UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA

FACULTAD DE INGENERIA DE SISTEMAS

MAESTRIA EN SOFTWARE LIBRE

"PROPUESTA E IMPLEMENTACIÓN DE UNA GUÍA PARA LA EVALUACIÓN DE CALIDAD DE APLICACIONES WEB ADAPTADA AL MODELO DE SOFTWARE LIBRE A TRAVÉS DE UN CASO DE ESTUDIO"

TESISTA: Camilo Ernesto Rodríguez Moreno

DIRECTOR: PHD. EDUARDO CARRILLO ZAMBRANO

FECHA DE PRESENTACIÓN: FEBRERO 24 DE 2010

CONTENIDO

pág.

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

LISTA DE ANEXOS

pág.

RESUMEN

La industria del software es grande y compleja. En la actualidad existe en el mundo una gran cantidad de empresas dedicadas al desarrollo y mantenimiento de software, además de otros modelos de negocio alrededor del mismo, por ejemplo, capacitación, adaptación, y mejora; específicamente en lo que respecta al software libre.

La industria del software es grande debido a la gran cantidad de oferta y demanda que existe en el mundo. Es imposible de concebir el mundo actual sin software, desde la empresa más pequeña que tiene un computador que cuenta mínimo con un sistema operativo y aplicaciones básicas para el procesamiento de información hasta la empresa más grande con software especializado para el control, almacenamiento, procesamiento e intercambio de información, de vital importancia para llevar a cabo sus procesos en forma rápida y eficaz.

La industria del software es compleja como se mencionó anteriormente, debido a las siguientes razones: 1) calidad insuficiente del producto final, 2) estimación de duración de proyectos y asignación de recursos inexactos, 3) retrasos en la entrega de los productos finales, 4) costos de desarrollo y mantenimiento de productos fuera de control, 5) escasez de personal calificado en un mercado de alta demanda y 6) tendencia del crecimiento de volumen y complejidad de los productos. Por lo tanto, este trabajo se centra en abordar una de las dificultades del software: calidad insuficiente del producto final. La calidad del software es de una altísima complejidad por eso se mide en tres momentos: en el proceso y en el producto. Haciendo un aporte a esta problemática se propone una guía metodológica que permita de forma práctica evaluar la calidad de aplicaciones Web de software libre, dirigida a empresas y profesionales en el área de las tecnologías de Información (TI).

Por otra parte, se consiguió documentar el estado de arte de los diferentes modelos y estándares de calidad a nivel del producto, además de hacer un análisis de la calidad tanto en la organización, como en el proceso, en el software y los datos. Además, se realizó una revisión bibliográfica de los diferentes modelos y estándares de calidad a nivel del proceso.

Por otro lado, se hizo la propuesta de una guía metodológica para evaluar la calidad de aplicaciones Web adaptada al modelo de software libre. Dicha propuesta está compuesta en total por nueve actividades distribuidas en tres

etapas: 1) evaluación, 2) implementación y 3) análisis. Es de resaltar que para esta guía se hace la propuesta básica de un árbol de características de calidad para aplicaciones de software libre a partir del modelo QSOS (Qualification and Selection of Open Source Software, por sus siglas en ingles).

Adicionalmente, es de mencionar que la hipótesis planteada para este caso de estudio no se vio cumplida. Dicha hipótesis planteaba que cualquier aplicación Web de cualquier dominio que se evaluara estaría sobre o por encima del punto crítico de aceptabilidad. Sólo una de las tres aplicaciones evaluadas estuvo por encima de dicho punto. Al parecer un factor fundamental al realizar una evaluación de calidad de aplicaciones de software libre tiene que ver con la madurez que tiene el proyecto, es decir, a mayor madurez existe mayor posibilidad de encontrar una aplicación con mayor nivel de calidad, por consiguiente, las otras dos aplicaciones provenían de proyectos muy recientes, que no contaban con la madurez suficiente.

Finalmente, este trabajo pretende convertirse en un aporte que sea de gran ayuda para las compañías o profesionales en el área de las TI y que tengan la necesidad de adquirir un instrumento práctico para la evaluación de la calidad de productos software para aplicaciones en entorno Web y de software libre. Además, incentivar a las compañías en la búsqueda de soluciones con alta calidad en TI de software libre, y por ende estimular los diferentes proyectos y la creación de nuevos bajo este movimiento.

INTRODUCCIÓN

ACERCA DEL PROBLEMA ABORDADO

El uso masivo de productos software a una escala global en casi todos los ámbitos del desempeño humano y el crecimiento de los servicios de Internet, ha creado la necesidad de que se observe con atención la calidad de los mismos, especialmente los de entorno Web bajo el modelo de software libre.

La academia y la industria son concientes de esta necesidad, por tanto, han realizado grandes esfuerzos por crear modelos y estándares de calidad. Lo anterior con el fin de que el software cumpla con las necesidades y expectativas del cliente, por medio del uso de las buenas prácticas en todo su ciclo de vida. La dificultad no está en la falta de herramientas que permitan medir y mejorar la calidad, la dificultad es de las compañías y/o profesionales en el área de TI. Debido a la complejidad que resulta seleccionar el modelo o estándar más adecuado para evaluar la calidad del software. Por otra parte, hay que sumarle que la calidad del software puede ser observada en tres momentos: en el proceso, en el producto o en el uso. Por consiguiente, en este trabajo se delimita el problema preocupándose por la calidad del producto y más aun concentrándose sólo en productos de entornos Web, que sean de software libre. Su motivación es hacer un aporte a la industria, a la académica, y a otros que sean de su interés, incrementando la productividad en la industria por medio de la automatización de procesos a través del uso de productos software de calidad, de estimular el uso de soluciones software bajo el modelo de software libre, y de esta forma, desmentir que el software libre carece de calidad atribuyéndole la calidad sólo al software propietario.

OBJETIVOS BÁSICOS DE LA TESIS

En este trabajo se definió un objetivo general y tres objetivos específicos. El objetivo general dice lo siguiente. “Adaptar e implementar una metodología para la evaluación de calidad para aplicaciones Web desarrolladas bajo el modelo de software libre tomando como caso de estudio una aplicación Web Open Source de interés para Universidades”. Dicho objetivo se cumplió a través de la realización de los tres objetivos específicos. A continuación se mencionan estos objetivos. Objetivo uno: documentar el estado de arte de los diferentes modelos y estándares para la calidad de software. Se dio cumplimiento en un 100% a este objetivo, ver capítulo uno. Objetivo dos:

desarrollar una guía para la implementación de una metodología de evaluación de calidad para aplicaciones Web adaptada al software libre. Dicho objetivo alcanza un 100% de cumplimiento, ver capítulo tres. Objetivo tres: realizar y documentar la evaluación de una aplicación Web Open Source de interés para Universidades a partir de la guía de evaluación propuesta. Este objetivo alcanza un 100% de cumplimiento, observar el capítulo dos y cuatro.

ORGANIZACIÓN DE ESTA TESIS

Esta tesis está organizada en seis capítulos. En el capítulo uno se encuentra el estudio de los diferentes modelos y estándares de calidad, dando cumplimiento al siguiente resultado esperado: informe acerca de los diferentes estándares y modelos de calidad para el software. En el capítulo dos se aprecia el estudio realizado correspondiente al dominio al que pertenecen las herramientas evaluadas. Dicho estudio permitió establecer el árbol de requerimiento de calidad que se observa en el apartado 4.3.2.6. En el capítulo tres se da cumplimiento al resultado esperado: guía de implementación para la evaluación de calidad para aplicaciones Web adaptada al modelo de software libre. Por otro lado, en este mismo capítulo en diferentes apartados se encuentra la siguiente información: 1) la descripción del problema que se aprecia en el apartado 3.1, 2) el señalamiento de la solución que se observa en el apartado 3.2, 3) la propuesta de la solución que se muestra en el apartado 3.3, y 4) las características, subcaracterísticas y atributos de calidad para evaluar aplicaciones de software libre que se observa en el apartado 3.5. En el capítulo cuatro se hace la implementación de la guía propuesta a través de un caso de estudio, y es de observar que en el apartado 4.3.4.3 está el análisis de los resultados de las tres herramientas evaluadas. Por otra parte, en este mismo capítulo se da cumplimiento al resultado esperado: documento acerca de la evaluación de calidad realizada a la aplicación Web de Open Source seleccionada. Finalmente, los dos últimos capítulos, el cinco y el seis corresponden a conclusiones y recomendaciones y trabajos futuros.

DESARROLLO DE LA TESIS

Alcanzar el objetivo general y los objetivos específicos de este trabajo se logra a través de los principios de una investigación documental, que según Alfonso[1], "es un procedimiento científico, un procedimiento sistemático de indagación, recolección, organización, análisis e interpretación de información, éste es conducente a la construcción de conocimiento". Con base en lo anterior para el desarrollo de este trabajo se definieron tres etapas. La primera etapa realizada correspondió al estudio de modelos y estándares de calidad de

[1]Morales, Oscar Alberto. Fundamentos de la Investigación Documental y la Monografía, dado por Alfonzo, I. Técnicas de investigación bibliográfica. Caracas: Contexto Ediciones, 1994.

software que corresponde al capítulo uno. En ella se hizo una revisión bibliográfica de los diferentes modelos y estándares de calidad, permitiendo clasificarlos de acuerdo a su campo de acción, proceso y producto, como también identificar las propuestas que están dirigidas al software libre. La segunda etapa corresponde a la propuesta de la guía metodológica para la evaluación de aplicativos Web adatada al modelo de software libre que corresponde al capítulo tres. Esta etapa se consigue por medio del estudio realizado en la primera, permitiendo identificar los requerimientos de calidad que ofrece cada modelo y estándar de calidad del producto que aquí sé proponen. Por otra parte, se establecen las tres principales etapas y sus actividades o pasos de la guía propuesta. La última etapa corresponde a la implementación de la guía propuesta que se hace a través del caso de estudio, a ella le corresponde el capítulo dos que es el estudio del dominio al que pertenecen las herramientas software evaluadas y el capítulo cuatro que corresponde al caso de estudio donde se hizo la implementación de dicha guía. Es de resaltar que para el desarrollo del caso de estudio fue de mucha importancia el estudio realizado en el capítulo dos, debido a qué permitió identificar los atributos, subcaracterísticas y características correspondientes a los requerimientos de calidad.

1. ESTÁNDARES Y MODELOS DE CALIDAD

1.1 INTRODUCIÓN

La calidad es uno de los aspectos más importantes que debe tener el software. Dicho aspecto no es sencillo de conseguir más en un mercado tan competitivo con clientes exigentes. Es por esto, que las empresas de software se ven obligadas a tomar medidas orientadas a cumplir las necesidades y expectativas de sus clientes, en otras palabras, ofrecer productos con alta calidad. Esta calidad es denominada calidad de software.

La calidad de software es compleja por su naturaleza. Un producto software tiene características muy particulares que lo hace diferente a otros productos industriales. A continuación se mencionan algunas particularidades especiales que tiene el software:

1. El software es un producto mental, no restringido por las leyes de la Física o por los límites de los procesos de fabricación. Es algo abstracto, y su calidad también lo es.
2. Se desarrolla, no se fabrica. El coste está fundamentalmente en el proceso de diseño, no en la producción y los errores se introducen también en el diseño, no en la producción.
3. El software no se deteriora con el tiempo. No es susceptible a los efectos del entorno, y su curva de fallos es muy diferente a la del hardware. Todos los problemas que surjan durante el mantenimiento estaban desde el principio, y afectan a todas las copias del mismo; no se generan nuevos errores.
4. Es artesanal en gran medida. El software, en su mayoría, se construye a medida, en vez de ser construido ensamblando componentes existentes ya probados, lo que dificulta aún más el control de su calidad. Aunque se ha escrito mucho sobre la reutilización del software, hasta ahora se han conseguido pocos éxitos tangibles.
5. El mantenimiento del software es mucho más complejo que el mantenimiento del hardware. Cuando un componente de hardware se deteriora se sustituye por una pieza de repuesto, pero cada fallo en el software implica un error en el diseño o en el proceso mediante el cual se tradujo el diseño en código de máquina ejecutable.

6. Es engañosamente fácil realizar cambios sobre un software, pero los efectos de estos cambios se pueden propagar de forma explosiva e incontrolada.
7. Como disciplina, el desarrollo de software es aún muy joven, por lo que las técnicas de las que disponemos aún no son totalmente efectivas o no están totalmente calibradas.
8. El software con errores no se rechaza. Se asume que es inevitable que el software presente errores[2].

Dificultades del software:

1. Aumento constante del tamaño y complejidad de los programas.
2. Carácter dinámico e iterativo a lo largo de su ciclo de vida, es decir que los programas de software a lo largo de su vida cambian o evolucionan de una versión a otra para mejorar las prestaciones con respecto a las anteriores.
3. Dificultad de conseguir productos totalmente depurados, ya que en ningún caso un programa será perfecto.
4. Se dedican elevados recursos monetarios a su mantenimiento, debido a la dificultad que los proyectos de software entrañan y a la no normalización a la hora de realizar los proyectos.
5. No suelen estar terminados en los plazos previstos, ni con los costes estipulados, ni cumpliendo los niveles deseables de los requisitos especificados por el usuario.
6. Incrementos constantes de los costes de desarrollo debido entre otros, a los bajos niveles de productividad.
7. Los clientes tienen una alta dependencia de sus proveedores por ser en muchos casos aplicaciones a "medida".
8. Procesos artesanales de producción con escasez de herramientas.
9. Insuficientes procedimientos normalizados para estipular y evaluar la calidad, costes y productividad[3].

La particularidad y los problemas generales que tiene el software hacen difícil la implementación de la calidad. Scalone[4] plantea la siguiente pregunta: se puede encontrar un conjunto de propiedades en un software que nos de una indicación de su calidad ?. La respuesta es: si, y se consigue a través de la implementación de modelos de calidad. Por lo tanto, para que una empresa de software implemente un modelo de calidad, lo haría por medio de la gestión de

[2] SCALONE, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software. Buenos Aires: Universidad Tecnológica Nacional, Facultad Regional Buenos Aires, 2006. p. 2
[3] Ibid., p 3.
[4] Ibid., p3.

calidad de software. Es decir, que para obtener calidad de software se necesario hacerlo a través de la gestión de calidad.

La gestión de calidad de software se compone básicamente por un conjunto de actividades orientadas a obtener productos de calidad que sean competitivos. Por otra parte, su aplicación está a nivel de organización, no obstante en cada proyecto se puede aplicar gestión de calidad. Sus actividades se organizan o agrupan en las siguientes categorías: 1) Planeación de Calidad de Software, 2) Control de Calidad de Software, 3) Aseguramientos de Calidad de Software y 4) Mejora de Calidad de Software. Dichas categorías se aplican durante todo el ciclo de vida del software por ende también la gestión de calidad.

La calidad de software es "la concordancia con los requerimientos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo documentados y con las características implícitas que se esperan de todo software desarrollado profesionalmente"[5]. Otra definición según ISO 8402[6]: Son las características de entidad que le confieren su aptitud para satisfacer las necesidades expresadas y las implícitas. Palabras más palabras menos, la calidad de software busca establecer una concordancia entre los requerimientos establecidos por el cliente con el cumplimiento de los mismos reflejados en el producto software final. Conseguir este objetivo se logra a través del uso de normas o estándares genéricos y procedimientos, ajustados a las particularidades de cada organización.

Por otra parte, la calidad de software se especifica por niveles: 1) a nivel de organización, 2) a nivel de proceso, 3) a nivel producto o software y 4) a nivel de datos. Dado lo anterior, es de mencionar la diferencia que existe entre cada nivel en relación a la forma de cómo se aplica la calidad. No obstante, existe una relación en doble vía entre cada nivel, es decir, si se tiene una alta calidad en el proceso, por consiguiente, será un producto con calidad similar a la del proceso, en caso contrario, de tener un producto de baja calidad implicaría revisar y mejorar la calidad en el proceso.

1.2 GESTIÓN DE CALIDAD DE SOFTWARE

Las empresas de software que desean producir software de calidad y ser competitivas, por ende, deben implantar la gestión de calidad. La cual es, "Aspectos de la función de gestión que determinan y aplican la política de la calidad, los objetivos y las responsabilidades y que lo realiza con medios tales como la planificación de la calidad, el control de la calidad, la garantía de

[5] SCALONE, Op. cit. p. 1, dado por Pressman, R.S: Ingeniería del Software. Un enfoque práctico. Mc Graw Hill, 2002.

[6] CUEVA LOVELLE, Juan Manuel. Calidad del Software. Oviedo: Departamento de Informática, Universidad de Oviedo, 1999. p.3, dado por, ISO 8402 (UNE 66-001-92).

calidad y la mejora de la calidad"[7]. Por otra parte, la definición según la ISO 9000:2000[8]. Es un conjunto de actividades de la dirección general que tiene como fin determinar la calidad, los objetivos y responsabilidades, y su implantación se debe hacer a través de medios tales como la planificación de calidad, el control de calidad, el aseguramiento de calidad y la mejora de calidad, enmarcado dentro de un sistema de calidad.

En todo el ciclo de vida del software se aplica la gestión de calidad de software, por lo cual, constantemente se debe evaluar y actualizar para dar garantía que al final se obtendrá un producto de calidad. Su forma de evaluación se basa en la recolección de métricas que permitan medir el rendimiento de las actividades y el proceso con el fin de mejorar, es decir, que la gestión calidad está en una dinámica constante de mejoramiento continúo. Por tanto, su propósito es establecer las necesidades y requerimientos de sus clientes en términos de calidad.

1.2.1 Planificación de calidad del software.

La calidad de software requiere de planificación, la cual, es una actividad que se da en la gestión de calidad de software. Según la norma ISO 9000:2000[9] la planificación de calidad se define como la parte de la gestión que se encarga de definir los objetivos de calidad, establecer la especificación de procesos operativos y el manejo de recursos, tanto humanos como materiales, necesarios para cumplir con los objetivos de calidad. La planificación de calidad de software, por otro lado, se encarga de realizar el proceso administrativo de desarrollar y mantener una relación entre los objetivos y recursos de la organización, además, de las oportunidades cambiantes del mercado[10].

Los aspectos que se deben tener en cuenta en la planificación de la calidad de software son: 1) modelo / estándar de calidad de software a utilizar, 2) costo de calidad de software y 3) los recursos humanos y materiales. Como uno de los aspectos de la planificación de calidad de software es la elección del modelo / estándar de calidad a usar, su determinación se puede hacer a través de factores. Scalone[11] realiza la siguiente propuesta de factores.

1. La complejidad del proceso de diseño.
2. La madurez del diseño.
3. La complejidad del proceso de producción.
4. Las características del producto o servicio.

---

[7] NORIEGA QUINTANA, Darcy Javier, HERNANDEZ PERES, Camilo, y GABINO MERINO, Nuirka San. Cuba: Calidad de Software. Universidad de Matanzas "Camilo Cienfuegos", Facultad de Informática. 2006. p. 3

[8] SCALONE, Op. cit. p. 4, dado por ISO 9000:2000.

[9] Ibid., p. 5

[10] SCALONE, Op. cit. p. 4, dado por ISO/IEC 90003:2004.

[11] SCALONE, Op. cit. p. 5

5. La seguridad del producto o servicio.
6. Económico.

Con base en estos factores se puede hacer la elección del modelo o estándar de calida más apropiado para ser implementado ya sea para la calidad a nivel de proceso o a nivel del producto.

Anteriormente se ha mencionado que la gestión de calidad de software se aplica durante todo el ciclo de vida de este. Por lo tanto, para que la gestión de calidad sea buena, tiene que tener objetivos claros y específicos, a los cuales, deben asignársele los recursos suficientes o de lo contrario muy difícilmente se podrán cumplir. Es ese el papel fundamental de la planificación de calidad de software; establecer una relación ágil y dinámica entre objetivos y recursos proporcionados por la organización. Además, debe tenerse en cuenta la influencia que generan los cambios del mercado. Por consiguiente, la planificación de calidad de software permite determinar los siguientes aspectos: "1) rol de la planificación, 2) requerimientos de la CS, 3) preparación de un plan de CS, 4) implementación de un plan de CS y 5) preparar un manual de calidad[12]".

El plan de calidad define los atributos de calidad más importantes del producto a ser desarrollado y define el proceso de evaluación de calidad.

1.2.2 Control de calidad del software.

Son las actividades orientadas a evaluar la calidad de los productos desarrollados. Según la norma ISO 9000:2000[13], es la parte de la gestión de la calidad orientada al cumplimiento de los requisitos de calidad. Pressman[14] dice, son las técnicas y actividades de carácter operativo utilizadas para satisfacer los requisitos relativos a la calidad, con base en dos objetivos fundamentales: 1) mantener bajo control un proceso, y 2) eliminar las causas de los defectos en las diferentes fases del ciclo de vida.

Uno de los aspectos a tener muy en cuenta del control de calidad software es la prueba de software. Debido a que por intermedio de ella se puede ir verificando que la calidad se esté cumpliendo, para tomar las medidas necesarias y eliminar los defectos presentados en las diferentes etapas del ciclo de vida del software.

1.2.3 Aseguramiento de calidad del software.

[12] Ibid., p. 7
[13] SCALONE, Op. cit. p. 7, dado por ISO 9000:2000.
[14] CUEVA LOVELLE, Op. cit. p6, dado por, R. S. Pressman. Ingeniería del software. Un enfoque práctico. 4ª Edición. McGrawHill (1998).

Es el conjunto de actividades orientadas a dar confianza en que el producto software satisface los requisitos dados de calidad.

La norma ISO 9000:2000[15] dice: es la parte de la gestión de calidad orientada a proporcionar confianza en que se cumplirán los requisitos de calidad. Pressman[16] la define como el conjunto de actividades planificadas y sistemáticas para aportar la confianza que el software satisfará los requisitos dados de calidad.

El aseguramiento de calidad se diseña para cada aplicación antes de comenzar a ser desarrollada y no después. Además está se presente en: 1) métodos y herramientas a lo largo del ciclo de vida del software, 2) inspecciones técnicas formales durante el ciclo de vida del software, 3) estrategia de pruebas multiescala, 4) control en la documentación del software y de los cambios realizados, 5) procedimientos para ajustarse a los estándares, 6) mecanismos de medida (métricas) y 7) registro de auditoria y realización de informes.

Por otra parte, la revisión técnica formal (RTF) es el filtro más efectivo desde el punto de vista del aseguramiento de calidad de software, por tanto es un medio para mejorar la calidad de software.

Actividades para el aseguramiento de calidad de software: "1) métricas de software para el control del proyecto, 2) verificación y validación del software a lo largo del ciclo de vida, y 3) la gestión de la configuración del software"[17].

1.2.4 Mejoramiento de Calidad del Software.

La norma ISO 9000:2000 dice: "es la parte de la gestión de calidad orientada a aumentar la capacidad de cumplir con los requisitos de la calidad. Los requisitos pueden estar relacionados con cualquier aspecto tal como la eficacia, la eficiencia o la trazabilidad[18]".

El mejoramiento de calidad de software consiste en mejorar la calidad a través evaluaciones constantes durante todo el proceso del ciclo de vida del software. La evaluaciones se realiza mediante mediciones, análisis de datos y auditorias.

La auditoria permite dar confianza en un producto o proceso debido a que se puede establecer el estado real del mismo a través de mediciones que confrontan sus resultados con los estándares, las guías, las especificaciones y los procedimientos.

---

[15] SCALONE, Op. cit. p. 16, dado por ISO 9000:2000.
[16] CUEVA LOVELLE, Op. cit. p4, dado por, R. S. Pressman. Ingeniería del software. Un enfoque práctico. 4ª Edición. McGrawHill (1998).
[17] NORIEGA QUINTANA, HERNANDEZ PERES, y GABINO, Op. cit. p. 4
[18] SCALONE, Op. cit. p. 22, dado por ISO 9000:2000.

Scalone menciona algunas razones para realizar una auditoria: 1) establecer el estado del proyecto, 2) verificar la capacidad de realizar o continuar un trabajo específico, 3) verificar qué elementos aplicables del programa o plan de aseguramiento de la calidad han sido desarrollados y documentados y 4) verificar la adherencia de esos elementos con el plan de aseguramiento de la calidad.

1.3 CALIDAD EN LAS EMPRESAS DE SOFTWARE

Las empresas dedicadas al desarrollo de software reconocen la importancia que tiene la calidad en sus productos. Por lo tanto, deben realizar grandes esfuerzos para lograrlo, más aun con clientes exigentes y un mercado agresivo donde sobreviven las empresas que son altamente competitivas. Hay que diferenciar en la oferta entre la empresa competidora y la empresa competitiva. La empresa competidora tiene como propósito una estrategia de supervivencia, dando la más mínima importancia a la calidad, por lo que se resaltan los siguientes aspectos: 1) no maneja la diferenciación, 2) basa su oferta en costos bajos y precios bajos, 3) compite y no le interesa crecer, 4) es una empresa más del montón, 5) no tiene estrategias de mejoras, 6) no tiene estrategias de calidad y 7) su único interés es vender. Diferente a la empresa competitiva que si le da gran importancia a la calidad y a sus clientes. Por lo cual, se le observan las siguientes características: 1) estrategia de negocios con: liderazgo empresarial, plan estratégico, visión de indicadores de clase mundial, toma de riesgos controlados, e identificación y ambición de oportunidades de mercado, 2) orientación al cliente con: valor agregado y efectivo a sus clientes, excedimiento de la expectativa del cliente, puntualidad, y cumplimiento de compromisos y acuerdos, 3) salud financiera que sea; rentable y que tenga precios atractivos, 4) recurso humano con personal: experimentado, motivado, capacitado y bilingüe 5) producción con: estándares, control operativo, y calidad en procesos y 6) gestión tecnológica que busqué la innovación de productos y servicios, y certificación en tecnologías y plataformas que usa.

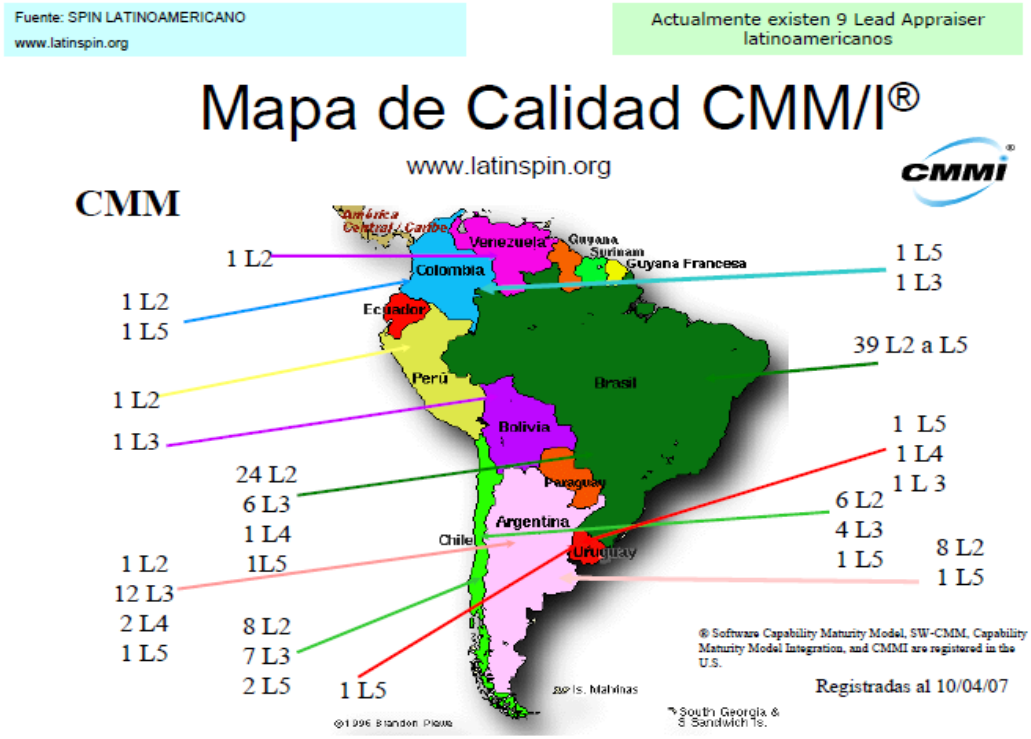
La empresa de software es conciente de la importancia que representa ser una empresa competitiva. Por esto, su estrategia de calidad se enfoca en la implementación de modelos y estándares de calidad. El desarrollar software de calidad requiere de una alta madurez en los procesos, y obtenerlo conlleva la implementación de un modelo o estándar de CS (Calidad de Software). Los países potencias en desarrollo de software tienen experiencia en el uso de estos modelos y estándares, creando un ambiente propicio a través de discusiones continuas entre los productores de software, la academia, las entidades de estandarización, y secretarías de gobierno, que finalmente incentivan a las empresas de software para que definan su estrategia

competitiva basada en la calidad, por consiguiente poseen la mayor parte del mercado.

Entonces para que las empresas de software de países del tercer mundo puedan estar al nivel de las potencias es necesario que el esfuerzo sea una responsabilidad colectiva, involucrando al sector empresarial, a la academia, y al estado, para que entre ellos se generen políticas, iniciativas, e incentivos financieros para mejorar la calidad del software. Por ejemplo, México con la unión de esfuerzos entre la Facultad de Ciencias de la Universidad Nacional Autónoma de México (UNAM) y la Secretaria de Economía han realizado la propuesta de un modelo que tiene como fin, mejorar y evaluar los procesos de desarrollo y mantenimiento de sistemas y productos software conocido con el nombre de Moprosoft. Así mismo en esta dinámica México se ha dado cuenta que Moprosoft es una propuesta que está orientada a los procesos de las empresas y no de las personas, para lo cual, la Secretaría de Economía a dado marcha a la iniciativa nacional TSP/PSP (Team Software Process/Personal Software Process, sus siglas en ingles), que está trabajando directamente con el SEI (Software Engineering Institute, sus siglas en ingles) y el Dr. Humphrey que tiene como objetivo crear una infraestructura humana que permita la introducción y expansión del uso de TSP, para que la industria del software de México alcance un desempeño superior en su competencia internacional.

Internacionalmente el modelo con más aceptación a nivel de proceso es CMMI (Capability Maturity Model Integration, sus siglas en ingles). Sin embargo, este modelo tiene una complejidad muy alta para ser implementado en empresas pequeñas. A pesar de esto, un grupo de empresas en Colombia y otros países de Latinoamérica tienen en su estrategia de calidad, estar certificados en su nivel 5. A continuación se muestra en la Figura 1 el mapa de Calidad CMMI monitoreado en abril de 2007 por Spin Latinoamérica (entidad encargada de fomentar en Latinoamérica la calidad orientada al proceso de software).

