfaz; en cambio, no se precisa definir ni conocer los detalles internos de su funcionamiento.

Se puede decir entonces que son pruebas funcionales que se aplican al sistema empleando un conjunto de datos de entrada y observando las salidas obtenidas para determinar si la función se está desarrollando correctamente por el sistema. Su realización parte de los requisitos funcionales. Algunos ejemplos básicos de pruebas de caja negra son la comprobación de valores límite, pruebas de integridad de la base de datos o pruebas de excepciones. Para las pruebas a la aplicación se introdujeron diferentes valores con el objetivo de verificar el correcto funcionamiento de la misma en diferentes escenarios, al introducir datos válidos e inválidos a partir de los requerimientos funcionales.

Con este tipo de pruebas se intenta detectar:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Problemas de rendimiento.

Para la realización de esta prueba se escogieron 4 clientes fundamentales, dos como autores del modelo y dos por parte del GESPRO que es donde se pondrá en práctica el módulo. Se realizaron 2 iteraciones revisando en cada una la aplicación contra los 9 casos de pruebas realizados. Durante la primera iteración se obtuvo un total de 13 no conformidades (NC) como muestra la ilustración 15 y especificados en el anexo 7, de las cuales 8 eran de complejidad baja y 5 de complejidad media, para luego en una segunda iteración no haberse encontrado ninguna no conformidad.

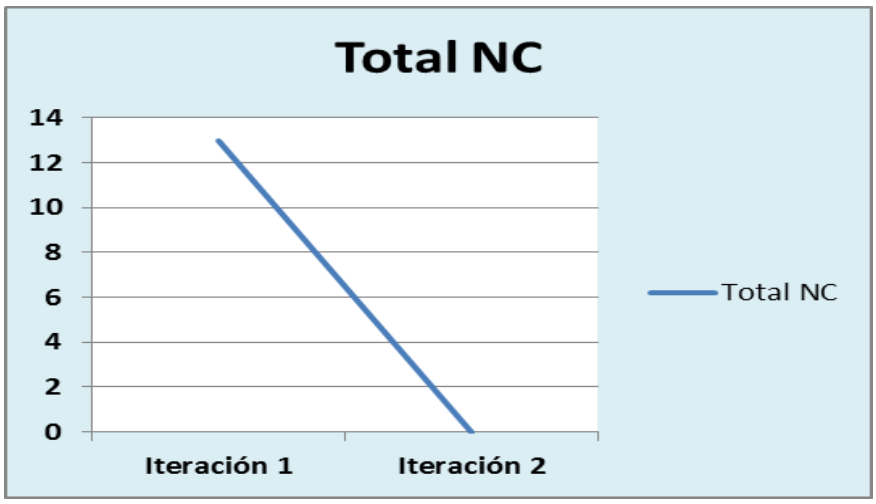


Ilustración 15: Total de no conformidades por iteración.

3.3.2 Casos de prueba

A continuación se muestran los casos de prueba correspondientes a los requisitos funcionales importar fichero de evaluaciones y adicionar integrantes al equipo de proyecto. El resto se encuentra en el anexo 5.

Tabla 20: Caso de prueba Importar.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Importar el fichero de evaluaciones de los miembros de un equipo al GESPRO.	Importar el fichero de evaluaciones de los miembros de un equipo al GESPRO.	EP 1.1: Importar el elemento al GESPRO con datos válidos.	-Se selecciona el icono de importar. -Se selecciona el botón **Examinar** -Se elige el fichero con las evaluaciones correspondientes. -Se presiona el botón **Cargar Archivo.** -Se muestra la

			coincidencia de columnas y se presiona **Importar**. -Se muestra un mensaje que dice que la cantidad de evaluaciones procesadas es la misma que la cantidad de evaluaciones importadas.
EP 1.2: Importar el fichero de evaluaciones de los miembros de un equipo al GESPRO con datos inválidos.	-Se selecciona el icono importar. -Se presiona el botón **Examinar.** -Se elige el fichero con las evaluaciones correspondientes y se presiona el botón **Cargar Archivo.** -Se muestra la coincidencia de columnas y se presiona **Importar**. -Se muestra un cartel de error que dice “X evaluaciones procesadas. Y evaluaciones importadas”. (X , Y se refieren a los números de evaluaciones procesadas e importadas respectivamente)		
EP 1.3: No seleccionar ningún fichero a importar.	Se presiona el botón **examinar.** Y se presiona el botón **Cargar Archivo.** Se muestra un mensaje de error que dice “Debe seleccionar un fichero”		

Tabla 21: Caso de prueba Adicionar Integrantes al equipo.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Añadirle integrantes al equipo.	Añadirle integrantes al equipo de proyecto.	EP 1.1: Añadirle integrantes al equipo de proyecto satisfactoriamente.	-Se selecciona el icono para añadir integrantes. -Se seleccionan los miembros del proyecto que se desean tener en el equipo siempre que sea menor que 7. -Se muestra un cartel que dice creación correcta.

EP 1.2: Añadirle miembros al equipo excediendo los 7 integrantes debidos.	-Se selecciona el icono para añadir integrantes. -Se seleccionan los miembros del proyecto que se desean tener en el equipo excediéndose de 7 integrantes. -Se muestra un cartel que dice el equipo solo debe tener 7 integrantes obteniéndose los 7 primeros de la selección.
EP 1.3: Añadirle integrantes al equipo dejando el campo de integrantes del equipo vacío.	-Se selecciona el icono para añadir integrantes. -No se seleccionan ninguno de los miembros del proyecto. -Se muestra el campo de integrantes del equipo con un error que dice “Por favor, selecciona un objeto de la lista”.

3.4 Encuesta de validación del módulo de evaluación de equipos de proyectos

La presente encuesta forma parte de la propuesta de validación, la misma se realizó a especialistas del centro CEIGE para confirmar la facilidad de uso del módulo respecto a la evaluación de equipos de proyectos teniendo en cuenta elementos de su composición, así como comprobar su aceptación por parte de los usuarios que harán uso del mismo.

Tabla 22: Encuesta de validación.

Encuesta de validación del módulo de evaluación de equipos.
1. Después de utilizar la aplicación evaluando un equipo determinado responda marcando con una x: **a) ¿La información que proporciona la aplicación es valiosa para el análisis de los equipos?** **___ Muy valiosa** **___ Valiosa** **___ Poco valiosa** **___ No valiosa** **b) ¿Se le facilita a usted analizar los equipos para tomar decisiones?** **___ Se facilita entre (75 y 100)%.** **___ Se facilita entre (50 y 75)%.** **___ Se facilita entre (25 y 50)%.** **___ Se facilita entre (0 y 25)%.**

2. Teniendo en cuenta que los resultados de la evaluación le es útil para el análisis de los equipos.

a) ¿Cree usted que el sistema disminuye los esfuerzos requeridos para el análisis de los equipos?

___ Disminuye entre (75 y 100)%.

___ Disminuye entre (50 y 75)%.

___ Disminuye entre (25 y 50)%.

___ Disminuye entre (0 y 25)%.

b) La utilización del módulo :
¿Facilita el trabajo de los especialistas en cuanto a la evaluación de equipos de proyecto?