Figura 1. Mapa de calidad CMMI



Fuente: http://www.latinspin.org/

Por otro lado, en la tabla 1 se observa la aceptación que ha tenido desde abril de 2007 hasta agosto de 2009 el modelo de CMMI en Colombia y Brasil. Es de apreciar la gran superioridad que sigue manteniendo Brasil sobre Colombia y los demás países de Latinoamérica.

Tabla 1. Calidad CMMI en Latinoamérica en el 2009

País	Nivel I	Nivel II	Nivel III	Nivel IV	Nivel V
Brasil		59	18	2	11

Colombia		4	10	1	1

"Los Modelos de Calidad son aquellos documentos que integran la mayor parte de las mejores prácticas, proponen temas de administración en los que cada organización debe hacer énfasis, integran diferentes prácticas dirigidas a los procesos clave y permiten medir los avances en calidad."[19]

"Los Estándares de Calidad son aquellos que permiten definir un conjunto de criterios de desarrollo que guían la forma en que se aplica la Ingeniería del Software. Los estándares suministran los medios para que todos los procesos se realicen de la misma forma y son una guía para lograr la productividad y la calidad. "[20]

Las empresas software que implementan modelos y/o estándares de calidad son empresas competitivas y tienen en claro que al cliente no sólo le interesa el precio, sino los servicios y la confiabilidad que puede brindar los productos software. Por lo tanto, la competencia entre las empresas no es a nivel de precios, sino a la que brinde mayor confianza, y esa confianza se consigue haciendo uso de las mejores costumbres en los procesos para el desarrollo de software, es decir, implementando un modelo y/o estándar de calidad, que deberá permitir: 1) unir la misión de la empresa y el esfuerzo de cada área en una sinergia de resultados hacia la competitividad y la calidad de clase mundial, 2) tener procesos y procedimientos ágiles; y compresibles para todos lo involucrados, pasando por las etapas del ciclo de vida del software, y 3) reducir costos.

Implementar un modelo y/o estándar de calidad trae consigo un cambio de mentalidad que incluye a todo el personal de la empresa, desde directivos hasta los empleados con más bajo rango. Por lo tanto, es un cambio cultural a nivel de organización que afecta a cada persona, en su forma de trabajar y de pensar, adquiriendo una responsabilidad individual que en su sumatoria tendría como resultado el esfuerzo colectivo en hacer las cosas a través de las mejores costumbres, es decir, con calidad.

Una de las cosas más difíciles es la selección del modelo y/o estándar de calidad a implementar. Es necesario que el equipo de trabajo tenga en claro las necesidades del cliente; actuales y potenciales, y las expectativas que la empresa tiene trazadas para alcanzar en un futuro no muy lejano. Por otra parte, los elementos de modelo y/o estándar de calidad deben estar estructurados de tal forma que permitan un control y aseguramiento de todos los procesos involucrados con la calidad.

[19] SCALONE, Op. cit. p. 26, dado por Piattini , García, *"C*alidad en el desarrollo y mantenimiento del software", RA-MA Editorial, Madrid, 2003.

[20] SCALONE, Op. cit. p. 26, dado por Piattini , García, *"C*alidad en el desarrollo y mantenimiento del software", RA-MA Editorial, Madrid, 2003.

Las ventajas que trae implementar un modelo y/o estándar de calidad, correspondientes a tener más control sobre el proyecto y los procesos implicados son: 1) disminución del número de defectos, 2) disminución de los tiempos de entrega, 3) disminución de los costos, 4) mayor satisfacción del cliente, y 5) mayores beneficios.

Por otra parte, Scalone plantea que para tener éxito en la implementación de un modelo y/o estándar de calidad de software se requiere que los directivos de la empresa de software comprenda la necesidad de fomentar en la empresa los siguientes conceptos:

> 1) Establecer una cultura de calidad en la empresa.
> 2) Establecer la atención centrada en el cliente creando el máximo valor.
> 3) Inculcar en todos la premisa de hacerlo bien, a la primera vez y siempre.
> 4) Crear constancia y ser perseverante con el propósito de mejorar los productos de software y servicios.
> 5) Realizar propuestas de innovación para mejorar la efectividad de la cadena de valor.
> 6) Establecer que los procesos, los métodos y sistemas deben estar sujetos a ciclos de mejora continua.
> 7) Establecer un programa para el diseño e implantación de los procesos y sistemas que integran el modelo / estándar de calidad del software.
> 8) Contribuir con la sociedad promoviendo los valores de calidad y generando un compromiso con el bienestar de la sociedad y con la conservación del medio ambiente[21].

Implementar la calidad de software en una empresa de software requiere de un conjunto de actividades y medios necesarios para definir el sistema de calidad por un lado, y por otra parte, de su control, aseguramiento y mejora continua. Esto se logra a través de la gestión de calidad de software, la cual, se implementa en tres niveles, independientemente de la actividad comercial de la empresa: a nivel organizacional, a nivel de proyecto, y a nivel de producto. Más sin embargo, las empresas de software debido a que su producto, el software, que trabaja con datos, estos pueden ser evaluados desde el punto de vista de calidad, por lo tanto, se considera otro nivel para aplicar la calidad, y es a nivel de datos.

1.3.1 Calidad a nivel organizacional.

[21] SCALONE, Op. cit. p. 27

En este nivel, la calidad tiene dos campos de trabajo. A nivel de entidad o organización, y a nivel de proyecto. La primera tiene como base una estructura, la cual, procura conseguir la calidad del producto a través de la mejora de actividades y/o procesos que competen: a la producción, comercialización y la interacción con el cliente. Además de fomentar el trabajo de calidad en todas las personas y de cada departamento de la empresa. Por lo tanto es necesario de un sistema de calidad que sus objetivos estén alineados a la empresa para que determinen: 1) responsabilidades, 2) actividades, 3) recursos, 4) procedimientos para llevar la gestión de calidad, y 5) la estructura de la organización.

El segundo se centra en lo correspondiente al desarrollo de software, el cual, se apoya en la infraestructura organizativa, la cual le proporciona estándares que guiarán las distintas actividades correspondientes al desarrollo y mantenimiento, las que deben adaptares a las particularidades de cada proyecto. Por otro lado, el aseguramiento de calidad de software se encarga de velar por que sean aplicados los procesos definidos a en la organización a partir del sistema de calidad y las guías de la misma adaptada a las características propias de cada proyecto.

La organización en su gestión de calidad se ve influenciada por, las entidades internacionales de estandarización para organizaciones de producción o servicios, especialmente por las directrices marcadas por la ISO (Internacional Organization for Standardización, sus siglas en ingles) a través de las diferentes normas. La norma ISO 9000 para la gestión de calidad que aplica de forma genérica para cualquier tipo de organización, para el software, la norma ISO 9001, y específicamente para las empresas que tiene como actividad principal la producción de software utilizan la norma ISO 9003. Por otra parte, el mundo del software tiene sus propias líneas de acción en la gestión de calidad que trabajan sobre los procesos de producción como elemento para garantizar la calidad del software. CMMI es un ejemplo un modelo de clasificación y mejora de los procesos empleados por las organizaciones que fue propuesto por miembros de la industria, el gobierno y SEI (Software Engineering Institute, sus siglas en ingles).

1.3.2 Calidad a nivel de proceso.

Obtener la calidad se logra a través de la gestión de calidad en todas las áreas de las que hace parte del proceso de una organización, por medio del uso de una metodología. Por lo tanto, el conseguir información de los procesos de modo que puedan controlarse y mejorarse, producen un aumento en la calidad de los productos y servicios. Así mismo, para alcanzar la calidad es necesario

satisfacer elementos del proceso tales como: 1) satisfacción de la alta dirección, 2) satisfacción del personal involucrado en el desarrollo del software, y 3) satisfacción del usuario final.

El control de calidad se aplica en cada etapa del proceso de desarrollo del software, por lo cual, si se presenta errores dependerán del funcionamiento y de la forma como ha venido evolucionado en todo el proceso. Por consiguiente, la calidad obedecerá a la forma de cómo se lleva el proyecto en cada proceso y subproceso, además, que la calidad de software se debe diseñar conjuntamente con el proyecto de software a desarrollar, y nunca al final.

Por otra parte, en la calidad a nivel del proceso se observa, en los sistemas de garantía de calidad la relación entre los precios y costos que generan fallas al producir software. Por ejemplo, 1) el alto costo que implica reparar los defectos de un software desarrollado, 2) la reducción del precio al obtener una calidad pobre del producto final, 3) el costo que demanda el proceso de inspección del software, 4) el costo del sistema de garantía de calidad y de los beneficios obtenidos. Por ende, se puede concluir que a mayor calidad, mayor es el costo de desarrollo, pero será mayor el beneficio en el mantenimiento del software.

1.3.3 Calidad a nivel de software.

Desarrollar software de calidad no es una tarea compleja, por el contrario, es fácil, siempre y cuando se tenga conocimiento y experiencia en la utilización de metodologías o procedimientos estándares en las etapas del ciclo de vida del software, como de análisis, de diseño, de programación, y de pruebas, obteniendo de esta forma una filosofía de trabajo uniforme que dará como resultado una mayor confiabilidad, facilidad de mantenimiento y facilidad de prueba, elevando así la productividad para el desarrollo y control de calidad.

La evaluación de calidad es lograda a través de la implementación de modelos o estándares de calidad de software. Los modelos de calidad se definen de forma jerárquica: nivel 1, factores de calidad, nivel 2, criterios de calidad del producto y nivel 3, métricas del producto.

El uso de modelos o estándares de calidad tiene algunas ventajas. La principal ventaja es que la calidad es algo concreto, por lo tanto, se puede definir, medir, y planificar. Otra ventaja es que se pueden comprender las relaciones que existen entre las diferentes características del producto.

Por otro lado, la calidad del producto de software abarca otros aspectos: la calidad interna, la calidad externa, y la calidad en uso. La calidad interna: mide las características internas, como el código fuente. La calidad externa: mide el comportamiento del producto, como en una prueba. La calidad en uso: mide el comportamiento del producto por parte del usuario final.

1.3.4 Calidad a nivel de datos.

Es un concepto multidimensional que comprende diferentes aspectos con base en las necesidades de los consumidores de datos o de los diseñadores de sistemas. Por lo tanto, la calidad de datos es catalogada por dimensiones. A continuación se mencionan algunas dimisiones de la calidad de datos.

- Exactitud: mide el grado en que la información refleja lo que esta pasando en el negocio (ej. exactitud de inventarios, exactitud de rutas de fabricación, de listas de materiales, etc.).
- Totalidad: medición que refleje el grado en que las bases de datos cuentan con toda la información critica para el negocio.
- Oportunidad: medición de que la información este disponible cuando se requiere para tomar una decisión.
- Relevancia: que la información le sirva a la persona que se la estas proporcionando.
- Nivel de detalle: que la información tenga el nivel de detalle requerido, dependiendo del nivel organizacional y el tipo de decisión al cual esté destinada la información.
- Consistencia: que la información sea la misma en todas las áreas o sistemas utilizados por la compañía.
- Facilidad de acceso: la disponibilidad de los datos debe ser fácil o rápidamente recuperables.
- Cantidad apropiada: el volumen de los datos debe ser adecuada en relación con la tarea que se está realizado.
- Seguridad: el acceso a los datos debe estar restringido apropiadamente para garantizar su seguridad.

1.4 MODELOS / ESTANDARES DE CALIDAD DE SOFTWARE

En la tabla 2 se observa el listado detallado de los diferentes modelos y estándares de calidad de software tanto a nivel de proceso como del

producto.

Tabla 2. Listado de los diferentes modelos y estándares de calidad para el proceso y producto

Nivel	***Modelo***	***Estándar***
PROCESO	CMMI	ISO 90003
	TickIT	ISO 12207
	Bootstrap	ISO 15504 (SPICE)
	Personal SW Process (PSP)	IEEE / EIA 12207
	Team SW Process (TSP)	ISO 20000
	Practical SW Measurement (PSM)	ITIL
	Six Sigma for Software	Cobit 4.0
	Gilb	ISO 9126-1
	GQM	ISO 25000 (SQUARE)
	Mc Call	IEEE Std 1061- 1998

Tabla 2 (Continuación)

Nivel	***Modelo***	***Estándar***
PRODUCTO	Furps	
	Boehm	
	SATC	
	Dromey	
	C-QM	
	Metodología SQAE	
	WebEQM	

Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software

Por otra parte, en la figura 2 se aprecia un cuadro comparativo entre las diferentes metodologías para valorar los procesos del software libre. Dichas metodologías han sido propuestas tanto por parte de la industria y la académica. Por ejemplo, OSMM es una propuesta hecha por Cappemini empresa proveedora de servicios de consultoría, tecnología y outsourcing. Otro ejemplo, OpenBRR que es una propuesta hecha por Carnegie Mellon Silicon

Valley, campus de investigación de la universidad Carnegie Mellon.

Figura 2. Listado de metodologías para valor el software libre

Criteria	OSMM Capgemini	OSMM Navica	QSOS	OpenBRR
Seniority	2003	2004	2004	2005
Original authors/sponsors	Capgemini	Navicasoft	Atos Origin	Carnegie Mellon Silicon Valley, SpikeSource, O'Reilly, Intel
License	Non-free license, but authorised distribution	Assessment models licensed under the Academic Free License	Methodology and assessments results licensed under the GNU Free Documentation License	Assessments results licensed under a Creative Commons license
Assessment model	Practical	Practical	Practical	Scientific
Detail levels	2 axes on 2 levels	3 levels	3 levels or more (functional grids)	2 levels
Predefined criteria	Yes	Yes	Yes	Yes
Technical/functional criteria	No	No	Yes	Yes
Scoring model	Flexible	Flexible	Flexible	Strict
Scoring scale by criterion	1 to 5	1 to 10	0 to 2	1 to 5
Iterative process	No	No	Yes	Yes
Criteria weighting	Yes	Yes	Yes	Yes
Comparison	Yes	No	Yes	No

Fuente:
http://en.wikipedia.org/wiki/Open_source_software_assessment_methodologies

1.4.1 Modelos de calidad de software a nivel de proceso.

1.4.1.1 Capability maturity model integration - CMMI.

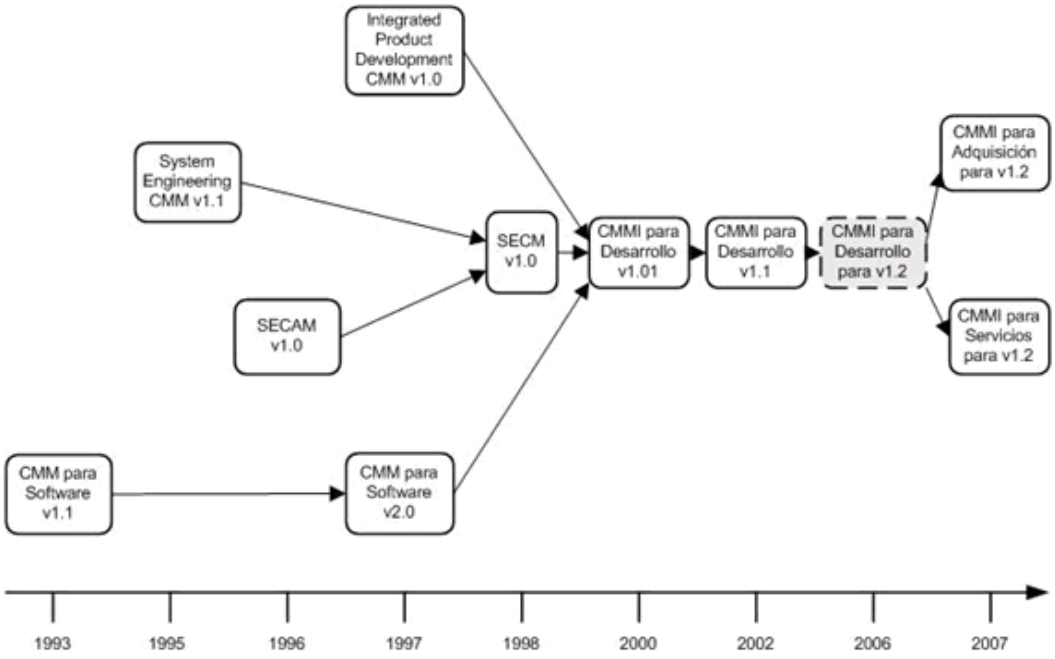
CMMI es un modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software. Por otra parte, CMMI es una evolución de CMM (Capability Maturity Model, sus siglas en ingles), que es un modelo de evaluación de los procesos de una organización, desarrollado entre 1987 y 1997, inicialmente para los procesos relativos al desarrollo e implementación de software por la Universidad Carnegie Mellon para el SEI (Software Engineering Institute, sus siglas en ingles). Es en el 2002 que se lanza CMMI 1.1 y en agosto de 2006 se lanza CMMI 1.2. Creado por

miembros de la industria, el gobierno de EE.UU. y SEI, del mismo modo que fue patrocinado por el ministerio de defensa de los Estados Unidos y NDIA (National Defense Industrial Association, sus siglas en ingles). Los creadores de CMMI lo crearon bajo la premisa que la calidad del producto o servicio está altamente influenciada por la calidad de los procesos que la producen y mantienen.

Una empresa tiene tres dimisiones críticas: las personas, los procesos, y la tecnología. Los procesos es la dimensión puente que se encarga de unir a las otras dos para alcanzar los objetivos del negocio. Por lo tanto, un enfoque centrado en los procesos ayuda a crear una plataforma de mejora continua, donde, las personas y la tecnología son cambiantes, pero los procesos transcienden en el tiempo y se adapta a las otras dos. Por consiguiente, la mejora continúa de los procesos debe ir paulatinamente incrementando el nivel de capacidades y madurez de una organización. En las empresas se puede encontrar un conjunto de procesos no definidos y procesos disciplinados, los primeros corresponden a procesos cuya organización cuenta con poca capacidad y madurez para realizarlos a diferencia de los segundos, donde, la organización cuenta con la capacidad y madurez suficiente para desarrollar con calidad probada. Luego una organización es capaz de definir su calidad total por medio del nivel de madurez de capacidades en que se encuentre sus procesos.

Uno de los propósitos de CMMI es unir en forma coherente varios modelos que eran utilizados en un conjunto dentro de una organización y que generaban repetición de contenido provocando que el proceso de mejora fuera una cosa difícil de implementar y a su vez altamente costoso. El la figura 3 se observa los procesos unificados en CMMI.

Figura 3. Procesos unificados de CMMi



Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software

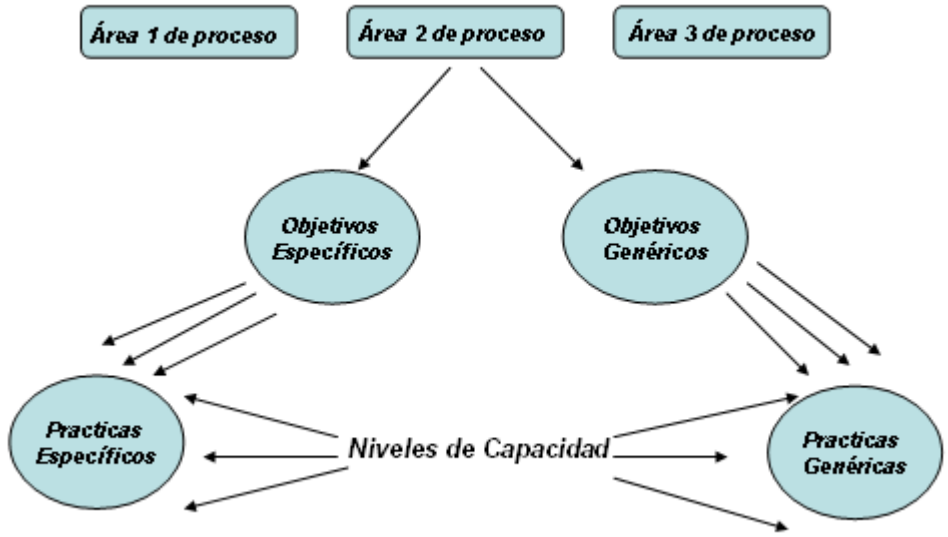
Por otro lado, CMMi tiene dos enfoques: 1) continúo y 2) escalonado. Los dos tienen la finalidad de atender las diversas necesidades de las organizaciones que quieran realizar mejoras en sus procesos. El continuo se caracteriza por concentrarse en ciertas áreas para realizar sus actividades de manera adecuada, y el escalonado por enfatizarse en el grado de madurez de los procesos. Además que ambos enfoques reconocen que las áreas del proceso se pueden agrupar en cuatro categorías: 1) gestión de proyectos, 2) gestión de procesos, 3) ingeniería, y 4) apoyo.

Enfoque continuo: este enfoque utiliza niveles de capacidad para medir el mejoramiento del proceso. Los niveles de capacidad se aplican en la realización del mejoramiento del proceso de la organización para cada área del proceso, ver figura 4. Por otro lado, hace uso de seis componentes del

CMMi:

- **Niveles de Capacidad – Capability Level (1)**: son prácticas genéricas y específicas de un área de proceso que puede mejorar los procesos de la organización asociados a esa área de procesos. A continuación se mencionan los niveles de capacidad: 0) incomplete, 1) performed, 2) managed, 3) defined, 4) quantitatively managed y 5) optimizing.
- **Área de Proceso - Process Area (2):** conjunto de prácticas de un área que satisface un conjunto de objetivos considerados importantes para el mejoramiento del área.
- **Objetivos específicos – Specific Goals (3):** son aquellos que se aplican a un área de proceso y consideran una única característica que describe que debe ser implementado para satisfacer el área de proceso. Por otro lado, ayuda a determinar si un área de trabajo cumple o no los objetivos.
- **Prácticas especificas – Specific Practices (4):** actividad que lleva a cabo un objetivo específico asociado.
- **Objetivos genéricos – Generic Goals (5):** componentes del modelo para determinar si un área de proceso está satisfecha. Cada nivel de capacidad tiene un sólo objetivo genérico.
- **Prácticas genéricas – Generic Practices (6):** son aquellas que están categorizadas por nivel de madurez y aseguran que los procesos asociados a las áreas del proceso serán efectivos, repetibles y duraderos.

Figura 4. Componentes del CMMi 1.1 según el enfoque continuo



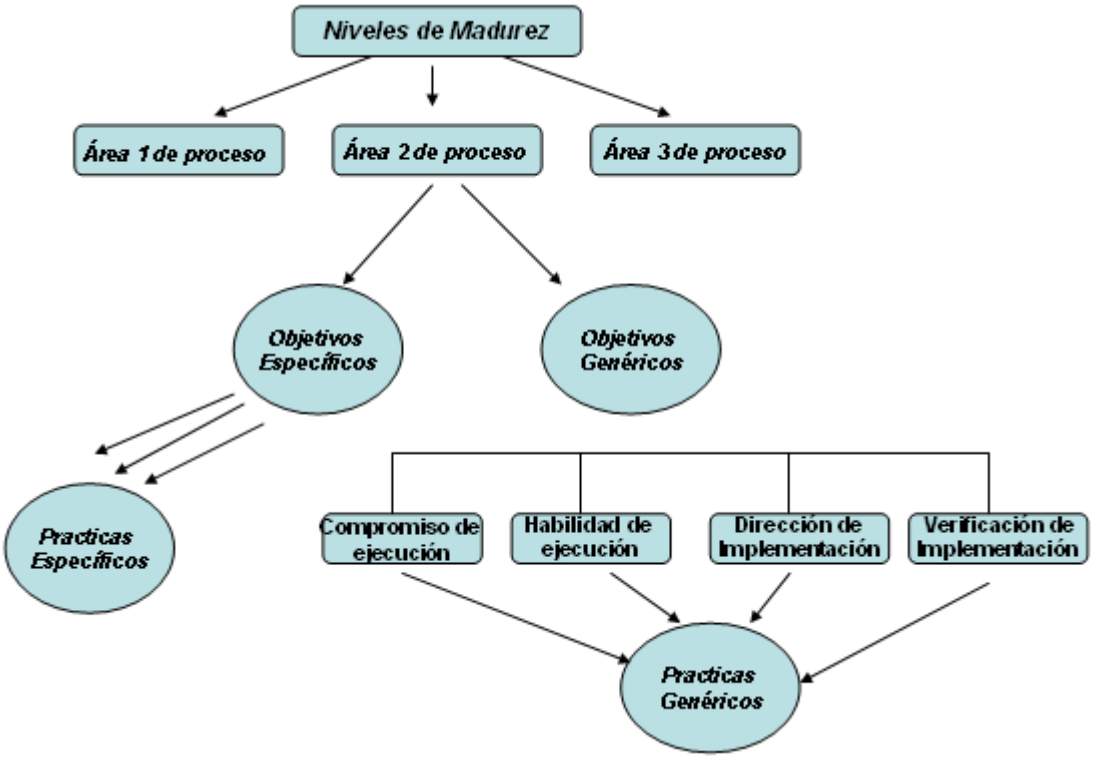
Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software

Enfoque escalonado: este enfoque utiliza niveles de madurez, los cuales se aplican a la madurez de la organización en su conjunto, ver figura 5. Por otro parte, hace uso de siete componentes del CMMi:

- **Niveles de Madurez – Maturity Level (1)**: permite predecir el futuro performance de la organización dentro de una disciplina dada o conjunto de disciplinas. A continuación se mencionan los niveles de madurez: 1) initial, 2) managed, 3) defined, 4) quantitativel, 5) managed y 6) optimizing.
- **Área de Proceso - Process Area (2):** conjunto de prácticas de un área que satisface un conjunto de objetivos considerados importantes para el mejoramiento del área.
- **Objetivos específicos – Specific Goals (3):** son aquellos que se aplican a un área de proceso y consideran una única característica que describe que debe ser implementado para satisfacer el área de proceso. Por otro lado, ayuda a determinar si un área de trabajo cumple o no los objetivos.
- **Prácticas especificas – Specific Practices (4):** actividad que lleva a cabo un objetivo específico asociado.
- **Objetivos genéricos – Generic Goals (5):** componentes del modelo para determinar si un área de proceso está satisfecha. Cada nivel de capacidad tiene un solo objetivo genérico.

- **Common Features Practices (6):** son aquellas que permiten organizar las prácticas de cada área de proceso.
- **Prácticas genéricas – Generic Practices (7):** son aquellas que están categorizadas por nivel de madurez y tienen asociado un objetivo genérico.

Figura 5. Componentes del CMMi 1.1 según el enfoque escalonado



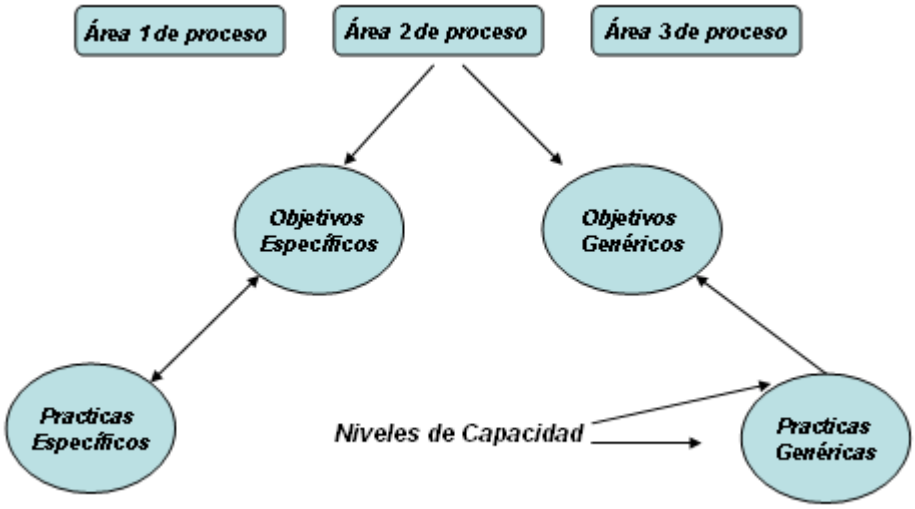
Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software

CMMi v 1.2

El modelo CMMi en la versión 1.2 surgida en el año de 2006 el planteamiento de sus enfoques varia un poco como se aprecia a continuación.

Enfoque continuo: plantea áreas de procesos que poseen objetivos específicos y objetivos genéricos. Los objetivos específicos tienen asociados prácticas específicas y los objetivos genéricos tienen asociados prácticas genéricas. Dichas prácticas genéricas constituyen los niveles de capacidad, observar figura 6.

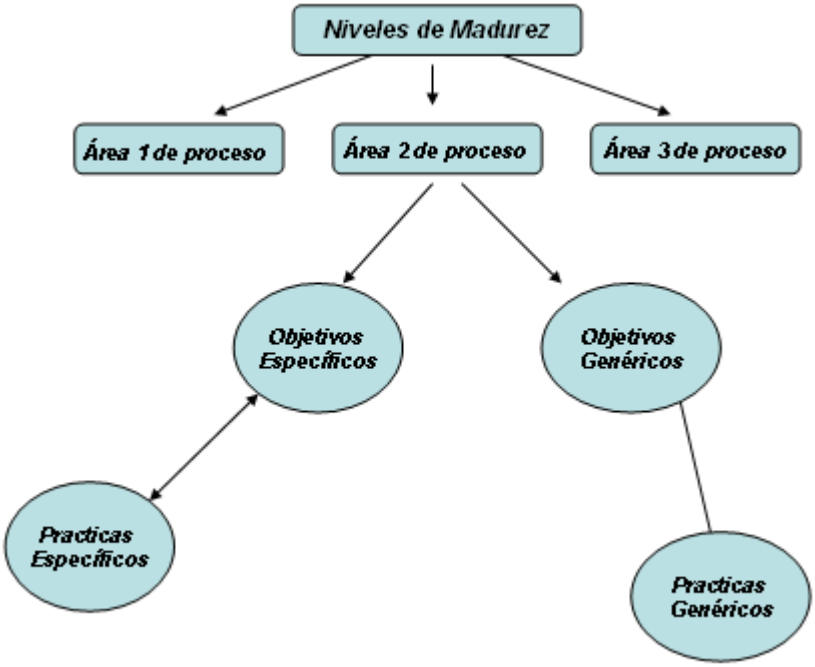
Figura 6. Enfoque continuo de CMMi V1.2



Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software

Enfoque por pasos: plantea niveles de madurez formados por áreas de proceso, las cuales poseen objetivos específicos y objetivos genéricos. Los objetivos específicos tienen asociados prácticas específicas y objetivos genéricos tiene asociados prácticas genéricas, ver figura 7.

Figura 7. Enfoques por pasos de CMMi V 1.2



Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software

1.4.1.2 Modelo Bootstrap.

Este modelo surge a través de un proyecto creado por la Comisión Europea como parte del programa ESPRIT (ESPRIT 5441 BOOTSTRAP: A European Assessment Method to Improve Software Development). Por otra parte, la administración y el mantenimiento del programa Bootstrap corresponde al Grupo Europeo de Interés Económico del Instituto Bootstrap (Bootstrap Institute European Economic Interest Group, su nombre en ingles) de Milán, Italia. Además de hacer parte de un programa estratégico Europeo para la investigación en tecnología de información, tiene como fin la evaluación de procesos de software que reduzcan los costos y mejoren la calidad previendo problemas.

Es una metodología como anteriormente se menciono para la mejora de procesos de software. Esta metodología mide, evalúa, y propone mejoras al proceso de desarrollo de software que siguen las unidades de producción del software de las empresas. Lo anterior lo consigue a través de prácticas, herramientas, y estándares de calidad internacional. Por otra parte, el modelo Bootstrap define el paradigma Organización- Metodología –Tecnología para los niveles de evaluación y agrupación de resultados. Dicho modelo ha sido estructurado en correspondencia con la arquitectura de procesos definida en la ISO 15504 V2.0.

Objetivos de la metodología Bootstrap:

- Proporcionar soporte para la evaluación de la capacidad de los procesos utilizando un conjunto de prácticas de ingeniería de software
- Incluir estándares de ingeniería del software reconocidos internacionalmente como fuentes para la identificación de las prácticas a considerar.
- Dar soporte a la evaluación, indicando como el estándar de referencia ha sido aplicado en la organización evaluada.
- Asegurar la fiabilidad y capacidad de repetición de la evaluación.
- Identificar las fortalezas y debilidades de los procesos de la organización evaluada.
- Dar soporte a la creación y aplicación de un plan de mejora que genere unos resultados aceptables y fiables, de forma que las acciones del plan de mejora permitan alcanzar los objetivos de la organización.
- Ayudar a incrementar la eficacia de los procesos poniendo en práctica los requisitos del estándar en la organización.

Es de mencionar que el enfoque de esta metodología es evaluar el proceso, no el producto. Por tanto, define un conjunto de características para el proceso, provee un análisis cuantitativo, produce vistas analíticas, evidencia fortalezas y debilidades, identifica áreas de mejora, provee recomendaciones y sugiere un plan de implementación.

1.4.1.3 Personal software process (PSP).

El PSP es un proceso del software definido para ser usado de forma individual, teniendo como objetivo guiar el planeamiento y desarrollo de los módulos de software o pequeños programas. Por otra parte, PSP se basa en los principios de mejoramiento del proceso para fomentar el mejoramiento a nivel personal, ofreciendo la administración y control del proceso al ingeniero de software para desarrollar software de forma estructurada y disciplinada. Por consiguiente, los ingenieros se ocupan: 1) seguir un proceso definido, 2) planificar, medir, y seguir su trabajo, 3) administrar la calidad del producto, y 4) aplicar aspectos cuantitativos para mejorar los procesos de trabajo personales.

Por otro lado, PSP lo conforman un conjunto de métodos, formularios y scripts que ayudan a los profesionales del software a cómo planificar, medir y administra su trabajo, diseñado para ser usado en cualquier lenguaje de programación o metodología de diseño.

1.4.1.4 Team software process (TSP).

Su objetivo es proporcionar un proceso operacional que ayude a los ingenieros hacer trabajos de calidad. Su motivación para ser implementado corresponde a que el equipo de ingenieros pueda realizar su trabajo de manera extraordinaria, siempre que sean formados y entrenados. Los objetivos de TSP son: 1) ayudar a los equipos de ingeniería de software a elaborar productos de calidad dentro de los costos y tiempos establecidos, 2) tener equipos confiables y rápidos, y 3) optimizar el rendimiento del equipo durante todo el proyecto.

Un aspecto a tener en cuenta para implementar TSP corresponde a que todos los miembros del equipo deben haber sido entrenados en PSP. Debido a que con PSP los desarrolladores alcanzan ciertas competencias determinantes para la utilización de TSP.

Por otra parte, TSP es una guía para ingenieros y gerentes para el uso de métodos que hace mucho más efectivo el trabajo de equipo, mejorando el rendimiento individual que a su vez impacta en la mejora del trabajo

colectivo. Un equipo para que sea efectivo debe tener como principio que el trabajo de todas sus unidades debe ser cohesivo. Por tanto, TSP proporciona las condiciones necesarias para conseguir que los equipos sean efectivos a través de procesos operacionales que permiten que sea más exigentes para alcanzar sus planes por medio del trabajo disciplinado. Finalmente, TSP comprende una serie de métodos que pueden ayudar al los ingenieros a desarrollar sistemas.

1.4.1.5 Practical software measurement (PSM).

Los grupos de software deben colocar mucha atención con respecto a su capacidades y al desarrollo de productos / servicios, por lo tanto, una forma de hacerlo es a través de los procesos de medición efectivos. Es la medición la herramienta que ayuda al proceso para detectar tendencias, anticipar problemas e identificar tendencias que pueden traer consecuencias en los procesos, mejorar el control de costos, reducir el riesgo, mejorar la calidad, y asegurar que se cumplan los objetivos del negocio. Asimismo, la medición es un proceso sistemático flexible que se aplica a la ingeniería de sistemas, al software, y a las actividades de administración. Es de mencionar que el proceso de medición puede ser adoptado a las necesidades particulares de cada proyecto.

El objetivo básico de PMS es proveer a los gerentes de proyecto de información cuantitativa vital para toma de decisiones, y su efecto se verá reflejado en los costos y el rendimiento de los objetivos técnicos y de programación. PMS se relaciona sobre manera con la norma ISO/IEC 15939 y con CMMI.

1.4.1.6 Six sigma for software.

Su finalidad es reducir los costos a través de la eliminación de defectos y la mejora de procesos aplicando los siguientes tres principios: 1) enfoque al cliente, 2) proceso de orientación, y 3) liderazgo en métricas. Por otra parte, Six Sigma es una filosofía, una métrica, y una estructura de mejoramiento. Esta filosofía aplica al dominio del software y la tecnología para lograr la satisfacción del cliente con productos novedosos y a un precio altamente competitivo. Además que hace gestión para evitar errores, y pérdidas innecesarias que conlleven repetir trabajo.

1.4.2 Estándares de calidad del software a nivel de proceso.

1.4.2.1 ISO 90003:2004.

Proporciona una guía para las organizaciones en relación a la aplicación de ISO/IEC 9001:2000 en la adquisición, suministro, desarrollo, operación y mantenimiento de software y servicios de soporte. Por otro lado, no cambia o agrega requerimientos de ISO/IEC 9001:2000.

La aplicación de ISO 90003:2004 es apropiada para un software que: 1) forme parte de un contrato comercial con otra organización, 2) es un producto disponible para el mercado, 3) es usado para soportar los procesos de una organización, y 4) está relacionado a servicios de software. Además es independiente de la tecnología, modelos del ciclo de vida, procesos de desarrollo, secuencias de actividades y estructura organizacional.

1.4.2.2 ISO/12207: 1995.

Proporciona un marco común que cubre todo el ciclo de vida del software desde la conceptualización hasta su retiro, consistiendo en procesos para adquirir y suministrar productos y servicios de software. Este marco controla y mejora los procesos.

Por otra parte, puede ser usado para: 1) suministrar, desarrollar, adquirir, operar, y mantener el software, 2) soportar las anteriores funciones a través del aseguramiento de calidad, administración de la configuración, revisiones conjuntas, auditorias, verificación, validación, resolución de problemas y documentación, 3) administrar y mejorar tanto el personal como los procesos de la organización, 4) establecer la administración del software y los ambientes de la ingeniería basados en los procesos del ciclo de vida que se adapten para servir a las necesidades del negocio, 5) ayudar a mejorar el entendimiento entre clientes , proveedores, y las partes involucradas en el ciclo de vida de un producto de software, y 6) facilitar la comercialización global de software.

Es también es utilizado para las actividades, procesos y tareas de adquisición de un sistema que contenga software, un producto software o un servicio de software. Por otra parte, este estándar está dirigido para los consumidores de sistemas, productores de software, y para los suministradores de software, desarrolladores, operadores, mantenedores, administradores, responsables de la calidad de software y usuarios de productos software. Además agrupa las actividades que deben ser realizadas en el del ciclo de vida del software en: 5

procesos principales, 8 procesos de soporte y 4 procesos organizacionales.

1.4.2.3 ISO/IEC 15504 – SPICE.

Su propósito para el que fue creado en el año de 1993 fue para la evaluación del proceso de software consistiendo básicamente en examinar el proceso que utilizan las organizaciones. Lo componen el objetivo: 1) describir los métodos que las organizaciones utilizan en la actualidad, señalando las fortalezas, las debilidades y los riesgos inherentes del proceso, 2) determinar en qué medida son eficaces para lograr las metas del proceso, y 3) determinar en qué medida forma un conjunto de métodos como punto de partida.

Spice (Software Process Improvement and Capability Determination, sus siglas en ingles) lo pueden usar organizaciones interesadas por la planificación, manejo, monitorización, control y mejora de la adquisición, suministro, desarrollo, operación y soporte de software. Además, que este modelo fue pensado como una iniciativa internacional para cubrir los métodos, prácticas y aplicaciones de valoración de procesos de adquisición, desarrollo, entrega, operación, evaluación y servicios de productos de software. Por lo tanto, proporciona un marco de referencia para la valoración de procesos de software que fomenta la calidad de los productos software y genera un proceso de valoración repetible, comparable y verificable.

1.4.2.4 ISO 20000:2005.

Permite que las organizaciones puedan mejorar su capacidad en estos tres aspectos: 1) entrega de los servicios administrados, 2) medir los niveles de servicios y 3) evaluar el rendimiento. Por otro lado, permite que los proveedores del servicio entiendan cómo aumentar la calidad del servicio entregado a los clientes internos y externos.

Además permitirá reducir el riesgo, cumplir requerimientos, y demostrar la calidad del servicio.

1.4.2.5 ITIL – Information technology infrastructure library.

Proporciona un conjunto de las mejores prácticas, extraídas de organismos punteros del sector público y privado a nivel internacional, que han sido recogidas por OGC (Office of Government Commerce). Es de resaltar que este

marco de trabajo esta siendo utilizado por cientos de organizaciones en el mundo y ha sido desarrollado reconociendo la dependencia creciente que tienen éstas en la tecnología para alcanzar sus objetivos.

Alinear el negocio con los sistemas de información es su objetivo principal. Por otra parte, es un conjunto de buenas prácticas de dirección y gestión de servicios de tecnologías de la información en lo referente a personas, procesos y tecnología, desarrollada por la OGC que cumple y desarrolla la norma BSC15000 (Bristh Standards Institution). La realización de las buenas prácticas especificadas en ITIL hace posible que los departamentos y organizaciones puedan reducir costos y mejorar la calidad del servicio tanto de clientes externos como de internos y aprovechar a lo máximo las habilidades y experiencia del personal, mejorando su productividad.

1.4.2.6 COBIT 4.0.

Está fundamentado en la alineación de los objetivos del negocio con los objetivos de tecnología de información (TI), proporcionando métricas y modelos de madurez para medir sus resultados, e identificar las responsabilidades asociadas al negocio y los responsables de TI.

Su funcionamiento es como un marco para la gestión TI de la organización dado que: 1) provee la información que la empresa requiere para lograr sus objetivos, 2) mediante la gestión y control de los recursos de TI utiliza una estructura de procesos para garantizar la entrega de los servicios de información requeridos.

1.4.3 Modelos de calidad de software a nivel de producto.

1.4.3.1 Modelo del Gilb.

El analista y el usuario juegan un papel muy importante debido a que ellos conjuntamente para este modelo deben especificar los requisitos de calidad para cada proyecto. Por otra parte, el modelo permite determinar una lista de características que definen la calidad de la aplicación. Las cuales pueden ser de dos tipos: 1) originales y 2) de modelos tradicionales. Por lo tanto, se caracteriza por su flexibilidad.

Es de mencionar que las características son medidas a través de varias subcaracterísticas o métricas. Además, que cada una de ellas debe tener los siguientes aspectos: 1) nombre y definición de la característica, 2) escala o

unidades de medición, 3) recopilación de datos o prueba, 4) valor previsto, 5) valor óptimo, 6) valor del sistema actual, y 7) comentarios.

1.4.3.2 Modelo GQM (Goal – Question –Metric).

El modelo GQM (Goal Question Metric, sus siglas en ingles) fue propuesto en el año de 1998 por Basili y Rombach, y se característica por su enfoque de objetivos y metas orientado a la definición de modelos de calidad. Esta propuesta se orienta en evaluar la calidad de cada proyecto. Su forma de implementación se basa en definir un modelo de calidad hasta obtener métricas respectivas con el análisis e interpretación de los datos correspondientes a las mediciones. De esta forma, se estipula un enfoque de medición para evaluar la calidad del software basado en la identificación de los objetivos a lograr.

GQM con la base de tener una mejora en la definición clara de procesos y productos, proporciona una estructura para obtener los objetivos cruciales del proyecto que consta de tres etapas: 1) listar objetivos principales correspondientes al desarrollo y mantenimiento, 2) cuestionar si está cumpliendo cada objetivo, y 3) decidir que medir para poder contestar las preguntas de manera adecuada, es decir, especificar un conjunto de métricas que permitan responder la pregunta.

1.4.3.3 Modelo McCall.

Es de mencionar que hace 25 años se definieron 25 factores de calidad como iniciativa hacia el desarrollo de métricas de calidad de software. El modelo McCall se enfoca en organizar los factores en tres ejes o puntos de vista de los cuales el usuario puede contemplar la calidad del producto, en relación a los siguientes puntos: 1) operación del producto, 2) revisión del producto, y 3) transición del producto.

Por otra parte, cada punto de vista se descompone en una serie de factores que determinan la calidad de cada uno de ellos, y a su vez cada factor determinante se decompone en una serie de criterios o propiedades que determinan su calidad, ver tabla 3. Dichos criterios son evaluados por un conjunto de métricas. Su forma de evaluación corresponde a que para cada criterio se debe fijar un valor máximo y un valor mínimo aceptable.

Tabla 3. Puntos de vistas o ejes de factores de calidad según el modelo MacCall

Puntos de vista o ejes	Factor	Criterios
OPERACIÓN DEL PRODUCTO	Facilidad de uso	- Facilidad de operación: Atributos del software que determinan la facilidad de operación del software. - Facilidad de comunicación: Atributos del software que proporcionan entradas y salidas fácilmente asimilables. - Facilidad de aprendizaje: Atributos del software que facilitan la familiarización inicial del usuario con el software y la transición del modo actual de operación. - Formación: el grado en que el software ayuda para permitir que nuevos usuarios apliquen el sistema.

	Corrección	- Control de accesos. Atributos del software que proporcionan control de acceso al software y los datos que maneja. - Facilidad de auditoría: Atributos del software que facilitan la auditoría de los accesos al software. - Seguridad: la disponibilidad de mecanismos que controlen o protejan los programas o los datos.

Tabla 3. (Continuación)

Puntos de vista o ejes	Factor	Criterios
OPERACIÓN DEL PRODUCTO	Integridad	Completitud: atributos del software que proporcionan la implementación completa de todas las funciones requeridas. - Consistencia: atributos del software que proporcionan uniformidad en las técnicas y notaciones de diseño e implementación. - Trazabilidad o rastreabilidad: atributos del software que proporcionan una traza desde los requisitos a la implementación con respecto a un entorno operativo concreto.

	Fiabilidad	- Precisión: atributos del software que proporcionan el grado de precisión requerido en los cálculos y los resultados. - Consistencia. - Tolerancia a fallos: atributos del software que posibilitan la continuidad del funcionamiento bajo condiciones no usuales. - Modularidad: atributos del software que proporcionan una estructura de módulos altamente independientes. - Simplicidad: atributos del software que posibilitan la implementación de funciones de la forma más comprensible posible. - Exactitud: la precisión de los cálculos y del control.

Tabla 3. (Continuación)

Puntos de vista o ejes	Factor	Criterios
OPERACIÓN DEL PRODUCTO	Eficiencia	- Eficiencia en ejecución: Atributos del software que minimizan el tiempo de procesamiento. - Eficiencia en almacenamiento: atributos del software que minimizan el espacio de almacenamiento necesario.

REVISION DEL PRODUCTO	Facilidad de mantenimiento	- Modularidad. - Simplicidad. - Consistencia. - Concisión: atributos del software que posibilitan la implementación de una función con la menor cantidad de códigos posible. - Auto descripción: atributos del software que proporcionan explicaciones sobre la implementación de las funciones.
	Facilidad de prueba	- Modularidad. - Simplicidad. - Auto descripción. - Instrumentación: atributos del software que posibilitan la observación del comportamiento del software durante su ejecución para facilitar las mediciones del uso o la identificación de errores.

Tabla 3. (Continuación)

Puntos de vista o ejes	Factor	Criterios

REVISION DEL PRODUCTO	Flexibilidad	- Auto descripción. - Capacidad de expansión: atributos del software que posibilitan la expansión del software en cuanto a capacidades funcionales y datos. - Generalidad: atributos del software que proporcionan amplitud a las funciones implementadas. - Modularidad.
	Reusabilidad	- Auto descripción. - Generalidad. - Modularidad. -Independencia entre sistema y software: atributos del software que determinan su dependencia del entorno operativo. - Independencia del hardware: atributos del software que determinan su dependencia del hardware.

Tabla 3. (Continuación)

Puntos de vista o ejes	Factor	Criterios

REVISION DEL PRODUCTO	Flexibilidad	- Auto descripción. - Capacidad de expansión: atributos del software que posibilitan la expansión del software en cuanto a capacidades funcionales y datos. - Generalidad: atributos del software que proporcionan amplitud a las funciones implementadas. - Modularidad.
	Interoperabilidad	- Modularidad. - Compatibilidad de comunicaciones: atributos del software que posibilitan el uso de protocolos de comunicación e interfaces estándar. - Compatibilidad de datos: atributos del software que posibilitan el uso representaciones de datos estándar. - Estandarización en los datos: el uso de estructuras de datos y de tipos estándar a lo largo de todo el programa.
	Portabilidad	- Auto descripción. - Modularidad. -Independencia entre sistema y software. - Independencia del hardware.

Fuente: Cervera Paz, Ángel. El modelo de mcCall como aplicación de la calidad a la revisión del software de gestión empresarial.

1.4.3.4 Modelo FURPS.

Hewllet-Packard en el año de 1987, realizan la propuesta de un conjunto de factores de calidad del software que fue denominado con el acrónimo de FURPS: funcionalidad, facilidad de uso, fiabilidad, rendimiento y capacidad de soporte.

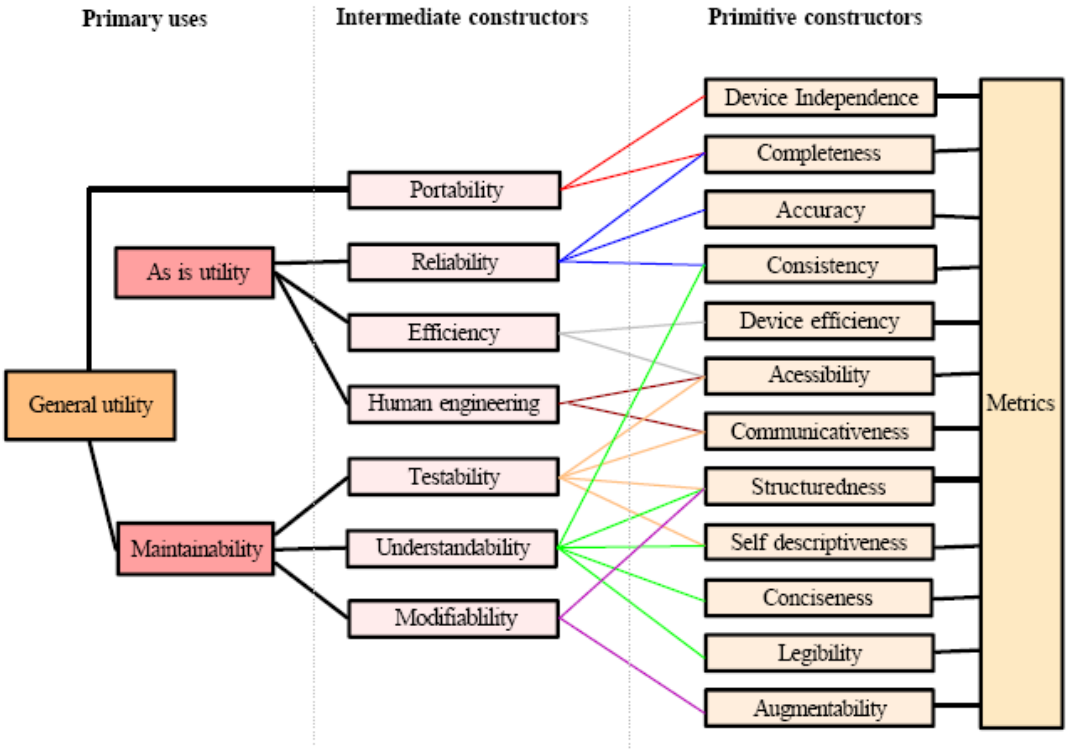
- **Funcionalidad**: se valora evaluando el conjunto de características y capacidades del programa, la generalidad de las funciones entregadas y la seguridad del sistema global.
- **Facilidad de uso**: se valora considerando los factores humanos, la estética, la consistencia y la documentación general.
- **Fiabilidad**: se evalúa midiendo la frecuencia y gravedad de fallos, la exactitud de las salidas, el tiempo de fallos, la capacidad de recuperación de un fallo y la capacidad de predicción del programa.
- **Rendimiento**: se mide por la velocidad de procesamiento, el tiempo de respuesta, consumo de recursos, rendimiento efectivo y eficacia.
- **Capacidad de soporte**: combina la capacidad de ampliar el programa, adaptabilidad y servicios, así como la capacidad de hacer pruebas, compatibilidad, capacidad de configuración, facilidad de instalación y la facilidad con que se pueda detectar errores.

Los factores FURPS y sus características pueden ser usados para crear métricas de calidad para ser utilizadas durante todo el proceso de software.

1.4.3.5 Modelo de BOEHM.

El modelo Boehm, describe la calidad del software a nivel de producto, es publicado en el año de 1978, y se caracteriza por tener un enfoque de descomposición top-down. Es de anotar que es muy similar al modelo McCall, a diferencia que incluye características de rendimiento de hardware. En la figura. 8 se puede apreciar la visión de modelo de calidad según Boehm.

Figura 8. Modelo de calidad de software Boehm



Fuente: Olsina, Luís Antonio. Metodología Cuantitativa para la Evaluación y Comparación de la Calidad de Sitios Web.

1.4.3.6 Modelo SACT (Software Assurance Technology Center).

Se caracteriza por ser un modelo dinámico que permite el desarrollo de varios proyectos al mismo tiempo. Por otra parte, se basa en proyecciones para identificar los riesgos y puntos de control de los proyectos, realizándolo a través de sus datos. Además, utiliza medidas o métricas que se relacionan con el proceso de desarrollo y con el producto. Este modelo trabaja con la definición de metas u objetivos relacionados al proceso y al producto de software con el fin de obtener indicaciones de probabilidad de éxito de los objetivos. A continuación en la tabla 4 se muestra las cuatro metas u objetivos con sus respectivos atributos y métricas.

Tabla 4. Tabla 4. Metas, atributos y métricas del modelo SACT.

Meta	Atributos	Métricas

Calidad de los Requerimientos	Ambigüedad. Integridad. Facilidad de entender. Volatilidad del requerimiento. Trazabilidad.	Nro de frases claras Nro de frases opcionales. Nro de TBDs/TBAs. Estructura del documento. Cantidad de cambios / cantidad de requerimientos. Etapa del ciclo de vida cuando se realiza un cambio. Nro de requerimientos del software que no se ajustan a los requerimientos del sistema Nro de requerimientos del software que no se ajustan al código y a las pruebas.
Calidad del Producto	Estructura / arquitectura. Facilidad de mantenimiento. Reusabilidad Documentación interna Documentación externa	Complejidad lógica uso del goto Tamaño Correlación de complejidad / tamaño Porcentaje de comentarios Índice legible
Efectividad de la implementación	Uso de los recursos Porcentaje de terminación	Horas staff dedicadas a las actividades del ciclo de vida Tareas terminadas, tareas terminadas planificadas.
Efectividad de la prueba	Corrección	Errores y criticidad, tiempo de encuentro de errores y tiempo de errores fijos. Ubicación del código de falla.

Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software

1.4.3.7 Modelo Dromey.

Dromey modelo de calidad del producto, propuesto en el año de 1995. Evalúa la calidad del producto en función de estándares de código, clasificación de

defectos, y el desarrollo de herramientas de auditoría. Asimismo resalta que la calidad del producto es altamente determinada por los componentes del mismo (incluyendo documentos de requerimientos, guías de usuarios, diseños, y código), las propiedades tangibles de los componentes y las propiedades tangibles de la composición de los componentes.

Por otra parte, clasifica las cualidades tangibles utilizando cuatro propiedades.

Correctitud: pueden ser internas (asociadas con los componentes individuales) o contextuales (asociadas con la manera en que los componentes son utilizados en el contexto).

Internas: miden que tan bien un componente ha sido entregado de acuerdo a su objetivo, implementación o que tan bien ha sido compuesto.

Contextuales: cómo los componentes son compuestos y las influencias que ejercen sobre la calidad del producto.

Descriptivas: para ser útil un software debe ser fácil de entender y de utilizar de acuerdo a su propósito. Estas propiedades descriptivas aplican a requerimientos, diseños, implementación y a las interfaces de usuario.

1.4.3.8 Web-site QEM (Web Site Quality Evaluation Method).

Es un modelo diseñado específicamente para evaluación y selección de aplicaciones Web con base al dominio y al perfil de usuario. Por lo tanto, se pueden evaluar aplicativos Web de diferentes dominios tales como, comercio electrónico, sistemas académico, financieros, entre otros. Por otro lado, la complejidad de la evaluación del producto se relaciona con respecto al número de características y atributos que puedan intervenir en los requerimientos de calidad y en las varias relaciones existentes entre atributos y subcaracterísticas.

Web-site QEM es una propuesta de tesis doctoral de Luis Olsina. En ella se plantea 4 características de calidad con sus respectivas subcaracterísticas y atributos, ver figura 9. Además, la componen las siguientes fases: 1) planificación y programación de la evaluación de calidad, 2) definición y especificación de requerimientos de calidad, 3) definición e implementación de la evaluación elemental, 4) definición e implementación de la evaluación global, y 5) análisis de resultados, conclusiones y documentación. Es de resaltar que según el alcance que se le quiera dar a la evaluación no todos los atributos son evaluados para cada evaluación.

Figura 9. Características, subcaracterísticas y atributos del modelo Web-site QEM.

1. Facilidad de Uso
1.1 Comprensibilidad Global del Sitio
1.1.1 Esquema de Organización Global
1.1.2 Calidad en el Sistema de Etiquetado
1.1.3 Visita Guiada Orientada al Estudiante
1.1.4 Mapa de Imagen (Campus/Edificio)
1.2 Mecanismos de Ayuda y Retroalimentación en línea
1.2.1 Calidad de la Ayuda
1.2.2 Indicador de Última Actualización
1.2.2.1 Global (de todo el sitio Web)
1.2.3 Directorio de Direcciones
1.2.4 Facilidad FAQ
1.2.5 Retroalimentación
1.3 Aspectos de Interfaces y Estéticos
1.3.1 Cohesividad al Agrupar los Objetos de Control Principales
1.3.2 Permanencia y Estabilidad en la Presentación de los Controles Principales
1.3.3 Aspectos de Estilo
1.3.4 Preferencia Estética
1.4 Misceláneas
1.4.1 Soporte a Lenguaje Extranjero
1.4.2 Atributo "Qué es lo Nuevo"
1.4.3 Indicador de Resolución de Pantalla
2. Funcionalidad
2.1 Aspectos de Búsqueda y Recuperación
2.1.1 Mecanismo de Búsqueda en el Sitio Web
2.1.2 Mecanismos de Recuperación
2.2 Aspectos de Navegación y Exploración
2.2.1 Navegabilidad
2.2.2 Objetos de Control Navegacional
2.2.3 Predicción Navegacional
2.3 Aspectos del Dominio orientados al Estudiante
2.3.1 Relevancia de Contenido
2.3.2 Servicios On- line
3. Confiabilidad
3.1 No Deficiencia
3.1.1 Errores de Enlaces
3.1.2 Errores o Deficiencias Varias
4. Eficiencia
4.1 Performance
4.1.1 Páginas de Acceso Rápido
4.2 Accesibilidad
4.2.1 Accesibilidad de Información
4.2.2 Accesibilidad de Ventanas

Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software.

1.4.4 Estándares de calidad del software a nivel de producto.

1.4.4.1 ISO/IEC 9126-1:2001 – Quality Model.

Este estándar evalúa la calidad del producto y esta especificado por 6 características de calidad interna y externa, las cuales se dividen en subcaracterísticas y se manifiestan cuando el software es utilizado como parte de un sistema, y son resultado de los atributos internos del software. La calidad interna se centra en que el software satisfaga las necesidades del usuario con base en las condiciones especificadas. La calidad externa evalúa el total de los atributos que un software deba tener con respecto a las condiciones especificadas. Es de mencionar que las características definidas son aplicables a cualquier tipo de software.

Por otro lado, este estándar permite especificar y evaluar la calidad del software de distintas perspectivas que están asociadas a la adquisición, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad, y auditoria del software.

La estructura del modelo de calidad ISO 9126-1 es de la siguiente forma. Tiene tres niveles: 1) características, 2) subcaracterísticas y 3) métricas. Métricas internas y externas. Las internas pueden ser aplicables a un software no ejecutable durante el diseño y la codificación. Las métricas externas se utilizan en el software ejecutable.

El modelo de calidad interna y externa está formado por las siguientes características:1) funcionalidad, 2) confiabilidad, 3) facilidad de uso, 4) eficiencia, 5) facilidad de mantenimiento y 6) portabilidad, ver la tabla 5.