___ Facilita entre el (75 y 100)%.

___Facilita entre el (50 y 75)%.

___Facilita entre el (25 y 50)%.

___Facilita entre el (0 y 25)%.

Como primer paso para llevar a cabo la encuesta se realizó la selección de los expertos y luego se procede con el lanzamiento de la misma, mediante la cual cada participante emite su criterio respecto a los indicadores evaluados. Finalmente se analizan los resultados obtenidos y se confecciona el balance de porciento.

3.4.1 Selección de los especialistas a realizar la encuesta

Para realizar la encuesta se escogieron a 10 profesionales que cumplieran con dos criterios fundamentales, ser jefe de proyecto o especialista del centro con 3 o más años de experiencia y estar involucrado en actividades relacionadas con la gestión de equipos de proyectos.

3.4.2 Lanzamiento de la encuesta y análisis de los resultados

Cuando cada profesional escogido acepta realizar la encuesta, primeramente se le realiza una explicación de la elaboración de una evaluación de equipo a mano, luego se le muestra el proceso de evaluar los equipos mediante la aplicación realizada y posteriormente realiza la encuesta.

Los criterios de los expertos estuvieron dados mediante la escala nominal donde 4(Excelente), 3(Bien), 2 (Regular) y 1 para (Mal).

En la siguiente tabla se exponen los indicadores evaluados y la respuesta a ellos dada por los profesionales seleccionados donde:

E1…E10: representan los encuestados.

TD: Toma de decisión.

TE: Trabajo de los especialistas.

Tabla 23: Respuesta a los indicadores evaluados.

Indicadores	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	Suma	%
Validez en el entorno	4	4	4	4	3	4	4	4	3	4	38	95%
Aporte a la TD	3	3	4	4	4	3	4	4	4	4	37	93%
Disminución del esfuerzo (horas/hombre)	4	4	4	4	4	4	4	4	4	4	40	100%
Facilidad del TE	4	4	4	4	3	4	4	4	4	4	39	98%

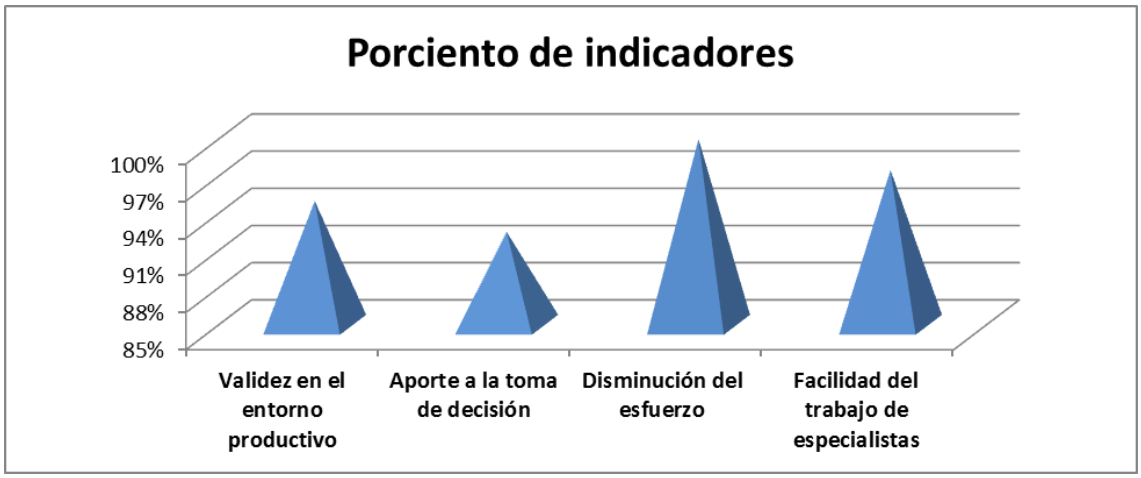
En dependencia de la respuesta que proporcionó el encuestado a cada pregunta, así será el rango de valores al que será asignado. A continuación en la tabla 24 se muestra como quedó estructurado el rango.

Tabla 24: Rango de valores.

Valor 4	Muy valiosa	(75 y 100)%
Valor 3	Valiosa	(50 y 75)%
Valor 2	Poco valiosa	(25 y 50)%
Valor 1	No valiosa	(0 y 25)%

Mediante la encuesta los profesionales emitieron sus criterios sobre la validez del sistema en el entorno productivo de acuerdo a la evaluación de equipos, el aporte que brinda el sistema para la toma de decisiones en la actividad de evaluación de equipos, la disminución del esfuerzo requerido para realizar la evaluación y la facilidad del trabajo de los especialistas con respecto a esta actividad. Estos criterios se muestran en la tabla 24, donde cada indicador es representado en porciento y la máxima puntuación que se podía obtener es de 40 puntos.

Tabla 25: Resultados de la encuesta en porciento.



Todos los indicadores recibieron puntuaciones por encima del 90%, indicando de manera general que la calidad de la aplicación es aceptada por los especialistas. La disminución del esfuerzo es un indicador que obtuvo el 100% corroborándose mediante la realización de un caso de estudio con seis equipos de tamaño siete que el esfuerzo realizado oscila entre 0.1 y 0.9 segundos por hombre, especificando que todos los profesionales consideran que el módulo para evaluar equipos de proyectos teniendo en cuenta elementos genéricos de composición disminuye el esfuerzo y el tiempo de los especialistas para llevar a cabo la evaluación de equipos. Los valores del resto de los indicadores evaluados superan el 90%, certificando así que su valor es también muy aceptable.

Los resultados finales de la encuesta demuestran una aceptación de la propuesta por parte de los expertos a los atributos sometidos a consideración.

3.5 Conclusiones

Durante el desarrollo del capítulo se validó la aplicación realizada mediante las pruebas de caja blanca y caja negra, así como por encuesta a los especialistas del centro donde se definió como muy adecuado el módulo.

Mediante las pruebas de funcionamiento realizadas se evaluaron los diferentes elementos de la aplicación y se comprobó que las funcionalidades implementadas mostraron un correcto funcionamiento y una respuesta a los requisitos funcionales garantizando las necesidades de los clientes.

También se realizaron los casos de pruebas para validar a través de la entrada de una serie de variables sus salidas y comprobar la respuesta de cada requisito implementado.

Conclusiones

Con la realización del siguiente trabajo se logró resolver los problemas de evaluación manual de equipos de proyectos informáticos basados en su composición, que existían en los proyectos pertenecientes al GESPRO. Se alcanzó la creación de un módulo que satisface las necesidades del cliente y cumple con las normas de diseño e implementación, logrando al final obtener una herramienta que evalúe equipos de proyectos informáticos basados en su composición.

Para lograr este resultado final se cumplieron los objetivos específicos trazados:

- Se definió el marco teórico de la investigación el cual ayudó a la compresión y desarrollo del sistema así como la elección de las herramientas usadas en el diseño e implementación.
- Se diseñó e implementó un módulo integrado al GESPRO para evaluar equipos de proyectos informáticos basados en su composición, logrando así una herramienta que permite dar criterio sobre los equipos con los que se cuenta en los proyectos, así como establecer de acuerdo a las evaluaciones estrategias de mejoras en los proyectos.
- Se validaron los resultados obtenidos mediante pruebas de caja blanca, pruebas de caja negra y encuestas a especialistas permitiendo mejorar la calidad del producto final.

Finalmente la investigación realizada y el trabajo llevado a cabo le proporcionan al GESPRO una nueva funcionalidad de la cual carecía dándole valor agregado a esta herramienta.