Tabla 5. Características y subcaracterísticas ISO/IEC 9126

Característica	Subcaracterística
Funcionalidad	Adecuación
	Exactitud

	Interoperabilidad
	Conformidad
	Seguridad de Acceso
Confiabilidad	Nivel de Madurez
	Tolerancia a fallas
	Recuperabilidad
Usabilidad	Comprensibilidad
	Facilidad de Aprender
	Operabilidad
Eficiencia	Comportamiento con respecto al Tiempo
	Comportamiento con respecto a Recursos
Mantenibilidad	Analizabilidad
	Modificabilidad
	Testeabilidad
Portabilidad	Adaptabilidad
	Instalabilidad
	Conformidad
	Reemplazabilidad

Fuente: OLSINA, Luis Antonio. Metodología Cuantitativa para la Evaluación y Comparación de la Calidad de Sitos Web

1.4.4.2 ISO/IEC 25000:2005 – SQuaRE.

SQuaRE (Software Quality Requirements and Evaluation, sus siglas en ingles) pertenece a una nueva serie de normas que con base en ISO 9126 y en ISO 14598. Su principal objetivo es la coordinación y armonización del contenido ISO 9126 y ISO 15939:2002. Por otro lado, SQuaRE incluye un estándar de requerimientos de calidad que se compone por 14 documentos agrupados en 5 tópicos: 1) administración de calidad, 2) modelo de calidad, 3) medidas de calidad, 4) requerimientos de calidad y 5) evaluación de calidad.

- **Administración de la calidad:** tiene dos componentes: 1) guía para Squire – Overview de la estructura y terminología, 2) planificación y administración, provee una guía para planificar y administrar las evaluaciones del software.
- **Modelo de calidad:** describe el modelo de calidad interno / externo y la capacidad en uso. Presenta características y

subcaracterísticas.

- **Medidas de calidad:** mediciones primitivas, medidas para la calidad interna, medidas para la calidad externa y medidas para la calidad en uso.
- **Requerimientos de calidad:** permite habilitar la calidad del software a ser especificado en términos de requerimientos de calidad durante todo el ciclo de vida de un proyecto de software, mantenimiento, y operación.
- **Evaluación de la calidad:** evaluación de la calidad, proceso para desarrolladores, proceso para compradores, proceso para evaluadores y documentación del módulo de evaluación.

1.4.4.3 IEEE-std 1061-1998: Standard for software quality metrics methodology.

Es una metodología que establece los requerimientos de calidad e identifica, implementa, analiza, y valida los procesos y las métricas de calidad del producto que son definidas. La metodología se extiende a todo el ciclo de vida del software. En la figura 10 se puede observar la estructura metodológica de este estándar.

Figura 10. Estructura metodológica de IEEE-std 1061-1998

Contenido

Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software.

1.4.5 Metodologías para la valorar el proceso del software libre.

1.4.5.1 Open source maturity model - Capgemini (OSMM).

Es un modelo para determinar la madurez de software libre, permitiéndolo hacer en corto tiempo y en pocos pasos. La estructura metodológica está compuesta por tres pasos: 1) determinar la madurez de los productos software evaluados, 2) comparar los productos software evaluados a través de los criterios unificados y objetivos, y 3) seleccionar el producto de software libre más apropiado para la organización. En la figura 11 pueden observarse sus 12 criterios de evaluación.

Figura 11. Criterios de evaluación de Capgemini (OSMM).

- Producto
  - Edad
  - Licencia
  - Organización humana
  - Comunidad de desarrolladores
  - Aspectos de ventas
- Integración
  - Modularidad
  - Interoperabilidad
  - Estándares
- Uso
  - Soporte
  - Fácil de implementar
- Adopción
  - Usuarios en la comunidad
  - Penetración de mercado

Fuente: http://www.osspartner.com/portail/sections/accueil-public/evaluation-osmm

1.4.5.2 Open source maturity model – Navica (OSMM).

El OSMM (Open source maturity model, sus siglas en ingles) fue desarrollado para ayudar a los gerentes en la adquisición de software libre para la comparación y evaluación. Esta compuesto por las siguientes fases.

- Primera fase: seleccionar software a evaluar.
- Segunda fase: asignación de factores de ponderación.
- Tercera fase: asignación de puntuación
- Cuarta fase: calculo de ponderación final de cada categoría.

1.4.5.3 Qualification and selection of open source software (QSOS).

QSOS (Qualification and selection of open source software, sus siglas en ingles) es una metodología para la valoración de software libre / open source. La metodología está protegida bajo la licencia libre GFDL (GNU Free Documentation Lincese, sus siglas en ingles). Su estructura metodológica está compuesta por 4 etapas: 1) definición, 2) evaluación, 3) clasificación, y 4) selección.

- **Definición:** definición y organización de los aspectos a evaluar: criterios y riesgos comunes del open source, y dominio técnico de funcionalidades específicas.
- **Evaluación:** evaluar los productos software con respecto a los criterios definidos anteriormente y establecer la puntación individual.
- **Clasificación:** clasificar la evaluación a través de los ejes de criterios de evaluación y el filtro de definición, relacionados con el contexto.
- **Selección:** selección de el software de open source más adecuado en relación al sistema de clasificación realizado en el paso 3.

La documentación que se genera en el proceso corresponde tanto a hojas de cálculo y grillas de comparación. Estas grillas de comparación pueden ser de mucha utilidad para la selección dependiendo el contexto.

El uso de esta metodología es apoyada con el uso de varias herramientas software que están bajo licencia GLP (General Public License, sus siglas en ingles). Dichas herramientas son: 1) plantilla de edición: QSOS XUL y 2) editores de hojas de calculo de valoración: QSOS QT y QSOS Java Editor.

1.5.6 Cuadro comparativo de modelos y estándares de calidad a nivel del producto.

A partir de los modelos y estándares de calidad definidos, las características planteadas en los diferentes modelos se presenta un cuadro comparativo, ver la tabla 6. Este es un análisis de Scalone y para este trabajo se le adiciono las características del modelo QSOS.

Tabla 6. Cuadro comparativo de modelos y estándares de calidad a nivel del producto.

Características Calidad	***B***	***D***	***MC***	***F***	***S***	***C***	***W***	***ISO***	***Q***
Facilidad de uso		X	X	X			X	X	
Integridad			X		X				
Corrección			X		X				
Confiabilidad	X	X	X	X			X	X	
Eficiencia	X	X	X				X	X	
Facilidad de mantenimiento		X	X		X	X		X	X
Facilidad de prueba	X		X						
Flexibilidad			X						
Facilidad de reutilización			X		X	X			
Interoperabilidad			X						X
Portabilidad	X	X	X					X	
Ingeniería humana	X								
Fácil de entender	X				X				
Fácil de modificar	X								X
Funcionalidad		X		X		X	X	X	
Performance				X					
Facilidad del soporte				X					X
Ambigüedad					X				
Trazabilidad					X				
Estructura/Arquitectura					X				
Documentación					X	X			X
Conformidad									
Aplicaciones Web							X		
Madurez									X
Adopción									X
Liderazgo en desarrollo									X
Actividad									X
Independencia de desarrollo									X
Servicios									X
Aseguramiento de Calidad									X
Empaquetado									X
Modularidad									X
Licencia									X
Derechos de autor									X
Modificación del código fuente									X
Estado actual de desarrollo del software									X
Patrocinador									X
Independencia Estratégica									X
Dominio del código									X

Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software.

B: Modelo de Boehm ISO: ISOIEC 9126-1
D: Modelo de Dromey F: Modelo FURPS
MC: Modelo McCall Q:QSOS
S: Modelo SATC…….. W: WebQEM
C: Modelo C-QM

Tabla 7. Concurrencias de características de calidad de los modelos y estándares comparados en la tabla 6.

Características Calidad	***Ocurrencias***
Facilidad de mantenimiento	6
Confiabilidad	6
Facilidad de uso	5
Eficiencia	5
Funcionalidad	5
Portabilidad	4
Facilidad de reutilización	3
Documentación	3
Integridad	2
Corrección	2
Facilidad de prueba	2
Interoperabilidad	2
Fácil de entender	2
Fácil de modificar	2
Facilidad del soporte	2
Flexibilidad	1
Ingeniería humana	1
Performance	1
Ambigüedad	1
Trazabilidad	1
Estructura/Arquitectura	1
Aplicaciones Web	1
Madurez	1
Adopción	1
Liderazgo en desarrollo	1
Actividad	1
Independencia de desarrollo	1
Servicios	1
Aseguramiento de Calidad	1
Empaquetado	1
Modularidad	1
Licencia	1
Derechos de autor	1
Modificación del código fuente	1
Estado actual de desarrollo del software	1
Patrocinador	1

Independencia Estratégica	1
Dominio del código	1
Conformidad	0

En una primera instancia se puede observar que las primeras 6 características (facilidad de mantenimiento, confiabilidad, facilidad de uso, eficiencia, funcionalidad y portabilidad) coincide con las características planteadas en ISO/IEC 9126-1:2001. Por otra parte, las características : madurez, adopción, liderazgo en desarrollo, actividad, independencia de desarrollo, servicios, aseguramiento de calidad, empaquetado, modularidad, licencia, derechos de autor, modificación del código fuente, estado actual de desarrollo del software, patrocinador, independencia estratégica y dominio del código son características que sólo están presentes en el modelo QSOS y que no lo están en los demás modelos, esto debido a que son exclusivos para la evaluación de la calidad de aplicaciones de software libre. Por otra parte, hay características como facilidad de mantenimiento, interoperabilidad, fácil de modificar, facilidad de soporte y documentación presente en modelos no exclusivos de software libre. Por tanto, se puede concluir que para la evaluación de calidad de software hay características comunes tanto para software propietario como para software no propietario que con fin común buscan mejorar la calidad del producto software.

1.5 CONCLUSIONES

El mercado del software es un mercado con una gran dinámica que día a día está ofreciendo nuevos productos software. Por consiguiente, dicha dinámica ha hecho de la industria del software muy competitiva con clientes exigentes. A pesar de la exigencia de los clientes y de la competividad del mercado está industria debería tener un solo tipo de productor de software que sea competitivo preocupado por la calidad de sus productos y sus clientes, pero la realidad no es así debido a que hay un tipo de productor competidor más preocupado por su supervivencia que por la calidad de sus productos dándole mayor importancia a conseguir costos bajos y precios bajos. Por lo tanto, se puede decir que el productor de tipo competidor está haciendo un gran daño a la industria del software incitando al uso de malas costumbres para el desarrollo de software permitiendo la liberación de muchos productos de muy baja calidad. Por otra parte, se observa la gran cantidad de modelos y estándares de calidad para el software, los hay para el proceso y el producto como también para el software libre. Por lo cual, el problema de seguir produciendo software de mala calidad no se debe a la falta de instrumentos que mejoren la calidad sino de la implementación de los mismo por parte de los

productores y los clientes. Estos últimos en especial al no hacer un proceso más minucioso de evaluación en el proceso de selección del software, de ser lo así los productos de baja calidad no tendrían cabida en el mercado.

Por otra parte, se hizo un cuadro comparativo de los diferentes modelos y estándares calidad a nivel del producto, dicho análisis será de gran utilidad para determinar cuales son los requerimientos de evaluación de calidad a ser usados en la propuesta de este trabajo, por consiguiente, fueron comparados los modelos y estándares de calidad a nivel del producto.

En este trabajo uno de los aspectos más importantes para poder hacer el caso de estudio fue el estudio realizado a la administración de proyectos y en especial a las herramientas que existen en el mercado para este fin debido a que este es el dominio al que pertenece las herramientas evaluadas en el caso de estudio realizado en el capítulo 4. Por tanto, el siguiente capitulo es un estudio completo sobre la administración de proyectos.

2. ADMINISTRACION DE PROYECTOS

2.1 INTRODUCIÓN

En este capítulo se presenta un estudio a profundidad sobre el dominio al que pertenecen los productos software a evaluar en el caso de estudio. Dicho dominio corresponde al software para la administración de proyectos o PMS (Project Management software, sus siglas en ingles). Por otra parte, la evaluación de calidad se realizará a partir de la propuesta de la guía metodológica realizada en este trabajo.

Un principio fundamental para realizar la evaluación de cualquier tipo de producto se hace a partir del conocimiento al detallado y claro de los aspectos a evaluar del producto. Por tanto, este capítulo es de suma importancia para alcanzar los objetivos de este proyecto permitiendo identificar lo siguiente: 1) productos candidatos a evaluar, 2) atributos, subcaracterísticas y características a evaluar que serán parte los requerimientos de la evaluación, 3) definición de métricas de evaluación, y 4) alcance de la evaluación.

Actualmente las organizaciones afrontan nuevos retos en relación al ambiente de negocios, el cual, es variante, dado que las condiciones del contexto actual no permiten tener una tendencia estacionaria en el mercado, más aun exigiendo un alto nivel de competitividad entre las mismas. Este dinamismo se debe a varios factores de cambio: en la tecnología, la sociedad, la competencia, las necesidades del cliente, la situación política, la estabilidad económica, y otros que corresponde a la gestión de las organizaciones. Por consiguiente, las organizaciones de hoy marcan su estrategia, involucrando y sincronizando el esfuerzo de todas sus unidades, procesos, departamentos, equipos y áreas funcionales en pro de alcanzar sus metas.

Este reciente paradigma de administración se conoce con el nombre de alineamiento estratégico. Consiste en que la estrategia es definida por la dirección, además que cuenta con el aporte de todos sus involucrados. Por otro lado, cuenta con la participación de forma sincronizada de todas sus unidades. Marchant dice:

> La estrategia es la definición de fines y medios que orientan a la organización y se traduce en la forma en que la empresa intenta crear valor para sus accionistas, dueños y la comunidad. Permite proyectar a la organización en el largo plazo, dándole un significado y dirección más trascendente que los objetivos financieros[22].

Dado todo este ambiente de negocios, las organizaciones buscan: 1) diferenciarse de sus competidores a través de nuevas ventajas competitivas y

[22] MARCHANT RAMÍREZ, Loreto. Hacia un Modelo de Implementación del Alineamiento Estratégico. Actualizaciones para el Desarrollo Organizacional Primer Seminario. 2005.

2) optimizar sus escasos recursos. Para Reynoso[23] esto se consigue a través de la administración basada en estrategia. Por lo cual, se ve acuñado los temas de estrategia y alineamiento estratégico como nuevas habilidades gerenciales.

Es así, que las organizaciones enfocan su alineamiento estratégico a la administración de sus actividades y procesos como proyectos. Esto permite, optimizar tiempo y recursos, debido a que los proyectos tienen características propias que los hacen diferentes de las actividades rutinarias. Por consiguiente, la administración de proyectos se vuelve fundamental en las organizaciones, ya que la tendencia marcada es conseguir los resultados de su estrategia a través de la realización de proyectos.

2.1.1 ¿Qué es un proyecto?.

Un proyecto se caracteriza básicamente por que es: 1) temporal, tiene una fecha de inicio y una fecha de finalización, 2) su resultado final puede ser: un producto, servicio o resultado único, 3) tiene una elaboración gradual, 4) pueden involucra una o varias personas, 5) su duración pude ser de semanas o varios años, 6) puede involucrar una o varias unidades, 7) tiene un alcance y 8) cuenta con un presupuesto.

La relación entre las organizaciones y proyectos se debe a que estas están llevando en su mayoría todas sus actividades y procesos a proyectos dentro de los límites operativos normales de la organización, esto quiere decir, que toda actividad o proceso que no haga parte de una actividad rutinaria organizacional sea gestionado como proyecto. Para una organización representa grandes beneficios administrar su actividades y procesos como proyectos, ya que estos permiten proyectar mejor el negocio, tener un monitoreo más cerca, y tomar mejor y a tiempo decisiones por lo cual, los proyectos se convierten en un medio fundamental para lograr el plan estratégico de una organización.

2.1.2 Administración de proyectos.

Es una disciplina que se caracteriza por reunir un conjunto de habilidades, técnicas, procedimientos y herramientas, que serán usadas durante todo el proceso de inicio, planificación y ejecución de un programa o proyecto. Además, en esencia busca encontrar la satisfacción del cliente, tanto interno como externo. Por otra parte, requiere hacer un buen uso de los recursos y un buen manejo los riesgos.

[23] REYNOSO, Álvaro. Alineamiento Estratégico – la eliminación de la teoría de la conspiración. Grupo Kaizen S.A.

La administración de proyectos se apoya en una metodología de proyectos, que sea adecuada, por ejemplo, PMBOK (Project Management Body of Knowledge) del la Guie Project Management Institute (PMI). Es de mencionar que dicha metodología tiene cinco procesos básicos: inicio, planeación, ejecución, control y cierre. Adicionalmente, tiene nueve áreas de conocimiento, además, de técnicas y herramientas. La implementación de una buena metodología es garantía de éxito del desarrollo de un proyecto.

El uso de la metodología de proyectos ha tenido un auge a nivel mundial en la industria como herramienta para ser más eficientes y eficaces en los proyectos. Es por eso se pueden encontrar varias instituciones y capítulos de ellas en distintas partes del mundo, tales como PMI (Project Management Institute, con base en Filadelfia, fundado en 1969), APM (Association for Project Management, con base en Inglaterra).

2.1.2.1 Definiciones varios autores.

La administración de proyectos, según Project Management[24], es la disciplina que se encarga de definir y alcanzar objetivos optimizando el uso de recursos: tiempo, dinero, recurso humano, espacio, entre otros.

La administración de proyectos es la forma de planear, organizar, dirigir y controlar una serie de actividades realizadas por un grupo de personas que tienen un objetivo especifico; el cual puede ser (crear, diseñar, elaborar, mejorar, analizar, entre otras) un problema o cosa.

2.1.2.1.1 Inicios.

La administración de proyectos hace su aparición en Estados Unidos, en el periodo, en que dicha nación salía de la segunda guerra mundial y entraba en la guerra fría. Para dicho gobierno, era de suma importancia su programa armamentista, ya que tenia en claro que la guerra se ganaba con el poder de las armas. Es por eso es que designa a una sola persona para que sea la responsable de todos los proyectos de construcción de armas en todas sus fases, y a su vez, sea la encargada de entregar los avances y resultados finales. De esta forma, surge el concepto de administración de proyectos (Kerzner, 2003)

Es así, que en todas las oficinas gubernamentales se empieza a implementar la administración de proyectos, especialmente en el departamento de defensa y la NASA. Por otra parte, se observa una disminución sobre-costos y exceso de

[24] Project Management Institute. Guía de los Fundamentos de la Dirección de Proyectos Tercera Edición.2005

tiempo, en los proyectos del gobierno.

Según Vega[25] en la década de los 70's y 80's el tamaño y complejidad de las actividades en las compañías crecen hasta tal punto donde su administración es imposible. Por tal motivo, la implementación de la administración de proyectos se ve formalizada. En los 90's las compañías aceptan que la administración de proyectos es una necesidad y no una elección según varios factores.

- La importancia en reducir el tiempo de programación y ser los primeros en el mercado.
- Hacer más trabajo y con menos gente.
- La administración de proyectos trabaja mejor si la toma de decisiones y la autoridad se descentraliza, se reconoció que se puede alcanzar el control si la alta dirección actuaba como patrocinador del proyecto.
- Un buen sistema para el control de costos del proyecto permitiría mejorar las estimaciones.
- Muy pocos proyectos eran terminados dentro del marco de los objetivos originales sin cambios de alcance, creando metodologías para una efectiva administración de cambios.

Factores a partir del año 2000

- Fusiones y adquisiciones creando más compañías multinacionales, donde la administración de proyectos se vuelve un reto mayor.
- Las corporaciones se encuentra bajo presión para alcanzar la madurez tan pronto como sea posible, ya que los modelos de madurez en al administración de proyectos ayuda las empresas alcanzar sus metas.
- Los modelos de madurez para la administración de proyectos da a las corporaciones una base para realizar la planeación estratégica para la administración de proyectos. La administración de proyectos es vista como una competencia estratégica en las organizaciones.

2.1.2.1.2 Profesional.

Inicialmente la administración de proyectos era aplicada en el departamento de defensa de los Estados Unidos y la NASA, en la década de los 50's e inicio de los 60's, básicamente consistía en aplicar normas, procedimientos y practicas promulgadas internamente. En seguida, viene una etapa de exploración y expansión, realizada a través de la publicación de libros y artículos, y realización de seminarios y programas de entrenamiento. Por otra parte, la

[25] VEGA DÍAZ, José Alberto. Estado y tendencia de la administración de proyectos en México. Tesis de Maestría. 2004. Universidad de las Américas Puebla, México. p 13.

literatura se centra en el uso de herramientas y técnicas. Ya, para finales de la década de los 60's y principios de la década de los 70's se comienzan a promover foros profesionales con el fin de comunicar y expandir la disciplina, a través de revista, conferencias y seminarios. A mediados de los 80's surgen las primeras entidades en vigilar el cumplimiento de los estándares requeridos por la administración de proyectos, como es el caso de PMI y APM.

PMI, fue la primera en crear un cuerpo de conocimiento para la administración de proyectos (BOK, por sus siglas en ingles Body of Knowledge), en 1976, permitiéndole a un profesional en administración de proyectos poder obtener una certificación, acreditándolo de su conocimiento del BOK. Pero fue hasta finales de la década de los 80's que su guía para la administración de proyectos (PMBOK ® GUIDE, por sus siglas en ingles A Guide to the Project Management Body of Knowledge) se convierte en la base de estándares y certificación. Dicha guía de conocimiento ha venido siendo revisado varias veces desde 1980 hasta su última versión (tercera edición de 2004).

Es de mencionar que PMI no fue la única en crear el único cuerpo de conocimiento, otras organizaciones también lo hicieron, como es el caso de APM, que lanzo su propio BOK, acogido en varios países de Europa (Austria, Francia, Alemania, Suiza y Holanda). Por otra parte, en 1998 el IPMA (por sus siglas en ingles, International Project Management Association) hizo una recopilación de varios libros, exceptuando el publicado por la PMI (debido a que ésta no es miembro de la APM), y de esta forma creo sus propias versiones en varios idiomas (Francés, Ingles y Alemán). Es fácil, que se presente a un gerente confusiones entre las guías de PMI y APM, la diferencia está, en que PMI se enfoca en procesos genéricos para terminar proyectos, como, el tiempo, el costo y alcance, a diferencia de APM que maneja un punto más amplio de la disciplina, manejando el contexto de la administración de proyectos en relación a temas como tecnológicos, comerciales y de administración en general.

2.1.2.1.3 Futuro.

El impacto positivo que ha tenido la administración de proyecto desde sus inicios en áreas de tecnología avanzada hasta nuestros días ha demostrado que también sus métodos de administración de proyectos pueden ser implementados en áreas menos tecnológicas como la sociología, psicología, educación y administración. Por otra parte, estudios y aportes sobre la perspectiva del desarrollo futuro de la administración de proyectos han sido bastantes. A continuación se mencionan algunos según varios autores

- Según Morris[26], actualmente el tema de la administración de proyectos está relativamente maduro y es reconocido por miles de administradores como vital importancia, en comparación con sus inicios en 1960.
- Jugdev[27], a la práctica de administración de proyectos solamente le hace falta el reconocimiento por parte del gobierno para que la práctica tenga suficiente impacto en los bienes públicos para que sea tratado como "profesional" como el caso de otras carreras (derecho, medicina, ingeniería y contabilidad).

Por otro lado, se muestra en seguida una serie de tendencias, predicciones y recomendaciones que encontraron Timothy J. Kloppernborg y Warren A. Opfer en un estudio que presentaron en la conferencia de investigación de PMI en Junio de 2002:

- Estandarización de procesos, herramientas y terminología, para contribuir al éxito en la administración de proyectos.
- Mayor uso de la tecnología Web para mejorar la comunicación y la colaboración empresarial.
- Incremento del outsourcing de la Administración de Proyectos por grandes empresas.
- Incremento de la ocurrencia de proyectos no tradicionales que tratarán con voluntarios, recursos y campañas para obtención de fondos.
- Creación de súper proyectos.
- La selección y priorización de proyectos se usará en las estrategias de los sectores de gobierno e industria.
- Incremento en la capacitación y adiestramiento formal en la administración de proyectos a través de certificaciones.
- Las empresas pondrán más énfasis en la capacitación y adiestramiento en la administración del riesgo, planeación de la contingencia, mitigación del riesgo y manejo de eventos del riesgo.

Además, es de importancia mencionar el gran número de razones directas e indirectas que han influido en el crecimiento de la administración de proyectos. En la década de 90's varios autores se preocuparon por este tema (V.gr. Stewar[28], 1995, Wirth[29], 1992), destacando las siguientes razones:

- Crecimiento de las presiones competitivas globales.
- Crecimiento exponencial de la tecnología de la información.
- El Internet.

---

[26] VEGA DÍAZ, Op. cit. p. 14, dado por MORRIS, P.W.G. The management of Projects. London: Thomas Telford.

[27] Ibid., p. 19, dado por JUGDEV, K. & Thomas, J. Project Management Maturity Models: The Silver Bullets of Competitive Advantage?. Project Management Journal. 2002.

[28] VEGA DÍAZ, Op. cit. p. 19, dado por STEWART, Wendy E. Balanced Scorecard for Projects. Project Management Journal. 2001.

[29] Ibid., p. 19, dado por Wirth: Project Management Education, 1992.

- Liberación y desregulación.
- Crecimiento de la cooperación.
- Preocupación por el ambiente.
- Altas expectativas de los clientes.
- Proyecto más complejos.
- Desarrollo social.

2.1.3 Clasificación de proyectos.

Para la administración de proyectos son varios los expertos que han manifestado la necesidad de clasificar los proyectos. Es de mencionar que los principios de la administración de proyectos pueden ser aplicados a cualquier tipo de proyecto de cualquier tipo de industria. Más sin embargo, debido a eso principios pueden variar de proyecto en proyecto y de industria a industria, debido a los diferentes requerimientos en la administración de proyectos y en los métodos que son utilizados. A continuación se menciona la clasificación de los proyectos según varios conceptos:

- Contenido del proyecto.
- Situación del cliente.
- Tamaño del cliente.
- Área de aplicación.
- Por sector.
- Unidades de organización involucradas.
- Factores de diferenciación.
- Por tipo de industria.

2.1.4 Administrador de proyectos.

Son varios los aportes de autores en relación a las habilidades que debe tener un administrador de proyectos. La administración de proyectos tiene cuatro áreas de experiencia clave, usadas en la práctica, siendo la base del conocimiento para el entrenamiento del personal del proyecto. A continuación se menciona las cuatro áreas de conocimiento.

- Base de experiencia.
- Habilidades sociales.
- Experiencias con trabajo de métodos.
- Habilidades organizacionales.

La práctica de la administración de proyectos requiere de varias habilidades que van más allá de la planeación y control, es importante que los individuos involucrados en el proyecto tenga una suficiente noción, tanto de lo social como

de lo técnico, en relación al funcionamiento de la organización, además, de su cultura y valores. Por otra parte, la industria ha realizado estudios correspondientes a la búsqueda del perfil ideal para el cargo de administrador de proyectos, y es así que se ha evidenciado la notable dificultad que tienen las organizaciones para encontrar el perfil ideal. Son varios los factores que marcan ésta tendencia, el más mencionado: la falta de una carrera exclusiva en la formación de administradores de proyectos, adicionalmente, del compromiso por parte de las organizaciones en la capacitación de sus empleados en ésta disciplina.
Según Kerzner[30], un factor para llevar acabo una buena ejecución de un proyecto, se debe ha la habilidad que debe tener el administrador de proyectos para integrar al personal hacer un trabajo en equipo. Así mismo, para la obtención de resultados, el administrador de proyectos debe relacionar una serie de factores entre si, tales como, 1) la gente que está administrado, 2) la tarea que se tiene que hacer, 3) las herramientas disponibles, 4) la estructura organizacional, y 5) el medio organizacional incluyendo la comunidad del cliente.

Por otro lado, Kerzner[31] relaciona tres habilidades necesarias que debe tener un administrador de proyectos para ser efectivo en el siglo XXI, son:

- Conocimiento del negocio.
- Administración del riesgo.
- Habilidades de integración.

Según Nehlsen[32] y otros autores se puede clasificar en perfiles y metodologías para la administración de proyectos de acuerdo al tipo de proyecto.

- Proyectos administrativos: habilidades administrativas.
- Proyectos de construcción: orientación al costo.
- Proyectos de desarrollo de software: comunicación y administración al riesgo.
- Proyectos de eventos: administración del tiempo.
- Desarrollo de productos: administración del tiempo.
- Proyectos de investigación: calidad.

2.1.5 Sistemas para proyectos.

2.1.5.1 Sistemas.

Es un conjunto integrado de componentes independientes o que interactúan entre si, creado para alcanzar un objetivo definido. Lo sistemas se pueden

[30] VEGA DÍAZ, Op. cit. p. 27, dado por Kerzner, Harold. Project Management a systems approach to planning, sheduling, and controlling Eighth edition: Wiley.
[31] Ibid., p. 27,
[32] VEGA DÍAZ, Op. cit. p. 26, dado por Nehlsen, T: Anpassungsbedarfs-Analyse, IPMI, Bremen 2001.

basar en un proceso físico, proceso de gestión o la combinación de los dos. Adicionalmente, los sistemas para la dirección de proyectos están formados por, procesos, técnicas, metodologías, y herramientas de dirección de proyectos, operadas por los miembros del equipo de dirección de proyectos.

2.1.5.2 Sistemas de gestión de proyectos.

Es una herramienta que se compone de procesos, herramientas, técnicas, metodologías y procedimientos, necesarios para la gestión de proyectos. El sistema es documentado en el plan de gestión de proyecto, el cual puede variar según, el área de aplicación, influencia de la organización, complejidad del proyecto y disponibilidad de los sistemas existentes. El sistema de gestión de proyectos pude ser formal o informal, que ayuda al director de proyectos a liderar el proyecto de forma efectiva desde el inicio hasta el final. Es conocido también como: sistemas de administración de proyectos, sistemas de dirección de proyectos, sistemas de gerencias de proyectos; o sistemas de gerenciamiento de proyectos.

2.1.5.3 Sistemas de información de la gestión de proyectos.

Es una herramienta basada en un sistema de información que está compuesto por herramientas y técnicas, utilizados para recolectar, integrar y difundir los resultados del proyecto. Además, de ser usado como respaldo de todos los procesos del proyecto desde el comienzo hasta el final a través de sistemas manuales o automatizados. Se conoce también como: Sistema de Información de Administración de Proyectos, Sistema de Información de la Dirección de proyectos, Sistema de Información de Gerencia de Proyectos; o Sistema de Información del Gerenciamiento de Proyectos.