Reconocimientos

- Se recomienda incluirle una funcionalidad al sistema que permita dado un número X de individuos saber cuál es el mejor equipo según su composición con una combinación Y de individuos a formar un equipo tomando como base este modelo.

Dónde: X represente un número cualquiera.

Y sea un subconjunto de X menor que 7.

- Incluir además nuevas características al perfil ideal basadas en competencias específicas y roles funcionales dentro de un proyecto.

Referencias Bibliográficas

1. Pressman, R., *Ingeniería del Software: Un enfoque práctico* Sexta edición ed. Revista de métodos cuantitativos para la economía y la empresa. 2006, México: Editorial Mc Graw Hill.
2. Prieto, M. (2010) *Método Cuantitativo para Integración y Comparación de Grupos de Trabajo de Instalación y Desarrollo de Software* 1-29.
3. André, M., *Un modelo para la asignación de recursos humanos a equipos de proyectos de software*, in *Facultad de Ingeniería Informática*. 2009, Instituto Superior Politécnico "José Antonio Echeverría": La Habana. p. 169.
4. Stiven, E.R., *Modelo para la Evaluación de la Composición Equipos de Proyectos Informáticos*, in *Facultad 3*. 2012, Universidad de las Ciencias Informáticas: La habana. p. 80.
5. PMI, *Guía de los Fundamentos para la Dirección de Proyectos*. 4ta. Edición. ed. 2009, Pennsylvania: Project Management Institute.
6. Katzenbach, J.R., *El trabajo en equipo*. 2000, Barcelona: Granica.
7. Humphrey, W.S., *Introduction to the Team Software Process*^{SM}. SEI Series in Software Engineering. 2000, Boston: Addison-Wesley.
8. Gil, F., R. Rico, and M. Sánchez, *EFICACIA DE EQUIPOS DE TRABAJO.* Papeles del Psicólogo, 2008. 29(1): p. 25-31.
9. Santos, A.C., *TECNOLOGÍA DE GESTIÓN DE RECURSOS HUMANOS*. Tercera edición corregida y ampliada ed. 2010, La Habana: Ed. "Félix Varela" y Academia.
10. Picacho, A.M. and A. Amengual, *Can Teamwork Management Help in Software Quality and Process Improvement?* Improving Quality in Business Processes, Products and Organizational Systems, 2009. X(9): p. 20-38.
11. *http://www.sei.cmu.edu/tsp/*. 2011.
12. Peña., M.A.A.y.M.G.B.d.l., *Un sistema de soporte a la decisión para la asignación de recursos humanos a equipos de proyectos de software.* Instituto Superior Politécnico "José Antonio Echeverría" (CUJAE), Ciudad de La Habana, Cuba, 2010.
13. Mas-Picacho., E.A.a.A. (december 2009) *Can Teamwork Management Help in Software Quality and Process Improvement.*
14. SourceForge.net., F.B.E.s.d.a. *OpenProj - Project Management* AplicacionesEmpresariales.com [Internet]. 2010 [cited 2012 29].
15. Centro de excelencia de software libre de Castilla La Mancha (Ceslcam, h.c.c., *Análisis de aplicación: Redmine*. 2010. 13.
16. Empresariales(CDAE)., L.d.S.d.G.d.P.G.C.d.C.y.D.d.A. *Resumen GESPRO 11.05 vista desde la toma de decisiones.* 2011.
17. González, P.R., *Estudio de la aplicación de metodologías ágiles para la evolución de productos software.* ¨2010.
18. PublicacionesMexico, *RUP vs.XP.* 2008.
19. Beck, K., *Extreme Programming Explained Embrace Change, Pearson Education.* 2007.
20. Perissé, M.C., *Una Metodología Simplificada.* 2006.
21. Anónimo. *Lenguaje de Programación Ruby. In: [online].* . [cited 2012 26 de noviembre].
22. Anónimo. *Ruby on Rails [online].* [cited 2012 26 de noviembre].
23. Programación., D.d.S.I.y.C.E.-F.U.P.d.V.M.y.T.d.l., *Introducción a Herramientas CASE y System Architect.* 2009: p. 13.
24. *Visual Paradigm*. [En línea] [Citado el: 12 de Mayo de 2011.] [cited 2012.
25. *UML, BPMN and Database Tool for Software Development.Visual Paradigm* [Internet]. [cited 2012 Nov 15]. [Citado el: 12 de Mayo de 2011.]].

26. Anónimo. *NetBeans IDE 6.9.1 Release Information [online].* . [cited 2012 20-11].
27. Anónimo. *Netbeans 6.9 final liberado [online].* [cited 2012 26-11].
28. Informáticas, U.d.l.C. *Comunidad Tecnica Cubana de PostgreSQL*. 2012.
29. Ridosbey Milián Iglesias, G.F.G.y.Y.R.R., *Herramientas para la gestión de proyectos de software.*
30. Xp., A. *https://Artefactos%20de%20XP/programacion-extrema2.shtml*. 2011.
31. 1471-2000., A.d.s.E.l.-I. *http://www.aprendeenlinea.udea.edu.co/lms/moodle*. 2011.
32. Pantoja., E.B., *El patrón de diseño Modelo-Vista-Controlador (MVC).* 2010.
33. Mestras, J.P., *Estructura de las Aplicaciones Orientadas a Objetos,El patrón Modelo-Vista-Controlador (MVC).* 2009: Dep. Ingeniería del Software e Inteligencia Artificial Universidad Computanción Madrid.
34. Hall., C.L.P., *"Applying UML and Patterns. An introduction to Object-Oriented Analysis and Design and Iterative Development. 3rd Edition.".* 2005.

Bibliografía

Pressman, R., Ingeniería del Software: Un enfoque práctico Sexta edición ed. Revista de métodos cuantitativos para la economía y la empresa. 2006, México: Editorial Mc Graw Hill.
André, M., Un modelo para la asignación de recursos humanos a equipos de proyectos de software, in Facultad de Ingeniería Informática. 2009, Instituto Superior Politécnico "José Antonio Echeverría": La Habana. p. 169.

Prieto, M. (2010) Método Cuantitativo para Integración y Comparación de Grupos de Trabajo de Instalación y Desarrollo de Software 1-29.

Stiven, E.R., Modelo para la Evaluación de la Composición Equipos de Proyectos Informáticos, in Facultad 3. 2012, Universidad de las Ciencias Informáticas: La habana. p. 80.

Gil, F., R. Rico, and M. Sánchez, Eficacia de equipos de trabajo. Papeles del Psicólogo, 2008. 29(1): p. 25-31.

Un sistema de soporte a la decisión para la asignación de recursos humanos a equipos de proyectos de software. Instituto Superior Politécnico "José Antonio Echeverría" (CUJAE), Ciudad de La Habana, Cuba, 2010.

González, P.R., Estudio de la aplicación de metodologías ágiles para la evolución de productos software. ¨2010.

Beck, K., Extreme Programming Explained Embrace Change, Pearson Education. 2007.