2.1.5.4 Software de gestión de proyectos.

Es una herramienta de tipo de aplicación de software, diseñada para ayudar al grupo de dirección de proyecto para la planificación, seguimiento y control, además, de la estimación de costos, planificación, comunicación, colaboración, gestión de la configuración, control de documentos, gestión de requisitos, y análisis de riesgos. Se conoce también como: Software de Administración de Proyectos, Software de Dirección de Proyectos, Software de Gerencia de Proyectos; o Software de Gerenciamiento de Proyectos.

2..5.5 Tecnologías de información que apoyan la administración de proyectos.

La administración de proyectos es muy cambiante en su forma de gestión. Se debe a las nuevas tecnologías, la competencia globalizada, y nuevas características, adoptadas en la administración de proyectos. Es así, que el papel que juega las tecnologías de información en la administración de proyectos es determinante.

La oferta de las nuevas herramientas que apoyan la administración de proyectos, son adoptadas en las organizaciones con base a los objetivos y las características de la metodología usada en su administración de proyectos.

Los cambios en la administración de proyectos van muy relacionados a las tecnologías de información. Esto hace que sea difícil poder determinar las herramientas disponibles que apoyen la administración de proyectos. Las empresas en la actualidad no solo usan herramientas que tengan diagramas de Gannt, PERT y capacidades para elaborar reportes rápidamente. Adicionalmente, se conocen como herramientas de apoyo y no se hacen llamar formalmente como herramientas de administración de proyectos. En seguida se mencionan algunos nombres.

- Herramientas de administración de proceso.
- Herramientas de administración de portafolios
- Herramientas de administración de conocimiento
- Portales
- Groupware
- Sistemas colaborativos integrados con otras aplicaciones

2.2 PROBLEMÁTICA

A continuación se menciona varias de las problemáticas que se presenta en la administración de proyectos especialmente en la industria de desarrollo de software correspondientes a varias tendencias para el desarrollo de proyectos, como, la administración de múltiples proyectos, la subcontratación, el desarrollo a distancia, y el concepto de la calidad en los proyectos. Dichas tendencias se han venido marcando de acuerdo al incremento competitivo y la evolución de la tecnología en el sector de desarrollo de software.

Para estas tendencias se requiere de herramientas eficaces. Administrar múltiples proyectos requiere de una herramienta que sea capaz de administrar

la alineación entre proyectos dentro de un mismo portafolio, manteniendo el control y la comunicación entre las partes. La subcontratación implica la interacción entre entidades externas con la organización necesitando de herramientas que faciliten el proceso de colaboración. El desarrollo a distancia, como su nombre lo dice, la interacción entre equipos de proyectos distribuidos geográficamente, por lo cual, necesita de herramientas de colaboración flexible basada en la Web. Según Barad[33] & Raz, el concepto de calidad de proyectos, requiere de herramientas que le permitan mejorar los productos y la satisfacción al cliente, específicamente mejorando el rendimiento en cuatro actividades: el control de procesos o actividades, el entrenamiento de personal, el enfoque dirigido al cliente y manejo de la información interna.

2.2.1 Software para la administración de proyectos.

La administración de proyectos incluye el uso de una gran variedad de tipo de software, principalmente para la programación de actividades, la asignación de recursos, administración de documentos y además del uso de software de colaboración. Por consiguiente, la administración de proyectos requiere de la ayuda de las aplicaciones software para automatizar y facilitar el uso de la metodología usada por la organización para su administración de proyectos.

Las organizaciones deben documentar sus necesidades y con base en esas necesidades realizar el proceso de selección de la herramienta o herramientas software más adecuadas para su administración de proyectos
.
Las nuevas tecnologías de información de administración de proyectos combinan las tres S's: scope, scheduling y status. En otras palabra son herramientas que permitan administrar el alcance, la programación de tares y el estado del proyecto. Por otro lado, según Murtagh[34], estas herramientas implementan tecnología basada en Internet. Se puede observar en la tabla 8, 9, 10 y 11 algunas aplicaciones de administración de proyectos, clasificadas en dos grandes categorías: software propietario y software libre, a su vez agrupadas por aplicaciones de tipo desktop y basadas en la Web. Por otra parte, dichas aplicaciones no exclusivas para trabajo de administración de proyectos, adicionalmente pueden ser usadas para otro tipo de trabajo, la cuales se clasifican de la siguiente forma:

- PM: Project management software
- C: Collaborative software
- IT: Issue tracking system
- PPM: Project portfolio management

[33] CABALLERO CERVANTES, Omar Higinio. Tecnología de Información y Herramientas para la Administración de Proyectos de Software. Revista Digital Universitaria, Volumen 7 Número 6, 2006. dado por BARAD, M., & RAZ, T. Contribution of quality management tools and practices to Project management performance. International Journal of Quality & Reliability Management.

[34] Ibid., dado por MURTAGH, J. IT Project management and software development methodology. Retrieved. 2005.

- ❖ RM: Resource management
- ❖ DM: Document management

Tabla 8. Aplicaciones desktop - software libre

Open-Source desktop applications	PM	C	IT	PPM	RM	DM
GanttProject	PM				RM	
KPlato	PM					
OpenProj	PM				RM	
Open Workbench	PM	C [1]			RM	
TaskJuggler	PM				RM	

Fuente: List of project management software. (2005). [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://en.wikipedia.org/wiki/List_of_project_management_software

Tabla 9. Aplicaciones basadas en la Web - software libre

Open-Source web-based applications	PM	C	IT	PPM	RM	DM
Bugzilla			IT			
eGroupWare	PM	C	IT	PPM	RM	DM
dotProject	PM		IT			DM
Mantis Bug Tracker			IT			
Project.net	PM	C	IT	PPM	RM	DM
ProjectPier	PM	C				
Trac	PM	C	IT			
SharpForge	PM	C	IT			

Fuente: List of project management software. (2005). [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://en.wikipedia.org/wiki/List_of_project_management_software

Tabla 10. Aplicaciones desktop – software propietario

Proprietary desktop applications	PM	C	IT	PPM	RM	DM
Artemis	PM	C	IT	PPM	RM	
Collanos Workplace		C				
Contactizer	PM	C			RM	
FastTrack Schedule	PM				RM	
LisaProject	PM				RM	
LiveProject	PM	C				DM
MacProject	PM				RM	
MicroPlanner X-Pert	PM	C	IT	PPM	RM	
Microsoft Project	PM				RM	
O3spaces	PM	C			DM	
OmniPlan	PM				RM	
OpenMind Business	PM			PPM	RM	
Planner Suite	PM			PPM	RM	DM
Planisware 5	PM	C	IT	PPM	RM	DM
Primavera Project Planner	PM	C	IT	PPM	RM	DM
Project KickStart	PM				RM	
RationalPlan	PM			PPM	RM	DM
RiskyProject	PM				RM	
Teamcenter	PM	C	IT	PPM	RM	
Tracker Suite	PM	C	IT	PPM	RM	

Fuente: List of project management software. (2005). [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://en.wikipedia.org/wiki/List_of_project_management_software

Tabla 11. Aplicaciones basadas en la Web – software propietario

Proprietary web-based applications	PM	C	IT	PPM	RM	DM
@task	PM	C	IT	PPM	RM	
24SevenOffice	PM	C				
AlterFiction-ISES	PM	C				
Basecamp		C				
Cardinis	PM	C	IT	PPM	RM	
Central Desktop	PM	C				
Clarity	PM	C	IT	PPM	RM	DM
Daptiv	PM	C	IT	PPM	RM	DM
EnterPlicity	PM	C	IT	PPM	RM	
eVisioner MetaTeam		C			RM	
Gatherspace	PM	C				
Genius Inside	PM	C		PPM	RM	DM
Instant Business Network	PM	C	IT	PPM	RM	
LiquidPlanner	PM	C		PPM	RM	
Microsoft Office Project Server	PM	C	IT	PPM	RM	
Mingle	PM	C	IT			
OpenAir	PM	C	IT	PPM	RM	DM
Oracle Projects	PM	C	IT	PPM	RM	
Planisware OPX2/Planisware 5	PM	C	IT	PPM	RM	DM
Project Insight	PM	C	IT	PPM	RM	

Fuente: List of project management software. (2005). [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://en.wikipedia.org/wiki/List_of_project_management_software

2.3 ESTUDIO COMPARATIVO DE SOFTWARE PARA ADMINISTRACIÓN DE PROYECTOS

2.3.1 Introducción.

En diciembre de 2007 fue realizado un estudio comparativo de varias aplicaciones software para la administración de proyectos. Dicho estudio fue realizado por el Instituto de Bioinformática de la Universidad de Jhannes Kepler Universität Linz para el Seminario Projektorganisation encabezado por la Dra. Elisabeth Kapsammer. La intención de mostrar este estudio es de realizar un comparativo de una evaluación de calidad que fue realizada de una forma muy subjetiva en comparación a la evaluación que se realizó para este trabajo que fue hecho a través de la guía metodológica propuesta, caracterizada por ser práctica e ingenieril. Por otra parte, este estudio fue de gran ayuda para la identificación de atributos, subcaracterística y categorías que en ambos tienen el mismo dominio sobre el producto evaluado que corresponde al software para la administración de proyectos.

Para la realización de éste trabajo se definieron varios criterios de evaluación y categorías, así mismo se seleccionaron ocho aplicaciones para la administración de proyectos. La muestra de herramientas fue bastante diversa entre aplicaciones de Open Source y propietarias. Por otra parte, se escogieron herramientas tanto en ambientes cliente/standalone (Microsoft Project 2007), cliente/servidor (Microsoft Project Server 2007) y cliente/servidor Web (dotproject). A continuación listan las aplicaciones seleccionadas en dicho estudio.

- Microsoft Project
- Trac
- Planner
- GanntProject
- doproject
- Merlin
- FastTrack Schedule 9
- Ovni Plan

2.3.1.1 Criterios de evaluación.

Para éste trabajo se definieron 25 criterios, agrupados en 6 categorías. Cada criterio fue medido en cada producto a través de la siguiente escala, -- (malo), ~ (regular), ++ (bueno) y n/a (no aplica). El resultado obtenido fue una medición subjetiva de cada criterio correspondiente cada producto evaluado. En la tabla 10, se observa el árbol de categorías con sus correspondientes

criterios. Es de mencionar que esto no son los únicos criterios de evaluación pueden existir otros y estos van en relación a los requerimientos de calidad que corresponden al dominio al que pertenece los productos a evaluar.

Tabla 12. Categorías y criterios de evaluación

Interfaz Grafica	Suficiencia de entrada
	Soporte Web
	Internacionalización
	Manejo intuitivo
	Estética
Económica	Costo
	Cuota de mercado
	Software libre
	Independencia plataforma
Interoperabilidad	Formatos estandarizados
	Automatización
	Cooperación
	Exportar, reportes
Manejo de datos	Versiones
	Sincronización
Administración proyectos	Recursos
	Tareas
	Ruta critica
	Administración de múltiples-escenarios
	Administración de múltiples-proyectos
Misceláneos	Documentación
	Soporte
	Rendimiento
	Estabilidad
	Transparencia

Fuente: Kapsammer, Elisabeth. Seminar Projektorganisation Comparison of Projectmanagement-Software.

A continuación se menciona las definiciones de cada criterio tenidas en cuenta para éste trabajo.

2.3.1.1.1 Interfaz Grafica.

- **Suficiencia de entrada**: PMS (*Project Management Software*, sus siglas en ingles) requiere de una cierta cantidad de información la cual debe estar disponible al usuario. Esto incluye (*Task Information*, por su nombre en ingles) como nombre, inicio, fin, trabajo, (*Resource Information*, por su nombre en ingles): nombre, disponibilidad, costo y relaciones entre ellos, además de la asignación y dependencias. Es así, que un buen método ingreso de datos masivo debe estar disponible para disminuir el esfuerzo de la entrada de datos.
- **Soporte Web**: la integración digital y el crecimiento de los datos, hace mucho más importante que la recuperación de la información se pueda hacer en línea a través de una interfaz Web. Para un PMS es relevante estar implementado en un ambiente Web.
- **Internacionalización**: la cultura de administración de proyectos puede ser beneficiada su localización a través del software, siendo más sencillo la compresión e intercambio de la información.
- **Manejo intuitivo**: una clara estructura en la interfaz de usuario hace que su comportamiento sea predecible. Esto hace que la productividad mejore y la curva de aprendizaje sea corta.
- **Estética**: los reportes claros y con buena presentación son requerimientos para un mejor entendimiento.

2.3.1.1.2 Económica.

- **Costo**: el retorno de la inversión depende de un justo y justificable modelo de licenciamiento respectivamente. La ganancia del rendimiento del software (*performance,* por su nombre en ingles) se obtiene a través del software libre.
- **Cuota de mercado**: la participación que tiene el producto en el mercado.
- **Software libre:** implica un costo de uso cero, el cual se convierte en un criterio de costo. Pero va más allá, permitiendo al usuario adaptar la aplicación según su entorno y necesidades propias, ideal para integrar con otros sistemas.
- **Independencia de plataforma**: aunque la virtualización de la tecnología está comenzando a reducir el impacto de la plataforma dependiente del software (platform-fixed, por su nombre en ingles): el costo fijo por rendimiento, el costo de licenciamiento, y la preparación a tiempo, no están siendo beneficiados. El uso de una plataforma independiente del software reduce la dependencia por parte del lado del cliente y puede evitar experiencias negativas.

2.3.1.1.3 Interoperabilidad.

- **Formatos estandarizados**: permitir el intercambio de información sin un formato definido entre diferentes productos. Esto no debe ser una restricción para un PMS, por ejemplo, la instancia del iCalendar, define un formato de datos el cual es soportado por una largar base de aplicaciones software.
- **Automatización**: es soportado a través de API (Application programming interfaces, sus siglas en ingles). La integración del software dentro de una compañía se hace a través de la sincronización entre los datos del proyecto con otras aplicaciones.
- **Cooperación**: obtener a través de otros canales: la creación y administración de un plan cooperativo, el intercambio de información entre el equipo de trabajo y la reducción del gasto fijo de la comunicación. La centralización entre varios planes de proyectos contribuye también a ésta idea.
- **Exportar y reportes:** Las graficas y tablas que hace parte de los diferentes *outputs* del PMS deben poder ser exportados a los diferentes formatos básicos esto contribuye a darle un valor agregado al software.

2.3.1.1.4 Manejo de datos.

- **Versiones**: es la posibilidad de llevar un control al cambio de versiones. Esto incluye la recuperación de documentos previos, relacionar los cambios de usuarios, observar las diferencias entre las diferentes versiones, backing up a versiones y características similares.
- **Sincronización**: es la capacidad de poder sincronizar la información entre varios planes de proyectos. De igual forma, es la importancia de levantarse en un ambiente donde el cliente y proveedor o administrador de múltiples proyectos coordina activamente los planes de proyectos o el trabajo en conjunto.

2.3.1.1.5 Administración proyectos.

- **Recursos:** éste criterio representa el manejo de los recursos. Incluye inicialmente la asignación automática y parcial. Indica el resumen de asignación, la lista de detalle de las tareas asignadas y el manejo de la no disponibilidad de los recursos manejados aquí

también.

- **Tareas:** son piezas del proyecto, de esa forma se agrupan y se arreglan dentro de la estructura de fallas del trabajo, es crítico y un requerimiento mínimo. Tiene diferentes presentaciones visuales, además de la grafica de Gantt.
- **Ruta critica**: el concepto de la ruta crítica ayuda a identificar las tareas críticas del proyecto. La disponibilidad de un plan de redes como MPM, CPM y el PERT que permite a los usuarios seleccionar un método al que esté acostumbrado o que sea usado por los usuarios de la compañía.
- **Administración de múltiples-escenarios**: la posibilidad de abrir varios escenarios desde un plan de proyecto puede resultar muy beneficioso para el análisis del riesgo, además que ayuda definir un plan de contingencia. Cuando el PMS lo trae resulta muy práctico, caso contrario de copiar y pegar un plan para recrear escenarios.
- **Administración de múltiples-proyectos**: permite la coordinación y la visualización del estado de varios proyectos. Además, de ayudar a administrar los proyectos por el lado de la compañía, también de ayudar a cargar el detalle global de los recursos.

2.3.1.1.6 Misceláneos.

- **Documentación**: para cualquier software es importante tener de manera clara y estructurada la documentación de ayuda, la cual debe estar a la mano al momento de ser necesitada y requerida por el usuario, tanto en línea como fuera de línea.
- **Soporte**: el soporte para productos software puede existir para productos de software propietario como de software libre. Sin embargo, la demanda de tal servicio será sólo asunto cuando PMS es muy usado y no es pertinente para usuarios infrecuentes.
- **Rendimiento:** no requiere de mucho poder de procesamiento y todas sus tareas debe ser cumplidas sobre el patrón *computer system* sin mucho retardo o uso de otros recursos.
- **Estabilidad**: mucho tiempo y datos pueden potencialmente ser perdidos por encima de software crashes o archivos corruptos. Los productos maduros no muestran tales problemas, pero frecuentemente el software libre muestran una ausencia de control de calidad y elaboración de testing.
- **Transparencia**: El argumento por cada decisión tomada por PMS debe ser transparente o tener una explicación. Varios algoritmos trabajan en el segundo plano para arreglar las tareas, calculo de costos, entre otros. Comprender estos algoritmos y a razón de que

va por detrás de la escena es el principal asunto para usar eficazmente el producto y obtener una importante reducción de la frustración.

2.3.2 Aplicaciones Evaluadas.

Como anteriormente se mencionó, en dicho estudio se evaluaron ocho herramientas software para la administración de proyectos o Project Management System (su nombre en ingles). La selección de dichas herramientas se realizó de la forma más diversa posible, tomando herramientas tanto de software propietario como de software libre. Por otra parte, se tuvo en cuenta que las herramientas a evaluar funcionaran en diferentes ambientes: cliente/escritorio, cliente/servidor y Web. En seguida se relaciona la evaluación realizada al Microsoft Project según los criterios de evaluación mencionados en el punto anterior.

2.3.2.1 Microsoft Project.

El Microsoft Project está a la cabeza en la categoría de software especializado para la administración de proyectos existente en el mercado, ha vendido 20 millones de licencias de usuario hasta el 2006. Similar a otras aplicaciones de Microsoft se beneficia a que está instalado sobre Windows, sistema operativo del mismo fabricante. Adicionalmente, tiene gran facilidad para la integración con otros productos de Microsoft, tales como el Microsoft Outlook y el Excel. La versión más reciente es el Microsoft Project 2007, disponible en cliente-escritorio o con la versión servidor – Microsoft Project Server 2007 para la centralización de los datos e integración con otros sistemas, tales como el Microsoft Sharepoint portal. En la tabla 13, se observa los resultados obtenidos de la evaluación, agrupados en las seis categorías.

Tabla 13. Evaluación de Microsoft Project 2007

Interfaz Grafica	Suficiencia de entrada	-	Problema principal en el uso básico
	Soporte Web	+	Desde la Web permite exportar y tiene acceso, más no permite la edición.
	Internacionalización	++	Software y documentación, pero costo extra.
	Manejo intuitivo	~	Interfaz estilo Windows pero no muy clara
	Estética	+	Buen diseño y coherencia con office
Económica	Costo	--	Mil dólares por una licencia para cada cliente
	Cuota de mercado	++	
	Software libre	--	No, pero libre SDK y compatibilidad con macros
	Independencia plataforma	n/a	Cierre completo a menos que use virtualización.
Interoperabilidad	Formatos estandarizados	-	Exporta a HTML, XML
	Automatización	+	Integración a través de macros
	Cooperación	~	Cuando se usa con software de Microsoft
	Exportar, reportes	+	Se exporta de la Web y reportes, pero requiere mucho de Excel
Manejo de datos	Versiones	-	Comparaciones y basado en servidor, sin plenos derechos
	Sincronización	~	Comparaciones o varias revisiones, manuales
Administración proyectos	Recursos	+	Cliente y servidor puesta en común disponible
	Tareas	+	Falta de claridad en muchos casos
	Ruta critica	++	De la red y putos de vista desde el diagrama de Gantt
	Administración de multiples-escenarios	-	En su mayoría manualmente
	Administración de multiples-proyectos	+	A través de acceso Web y un servidor de portafolio
Misceláneos	Documentación	++	No bien estructurada
	Soporte	+	Disponible, pero costosa
	Rendimiento	+	La versión de servidor requiere muchos recursos
	Estabilidad	++	El sistema de acceso Web causa que se recargue el sistema
	Transparencia	-	El cliente necesita mejora

Fuente: Kapsammer, Elisabeth. Seminar Projektorganisation Comparison of Projectmanagement-Software.

2.3.2.1.1 Interfaz Grafica.

La evaluación obtenía de ésta categoría es regular en términos generales. Los resultados completos se pueden observar en la Tabla 13. La interfaz gráfica es una categoría (ver Tabla 12.) compuesta por cuatro criterios, los cuales, son evaluados. El Soporte Web o *Web Support* (su nombre en ingles) tiene una evaluación de + (medio bueno), a pesar de tener acceso Web pero su gran desventaja está en relación a no permitir la edición en línea. Internacionalización o Internationalization (su nombre en ingles) fue evaluada con ++ (bueno) debido al software y su documentación. Por otra parte, el resultado no fue el más favorable correspondiente a los dos criterios restantes: el ajuste de entrada o *Input adequacy* (su nombre en ingles) y el manejo intuitivo o *Intuitive Handling* (su nombre en ingles). El primero obtuvo – (medio malo) debido a problemas básicos de usabilidad y el segundo obtuvo ˜ (regular) por su presentación al estilo Windows, la cual pierde claridad. A continuación se mencionan algunos aspectos tenidos en cuenta para la evaluación de dicha categoría.

- Diagrama de Gantt o vista del grupo de tareas. Es presentado de forma más radiante y clara, como era de espera de un producto de Microsoft.
- Barra de tareas. A pesar de tener una muy buena impresión con sus bordes redondos, se puede evidenciar que al trabajar con varios puntos de vista y cuadros de diálogos de entrada, la herramienta hace parte de la familia de la Suite de Office, heredando problemas de usabilidad e ineficiencias. Por ejemplo, la pérdida de datos, que no es tenida en cuenta en varias situaciones. Por otro lado, el ingreso de datos de forma masiva que es obstaculizada por el rechazo de datos no validos en los campos, debido a errores tipográficos.
- Opacidad. Problema en general debido a que no se presenta la información, específicamente en algunas tareas que no se pueden mover.
- Aprendizaje intuitivo de símbolos. Requiere de una amplia búsqueda en la documentación. Esto garantiza que la curva de aprendizaje tenga puntos de frustración.
- Coherencia con Windows e interfaces de usuario de Office. Proporciona cierta familiaridad con el funcionamiento de la aplicación en relación a la normalización de la barra de herramientas e iconos.
- Acceso Web. Es compresivo en relación a la consulta de proyectos, almacenados en el Microsoft Project Server, además de su disponible al usuario, sin la necesidad de tener la funcionalidad completa, más sin embargo, proporciona un completo resumen del proyecto, permitiendo consultar el riesgo general, los enlaces entres proyectos y el reporte de estado y de gestión.
- Idioma. Cuenta con versiones regionalizadas para 35 idiomas, así como también paquetes de lenguajes que pueden ser comprados para permitir

la múltiple regionalización.

2.3.2.1.2 Económica.

Económico o *Economical* (su nombre en ingles) es una categoría compuesta por cuatro criterios (ver Tabla 12). Su evaluación fue muy mala especialmente en dos criterios: costo o *cost* (su nombre en ingles) y software libre o *Open Source* (su nombre en ingles), obteniendo como resultado – (malo). Por otra parte, caso contrario sucedió para el criterio de cuota de mercado o *Market Share* (su nombre en ingles) que obtuvo un resultado de ++ (bueno). Por otro lado, el criterio de plataforma independiente o *Platforms Independence* (su nombre en ingles) no aplico para la evaluación de ésta herramienta. En seguida se mencionan aspecto tenidos en cuenta para la evaluación de dicho criterio.

- Costo de licencia. El Microsoft Project 2007 tiene un costo de licenciamiento. El precio para la versión cliente está alrededor de los $1000 dólares US, dicho precio en relación al año 2007. Por otra parte, por cada paquete de lenguaje de regionalización tiene un valor de 25 dólares US cada uno. Las licencias para el servidor son vendidas como CAL (Cliente Access Licenses, su nombre en ingles), es decir, que por cada usuario o por cada dispositivo.
- Software propietario versus software libre. Aquí se marca una gran diferencia entre el costo de adquisición en relación a aplicaciones de software libre (no tiene costo de licenciamiento) y las aplicaciones de software propietario (tiene costo de licenciamiento), por ejemplo, para el Microsoft Project. En dicho caso, para el uso individual de forma no profesional es muy probable que no se recupere la inversión, y una hoja de cálculo de software libre sería suficiente para éste tipo de trabajo.
- Costo de plataforma. El Microsoft Project 2007 Server sólo corre en Windows Server, por lo cual, hay que sumar el costo de la plataforma.

2.3.2.1.3 Interoperabilidad.

La interoperabilidad o *Interoperability* (su nombre en ingles) es una categoría que obtuvo una evaluación regular en términos generales, dicha categoría la componen cuatro criterios (ver Tabla 12.). Normalización de formatos o *Standardized Formats* (su nombre en ingles), obtuvo un resultado de – (medio malo) debido a que sólo puede exportar datos a HTML y XML. Cooperación o *Cooperation* (su nombre en ingles) dicho criterio obtuvo un resultado de ˜ (regular) a razón de qué tiene una alta cooperación cuando es usado con software del mismo fabricante, en este caso Microsoft. Por otro lado, los dos

criterios restantes: automatización o *Automation* (su nombre en ingles) y reportes o *Reports* (su nombre en ingles), ambas obtuvieron un resultado de + (medio bueno). La primera debido a su integración y trabajo de macros, por otra parte, la segunda, se debió a la posibilidad de exportar los reportes a través de la Web, pero con la desventaja de que se requiere mucho de Excel (herramienta de la suite de office de Microsoft). A continuación se mencionan algunos aspectos tenidos en cuenta para la evaluación de dicha categoría.

- Integración con herramientas de Microsoft. Es una de las principales características del Microsoft Project 2007 debido a su facilidad para la integración con otras herramientas de Microsoft, tales como, Microsoft Exchange para la sincronización del calendario y recurso humano, Sharepoint para la combinación del acceso Web con el Project Web portal, SQL Server como servidor de respaldo, entre otras. Por otro lado, el cliente se puede exportar una gran cantidad de elementos a Excel y Access.

- Integración con proveedores independientes. Permite exportar los datos en texto plano, HTML, XML. Adicionalmente, para una automatización más sofisticada o integración dentro de otros sistemas, se proporciona un SDK (Software Development Kit, sus siglas en ingles) con soporte para el cliente y el servidor, dicha versión se puede descargada sin costo alguno.

2.3.2.1.4 Manejo de datos.

El manejo de datos o *Data handling* (su nombre en ingles) es una categoría compuesta por dos criterios: versión o *Versioning* (su nombre en ingles) y sincronización o *Synchronization* (su nombre en ingles). Por otro lado, su evaluación obtuvo un resultado en promedio muy malo, obteniendo un – (medio malo) y un ~ (regular) respectivamente para cada criterio. A continuación se menciona algunos aspectos que se tuvieron en cuenta para la evaluación de dicha categoría.

- Funcionalidad del sistema de versiones y funcionalidad de elaboración. Mientras el primero no puede comprender, el segundo compara dos archivos del proyecto, esto de debe ser capaz manejar todos los asuntos que puedan surgir durante la sincronización de un plan de proyecto.
- Enlaces entre diferentes proyectos en la parte del cliente. Se pueden establecer diferentes enlaces entre varios planes de proyectos, aunque ésta característica no funcionó como se esperaba al momento de hacer las pruebas. Por otra parte, se puede crear un pool de recursos,

guardándolo en un archivo de proyecto para ser referenciado desde otros archivos.

- Expansión de opciones con la versión de servidor. El servidor puede ser usado como almacenamiento central de planes globales de proyectos con un simple sistema de bloqueo con chequeo de entra y salida. Por otro lado, a través de los backups del servidor, implícitamente se puede usar también las herramientas existentes para la recuperación de proyectos de forma individual desde backups. Adicionalmente, actúa como punto central para los recursos, los documentos, y los enlaces entre proyectos.
- Soporte a *OLAP cube*. El Project Server soporta la generación del llamado al *OLAP cube* para ser almacenado en Microsoft SQL Server y ser procesado por la compañía de análisis de servicio. Por otra parte, Microsoft SQL Server 2005 Analysis Services hace entrega en línea del procesamiento analítico (OLAP) y la funcionalidad para minería de datos para las aplicaciones de inteligencia de negocios.

2.3.2.1.5 Administración proyectos.

En esta categoría se agrupan los criterios más representativos con la administración de proyectos. Esta compuesta por cinco criterios (ver Tabla 12.). Su evaluación en términos generales fue bastante buena, a diferencia del criterio de manejo múltiple de escenario o Multi-scenario management (su nombre en ingles) que obtuvo un – (medio malo), las restantes fueron ++ (bueno) y + (medio bueno). En seguida se mencionan algunos aspectos que se tuvieron en cuenta para la evaluación de esta categoría.

- Las dependencias y enlaces se puede crear con todas las combinaciones y puede ser gestionados a través de archivos o de un servidor central.
- Dependencia y enlaces. Se puede crear con todas las combinaciones y puede ser gestionados a través de archivos o de un servidor central.
- La ruta o trayectoria crítica (*Critical Path*, su nombre en ingles). Se puede colocar en relieve y su vista puede cambiar a un estilo de diagrama de red que puede ser personalizado con muchas apariencias.
- Funcionalidad de Multiple-scenario management. No esta disponible, por otra parte, el administrador múltiple de proyectos o Multi-project-management (su nombre en ingles) es soportado por parte del servidor con un alto nivel, el acceso de esta característica se hace a través de la Web, permitiendo observar el estado de todos los proyectos.
- Para la coordinación en un alto nivel, Microsoft entrega Project Portafolio Server para el Project portafolio management. Dicho producto no esta

disponible por los autores, por lo cual no pudo ser evaluado.

- Coordinación a un alto nivel. El Microsoft entrega el Project Portafolio Server para el administrador de portafolio de proyecto o *Project portafolio management* (su nombre en ingles).

2.3.2.1.6 Misceláneos.

La categoría de misceláneos o *Miscellaneous* (su nombre en ingles) es una categoría compuesta por cinco criterios (ver Tabla 12.). Su evaluación en promedio fue muy buena, a diferencia del criterio transparencia o *Transparency* () que obtuvo – (medio malo), las restantes tuvieron ++ (bueno) y + (medio bueno). En seguida se mencionan algunos aspectos tenidos en cuenta para la evaluación de dicha categoría.