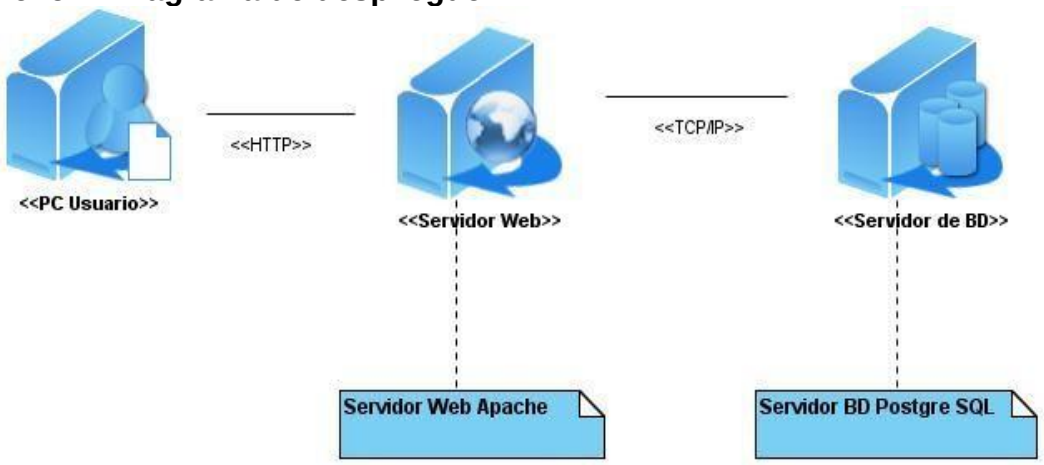
González, P.R., Estudio de la aplicación de metodologías ágiles para la evolución de productos software. ¨2010.

Mestras, J.P., Estructura de las Aplicaciones Orientadas a Objetos,El patrón Modelo Vista-Controlador (MVC). 2009: Dep. Ingeniería del Software e Inteligencia Artificial Universidad Computanción Madrid.

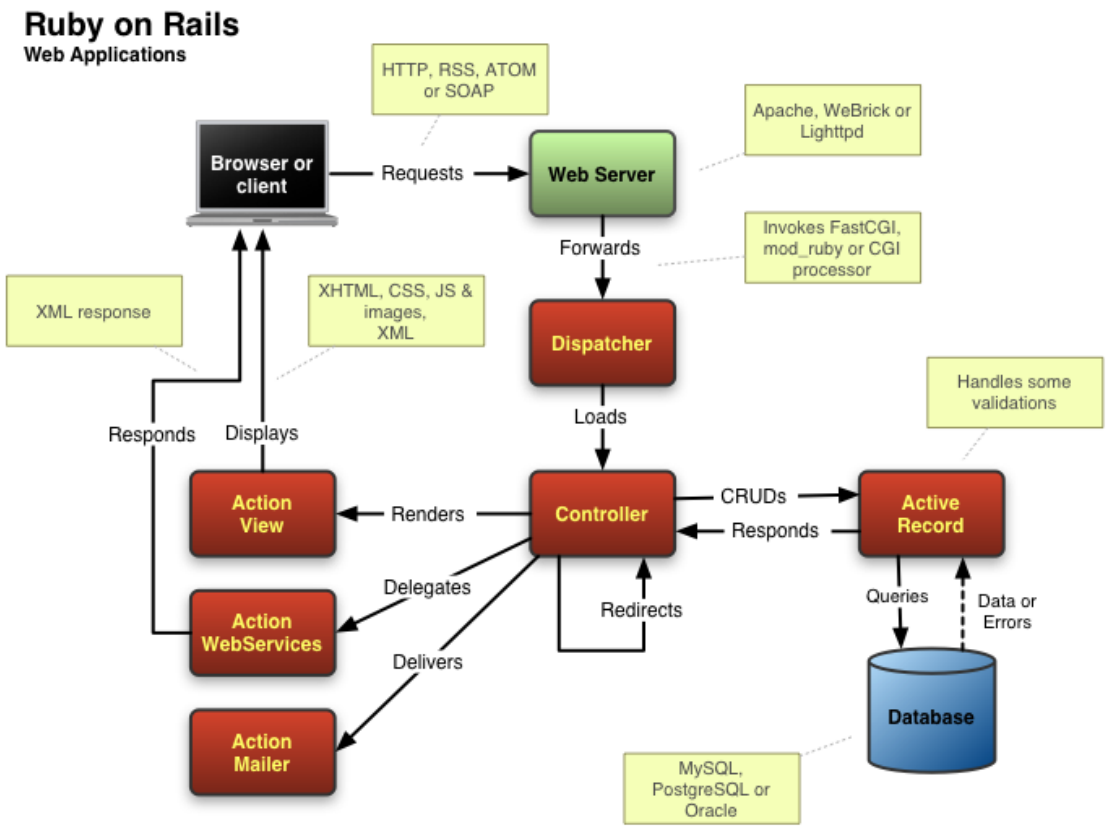
Anexos

Anexo 1: Patrón de arquitectura MVC

Anexo 2: Diagrama de despliegue



Anexo 3: Estructura del framework RoR.



Anexo 4: Historias de usuario

Tabla 26: Historia de usuario. Resultado de la evaluación.

Historia de Usuario	
Código: HU8	**Nombre Historia de Usuario**: Mostrar interfaz con los resultados de la evaluación del equipo de proyecto.
Modificación de Historia de Usuario: Primera	
Referencia: RF3.5	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Media	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Baja	**Puntos Reales:** 1semana
Descripción: La funcionalidad permite mostrar los resultados de la evaluación realizada al equipo de proyecto a través de los índices de correspondencia de los métodos matemáticos distancia de Hamming y Coeficiente de Adecuación.	

Observaciones:

Tabla 27: Historia de usuario. Matriz de competencias genéricas.

Historia de Usuario	
Código: HU4	**Nombre Historia de Usuario:** Mostrar matriz de competencias genéricas de los miembros del equipo.
Modificación de Historia de Usuario: Primera	
Referencia: RF4.2	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Alta	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 1 semana
Descripción: La funcionalidad permite mostrar la matriz que corresponde a las evaluaciones de las competencias genéricas por cada uno de los miembros del equipo.	
Observaciones:	

Tabla 28: Historia de usuario. Listado de las evaluaciones.

Historia de Usuario	
Código: HU9	**Nombre Historia de Usuario**: Listar historial de las evaluaciones de todos los equipos.
Modificación de Historia de Usuario: Primera	
Referencia: RF4.3	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Media	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Baja	**Puntos Reales:** 1 semana
Descripción: La funcionalidad permite listar las evaluaciones de los equipos con sus resultados respectivos.	
Observaciones:	

Tabla 29: Historia de usuario. Filtrar evaluación.

Historia de Usuario

Código: HU10	**Nombre Historia de Usuario**: Filtrar por identificador del equipo los resultados de sus evaluaciones anteriores.
Modificación de Historia de Usuario: Primera	
Referencia: RF4.5	
Programador: Leiza Causilla Rojas	**Iteración Asignada:** Primera
Prioridad : Alta	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 1 semana
Descripción: La funcionalidad permite filtrar mediante el identificador de un equipo los resultados de sus evaluaciones anteriores.	
Observaciones:	

Anexo 5: Casos de pruebas

Tabla 30: Caso de prueba. Evaluar equipo de proyecto.

Nombre del requisito	**Descripción general**	**Escenarios de pruebas**	**Flujo del escenario**
1: Evaluar un equipo del proyecto.	Evaluar un equipo del proyecto de acuerdo a elementos de su composición.	EP 1.1: Evaluar un equipo del proyecto de acuerdo a elementos de su composición satisfactoriamente.	Se selecciona el icono del equipo que dice **Evaluar equipo**. Se muestra los resultados de las evaluaciones correspondientes al equipo en una tabla donde aparecen las evaluaciones de los indicadores de cada método matemático utilizado.
EP 1.2: Evaluar un equipo del proyecto de acuerdo a elementos de su composición sin haber evaluado a todos sus integrantes.		Se selecciona el icono del equipo correspondiente a **Evaluar equipo**. Si las evaluaciones individuales no coinciden con la cantidad de integrantes en el equipo. Se muestra un mensaje de error que dice “Deben estar todos los miembros en las evaluaciones.”	

Tabla 31: Caso de prueba: Adicionar equipo al proyecto.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Adicionar un equipo al proyecto.	Adicionar un equipo al proyecto.	EP 1.1: Adicionar un equipo al proyecto satisfactoriamente.	Se selecciona el icono **Nuevo Equipo**. Se muestra una pantalla para escribir el nombre y la descripción del nuevo equipo. Se llenan todos los campos. Se presiona el botón crear. Se muestran un mensaje que dice “Creación correcta”.
EP 1.2: Adicionar un equipo al proyecto dejando campos requeridos vacíos.		Se selecciona el icono **Nuevo Equipo**. Se muestra una pantalla para escribir el nombre y la descripción del nuevo equipo. Se deja el campo nombre vacío y se presiona el botón **Crear**. Se muestra un mensaje que dice “Nombre es requerido”.	
EP 1.3: Cancelar la operación Adicionar un equipo al proyecto.		Se selecciona el icono nuevo equipo. Se muestra una pantalla para escribir el nombre y la descripción del equipo. Se selecciona Equipos del proyecto cancelándose así la operación.	
EP 1.4: Cancelar la operación Adicionar un equipo al proyecto habiendo escrito algo en la descripción.		Se selecciona el icono nuevo equipo. Se muestra una pantalla con nombre y la descripción del equipo. Se escribe la descripción del equipo. Se selecciona Equipos del proyecto. Se muestra un mensaje “Esta página te pide que confirmes que deseas salir - los datos que has introducido se perderán.” Y se presiona el botón Abandonar Página. Se cancela la operación.	

Tabla 32: Caso de prueba: Mostrar evaluaciones de todos los equipos.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario

1: Mostrar las evaluaciones de todos los equipos.	Mostrar las evaluaciones de todos los equipos sobre los indicadores de los métodos utilizados.	EP 1.1: Mostrar las evaluaciones de todos los equipos a través de la página Historial.	✓ Se selecciona la página Historial. ✓ Se muestran las evaluaciones de todos los equipos.
EP 1.2: Mostrar evaluaciones de los integrantes de un equipo a través de la página Evaluaciones de equipos.		✓ Se presiona la página correspondiente a Evaluaciones de equipos. ✓ Se muestran las evaluaciones de todos los equipos.	

Tabla 33: Caso de prueba: Mostrar evaluación de un equipo.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Mostrar la evaluación de un equipo de proyecto.	Mostrar la evaluación de un equipo de proyecto.	EP 1.1: Mostrar la evaluación de un equipo de proyecto utilizando la página Historial.	✓ Se selecciona la página Historial**.** ✓ Se muestran todas las evaluaciones correspondientes a los equipos de proyectos. ✓ Se filtra por el identificador del equipo su evaluación.
EP 1.2: Mostrar todas las evaluaciones de los equipos de proyecto utilizando la página Evaluaciones de equipos.		✓ Se selecciona la página Evaluaciones de equipos. ✓ Se muestran las evaluaciones correspondientes a los equipos de proyectos. ✓ Se busca la evaluación del equipo que se desea saber.	

Tabla 34: Caso de prueba: Subida de evaluaciones no usando fichero csv.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Subida de la evaluación individual de un miembro del equipo.	Subida de la evaluación individual de un miembro del equipo.	EP 1.1: Subida de la evaluación individual de un miembro del equipo satisfactoriamente.	✓ Se selecciona el icono **Nueva evaluación.** ✓ Se llenan todos los campos y se presiona el botón **Crear**. ✓ Se muestra un

			mensaje que dice “Creación correcta”.
EP 1.2: Subida de la evaluación individual de un miembro del equipo dejando campos en blanco.		✓ Se selecciona el icono **Nueva evaluación.** ✓ Se llenan algunos campos y se dejan otros en blanco ✓ Se presiona el botón **Crear**. ✓ Se muestra un mensaje que dice “Campo N es un campo requerido”, donde N representa el nombre del campo requerido.	
EP 1.2: Subida de la evaluación individual de un miembro del equipo llenando los campos de los roles de Belbin incorrectamente.		✓ Se selecciona el icono **Nueva evaluación.** ✓ Se llenan los campos correspondientes a los roles de Belbin incorrectamente. ✓ Se presiona el botón **Crear**. ✓ Se muestra un mensaje que dice “ N es de la forma #:# ”, donde N representa el nombre del campo requerido.	

Tabla 35: Caso de prueba: Eliminar equipo de proyecto.

Nombre del requisito	**Descripción general**	**Escenarios de pruebas**	**Flujo del escenario**
1: Eliminar equipo de proyecto.	Eliminar un determinado equipo de proyecto.	EP 1.1: Eliminar un equipo de proyecto satisfactoriamente.	✓ Se selecciona el enlace eliminar que se encuentra en la parte derecha correspondiente al equipo que se desea eliminar. ✓ Se muestra un cartel que dice: N será borrado .Está seguro?(Donde N es el nombre del equipo a eliminar) ✓ Se presiona el botón OK. ✓ Se elimina el equipo.