- Documentación. La documentación se integra dentro del proyecto y la información se proporciona en línea y fuera de línea. Lo anterior ayuda a que la documentación esté siempre actualizada, pero por otra parte, aumenta los asuntos de privacidad. Es de mencionar que Microsoft es capaz de recolectar datos de uso personalizado y que el software tiene una conexión integrada con Microsoft, donde no se puede determinar, lo que esta siendo transmitido. Por lo tanto se debe confiar en el proveedor para la recuperación interna del proyecto o el bloqueo del software con el firewall systems.
- Ayudas en video o *Webcasts* (su nombre en ingles). Se encuentran disponibles y pueden ser vistos y encontrados, otra vez a través del sistema de ayuda, así como también a través de los recursos generales de la Web como *homepage* del *Project-related products*. Así la integración de los recursos sea en línea o fuera de línea, se ha realizado de una manera agradable que incluye la colección de la información del centro de ayuda de lado del cliente.
- Rendimiento o *Performance* (su nombre en ingles) y estabilidad o *Stability* (su nombre en ingles). No presento problemas durante la evaluación de los productos. Es de anotar, que se observo que la versión del *Project Server* junto con el Windows Server requiere de muchos más recursos para correr de manera ligera.
- Transparencia o *Transparency* (su nombre en ingles). Se discutió durante la evaluación del la interfaz gráfica, donde no quedo muy claro. A lo anterior se le debe sumar la no divulgación del código fuente impidiendo determinar la falta de mecanismo de trabajo, necesitados o deseados, esto es una desventaja en relación a los productos de software libre.

2.3.3 Resumen.

En resumen de todos los resultados se puede decir que el Microsoft Project 2007 es una aplicación profesional, pero muy costosa para la administración de proyectos, a pesar de contar con la extensión e integración con otros productos de Microsoft. Junto con el Project Server 2007 y Project Portafolio Server, proporcionando un solución empresarial calificada.

En un máximo de escala ésta solución muestra haber arrastrado el foco muy poco fuera del software del cliente; central para todos los project management. Al sugerir algo como una imagen heredada en comparación a los otros productos de Office en términos básicos de usabilidad y claridad.

Los mejores argumentos en pro y en contra para tomar una decisión a incluir son: el costo, el ambiente de los productos Microsoft existente en la compañía y la necesidad propia del software a nivel corporativo.

2.3.4 Análisis de resultados.

Los resultados de la evaluación de todos los productos se puede observar en la tabla 14. De acuerdo a estos resultados se puede sacar varias conclusiones en relación a las diferentes características evaluadas. Por ejemplo, la característica de interfaz gráfica se evalúan los siguientes atributos: suficiencia de entrada, soporte Web, internacionalización, manejo intuitivo y estética. Es de observa que las herramientas: Grannt Project y dotProject tiene la mayor valoración en relación a las demás siendo las mejores en ésta característica, caso contrario de las herramientas: MS Project, Trac y Planner que les sigue en un lugar intermedio y las peor valoración les correspondió a las herramientas: Merlin, Fasttrack9 y Omniplan. La característica de economía evalúa los atributos: costo, cuota de mercado, software libre e independencia de plataforma. Los resultados de está característica refleja que las herramientas de software libre (Trac, Planner, Grannt Project, y dotProject) son las que han tenido la mejor valoración en relación a las demás herramientas, la de peor valoración le correspondió a MS Project, debió a su valoración en los atributos: software libre e independencia de plataforma. La interoperabilidad otra característica evaluada muestra en sus resultados que la herramienta que fue mejor evaluada correspondió a la herramienta Trac seguida de la herramienta Grannt Project y la de peor valoración les correspondió a las herramientas: MS Project y Merlin. Por otro lado, la característica, manejo de datos sorprende por sus resultados debido a que la mejor herramienta es Omniplan y la de pero valoración fue para la herramienta MS Project. La característica,

administración de proyecto que tiene los atributos específicos correspondientes a la gestión de proyectos como tareas, ruta critica, administración de múltiples escenarios y administración de múltiples proyectos, se observa en su evaluación que la herramienta mejor evaluada fue MS Project seguida por las herramientas: Trac y dotProject. Finalmente, la última característica, misceláneos, se aprecia que la herramienta con mayor valoración fue para la herramienta Trac seguida por las herramientas: Merlin y Omniplan, estando por encima de MS Project y de dotProject. De acuerdo al resultado promedio se observa que la mejor herramienta corresponde a la herramienta Trac. Las demás herramientas tuvieron una valoración promedio de regular. Es de aclarar que en este caso se está buscando una herramienta para la gestión de proyecto, por tanto, es de suma importancia los resultados obtenidos en la característica de administración de proyectos, y la mejor valoración fue para la herramienta MS Project seguida de la herramienta Trac. Se puede concluir que si las herramientas con mejor valoración en la característica de administración de proyectos su valoración promedio sea una valoración buena o excelente, esto depende de las otras características y caso contrario ocurre que una buena valoración en las demás características no garantiza una herramienta buena en la gestión de proyectos.

Tabla 14. Comparación de los resultados de todos los productos

		MS Project	Trac	Planner	Gantt Project	dotProject	Merlin	Fasttrack9	Omniplan
Interfaz Grafica	Suficiencia de entrada	-	+	-	++	+	+	~	~
	Soporte Web	+	++	--	+	++	-	-	+
	Internacionalización	++	n/a	++	++	++	-	+	--
	Manejo intuitivo	~	~	+	+	+	+	+	+
	Estética	+	+	+	++	++	+	+	+
Económica	Costo	--	++	++	++	++	~	-	+
	Cuota de mercado	++	+	--	n/a	n/a	-	+	-

Tabla 14. (Continuación)

		MS Project	Trac	Planner	Gantt Project	dotProject	Merlin	Fasttrack9	Omniplan
	Software libre	--	++	++	++	++	--	--	--
	Independencia plataforma	n/a	++	++	++	++	~	~	--
Interoperabilidad	Formatos estandarizados	-	+	-	++	~	~	~	~
	Automatización	+	++	n/a	~	++	+	+	-
	Cooperación	~	++	n/a	+	++	n/a	n/a	--
	Exportar, reportes	+	+	n/a	++	~	~	~	~
Manejo de datos	Versiones	-	++	n/a	n/a	~	n/a	--	--
	Sincronización	~	+	n/a	+	++	n/a	n/a	--
Administración proyectos	Recursos	+	+	+	+	+	+	+	+
	Tareas	+	~	+	++	+	~	+	+
	Ruta critica	++	+	+	n/a	n/a	+	+	+
	Administración de múltiples-escenarios	-	n/a	~	n/a	n/a	--	--	-
	Administración de múltiples-proyectos	+	++	n/a	n/a	++	+	+	n/a
Misceláneos	Documentación	++	++	++	++	+	+	~	++
	Soporte	+	+	~	+	+	++	+	+
	Rendimiento	+	++	++	+	n/a	++	++	++
	Estabilidad	++	++	++	++	n/a	++	++	++
	Transparencia	-	++	+	n/a	n/a	+	+	+
Media:		~	+	~	~	~	~	~	~

Fuente: Kapsammer, Elisabeth. Seminar Projektorganisation Comparison of Projectmanagement-Software.

2.4 CONCLUSIONES

Es evidente la importancia que tiene la administración proyectos en las organizaciones debido a que su estrategia se traza con base en la realización de actividades que se gestionan como proyectos. Adicionalmente, las organizaciones tienen múltiples proyectos en curso que en muchos de los casos son desarrollados a distancia y con la participación de externos. Todas estás particularidades hacen que las organizaciones necesiten de un sistema

de gestión de proyectos que esté apoyado por las nuevas tecnologías de información. Por consiguiente, la industria del software hace una variada oferta de diferentes productos para la gestión de proyectos que van desde el software propietario hasta el software libre en ambientes de escritorio y Web.

La selección del software adecuado para una organización es una tarea compleja debido al análisis de los requerimientos y de la evaluación de los diferentes productos. Por lo cual, se vuelve una necesidad el poder medir y comparar las diferentes soluciones del mercado con el fin de hacer la mejor selección en relación a las necesidades de la organización.

Por otra parte, de acuerdo al análisis realizado en el apartado 2.3.4 Análisis de resultados, se puede decir que los productos ofertados que están dentro del dominio de gestión de proyectos no cumplen siempre con las necesidades básicas para la administración de proyectos.

Finalmente, unas de las dificultades que tienen las organizaciones es la selección del producto software en relación a sus requerimientos y la calidad del producto, mencionado anteriormente. Por eso en el capítulo siguiente se hace la propuesta de un instrumento metodológico que será de gran utilidad para la evaluación y comparación de calidad para el producto software

3. GUIA PARA LA EVALUACIÓN DE CALIDAD DE APLICACIONES WEB ADAPTADA AL MODELO DE

SOFTWARE LIBRE.

3.1 DESCRIPCIÓN DEL PROBLEMA

La industria del software no es ajena a los problemas. En los últimos años la constante de sus dificultades se relaciona a su baja calidad y productividad, a demás de su alto costo en el desarrollo y mantenimiento. La calidad no representa la solución definitiva al problema, pero si disminuye la brecha de los mismos. Dicho problema tiene uno de sus máximos alcances en el producto final, debido a que requiere de un esfuerzo constante, desgastador y altamente costoso. Por otra parte, su dependencia tecnología, metodológica y la falta de formación sobre los procesos del ciclo de vida, hacen que la construcción del software sea realizada de forma artesanal, y al final como consecuencia el producto liberado no satisface las necesidades del cliente.

Según Scalone[35], los principales problemas en el área del software son: 1) calidad insuficiente de producto final, 2) estimaciones de duración de proyectos y asignación de recursos inexactos, 3) retrasos para la entrega de los productos finales, 4) costos de desarrollo y mantenimiento de productos fuera de control, 5) escasez del personal calificado en un mercado laboral de alta demanda; y 6) tendencia del crecimiento del volumen y complejidad de los productos.

El software se ha convertido en la columna vertebral de las organizaciones. En la actualidad la gran mayoría de ellas tienen automatizados todos sus procesos para poder subsistir y ser competitivos. Dado esto, la industria del software libera una gran cantidad de soluciones software, tanto open source como propietarias, las cuales están dirigidas a diferentes tipos de clientes, dando soporte a los procesos: de negocio, productivo, administrativo, que a su vez realizan un aporte integral a la estrategia corporativa, la cual, genera las ventajas competitivas. Por ende, el software es crítico para la misión de la organización, además ser base total de sus activos.

Por otra parte, según un estudio realizado en 1995 por parte de Gartner Group, índico que más del 90% de las grandes organizaciones incrementaron sus inversiones en TI. Adicionalmente, las inversiones se realizan sin el acompañamiento de algún proceso de control. Lo anterior, refleja el alto riesgo al que se expone las organizaciones al momento de seleccionar un producto software. El problema se debe a la gran disponibilidad de software de baja calidad. Esta causa se debe a, la pequeña, mediana y grande empresa, dedicada a la producción y comercialización de software, qué en una gran parte de ella se acostumbro hacer uso de malas prácticas en el proceso. Por

[35] SCALONE, Op. cit. p. 273

consiguiente, dicha tendencia se debe a la no implementación de modelos o estándares de calidad a nivel del proceso del software. Adicionalmente, en la selección de un producto software, las organizaciones, no evalúan y comparan la calidad del software a través del uso de una metodología ingenieril. Lo anterior se debe al alto desconocimiento y complejo que puede resultar el uso de un modelo o estándar de calidad a nivel del producto.

Las organizaciones internacionales de estandarización, la academia y la industria, han realizado esfuerzos grandes para enfrentar el problema, resultado de esto, son las diferentes propuestas de modelos y estándares de calidad, dirigidas tanto al proceso y el producto software. La gran cantidad y complejidad de los mismos han aumentado la problemática, debido a qué, las organizaciones no tienen a la mano una guía metodológica que les permita elegir de manera sencilla y precisa el modelo o estándar con base en requerimientos. Ante estas necesidades para mediados del año 2006 Scalone Fernanda realizó la propuesta de una metodología para la selección de un modelo o estándar de calidad a nivel del proceso de software basado en requerimientos. A pesar de este gran aporte, no resulta suficiente, aun falta una guía metodológica que permita seleccionar el modelo o estándar más adecuado para evaluar la calidad a nivel del producto y más aun que se enfoque a desarrollos en entorno Web.

Es de resaltar que los nuevos desarrollos correspondientes a soluciones en TI se centran en la Web. Las hay desde aplicaciones de comercio electrónico, portales de contenido, hasta desarrollos a la medida y con altos niveles de complejidad. El boom de la Internet y la Web trajo consigo unas necesidades, una de ellas era el cómo evaluar la calidad de los productos Web. A raíz de esto para el mismo periodo de tiempo comenzaron a surgir varias propuestas que se centraron en trabajos desarrollados en entornos Web, los cuales, fueron los primeros aportes que permitieron evaluar su calidad.

Con esta propuesta de una guía metodológica para la evaluación de la calidad de aplicaciones Web para aplicaciones de software libre se pretende realizar un aporte a las organizaciones de una herramienta práctica que permita elegir el modelo o estándar de calidad con base a requerimientos. Dicha propuesta no es la solución definitiva a los problemas que tiene el software. Por otra parte, será un instrumento que va a disminuir el riesgo que representa la elección de un producto software, específicamente que sea en entorno Web y de software libre.

3.2 SEÑALAMIENTO DE LA SOLUCIÓN DEL PROBLEMA

3.2.1 Solución propuesta.

Una guía metodológica puede ayudar a las empresas a seleccionar el modelo o estándar más conveniente para evaluar la calidad de los desarrollos software; implantados o por implantar. Esto mejora la competitividad de las mismas, maximizando el rendimiento de varios de sus recursos, tales como, procesos transaccionales, capital, recurso humano, entre otros. Además, que sus unidades aumentan su eficiencia global. Por otra parte, se verán incrementadas sus utilidades teniendo un mayor posicionamiento en el mercado y mejorando sus procesos de negocio.

Esta propuesta ofrece una herramienta sencilla y metodológica que permite la elección adecuada del método de evaluación y la evaluación del producto. Dirigida a gerentes, directores de tecnología o profesionales en el área de las Tecnologías de Información. De esta forma, la toma decisión no se realiza con base en la subjetividad, al contrario será un acto objetivo y responsable. Y por ende, muy beneficioso para la empresa.

Evaluar la calidad de un producto antes de ser implantado trae consigo una gran cantidad de beneficios. En primer lugar, evita el costo de repetición de trabajo, tales como, correcciones de errores y pérdidas en productividad. En segundo lugar, la funcionalidad del producto es la esperada. Y por último, permite identificar carencias del producto, y dado el caso, si es viable se puede modificar de forma puntual.

Con éste trabajo se pretende que las empresas cuenten con una herramienta de uso práctico e ingenieril que permita seleccionar el más adecuado modelo o estándar de calidad para la evaluación de aplicaciones Web de software libre. El beneficio será tanto en la mejora de la calidad de software libre como en el incremento de la competitividad de las empresas. Lo primero se consigue a través de un proceso de retroalimentación entre el resultado de la evaluación del producto y el proceso de desarrollo, dándose, entre la comunidad libre y la entidad evaluadora. Lo segundo se obtiene a través de la implementación de software de calidad; para la gestión, administración y automatización de procesos transaccionales de negocio. Por consiguiente, las empresas podrán incrementar su competitividad a través del software libre a un costo mínimo, correspondientes a implementación, capacitación, y adaptación, este último de ser necesario. Además, de tener cero costo por concepto de licenciamiento.

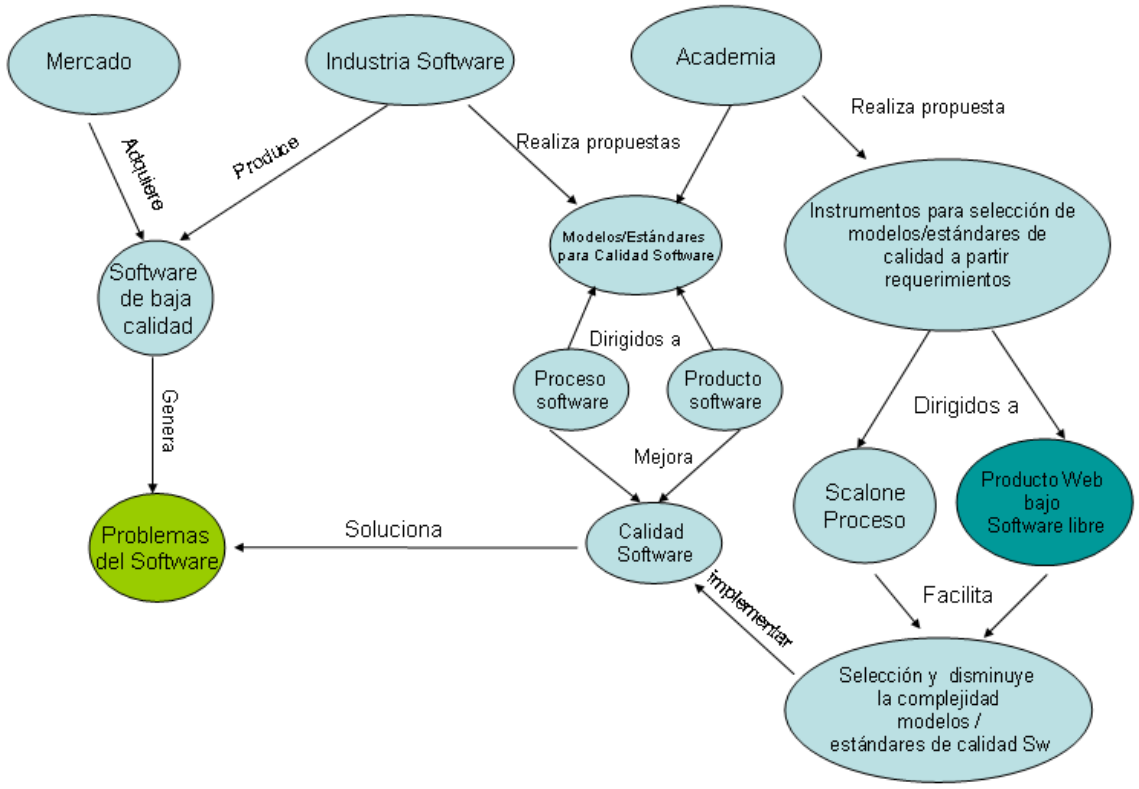
Otro de los propósitos de este trabajo es incrementar el uso de software libre.

Esta guía en su primera etapa será de gran ayuda en la búsqueda de soluciones informáticas libres a través de repositorios y gestores de proyectos. Con esto se pretende que las empresas disminuyan la brecha que existe entre sus necesidades y la disponibilidad de soluciones. Además, que las empresas podrán aprovechar las cuatro libertades que ofrece el software libre; libertad de uso, libertad de estudiarlo y adaptarlo, libertad de distribuirlo y libertad de mejorarlo y publicar las mejoras.

3.2.2 Mapa conceptual de problema y la solución.

El la figura 12 se puede apreciar un mapa conceptual del problema y de la solución de este trabajo. Son tres los entes involucrados: el mercado, la industria del software y la academia. El mercado representa a los clientes consumidores de software, la industria del software representa a las compañías desarrolladores de software como también a los desarrolladores independientes que no están representados por una compañía y que en muchos de los casos corresponde a un grupo menor de cinco personas y por último la academia que son todas las instituciones educativas que tienen proyectos de investigación en relación al software. En el medio existe una gran cantidad de software de baja calidad que genera bastantes problemas y que es producido por la industria, claro que no corresponde a todo el software producido sino una parte que es adquirido por el mercado. Por ende, la industria y la academia ven la necesidad de hacer esfuerzos que generen aportes para mitigar el problema y lo hacen con la liberación de modelos y estándares de calidad dirigiditos a todo el ciclo de vida del software específicamente al proceso y al producto. Pero aun así es muy complejo la implementación de los mismos por esto la academia a través de varios trabajos de investigación han realizado guías metodológicas para la uso de estos instrumentos de calidad como lo son: guía para la selección del modelo y estándar de calidad para el proceso del software, propuesta realizada por Scalone Fernanda en su trabajo “Estudio Comparativo de los Modelos y Estándares de Calidad del Software” y la otra propuesta es la realizada en este trabajo que corresponde a una guía metodológica para la evaluación de calidad de aplicaciones Web adaptada al modelo de software libre. La implementación de estas dos guías van ha impactar en la calidad del software permitiendo elegir el instrumento de calidad más adecuado para usar que conllevará a la disminución de los problemas del mismo, no es su totalidad debido a que no todos corresponden a la calidad.

Figura 12 . Mapa conceptual del problema y la solución



3.3 GUÍA PARA LA SELECCIÓN DEL MODELO / ESTÁNDAR PARA LA EVALUACIÓN DE CALIDAD DE APLICACIONES WEB

En la tabla 15 se observa la propuesta realizada para la selección del modelo o estándar de evaluación calidad del producto a través de requerimientos. En este trabajo se identificaron los requerimientos necesarios que podrían solventar los modelos y estándares ahí sugeridos. Por otra parte, está propuesta está abierta a seguir creciendo en relación a nuevas propuestas realizadas tanto por la industria como por la academia y a otro tipo de organizaciones que velen por el bienestar de la calidad del software. Es de resaltar que con la creciente oferta de soluciones de software libre han surgido metodologías que permiten evaluar los principios de la libertad del software libre como es el caso de QSOS (Qualification and Selection of Open Source Software, sus siglas en ingles).

Tabla 15. Determina el Modelo / Estándar según requerimiento

Modelo / Estándar	Requerimientos
GQM (Goal – Question – Metric)	Mide y evalúa la calidad del software a través de objetivos / metas a lograr. Tiene tres niveles: 1) conceptual (Objetivos / metas) definido en base a la necesidad de la empresa, 2) operacional (Preguntas), 3) cuantitativo (métricas). Es un enfoque útil para decidir que medir.
McCall	Mide la calidad del software según la visión del usuario y la visión del desarrollador.
FURPS	Evalúa la calidad en todo el ciclo de vida del producto con base en características de calidad y dos categorías de requerimientos.
BOEHM	Mide la calidad del software para que el software logre: 1) realice lo que el usuario desea, 2) utilice los recursos informáticos de manera correcta y eficiente, 3) sea fácil de utilizar y aprender, 4) sea bien diseñado, codificado, probado y mantenido.
SACT (Software Assurance Technology Center)	Permite la producción de varios proyectos en desarrollo. Evalúa la calidad del software a través de métricas relacionadas al proceso y al producto, además de permite realizar indicaciones de probabilidad de éxito de objetivos o metas relacionados al producto y atributos del proceso.
Dromey	Realiza la evaluación de tres etapas: 1) determinación de requerimientos, 2) diseño e 3) implementación, las cuales pueden ser usadas para elaborar, comparar, y evaluar la calidad de los productos de software.
C-QM	Propone un modelo de calidad comprensivo que puede ser aplicado efectivamente para evaluar diversos aspectos de la calidad de software.
SQAE (Software Quality Assessment Exercise)	Permite cuantificar los riesgos asociados al software y la calidad del software. Además, se aplica en todas las etapas del ciclo de vida del software.
WebSite QEM (Web Quality Evaluation Method)	Evalúa y compara la calidad de aplicaciones Web de cualquier tipo de dominio.
ISO/IEC 9126-1 Quality Model	Mide la calidad del producto de cualquier tipo de software a través de la calidad interna, calidad externa y calidad en uso.

Tabla 15 (Continuación)

Modelo / Estándar	Requerimientos
QSOS (Qualification and Selection of Open Source Software)	Evalúa, selecciona y compara aplicaciones software de open sources o software libre.

A partir de la tabla 15 si un cliente quiere evaluar varios productos software debe primero identificar los requerimientos esenciales de la evaluación. Esto sería, identificar el dominio al que pertenece el producto, determinar el ambiente de implementación, establecer si la evaluación se hace en todo el ciclo de vida del software o en especial en una fase, fijar si la evaluación se va tener en cuenta el punto de vista del usuario y del desarrollador y si se va ha determinar la calidad del producto a través de la calidad interna, externa o en uso, con base en estos cuestionamientos se podrá determinar el modelo o estándar más adecuado. Por ejemplo, el domino al que pertenece los productos a evaluar es comercio electrónico y su ambiente de implementación es Web y su evaluación es sobre el producto y no sobre el proceso en el ciclo de vida del software. Dado ésta información se podría decir que el modelo más indicado para evaluar la calidad del software es WebSite QEM.

Por otro lado, se puede determinar en general que los modelos y estándares de calidad del producto tiene como objetivo evaluar la calidad del software a partir de unos requerimientos de calidad preestablecidos por estándares específicos que son ajustados a las necesidades particulares del cliente. Dichos requerimientos son medidos a través de métricas que permiten cuantificar el grado de calidad de un producto software para hacer su comparación y selección. Estos modelos y estándares miden la calidad del producto con fin de establecer el nivel de complacencia del cliente en relación de las expectativas que tiene con respecto a sus necesidades. Por otra parte, existen modelos específicos para evaluar la calidad del producto según sea su entorno o ambiente de implementación como es el caso de las aplicaciones Web, y por otra parte existen otros modelos para evaluar productos que están bajo el moviendo del software libre o código abierto.

3.4 ETAPAS DE LA GUÍA

La guía metodológica pretende ser un instrumento sencillo que permite elegir de forma adecuada el modelo o estándar de calidad para evaluar la calidad de

un producto. Dicha guía se compone de tres etapas: evaluación, implementación y análisis, las cuales a su vez están compuesta de varias actividades. A continuación se muestra las etapas y actividades de esta propuesta. Ver figura 13.

Etapa1 – Evaluación:

- Determinar requerimientos.
- Estudiar el dominio al que pertenece el producto.
- Buscar soluciones en comunidades libres.
- Depurar listado.
- Seleccionar modelo / estándar.

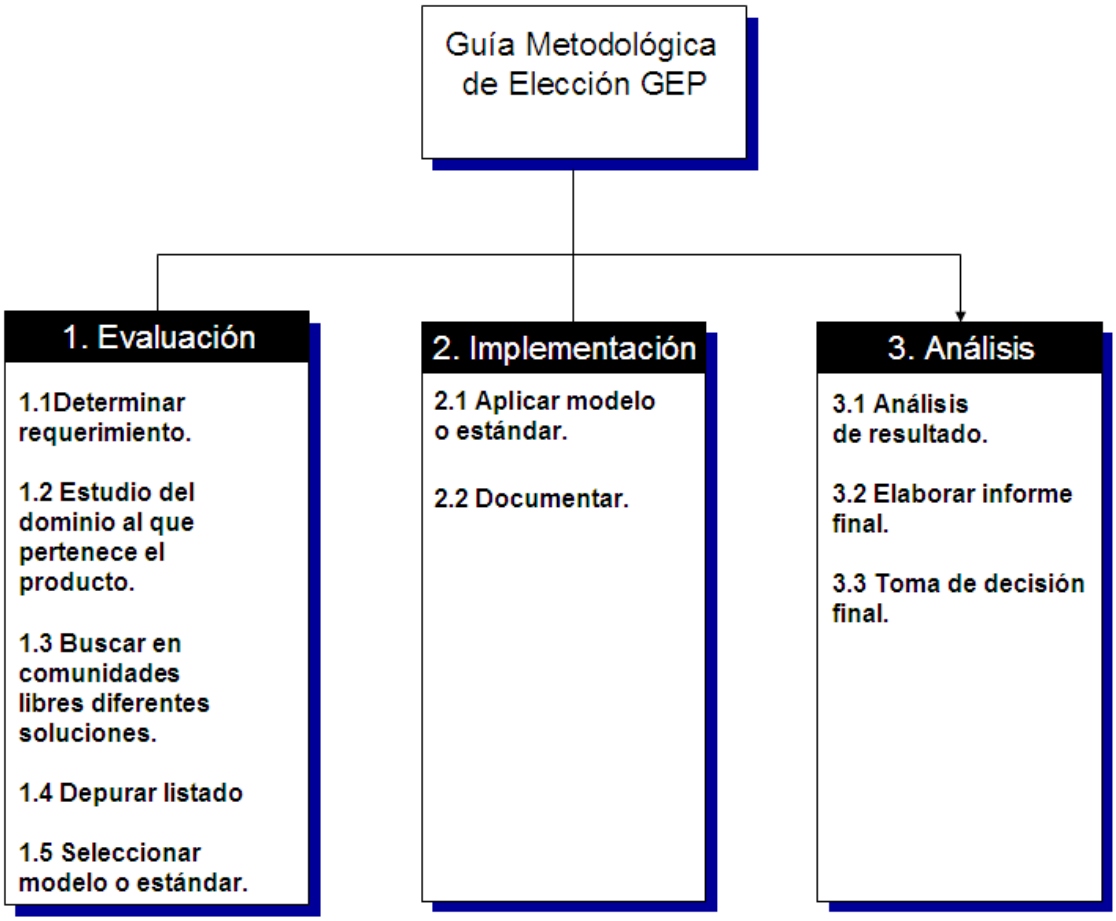
Etapa 2 – Implementación:

- Aplicar modelo / estándar de calidad
- Documentación.

Etapa 3 – Análisis

- Analizar resultados.
- Elaborar informe final.
- Tomar decisión final.

Figura 13. Guía Metodológica de elección GEP



ETAPA 1 – EVALUACIÓN

El objetivo de esta etapa es evaluar el proyecto en todas sus directrices

Paso 1 – Determinar requerimientos (1.1).

De acuerdo a las necesidades del cliente este debe determinar los aspectos generales que debe tener el producto. Dicha información debe ser registrada en el siguiente formulario, ver la figura 14.

Figura 14. Formulario de aspectos generales del producto

SELECCION DEL MODELO / ESTANDAR DE CALIDAD PARA EVALUAR EL PRODUCTO SOFWARE

FORMULARIO DE REQUERIMIENTOS **FECHA:**

Actividad o Proceso que se quiere embestir:

_____
_____
_____
_____
_____
_____
_____

Nombre Genérico del Producto Software que se con el que se conoce en el medio:_____

Arquitectura:	**Usuarios:**
• Escritorio __ • Escritorio Cliente/Servidor__ • Web__	• Monousuario__ • Multiusuario__
Licencia:	**Plataforma:**

• Software Libre____
  • Con Protección Heredada__
  • Sin Protección Heredada__
  • Dominio publico___
• Software propietario____
	• Multiplataforma_____
	• Plataforma tecnológica:

Paso 2 – Estudiar el dominio al que pertenece el producto. (1.2).