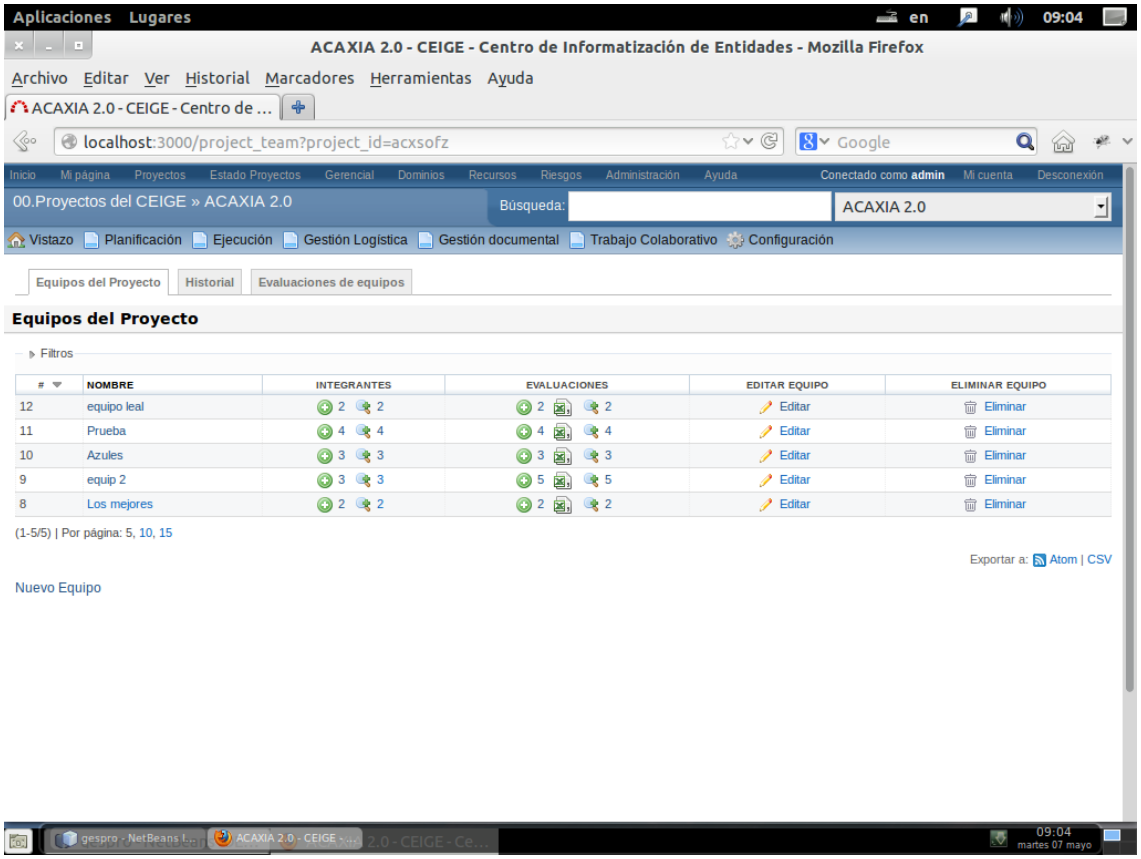
EP 1.2: Cancelar la operación de eliminar equipo.	✓ Se selecciona el enlace eliminar que se encuentra en la parte derecha correspondiente al equipo que se desea eliminar. ✓ Se muestra un cartel que dice: N será borrado .Está seguro?(Donde N es el nombre del equipo a eliminar) ✓ Se presiona el botón Cancel. ✓ Se cancela la operación.

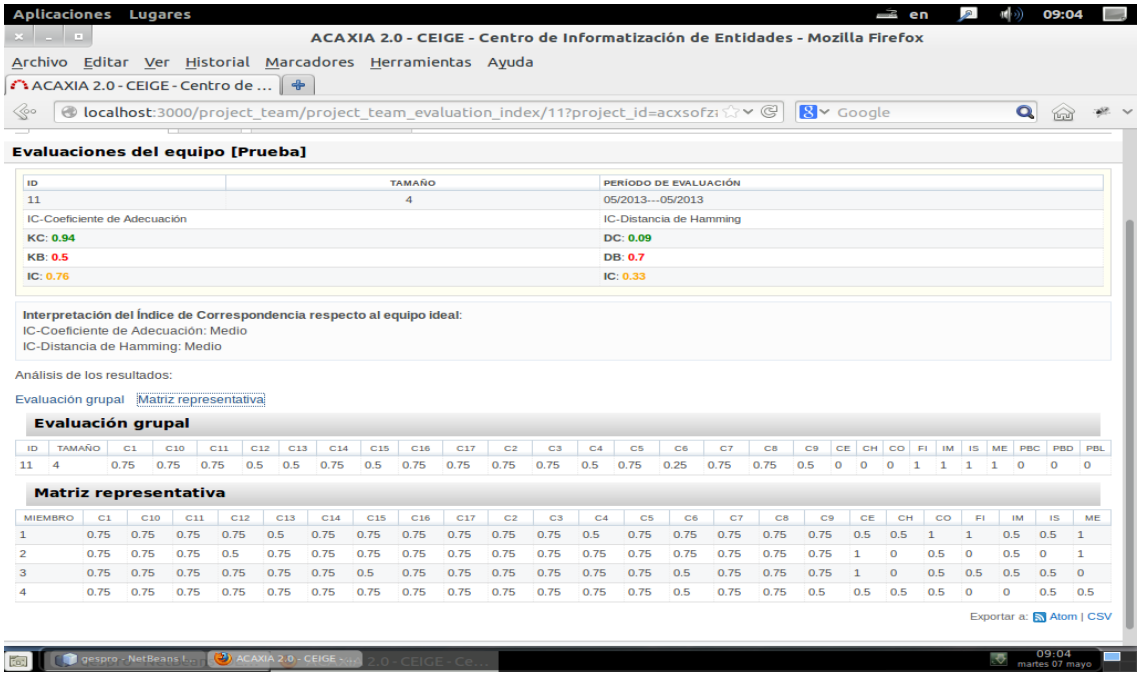
Tabla 36: Caso de prueba: Eliminar integrante del equipo de proyecto.

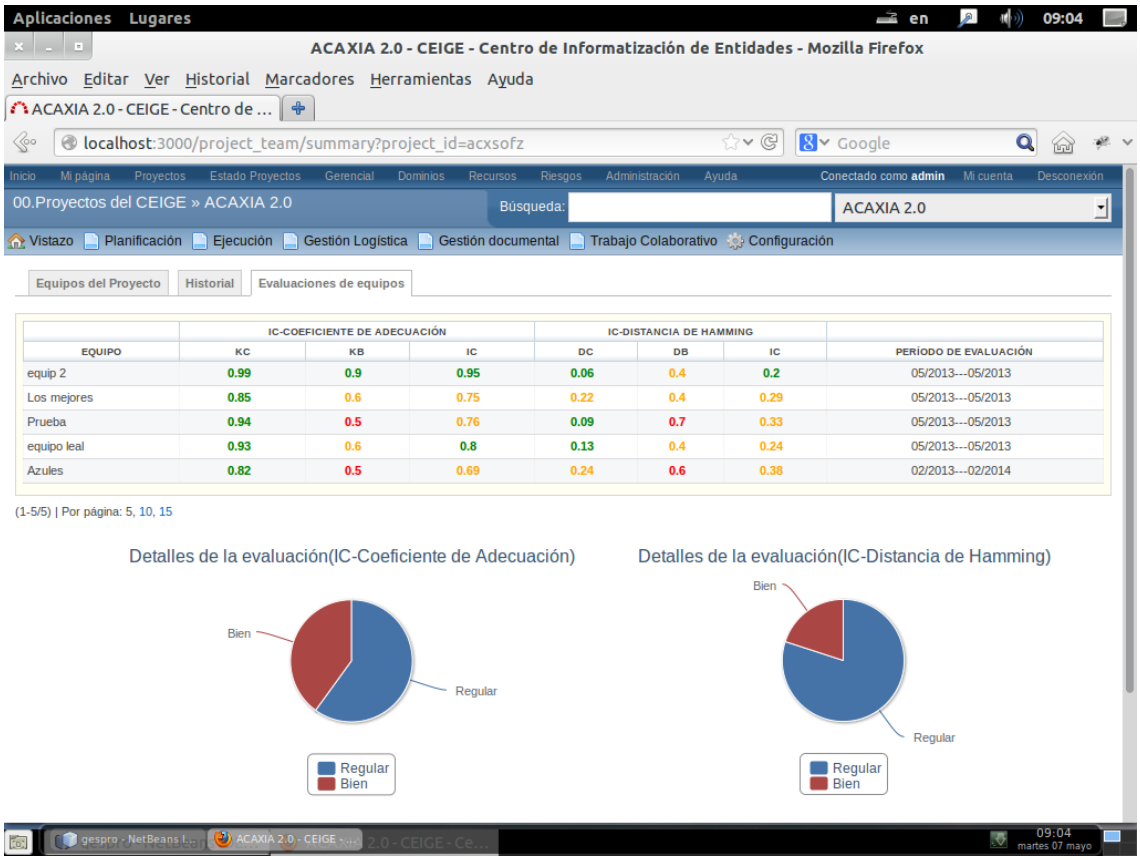
Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Eliminar integrantes del equipo de proyecto.	Eliminar un determinado integrante del equipo de proyecto.	EP 1.1: Eliminar un integrante del equipo de proyecto satisfactoriamente.	✓ Se selecciona el icono que dice mostrar miembro perteneciente a la columna integrantes. ✓ Se presiona el enlace eliminar que está a la derecha del miembro a eliminar del equipo. ✓ Se muestra un cartel que dice: N será borrado.Está seguro?(Donde N es el nombre del integrante a eliminar) ✓ Se presiona el botón OK. ✓ Se elimina el miembro de ese equipo.
EP 1.2: Cancelar la operación de eliminar integrante del equipo.			✓ Se selecciona el icono que dice mostrar miembro perteneciente a la columna integrantes. ✓ Se presiona el enlace eliminar que está a la derecha del miembro a eliminar del equipo. ✓ Se muestra un cartel que dice: N será borrado .Está seguro?(Donde N es el nombre del integrante a eliminar)

	✓ Se presiona el botón Cancel. ✓ Se cancela la operación de eliminar el miembro de ese equipo.

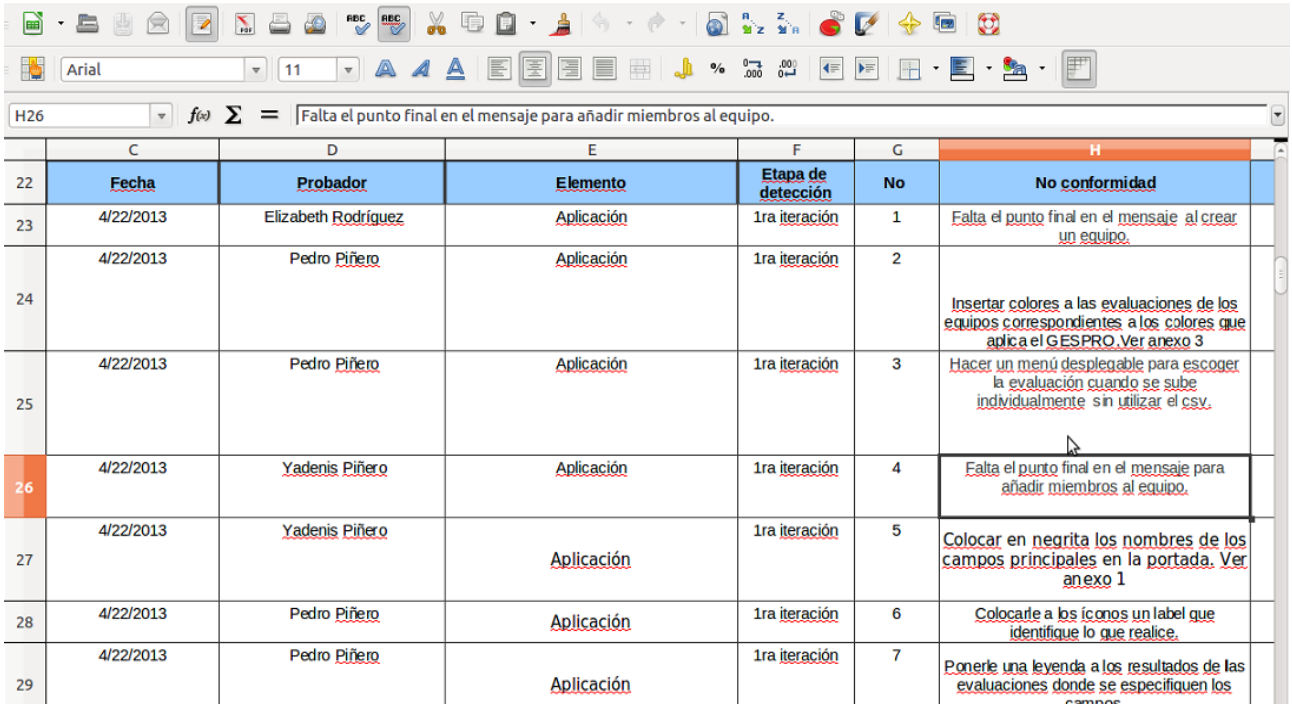
Anexo 6: Imágenes de la aplicación




Aplicaciones Lugares
en
09:04
ACAXIA 2.0 - CEIGE - Centro de Informatización de Entidades - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
ACAXIA 2.0 - CEIGE - Centro de ...
localhost:3000/project_team/project_team_evaluation_index/11?project_id=acxsofz
Google
Evaluaciones del equipo [Prueba]
ID
TAMAÑO
PERÍODO DE EVALUACIÓN
11
4
05/2013---05/2013
IC-Coeficiente de Adecuación
IC-Distancia de Hamming
KC: 0.94
DC: 0.09
KB: 0.5
DB: 0.7
IC: 0.76
IC: 0.33
Interpretación del Índice de Correspondencia respecto al equipo ideal:
IC-Coeficiente de Adecuación: Medio
IC-Distancia de Hamming: Medio
Análisis de los resultados:
Evaluación grupal
Matriz representativa
Evaluación grupal
Matriz representativa
Exportar a: Atom | CSV