De acuerdo a las necesidades expresadas por el cliente en el paso 1, se debe identificar el domino al que pertenecen las necesidades y realizar un estudio sobre el mismo, con el fin de contextualizar el alcance que puede tener la solución. Por otra parte, se debe diligenciar el respectivo formulario correspondiente a la definición del dominio, ver la figura 15.

Figura 15.. Lista de productos software

SELECCION DEL MODELO / ESTANDAR DE CALIDA PARA EVALUAR EL PRODUCTO SOFWARE

DEFINICIÓN DEL DOMINIO **FECHA:**

Descripción

Dominio:

Estudio sobre dominio:

Paso 3 – Buscar soluciones en comunidades software libres. **(1.3).**

A partir de los requerimientos definidos en el paso 1 y el paso 2 se procede ha realizar una búsqueda de productos software en los diferentes repositorios de código fuente de software libre. A continuación se mencionan algunos repositorios.

- SourceForget.net

- Google Code
- BerliOs
- GForge
- Freshmeat.net

Por otra parte, también se puede realizar la búsqueda de los diferentes estudios comparativos del tipo de de soluciones software que se busca. Esto se puede hacer directamente sobre un buscador Web a través de las palabras claves: list of o lista de y nombre o siglas del software. Por ejemplo, list of ERP, busca el listado de los diferentes herramientas Enterprise Resource Planning.

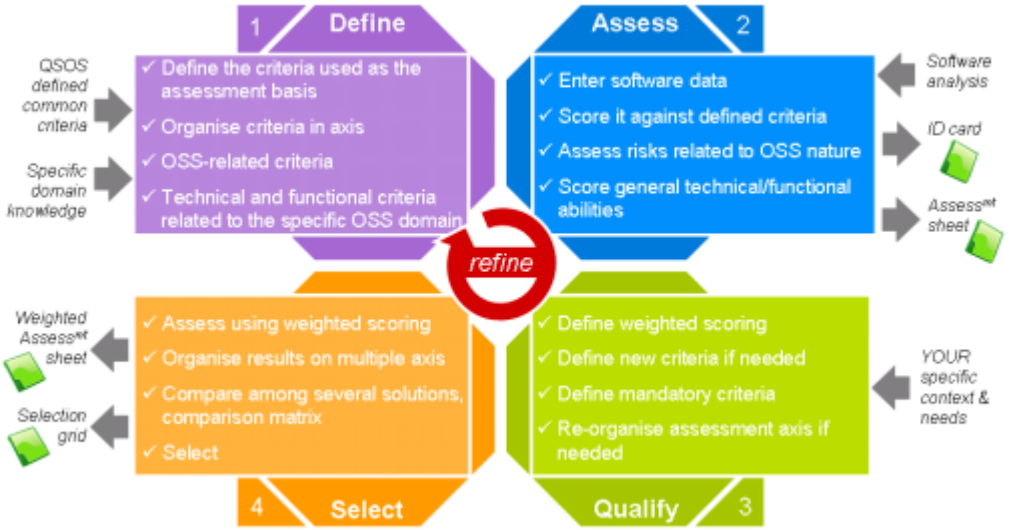
3.5 PROPUESTA DE CARACTERÍSTICAS, SUBCARACTERÍSTICAS Y ATRIBUTOS PARA EVALUAR APLICACIONES DE SOFTWARE LIBRE

Las compañías tienen a disposición una gran variedad de soluciones informáticas que pueden ser del orden de software privativo o de software libre. Es de anotar que este último es muy rico y tiene una alta cobertura. Por otro lado, independientemente del movimiento al que pertenezca el software las compañías tienen la necesidad de evaluarlo para elegir el más adecuado con respecto a sus necesidades y requerimientos. Por tanto, en el caso del software libre se requiere de un método ingenieril altamente objetivo que permita medir la particularidad, es decir, las características propias de este tipo de software, con el fin determinar los requerimientos y el riesgo especifico que trae el uso del mismo. Por consiguiente, las compañías deben formularse las siguientes preguntas en relación al uso del software libre. 1) cual es la durabilidad del software?, 2) cuales son los riegos de una bifurcación?, y como se anticipa para su manejo?, 3) cual es el nivel de estabilidad esperada del software, y como maneja su disfunción? 4) cuál es el nivel de disponibilidad esperado del software?, y 5) es posible la influencia de futuros desarrollos con nuevas funcionalidades?.

La mejor forma de encontrar respuesta al anterior grupo de preguntas es a través de un método que permita evaluar los requerimientos y riesgos del software libre. La compañía Atos Origin es líder internacional en la prestación de servicios en el área de tecnologías de información, y por iniciativa propia propuso un método que permite evaluar y seleccionar software libre. Dicho método se conoce con el nombre QSOS (Method for Qualification and Selection of Open Source Software, sus siglas en ingles). QSOS está bajo licenciamiento libre, específicamente bajo una licencia GFDL (GNU Free Documentation License, sus siglas en ingles).

Las compañías que implementan QSOS pueden alcanzar el objetivo de evaluar y seleccionar el software libre más adecuado para sus requerimientos y necesidades. Dicho proceso, se realiza con base en la evaluación de tres aspectos: 1) definición grado de funcionalidad, 2) riesgo desde la perspectiva del usuario, y 3) riesgo desde la perspectiva del servicio proporcionado. Además es un método interactivo compuesto por tres etapas. Etapa1: definición, etapa 2: evaluación, etapa 3: calificación y etapa 4: selección, ver la figura 16.

Figura 16. Etapas de QSOS



Fuente: Method for Qualification and Selection of Open Source software (QSOS)-

Por otro lado, QSOS propone características, subcaracterísticas y atributos para medir la calidad del software libre en relación a dos aspectos anteriormente mencionados: el riesgo desde la perspectiva del usuario, y el riesgo desde la perspectiva del servicio proporcionado. A continuación se observa las características principales según los dos aspectos anteriormente mencionados.

- Características según el riesgo desde la perspectiva del usuario:
  - Durabilidad intrínseca
  - Solución industrializada
  - Integración
  - Capacidad de adaptación técnica

- Estrategia

- Características según riesgo desde la perspectiva del servicio proporcionado:
  - Mantenimiento
  - Código maestro

En la tabla 16 se aprecian todas las características, subcaracterísticas y atributos propuestos por QSOS para medir la calidad correspondiente al software libre. Es de apreciar que el score o puntaje de evaluación para este método están entre el rango de 0 a 2, siendo 0 la peor calificación y 2 la mejor calificación.

Tabla 16. Características, subcaracterísticas y atributos QSOS

a) Características, subcaracterísticas y atributos desde la perspectiva del usuario

Durabilidad intrínseca		Puntuación		
		0	***1***	***2***
Madurez	Edad	Menos de 3 meses.	Entre 3 meses y 3 años.	Más de 3 años.
	Estabilidad	Software inestable con numerosos lanzamientos o parches y generación de efectos secundarios.	Estable el producto actualmente liberado pero antiguo. Dificultades para estabilizar las versiones siguientes.	Software estable. Liberaciones siguientes presenta errores con correcciones correspondientes a funcionalidades nuevas.
Madurez	Historia, conocimiento de problemas	Se tiene conocimiento de varios problemas del software el cual puede ser prohibitivo	No se conoce de problemas principales o crisis.	Se tiene historia de la buena gestión de situaciones de crisis.

Tabla 16. (Continuación)

Madurez	Probabilidad de bifurcación	El software es muy probable que se bifurque en el futuro.	El software viene de una bifurcación, pero tiene muy pocas posibilidades que se bifurque en el futuro.	El software tiene muy pocas posibilidades de que se bifurque. No procede de una bifurcación.
Adopción	Popularidad (relacionados a: publico general, nicho, ...)	Muy pocos usuarios identificados	Detectable sobre el uso Internet (SourceForge, Freshmeat, Google, ...)	Numerosos usuarios, numerosas referencias
	Referencias	Ninguna	Pocas referencias, los usos no críticos	Frecuentemente implementada por aplicaciones criticas.
Durabilidad intrínseca		Puntuación		
		0	***1***	***2***
	Contribución a la comunidad	Ninguna comunidad o sin actividad real (foro, correo lista,...)	Existe en la comunidad con una actividad notable	Comunidad fuerte: gran actividad en foros, numerosas contribuyentes y defensores
	Books	No tiene libros acerca del software.	Menos de 5 libros acerca del software que están disponibles.	Más de 5 libros acerca del software que están disponibles, en varios lenguajes.
Liderazgo en desarrollo	Equipo líder	1 a 2 personas involucradas, no claramente identificadas	Entre 2 y 5 personas independientes	Más de 5 personas
	Estilo de Gestión	Dictadura completa	Iluminado despotismo	Consejo de Arquitectos con identificación líder(e.g: ASF, ...)
Actividad	Desarrolladores, identificadores, rendimiento.	Menos de 3 desarrolladores, claramente no identificación.	Entre 4 y 7 desarrolladores, o más desarrolladores no identificados con importante rendimiento.	Más de 7 desarrolladores claramente identificados, un equipo muy estable.

Tabla 16. (Continuación)

Actividad	Actividad de errores	Lenta reacción en el foro o en lista de correo, o de nada sobre estimación de error notas de la versión	Detectables pero sin proceso claramente expuestas, larga reacción /resolución tiempo	Herramientas para le manejo de errores, con una fuerte interacción con el estado de evolución del software
	Actividad de liberación	Actividad muy débil en tanto la producción y liberaciones de nuevas versiones	Actividad en la producción y liberación de nuevas versiones. Frecuente liberaciones pequeñas correspondientes a soluciones de errores.	Importante actividad, con frecuentes lanzamientos de menor importancia para errores y los lanzamientos previstos de versiones principales relativos a la predicción del plan de trabajo
Durabilidad intrínseca		Puntuación		
		0	***1***	***2***
Independencia de desarrollo	Independencia de desarrollo	Desarrollo realizado de 100% por empleados de una sola compañía	60% Máximo	20% Máximo
Solución Industrializada		Puntuación		
		0	***1***	***2***
Servicios	Entrenamiento	No oferta de entrenamiento identificadas	Oferta existe, pero se limita geográficamente y con una lengua o es proporcionado por un solo fuente	Proporciona una rica y variada Ofrece una rica y variada ayuda de varias fuentes, en varios lenguajes divida en varios niveles.
	Soporte	Ninguna oferta de apoyo excepto a través de foros públicos y lista de correos	Oferta existe, pero es proporcionada por un solo contratista sin fuertes compromisos de calidad de servicios	Varios proveedores de servicios con un fuerte compromiso (por ejemplo: el tiempo de resolución garantizada)

Tabla 16. (Continuación)

	Consulta	Ninguna oferta de servicios de consultoría.	Oferta existe, pero se limita geográficamente y con una lengua o es proporcionado por un solo contratista.	Los servicios de consultoría prestados por contratistas son en diferentes idiomas.
Documentación	Documentación	No usa documentación	Documentación existe sino que se desplaza en el tiempo, se limita a un idioma o no está bien detallado	Documentación siempre al día, traducido y posiblemente adaptada a diferentes lectores (usuario final, administración de sistemas, administrador,?)
Aseguramiento de Calidad	Aseguramiento de Calidad	No tiene ningún proceso AC	Tiene identificado AC pero no esta formalizado y sin herramienta.	Proceso de pruebas automáticas de código incluido en la vida de ciclo con la publicación de los resultados
Solución Industrializada		Puntuación		
		0	***1***	***2***
Aseguramiento de Calidad	Aseguramiento de Calidad	No tiene ningún proceso AC	Tiene identificado AC pero no esta formalizado y sin herramienta.	Proceso de pruebas automáticas de código incluido en la vida de ciclo con la publicación de los resultados
	Herramientas	Herramienta para le manejo de errores.	Herramientas estándar proporcionado (por ejemplo: hosting forge), pero mal usados.	Muy activo en el uso de herramientas para las funciones y tareas de asignación y monitoreando el progreso de errores

Tabla 16. (Continuación)

Empaquetado	Código fuente	El software no se puede instalar desde el código fuente, sin un cúmulo de trabajo.	La instalación de los fuentes es limitada y depende de condiciones muy estrictas (OS, arco, lib, ...)	La instalación desde el código fuente es sencilla.
	Debian	El software no está empaquetado por Debian.	Un paquete de Debian existe, pero tiene problemas importantes o que no tiene apoyo financiero.	El software está empaquetado en la distribución.
	FreeBSD	El software no está empaquetado por FreeBSD.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	Existe un paquete oficial en FreeBSD.
	HP-UX	El software no está empaquetado por HP-UX.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	Un paquete probado es proporcionado por HP-UX.
	MacOSX	El software no está empaquetado por MacOSX.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	El software es empaquetado en la distribución.
Solución Industrializada		Puntuación		
		0	***1***	***2***
Empaquetado	Mandrive	El software no está empaquetado por MacOSX.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	El software es empaquetado en la distribución.
	NetBSD	El software no está empaquetado por MacOSX.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	Existe un paquete oficial en NetBSD.

Tabla 16. (Continuación)

Empaquetado	OpenBSD	El software no está empaquetado por OpenBSD.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	Existe un paquete oficial en OpenBSD.
	RedHat / Fedora	El software no está empaquetado por RedHat / Fedora.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	El software es empaquetado en la distribución.
	Solaris	El software no está empaquetado por Solaris.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	El software es soportado por Sun de Solaris.
	SuSE	El software no está empaquetado por SuSE.	Existe un paquete, pero tiene problemas importantes y no tiene un soporte oficial.	El software es empaquetado en la distribución.
	Windows	El proyecto no puede ser instalado en Windows.	Existe un paquete pero limitado o con importantes publicaciones o con cubrimientos específicos para las liberaciones de Windows.	Windows soporta completamente y un paquete es proporcionado.

Tabla 16. (Continuación)

Solución Industrializada		Puntuación		
		0	***1***	***2***
Explotabilidad	Facilidad de uso, ergonomía.	Difícil de usar, requiere un conocimiento en profundidad de la funcionalidad del software.	Austera y muy técnico ergonomía.	GUI incluye ayudas de funciones y ergonomías elaboradas (e.g: skins/themes management).
	Administración / Control.	No administración o monitoreo de funcionalidades.	Existen funcionalidades pero incompletas y requiere de mejoramiento.	Administración completa y de fácil uso y monitoreo de funcionalidades. Posible integración con herramientas externas. (e.g: via SNMP, ...).
Adaptabilidad Técnica				
Modularidad	Modularidad	Software monolítico.	Presencia de módulos de alto nivel que permite un primer nivel de adaptación de software.	Concepción modular, que permite la fácil adaptación del software mediante la selección de módulos o nuevos desarrollos.
Productos	Modificación del Código	Todo por mano.	Posible recompilación, aunque compleja, sin necesidad de herramientas o documentación.	Recopilación con las herramientas (por ejemplo: marca, ANT, ...) y siempre con documentación.
	Extensión del Código	Cualquier modificación requiere recompilar el código	Arquitectura diseñada para la extensión estática, sino que requiere la recompilación	Principio de extensión, arquitectura diseñada para la extensión dinámica sin recompilación

Tabla 16. (Continuación)

Estrategia		Puntuación		
		0	***1***	***2***
Licencia	Permisividad (que se ponderarán solo si el usuario quiere convertirse en propietario de código)	Licencia muy estricta, como GPL	Licencia moderadamente permisiva entre los dos extremos (GPL y BSD), doble licenciamiento dependiendo del tipo de usuario (persona, compañía) o sus actividades.	Muy permisiva como las licencias BSD o Apache
	Protección propietaria contra bifurcaciones	Muy permisiva como las licencias BSD o Apache	Licencia moderadamente permisiva entre los dos extremos (GPL y BSD), doble licenciamiento dependiendo del tipo de usuario (persona, compañía) o sus actividades.	Licencia muy estricta, como GPL
Derechos de autor	Derechos de autor	Derechos de autor en manos de unos pocos individuos o entidades, por lo que es más fácil de cambiar la licencia	Derechos autor en manos de muchas personas que poseen el código de una manera homogénea, lo que hace muy difícil su renovación.	Derechos autor en poder de una persona jurídica en quien confía en la comunidad (por ejemplo, la FSF o ASF)
Modificación del código fuente	Modificación del código fuente	No hay forma práctica de proponer modificación del código.	Herramientas proporcionadas para acceder y modificar el código (como CVS o SVN), pero no se ha usado para desarrollar el software.	El proceso de modificación del código está bien definido, expuesto y respetado, sobre la base de las funciones de asignación.

Tabla 16. (Continuación)

Estrategia		Puntuación		
		0	***1***	***2***
Estado actual de desarrollo del software (roadmap)	Estado actual de desarrollo del software (roadmap)	No tiene publicado estado actual de software(roadmap)	Plan de trabajo existente no tiene ninguna planificación.	Estado actual de desarrollo del software (roadmap) con planificación y medición de retrasos.
Patrocinador	Patrocinador	Software no tiene un patrocinador, el equipo central no se paga.	El software tiene un patrocinador único que puede determinar su estrategia.	El software es patrocinado por la industria.
Independencia Estratégica	Independencia Estratégica	Ninguna estrategia detectables o una fuerte dependencia de un actor único (persona, empresa, patrocinador, ...)	Visión estratégica compartida con varios u otros proyectos de código libre y abierto, pero sin un fuerte compromiso de los propietarios de los derechos de autor	Una fuerte independencia del equipo básico, los derechos de persona jurídica, tenencia, y una fuerte participación en el proceso de normalización.

Fuente: Method for Qualification and Selection of Open Source software (QSOS).

b) Características, subcaracterísticas y atributos desde la perspectiva suministrada por el servicio.

Tabla 16. (Continuación)

Prestación servicios		Puntuación		
		0	***1***	***2***
Mantenimiento	Calidad del código fuente	Código no muy legible o de mala calidad, la incoherencia en la codificación de estilos	Código legible, pero no comentado al detalle.	Legible y código comentado, con implementación de patrones clásicos y con una política de codificación coherente.
	Dispersión tecnológica	Uso de lenguajes diferentes.	Un lenguaje principal, con ciertos módulos codificados en otros lenguajes para los requisitos limitados y específicos.	Un único lenguaje.
	Complejidad intrínseca	Código muy complejo que requiere de alto nivel de conocimientos para llevar a cabo modificaciones sin la generación de efectos secundarios	No es un código muy complejo, pero que aún requieren conocimientos en lenguajes de programación y de diseño de software.	Código y diseño simple, fácil de modificar.

		No hay documentación (guía de desarrollo o generado automáticamente como javadoc)	Documentación incompleta o desactualizado, sin concepción y consideraciones de la arquitectura.	Detallada y actualizada la documentación, incluye la concepción, diseño y consideraciones de la arquitectura.

Tabla 16. (Continuación)

Prestación servicios		Puntuación		
		0	***1***	***2***
Dominio del código	Directo	Ninguna experticia directa del código fuente.	Dominio de código, pero limitada a una sola persona o sólo una parte del código fuente	Dominio de código por varios individuos que cubren en conjunto la totalidad del código fuente.
	Indirecto	Ninguna experticia indirecta del código fuente.	Fuerte dominio a través de expertos externos proporcionados por los socios	Asociación con el dueño de los derechos de autor y / o el equipo básico

Fuente: Method for Qualification and Selection of Open Source software (QSOS).

Parte del objetivo principal de este trabajo como se ha mencionado anteriormente es hacer la propuesta de una guía metodológica para evaluar la calidad de aplicaciones en entorno Web adaptada al modelo de software libre. Para hacer cumplimiento a esto último: adaptada al software libre, se hace la propuesta de características, subcaracterísticas y atributos para medir la calidad en relación a aspectos particulares correspondientes a este tipo de software. Dicha propuesta se realizó con base en el método QSOS. Por otra parte, el criterio de selección fue en relación a la experiencia del evaluador, por tanto es flexible a modificar según sea el caso de evaluación. En la tabla 17 se pueden observar las características, subcaracterísticas y atributos propuestos. Adicionalmente, la propuesta que se hace en este trabajo correspondiente a la definición de evaluación de cada atributo que correspondería al criterio de tipo elemental, más sin embargo, se dejo con el mismo score o criterio de evaluación pero con una adaptación en sus valores que van de 0 a 100, siendo 0 la peor calificación y 100 la mejor.

Tabla 17. Propuesta de características, subcaracterísticas y atributos

Durabilidad intrínseca		Puntuación		
		0	***1***	***2***
Madurez	Edad	Por ejemplo, menos de 3 meses.	Por ejemplo, entre 3 meses y 3 años.	Por ejemplo, más de 3 años.
	Estabilidad	Software inestable con numerosos lanzamientos o parches y generación de efectos secundarios.	Estable el producto actualmente liberado pero antiguo. Dificultades para estabilizar las versiones siguientes.	Software estable. Liberaciones siguientes presenta errores con correcciones correspondientes a funcionalidades nuevas.
	Probabilidad de bifurcación	El software es muy probable que se bifurque en el futuro.	El software viene de una bifurcación, pero tiene muy pocas posibilidades que se bifurque en el futuro.	El software tiene muy pocas posibilidades de que se bifurque. No procede de una bifurcación.
Durabilidad intrínseca				
Adopción	Popularidad (relacionados a: publico general, nicho, ...)	Muy pocos usuarios identificados	Detectable sobre el uso Internet (SourceForge, Freshmeat, Google, ...)	Numerosos usuarios, numerosas referencias
	Referencias	Ninguna	Pocas referencias, los usos no críticos	Frecuentemente implementada por aplicaciones criticas.

	Contribución a la comunidad	Ninguna comunidad o sin actividad real (foro, correo lista,...)	Existe en la comunidad con una actividad notable	Comunidad fuerte: gran actividad en foros, numerosas contribuyentes y defensores
	Books	No tiene libros acerca del software.	Menos de 5 libros acerca del software que están disponibles.	Más de 5 libros acerca del software que están disponibles, en varios lenguajes.

Tabla 17 (Continuación)

Durabilidad intrínseca		Puntuación		
		0	***1***	***2***
Liderazgo en desarrollo	Equipo líder	1 a 2 personas involucradas, no claramente identificadas	Entre 2 y 5 personas independientes	Más de 5 personas
Solución Industrializada				
Servicios	Entrenamiento	No oferta de entrenamiento identificadas	Oferta existe, pero se limita geográficamente y con una lengua o es proporcionado por un solo fuente	Proporciona una rica y variada Ofrece una rica y variada ayuda de varias fuentes, en varios lenguajes divida en varios niveles.
	Soporte	Ninguna oferta de apoyo excepto a través de foros públicos y lista de correos	Oferta existe, pero es proporcionada por un solo contratista sin fuertes compromisos de calidad de servicios	Varios proveedores de servicios con un fuerte compromiso (por ejemplo: el tiempo de resolución garantizada)
Aseguramiento de Calidad	Aseguramiento de Calidad	No tiene ningún proceso AC	Tiene identificado AC pero no esta formalizado y sin herramienta.	Proceso de pruebas automáticas de código incluido en la vida de ciclo con la publicación de los resultados
Estrategia				

Licencia	Permisividad (que se ponderarán solo si el usuario quiere convertirse en propietario de código)	Licencia muy estricta, como GPL	Licencia moderadamente permisiva entre los dos extremos (GPL y BSD), doble licenciamiento dependiendo del tipo de usuario (persona, compañía) o sus actividades.	Muy permisiva como las licencias BSD o Apache

Tabla 17 (Continuación)

Estrategia		Puntuación		
		0	***1***	***2***
Licencia	Protección propietaria contra bifurcaciones	Muy permisiva como las licencias BSD o Apache	Licencia moderadamente permisiva entre los dos extremos (GPL y BSD), doble licenciamiento dependiendo del tipo de usuario (persona, compañía) o sus actividades.	Licencia muy estricta, como GPL
Derechos de autor	Derechos de autor	Derechos de autor en manos de unos pocos individuos o entidades, por lo que es más fácil de cambiar la licencia	Derechos autor en manos de muchas personas que poseen el código de una manera homogénea, lo que hace muy difícil su renovación.	Derechos autor en poder de una persona jurídica en quien confía en la comunidad (por ejemplo, la FSF o ASF)

Modificación del código fuente	Modificación del código fuente	No hay forma práctica de proponer modificación del código.	Herramientas proporcionadas para acceder y modificar el código (como CVS o SVN), pero no se ha usado para desarrollar el software.	El proceso de modificación del código está bien definido, expuesto y respetado, sobre la base de las funciones de asignación.
Patrocinador	Patrocinador	Software no tiene un patrocinador, el equipo central no se paga.	El software tiene un patrocinador único que puede determinar su estrategia.	El software es patrocinado por la industria.

Tabla 17 (Continuación)

Estrategia		Puntuación		
		0	***1***	***2***
Independencia Estratégica	Independencia Estratégica	Ninguna estrategia detectables o una fuerte dependencia de un actor único (persona, empresa, patrocinador, ...)	Visión estratégica compartida con varios u otros proyectos de código libre y abierto, pero sin un fuerte compromiso de los propietarios de los derechos de autor	Una fuerte independencia del equipo básico, los derechos de persona jurídica, tenencia, y una fuerte participación en el proceso de normalización.

Fuente: Method for Qualification and Selection of Open Source software (QSOS)-

3.6 CONCLUSIONES

Los problemas que menciona Scalone en la sección 3.1 en relación al software y la baja calidad en algunos productos software, determinan que esta industria tiene grandes dificultades con respecto a la calidad de sus productos y la

calidad en sus procesos. Debido a lo anterior la industria y la academia han realizado grandes esfuerzo por desarrollar modelos y estándares de calidad dirigidos a la calidad del software tanto para el producto como para el proceso, pero aun así no es suficiente y no es garantía que hacer uso de estos medios mejore la calidad y solucione los problemas actuales del software.

Adicionalmente, seleccionar el modelo o estándar adecuado para evaluar la calidad del producto es una actividad compleja debido a la gran oferta de los mismos, y hacer una buena selección reduce costos, tiempo, y tiene un mayor enfoque en relación a los requerimientos del cliente con respecto a la evaluación. Por ende, la guía metodológica propuesta en este capítulo es un instrumento que ayuda y orienta la evaluación de un producto software Web bajo el movimiento de software libre. Lo anterior no va mejorar la calidad del producto sino se hacen los ajustes correspondientes y se revalúa el proceso, pero si ayuda a las organizaciones a hacer una buen proceso evaluación y selección.

En el siguiente capítulo se aprecia en detalle el uso de la guía metodológica a través de un caso de estudio. En dicho caso de estudio se van a evaluar tres herramientas software libres en ambiente Web que su dominio es la administración de proyectos.

4. CASO DE ESTUDIO APLICANDO LA GUÍA METODOLOGÍA PARA LA EVALUACIÓN DE CALIDAD DE APLICACIONES WEB ADAPTADA AL MODELO DE SOFTWARE LIBRE

4.1 INTRODUCCIÓN

En este capítulo se presenta el desarrollo del caso de estudio aplicando la guía metodológica propuesta que comprende tres etapas: (1) evaluación, (2) implementación y (3) análisis, las cuales en cada una de ellas tiene una serie de actividades o pasos. La etapa de evaluación esta compuesta por cinco actividades relacionadas en el apartado 4.2 que corresponde a los siguientes pasos, paso 1: determinar requerimiento, paso 2: estudio del dominio al que pertenece el producto, paso 3: buscar en comunidades libres diferentes soluciones, paso 4: depurar listado, y paso 5: seleccionar método o estándar, ver figura 13. En la etapa siguiente de implementación la componen básicamente dos actividades: (1) aplicar modelo o estándar y (2) documentar. La primera actividad puede estar compuesta por muchas otras actividades y esto varia con respecto al modelo o estándar seleccionado, en este caso el modelo seleccionado es Web-site QEM, por lo tanto la componen tres principales fases: (1) definición y especificación de requerimientos, referenciado en el apartado 4.3.2, (2) evaluación elemental: definición e

implementación correspondiente al apartado 4.3.3, y (3) evaluación global, referenciado en el apartado 4.3.4. A su vez, cada fase está compuesta por varias actividades. Las actividades que componen la primera fase son: (1) selección del dominio, referenciado en el apartado 4.3.2.1, (2) planificar y programar la evaluación de calidad, referenciado en el apartado 4.3.2.2, (3) perfil de audiencia, referenciado en el apartado 4.3.2.3, (4) características principales del dominio para PMS, referenciado en el apartado 4.3.2.4, y (5) árbol de requerimientos de calidad, referenciado en el apartado 4.3.2.5. Por otra parte, las actividades que componen la fase 2 son: (1) plantillas de atributos de calidad que se referencian en el apartado 4.3.3.1, (2) evaluación elemental, referenciado en el apartado 4.3.4.2. Y por último, la tercera fase, la cual tiene las siguientes actividades: (1) estructura de agregación de referencias parciales usando el modelo LSP, referenciado en el apartado 4.3.4.1, (2) resultados de los valores de las referencias parciales y globales de calidad, para los tres sitios PMS evaluados que se referencian en el apartado 4.3.4.2. Por último, la etapa de análisis, la cual está compuesta por: (1) análisis de resultados, referenciado en el apartado 4.3.4.3, (2) elaboración de informe final, (3) toma de decisión final. Los resultados obtenidos en está última etapa corresponde a que la herramienta dotProject que obtuvo la mayor calificación de calidad superando el punto critico de calidad con un puntaje de 73%, seguida por ProjectNet que obtuvo un puntaje de 57,9% no alcanzando superar el punto critico de calidad, y la de peor calificación fue para ProjectPier con un puntaje de 31.54% muy por debajo del punto critico de calidad siendo la herramienta con menos calidad.

Por otra parte, es de mencionar que para la realización de este caso de estudio contó con la única participación del proponente de esta guía metodológica. Adicionalmente, se descargo el código fuente y se hizo la instalación correspondiente de las herramientas evaluadas. Adicionalmente, los datos usados para evaluar cada herramienta correspondían al anteproyecto de este trabajo.

4.2 ETAPA DE EVALUACIÓN

Paso 1 – Determinar requerimientos (1.1).