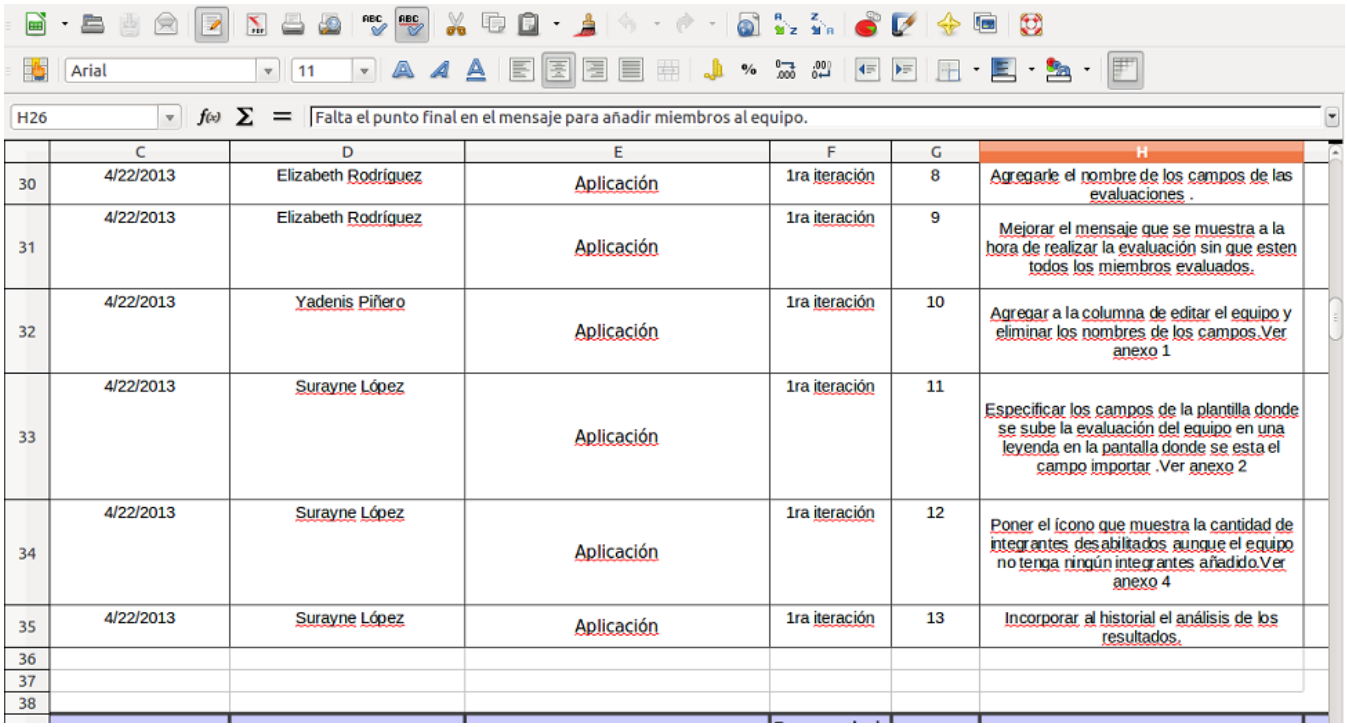

Aplicaciones Lugares
en
09:04
ACAXIA 2.0 - CEIGE - Centro de Informatización de Entidades - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
ACAXIA 2.0 - CEIGE - Centro de ...
localhost:3000/project_team/summary?project_id=acxsofz
Google
Inicio Mi página Proyectos Estado Proyectos Gerencial Dominios Recursos Riesgos Administración Ayuda
Conectado como admin Mi cuenta Desconexión
00.Proyectos del CEIGE » ACAXIA 2.0
Búsqueda:
ACAXIA 2.0
Vistazo Planificación Ejecución Gestión Logística Gestión documental Trabajo Colaborativo Configuración
Equipos del Proyecto
Historial
Evaluaciones de equipos
IC-COEFICIENTE DE ADECUACIÓN
IC-DISTANCIA DE HAMMING
EQUIPO
KC
KB
IC
DC
DB
IC
PERÍODO DE EVALUACIÓN
equip 2
0.99
0.9
0.95
0.06
0.4
0.2
05/2013---05/2013
Los mejores
0.85
0.6
0.75
0.22
0.4
0.29
05/2013---05/2013
Prueba
0.94
0.5
0.76
0.09
0.7
0.33
05/2013---05/2013
equipo leal
0.93
0.6
0.8
0.13
0.4
0.24
05/2013---05/2013
Azules
0.82
0.5
0.69
0.24
0.6
0.38
02/2013---02/2014
(1-5/5) | Por página: 5, 10, 15
Detalles de la evaluación(IC-Coeficiente de Adecuación)
Bien
Regular
Regular
Bien
Detalles de la evaluación(IC-Distancia de Hamming)
Bien
Regular
Regular
Bien

Anexo 7:



H26 | Falta el punto final en el mensaje para añadir miembros al equipo.

	C	D	E	F	G	H
22	**Fecha**	**Probador**	**Elemento**	**Etapa de detección**	**No**	**No conformidad**
23	4/22/2013	Elizabeth Rodríguez	Aplicación	1ra iteración	1	Falta el punto final en el mensaje al crear un equipo.
24	4/22/2013	Pedro Piñero	Aplicación	1ra iteración	2	Insertar colores a las evaluaciones de los equipos correspondientes a los colores que aplica el GESPRO.Ver anexo 3
25	4/22/2013	Pedro Piñero	Aplicación	1ra iteración	3	Hacer un menú desplegable para escoger la evaluación cuando se sube individualmente sin utilizar el csv.
26	4/22/2013	Yadenis Piñero	Aplicación	1ra iteración	4	Falta el punto final en el mensaje para añadir miembros al equipo.
27	4/22/2013	Yadenis Piñero	Aplicación	1ra iteración	5	Colocar en negrita los nombres de los campos principales en la portada. Ver anexo 1
28	4/22/2013	Pedro Piñero	Aplicación	1ra iteración	6	Colocarle a los íconos un label que identifique lo que realice.
29	4/22/2013	Pedro Piñero	Aplicación	1ra iteración	7	Ponerle una leyenda a los resultados de las evaluaciones donde se especifiquen los



H26 | Falta el punto final en el mensaje para añadir miembros al equipo.

	C	D	E	F	G	H
30	4/22/2013	Elizabeth Rodríguez	Aplicación	1ra iteración	8	Agregarle el nombre de los campos de las evaluaciones .
31	4/22/2013	Elizabeth Rodríguez	Aplicación	1ra iteración	9	Mejorar el mensaje que se muestra a la hora de realizar la evaluación sin que esten todos los miembros evaluados.
32	4/22/2013	Yadenis Piñero	Aplicación	1ra iteración	10	Agregar a la columna de editar el equipo y eliminar los nombres de los campos.Ver anexo 1
33	4/22/2013	Surayne López	Aplicación	1ra iteración	11	Especificar los campos de la plantilla donde se sube la evaluación del equipo en una leyenda en la pantalla donde se esta el campo importar .Ver anexo 2
34	4/22/2013	Surayne López	Aplicación	1ra iteración	12	Poner el ícono que muestra la cantidad de integrantes desabilitados aunque el equipo no tenga ningún integrantes añadido.Ver anexo 4
35	4/22/2013	Surayne López	Aplicación	1ra iteración	13	Incorporar al historial el análisis de los resultados.
36						
37						
38						

Glosario de términos

Aplicación Web: Aplicación informática que los usuarios utilizan accediendo a un servidor Web a través de un navegador o browser. Estas son muy populares debido a la habilidad para actualizar y mantener la información manipulada sin distribuir e instalar el software en miles de potenciales clientes.

Proyecto: planificación que consiste en un conjunto de actividades que se encuentran interrelacionadas y coordinadas, surge como respuesta a una necesidad, finaliza cuando se obtiene el resultado deseado, desaparece la necesidad inicial, o se agotan los recursos disponibles.

Gestión de proyectos: La Gestión de Proyectos tiene como finalidad principal la planificación, el seguimiento y control de las actividades y de los recursos humanos y materiales que intervienen en el desarrollo de un Sistema de Información o en la vida de un proyecto.

Equipo de proyecto: es un conjunto pequeño de personas, enfocado en tareas específicas, con habilidades, actitudes y conocimientos complementarios, comprometidos con un propósito y objetivo común; consciente que la única forma de completar la misión es trabajar en conjunto.

Plugin: complemento de una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica.

Requerimiento: es una característica que el sistema debe tener o es una restricción que el sistema debe satisfacer para ser aceptada por el cliente.

Rol: Define el comportamiento y responsabilidades de un individuo.

Ruby: Lenguaje de programación interpretado, reflexivo y orientado a objetos.

PostgreSQL: SGBD relacional, que posee una arquitectura cliente-servidor.

Open Source: Código abierto, es el término con el que se conoce al software distribuido y desarrollado libremente.

TCP/IP: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP). El TCP/IP es la base de Internet, y sirve para enlazar computadoras que utilizan diferentes sistemas operativos.

Roles de Belbin: Según el Dr. Meredith Belbin el rol de un equipo es la particular forma de comportarse, contribuir y relacionarse socialmente. Clasificándolos en tres categorías: roles de acción, roles mentales y roles sociales.