SELECCION DEL MODELO / ESTANDAR DE CALIDA PARA EVALUAR EL PRODUCTO SOFWARE
FORMULARIO DE REQUERIMIENTOS **FECHA: 05/05/2009**

Actividad o Proceso que se quiere embestir:

La Universidad Autónoma de Bucaramanga, UNAB, en su quehacer diario genera una gran cantidad de proyectos tanto en la parte académica como administrativa. La academia tiene una estructura conformada por nueve áreas del conocimiento, las cuales, se componen por centros de investigación y conocimiento, siendo su insumo principal los 31 grupos de investigación (información suministrada por la Dirección de Investigación de la UNAB vigente a la fecha de febrero de 2010). Dicha cantidad de grupos incrementa la demanda en la gestión de proyectos, a eso, también se le debería sumar los proyectos de pregrado correspondiente a los 21 diferentes programas que ofrece la Universidad en la modalidad presencial y virtual.(información tomada de sitio Web oficial de la UNAB vigente a la fecha de febrero 2010) Por otra parte, el área administrativa no es ajena. Sus unidades de acuerdo al lineamiento estratégico llevan en gran parte la administración de actividades y procesos como proyectos. Dado lo anterior, para la UNAB es un punto critico el llevar el control que ello le genera, por lo cual, necesita de una herramienta tecnológica que sirva de repositorio y a su vez que permita la gestión de los mismos, tanto para ser aplicado en el área académica como administrativa.

El grupo PRISMA (Preservación e Intercambio Digital de Información y Conocimiento) de la UNAB en búsqueda de atender esta necesidad plantea la búsqueda, evaluación y selección de una herramienta software que tenga en cuenta los siguientes requerimientos: 1. libre licenciamiento y código abierto, 2. llevar el control y gestión de proyectos, 3. entorno Web, 4 que sea un aplicativo estable, 5. que cuente con suficiente documentación.

Nombre Genérico del Producto Software cómo se conoce en el medio: Software para la Administración de Proyectos o PMS (Project Management Software , sus siglas en Ingles)

Arquitectura:	**Usuarios:**
• Escritorio: __ • Escritorio Cliente/Servidor:__ • Web:_X_	• Monousuario__ • Multiusuarios _X

Licencia:
 	
• Software Libre__X__
  • Con Protección Heredada:_X_
  • Sin Protección Heredada:_
  • Dominio publico:__
• Software propietario____
	Plataforma:
	
	• Multiplataforma:____x__
	
	• Plataforma tecnológica:

Módulos: • Administración Múltiples Proyecto • Administración de Múltiples Tareas	

Paso 2 – Estudio del dominio al que pertenece el producto. (1.2).

De acuerdo a las necesidades expresadas por el cliente en el paso 1, se debe identificar el domino al que pertenecen las necesidades y realizar un estudio sobre el mismo, con el fin de contextualizar el alcance que puede tener la solución.

SELECCION DEL MODELO / ESTANDAR DE CALIDA PARA EVALUAR EL PRODUCTO SOFWARE	
DEFINICIÓN DEL DOMINIO	**FECHA: 06/05/2009**
Descripción La principal necesidad del cliente se centra la administración de proyectos. Para lo cual, requiere de una herramienta tecnológica específicamente de un sistema de información que permita el almacenamiento, control y gestión de proyectos. **Dominio:** Software para la gestión de proyectos **Estudio sobre dominio:** dicho estudio se encuentra el capítulo 2 de este documento. Este trabajo permitió identificar los principales entes y relaciones del dominio. Por otra parte, se encontraron trabajos de evaluación de herramientas para la gestión de proyectos y así mismo un listado actualizado de las diferentes ofertas que hay en el mercado de este producto tanto de software libre como propietario.	

Paso 3 – Buscar soluciones en comunidades software libres. **(1.3).**

A partir del estudio del dominio realizado en el paso 1.2 se encontró un listado actualizado de las diferentes soluciones para la gestión de proyecto

SELECCION DEL MODELO / ESTANDAR DE CALIDA PARA EVALUAR EL PRODUCTO SOFWARE

LISTADO DE SOLUCIONES **FECHA: 06/05/2009**

Nombre	Licencia		Entorno		Url
	Libre	Propietario	Web	Escritorio	
GanttProject	x			x	http://www.ganttproject.biz/
KPlato	x			x	http://www.koffice.org/kplato/
OpenProj	x			x	http://openproj.org/
Open Workbench	x			x	http://www.openworkbench.org/
Planner	x			x	http://live.gnome.org/Planner
TaskJuggler	x			x	http://www.taskjuggler.org/
Codendi	x		x		http://blog.codendi.com/
Collabtive	x		x		http://collabtive.o-dyn.de/
dotProject	x		x		http://www.dotproject.net/
eGroupWare	x		x		http://www.egroupware.org/
KForge	x		x		http://en.wikipedia.org/wiki/KForge
OpenGoo	x		x		http://www.opengoo.org/
Project.net	x		x		http://www.project.net/
ProjectPier	x		x		http://www.projectpier.org/
Redmine	x		x		http://www.redmine.org/
Trac	x		x		http://trac.edgewall.org/
SharpForge	x		x		http://cvsdude.com/sharpforge.html
jotBug	x		x		http://www.jotbug.org/
A-Plan		x		x	http://www.braintool.com/
AMS REALTIME 7		x		x	http://www.amsrealtime.com
Artemis		x		x	
CAMeLEAN/PM		x		x	http://www.ranal.com/softwaresolutions/camelean-pm.htm
Contactizer		x		x	http://objective-decision.com/en/products/contactizerpro/
FastTrack Schedule		x		x	http://www.aecsoftware.com/products/fasttrack/
InLoox		x		x	http://www.inloox.com/
LisaProject		x		x	
MacProject		x		x	http://www.mac512.com/macwebpages/macproje.htm
MatchWare MindView Business		x		x	http://www.mac512.com/macwebpages/macproje.htm
MicroPlanner X-Pert		x		x	http://www.microplanning.com/

Microsoft Project		x		x	http://office.microsoft.com/en-us/project/default.aspx
O3spaces		x		x	http://www.o3spaces.com/
OmniPlan		x		x	http://www.omnigroup.com/applications/omniplan/
P2ware Planner		x		x	http://www.p2ware.com/
Planner Suite		x		x	http://www.plannersuite.com/
PlanningPME		x		x	http://www.planningpme.com/
Planisware 5		x		x	http://www.planisware.com/
Primavera Project Planner		x		x	http://www.oracle.com/primavera/index.html
Project KickStart		x		x	http://www.projectkickstart.com/
RationalPlan		x		x	http://www.rationalplan.com/
RiskyProject		x		x	http://www.intaver.com/
Teamcenter		x		x	http://www.plm.automation.siemens.com/en_us/products/teamcenter/
Tracker Suite		x		x	http://www.acentre.com/
@task		x	x		http://en.wikipedia.org/wiki/Project_management_software
24SevenOffice		x	x		http://www.24sevenoffice.com/webpage/int/
5PM		x	x		http://www.5pmweb.com/
Artifact Software		x	x		http://www.artifactsoftware.com/
Assembla		x	x		http://www.assembla.com/
Basecamp		x	x		http://www.basecamphq.com
Blue Ant		x	x		http://www.proventis.net/website/live/index_en.html
CAMeLEAN/PM		x	x		http://www.ranal.com/softwaresolutions/camelean-pm.htm
Cardinis		x	x		http://www.cardinis.com/
Central Desktop		x	x		http://www.centraldesktop.com/
Clarity		x	x		http://www.clarizen.com/
Clarizen		x	x		
CreationFlow		x	x		http://www.creationflow.com/
DeskAway		x	x		http://www.deskaway.com/
Daptiv		x	x		http://www.daptiv.com/
EPM Live		x	x		http://www.epmlive.com/
Easy Projects .NET		x	x		http://www.easyprojects.net/
EnterPlicity		x	x		http://www.teaminteractions.com/
FogBugz		x	x		http://www.fogcreek.com/FogBugz/
Gemini		x	x		http://www.countersoft.com/home.aspx
FreeTime		x	x		http://www.freetimemanager.com/
Huddle		x	x		http://www.huddle.net/
JIRA		x	x		http://www.atlassian.com/software/jira/

Journyx		x	x		http://www.journyx.com/
Kayako helpdesk software		x	x		http://www.kayako.com/
eVisioner MetaTeam		x	x		http://www.evisioner.com/
Genius Inside		x	x		http://www.geniusinside.com/
Goplan					http://goplanapp.com/
Instant Business Network					
Launchpad					https://launchpad.net/launchpad-project
LiquidPlanner					http://www.liquidplanner.com/
Microsoft Office Project Server					http://office.microsoft.com/en-us/project/default.aspx
Mingle					
mpower					http://www.monitor-mpower.com/
OpenAir					http://www.openair.com/
Oracle Project Portfolio Management					http://www.oracle.com/us/products/applications/index.htm
P2ware Planner Server					http://www.p2ware.com/
Planisware OPX2/Planisware 5					http://www.planisware.com/
Projektron BCS					http://en.wikipedia.org/wiki/ProjektronBCS
Project Insight					http://www.projectinsight.net/
ProjectPartner					http://www.projectpartner.com/
Projectplace					
ProjectVision					
QuickArrow					http://www.quickarrow.com/
QuickBase					
Rplan					http://www.santexq.com/
Santexq					
Sciforma PSNext					http://www.sciforma.com/page?id=318
Smartsheet					http://www.smartsheet.com/
TargetProcess					http://targetprocess.com/
Time Activity Planning system (TAPs)					http://www.velerotech.com/
Teamwork					http://www.twproject.com/
Tenrox					http://www.twproject.com/

ValleySpeak Project Server					
Vertabase Pro					http://www.vertabase.com/
Viewpath					http://www.viewpath.com/
VPMi					http://www.vcsonline.com/
WorkLenz					http://www.metier.com/
Wrike					http://www.wrike.com/
Zoho Projects					http://www.zoho.com/

En esta oportunidad no hubo necesidad de usar los repositorios de proyectos de software libre, debido que a partir del estudio del dominio se encontró un generoso listado de soluciones para la gestión de proyectos. De todas formas, es importante resaltar que una inicial fuente para la búsqueda de proyectos relacionados a las necesidades del cliente son estos repositorios.

Paso 4 – Depurar listado. **(1.4).**

Con base en los requerimientos del cliente expresados en el paso 1, se determina que la solución debe ser una herramienta que este bajo licenciamiento de software libre y entorno Web. A partir de estos requerimientos se depuro el anterior listado.

SELECCION DEL MODELO / ESTANDAR DE CALIDA PARA EVALUAR EL PRODUCTO SOFWARE

LISTADO DE SOLUCIONES DEPURADO **FECHA: 06/05/2009**

Nombre	Evaluar	Licencia		Entorno		Url
		Libre	Propietario	Web	Escritorio	
Codendi		x		x		http://blog.codendi.com/
Collabtive		x		x		http://collabtive.o-dyn.de/
dotProject	x	x		x		http://www.dotproject.net/
eGroupWare		x		x		http://www.egroupware.org/
KForge		x		x		http://en.wikipedia.org/wiki/KForge
OpenGoo		x		x		http://www.opengoo.org/
Project.net	x	x		x		http://www.project.net/
ProjectPier		x		x		http://www.projectpier.org/
Redmine		x		x		http://www.redmine.org/
Trac		x		x		http://trac.edgewall.org/

SharpForge	x	x		x		http://cvsdude.com/sharpforge.html
jotBug		x		x		http://www.jotbug.org/

Para este caso de estudio se escogerán tres soluciones para ser evaluadas, como aun, el listado es muy grande se procede a depurar en relación a la popularidad que tiene cada proyecto. Por tanto, se seleccionaron las siguientes tres herramientas: dotProject, Project.net y ProjectPier

Paso 5 – Seleccionar método o estándar. **(1.5).**

En relación a los requerimientos expresados en los pasos anteriores se requiere de la búsqueda, evaluación y selección de una herramienta para la gestión de proyectos. Dicha herramienta será evaluada en uno de los procesos del ciclo de vida, del producto. Además, debe ser una aplicación Web.

SELECCION DEL MODELO / ESTANDAR DE CALIDA PARA EVALUAR EL PRODUCTO SOFWARE **SELECCIÓN MODELO / ESTANDAR** **FECHA: 06/05/2009**
MODELO / ESTANDAR: Web QEM (Web Quality Evaluation Method) **JUSTIFICACIÓN:** Evaluar, comparar la calidad de un producto Web aplicable a cualquier dominio.

4.3 ETAPA DE IMPLEMENTACIÓN

En está etapa se procede a realizar la evaluación y documentación de calidad a través de la implementación del modelo o estándar de evaluación seleccionado en la etapa de evaluación, específicamente en el paso 5. Para este caso de estudio se seleccionó el modelo Web QEM (Quality Evaluation Method, sus siglas en ingles). A continuación se procede a realizar los pasos del modelo seleccionado.

4.3.1 Introducción.

En este trabajo se hizo uso de una técnica de investigación con el fin de

corroborar o refutar a la hipótesis nula. En nuestro caso se parte de una hipótesis que declara: que en aplicaciones Web de software libre, de un dominio determinado, como PMS, LMS, CMS, entre otros, la calidad de los artefactos (las aplicaciones) satisfacen en general los requerimientos de calidad en relación a un perfil de usuario. Particularmente, que cada aplicación Web de software libre satisface al menos el punto crítico de aceptabilidad del 60% de la preferencia global, conforme a los requerimientos de calidad acordado. La anterior es una técnica de investigación denominada Caso de Estudio.

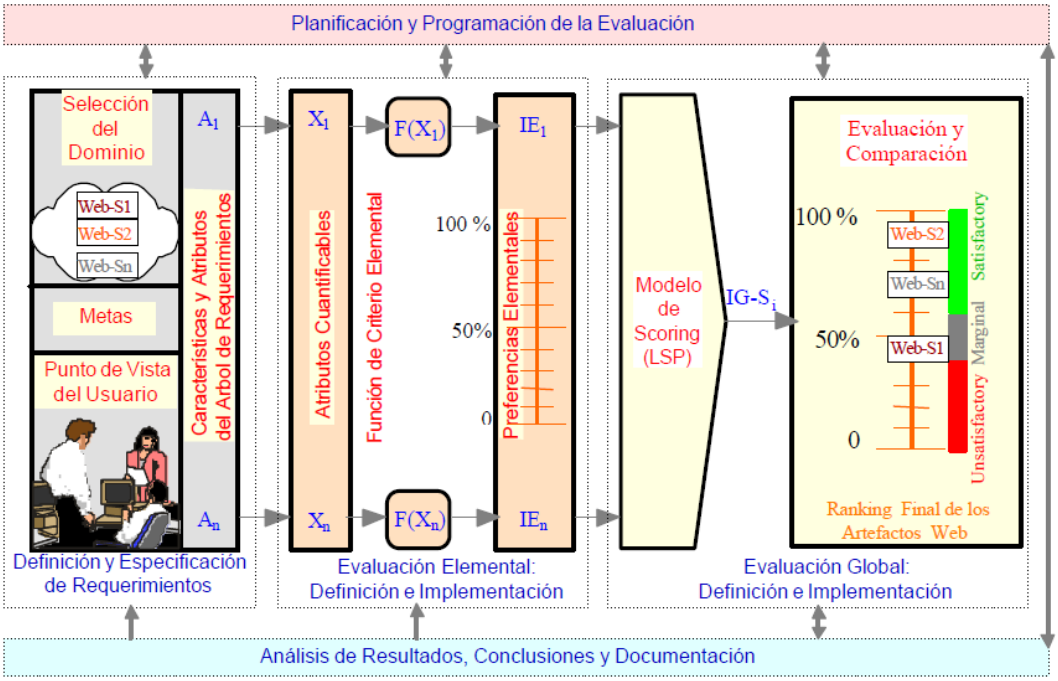
El modelo WebSite-QEM tiene tres fases que corresponden a la fase 1, definición y especificación de requerimientos, la fase 2, evaluación elemental: definición e implementación, y la fase 3, evaluación global: definición e implementación.

En la fase 1 trata con varias actividades y procedimientos para la elicitación, modelado y especificación de los requerimientos de calidad. Dichos requerimientos deben responder a las necesidades y deseos de un perfil o perfiles de usuario y dominios establecidos. En esta fase la componen las siguientes actividades: (1). selección del dominio, (2) definición de metas, (3) definición del perfil de usuario de audiencia, y (4) definición de las características y atributos del árbol de requerimientos. El resultado de las anteriores actividades fueron establecidas con base en los requerimientos definidos en la etapa de evaluación correspondiente a este caso de estudio, ver sección 4.2.

En la fase 2 trata con actividades, modelos, técnicas, heurística y herramientas para determinar criterios de evaluación para cada atributo cuantificable y realizar proceso de medición. En esta fase se compone de dos actividades: (1) definición de métricas para cada atributo y (2) medición de cada atributo.

En la fase 3 trata con actividades, modelos, procedimientos y herramientas para determinar los criterios de agregación de las preferencias de calidad elemental, obtenidas en la fase anterior, a partir del árbol de requerimientos con el fin de producir la preferencia global para cada sistema evaluado. Esta fase la componen las siguientes actividades: (1) definición del modelo LSP, y (2) medición de las subcaracterísticas y características de más alto nivel para obtener el indicador de calidad para cada sistema involucrado. El la figura 17 se pueden observar las fases y actividades intervinientes que hacen parte del proceso de evaluación y comparación usando WebSite-QEM.

Figura 17. Módulos que intervienen en el proceso de evaluación y comparación usando Web-site QEM



Fuente: OLSINA, Luis Antonio. Metodología Cuantitativa para la Evaluación y Comparación de la Calidad de Sitos Web

4.3.2 Fase de definición y especificación de los requerimientos de calidad.

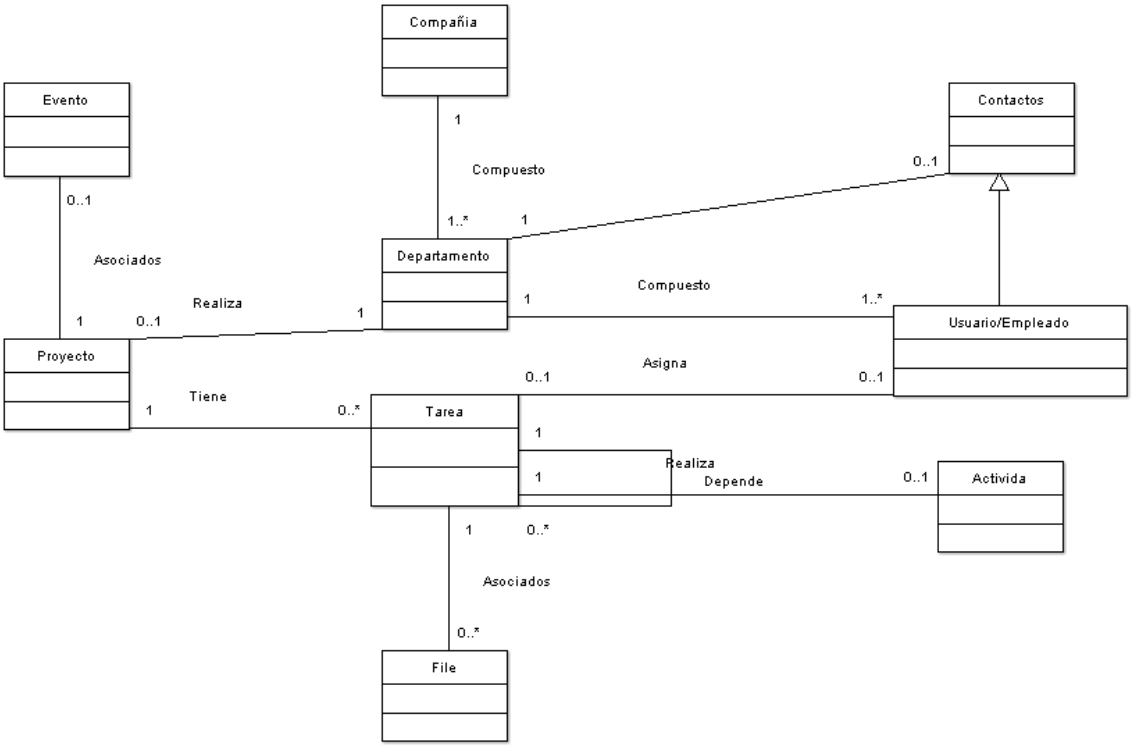
4.3.2.1 El Dominio de Aplicaciones Web para la Administración de Proyectos (PMS).

Como parte de unos de los objetivos de este trabajo es el estudió de los diferentes proyectos de software libre correspondiente a la administración de proyectos, en el capítulo 2 se presenta la importancia y la evolución de la administración de proyectos, además de las diferentes herramientas tecnológicas que apoyan está disciplina.

Dichas herramientas tecnológicas se pueden clasificar en dos grandes grupos, las de software libre y las de software propietario, adicionalmente se tiene dos subgrupos en relación al ambiente de uso: escritorio (Desktop) y basadas en la Web. En el capítulo 2 en la sección 2.1.4.2 se observa las características propias que deben tener un Software para la Administración de Proyectos o PMS (Project Management Software, sus siglas en ingles), además del listado

más reciente de este tipo de herramientas disponibles en el mercado.
Por otro lado, la definición del dominio se puede hacer desde el punto de vista de la evaluación como un sistema real o abstracto del universo que existe independientemente desde el sistema de evaluación. Esto consiste en tener un conjunto de entes a los que se les atribuye propiedades (atributos, características), manifiestan un comportamiento y se relacionan. En este trabajo el dominio es para las aplicaciones de Administración de Proyectos, los entes o clases a considerar son: proyecto, tarea, usuario, contacto, actividad, departamento y las diferentes relaciones entre ellas, tales como pertenece, tiene, compuesto, entre otras. Se puede observar la figura 18.

Figura 18. Diagrama reducido de clases y relaciones para el dominio de un sistema para la administración de proyectos



Es de mucha importancia el modelo conceptual para entender y comprender el dominio de un sistema de información, tanto para el proceso de desarrollo como para la evaluación.

4.3.2.2 Planificar y Programar la Evaluación de Calidad.

En esta fase son dos las actividades primordiales a toda evaluación de calidad siguiendo la metodología WebQEM. La primera actividad consiste en la definición de las metas de evaluación, y la segunda actividad corresponde a la selección del perfil de usuario.

4.3.2.3 Metas de la Evaluación del Caso de Estudio.

- La evaluación de calidad se va realizar sobre proyectos que se encuentre en la fase operativa.
- Se realizará una evaluación comparativa entre las siguientes herramientas: dotProject, Project.net y ProjectPier. Dichas herramientas fueron seleccionadas en el paso 4 de la etapa de evaluación de está guía metodológica, ver sección 4.2.
- Permitir conocer la calidad de las aplicaciones de software libre para la administración de proyectos desde el perfil de visitante experto.

4.3.2.4 Perfil de la audiencia.

Un meta típica de evaluación es determinar el cumplimiento de requerimientos elementales, parciales y global de calidad para las aplicaciones Web operativa, considerando el perfil de visitante experto.

El visitante experto se define como la audiencia que es especialista en el dominio de la administración de proyectos, tales como presidentes de compañía, directores de proyecto y empleados. Su permanencia en el sitio es generalmente mayor que el de la audiencia casual. Para este caso de estudio se va a trabajar en especial con el rol de director de proyecto, el cual va tener por defecto todos los permisos sobre las aplicaciones a evaluar.

4.3.2.5 Características Principales del Dominio para PMS.

En el apartado 2.1.4.2 se menciona las tres principales características que en una herramienta software se deben tener para permitir la administración de proyectos, estas son: gestión del alcance del proyecto, programación de tareas y supervisión del estado del proyecto, también conocidas como la combinación de las tres S's: scope, scheduling y status.

El estudio comparativo de software para la administración de proyectos

realizado en cabeza de la Dra. Elisabeth Kapsammer (mencionado en la sección 2.1.5), se establecen 25 criterios de evaluación, agrupados en 6 categorías. En la categoría de administración de proyecto se definieron seis criterios de evaluación: resources, task, critical path, multi-scenario management, multi-project Management, los cuales fueron tenidos en cuenta en la definición de Árbol de Requerimientos de Calidad.

4.3.2.6 Árbol de Requerimientos de Calidad.

1. Usabilidad
1.1. Capacidad de compresión del sitio global
1.1.1 Menú principal de Acceso
1.2 Mecanismos de Ayuda y Retroalimentación
2.2.1 Calidad de Ayuda
2.2.1.1 Sistema de Búsqueda de Ayuda
2.2.1.2 Facilidad de FQA
2.2.1.3 Manual de uso de la aplicación
2.3 Capacidades estéticas y de interfaz
2.3.1 Cohesividad al Agrupar los Objetos de Control Principales
2.3.2 Cohesividad al Agrupar los Objetos de Control Principales
2.3.2.1 Permanencia de Controles Directos
2.3.2.2 Permanencia de Controles Indirectos
2.3.3 Aspectos del Estilo
2.3.3.1 Uniformidad en el Color de Enlaces
2.3.3.2 Uniformidad en el Estilo Global
2.4 Misceláneas
2.4.1 Soporte de Lenguaje Extranjero

2. Funcionalidad
2.1. Administración múltiple de proyectos
2.1.1 Filtro de proyectos
2.1.1.1 Por usuario
2.1.1.2 Por departamento
2.1.1.3 Listar por estados
2.1.2Características de un proyecto
2.1.2.1 Nombre
2.1.2.2 Propietario
2.1.2.3 Compañía
2.1.2.4 Asignar contacto
2.1.2.5 Asignar departamentos
2.1.2.6 Fecha de inicio y fecha final
2.1.2.7 Prioridad
2.1.2.8 Nombre corto
2.1.2.9 Color de identificación
2.1.2.10 Tipo de proyecto
2.1.2.11 Estado
2.1.2.12 Importa tareas
2.1.2.13 Descripción
2.1.3Plantilla de proyecto

2.1.3.1 Documento
2.1.3.2 Archivos adjuntos de tareas
2.1.3.3 Contactos
2.1.3.4 Configuración
2.2 Administración múltiple de tareas
2.2.1Filtro de tareas
2.2.1.1 Por usuario
2.2.1.2 Por estado
2.2.1.3 Por prioridad
2.2.1.3.1 Por compañía
2.2.1.4 Búsqueda por nombre
2.2.2 Características de una tarea
2.2.2.1 Descripción
2.2.2.2 Estado
2.2.2.3 Prioridad
2.2.2.4 Progreso
2.2.2.5 Asignar a un grupo o departamento
2.2.2.6 Asignar a varios contactos o revisores
2.2.2.7 Tipo tarea
2.2.2.8 Fecha inicio y fecha final
2.2.2.9 Tiempo estimado
2.2.2.10 Dependencia de otras tareas
2.2.2.10.1 Asignar dependencias a una tarea
2.2.2.10.2 Establecer la fecha de inicio de tareas con base de la dependencia
2.2.2.11 Asignación de recursos humanos o usuarios
2.2.2.11.1 Asignación de recurso humano
2.2.2.11.2 Notificación vía erial
2.2.2.12 Adjuntar archivos
2.2.2.12.1 Registro de versión
2.2.2.12.2 Asignar a una tarea de un proyecto
2.2.2.12.3 Notificación a todos los miembros
2.2.2.13 Actividad de tarea
2.2.2.13.1 Fecha finalización de tarea
2.2.2.13.2 Progreso
2.2.2.13.3 Nombre
2.2.2.13.4 Descripción
2.2.2.13.5 Notificación vía email
2.2.3 Progreso del proyecto
3. Software libre
3.1. Durabilidad
3.1.1 Madurez
3.1.1.1 Edad
3.1.1.2 Estabilidad
3.1.1.3 Probabilidad de bifurcación, bifurcación de código fuente
3.1.2 Aceptación
3.1.2.1 Popularidad
3.1.2.2 Referencias
3.1.2.3 Contribución de la comunidad
3.1.2.4 Libros
3.1.3 Desarrollo y liderazgo
3.1.3.1 Líder de equipo
3.2. Solución industrializada

3.2.1 Servicios
3.2.1.1 Entrenamiento
3.2.1.2 Soporte
3.2.2 Aseguramiento de calidad
3.2.2.1 Herramientas
3.3 Estrategia
3.3.1 Licencia
3.3.1.1 Permisiva
3.3.1.2 Protección contra bifurcaciones
3.3.2 Propietarios de los derechos de autor
3.3.3 Modificación del código fuente
3.3.4 Patrocinadores
3.3.5 Estrategia independiente

El árbol de requerimientos de calidad para este trabajo consta de tres principales características de calidad: usabilidad, funcionalidad y software libre, de las cuales, la componen 9 subcaracterísticas de más alto nivel y 74 atributos. Es de mencionar que los modelos de calidad prescritos en los estándares ISO 9126 e IEEE 1061 están compuestos por 6 principales características y 21 subcaracterísticas como se observa en la tabla 18.

Por otra parte, para este trabajo la propuesta de evaluar tres características se debe a que en este caso de estudio se pretende hacer una evaluación comparativa de la usabilidad, la funcionalidad, y el software libre entre las herramientas involucradas.

Por lo tanto, con la usabilidad se pretende determinar la facilidad de uso y la facilidad de aprendizaje. Así mismo, con la funcionalidad se quiere establecer el nivel de satisfacción de las funciones y propiedades de todas las necesidades explicitas e implícitas de los usuarios. Y con el software libre se requiere determinar el nivel de libertad que tiene el software en relación a las libertades del software según la Fundación de Software Libre (Free Software Foundation, sus siglas en ingles) que reza que una vez el usuario ha adquirido el producto, tiene la libertad de ejecutar, distribuir, estudiar, cambiar y mejorar el software.

Finalmente, no se tuvieron en cuenta otras características como eficiencia, mantenibilidad, confiabilidad y portabilidad propuestas en el estándar ISO/IEC 9126 y IEEE1061, ver la tabla 18, debido que para alcanzar el objetivo de esta evaluación se encuentra bien representado en las características, subcaracterísticas y atributos definidos en el árbol de requerimientos de calidad. Por otro lado, es de resaltar que el objetivo de este trabajo es hacer la propuesta de una guía para la evaluación de calidad de aplicaciones Web adaptada al modelo de software libre. Por lo tanto, se hace la propuesta de las características de software libre, destinadas a evaluar aspectos de importancia que deben tenerse en cuenta para la evaluación de aplicaciones de software

libre.

Tabla 18. Características y Subcaracterísticas propuestos en los modelos ISO 9126 y IEEE1061

ISO 9126		***IEE 1061***	
Características	***Subcaracterísticas***	***Factores***	***Subfactores***
Funcionalidad	Adecuación, Exactitud, Interoperabilidad, Conformidad y Seguridad de Acceso.	Eficiencia	Economía con respecto al tiempo y Economía con respecto a Recursos
Confiabilidad	Nivel de madurez, Tolerancia a fallas y Recuperabilidad.	Funcionalidad	Completitud, Correctitud, Seguridad, Compatibilidad e Interoperabilidad.
Usabilidad	Comprensibilidad, Facilidad de aprender y Operabilidad.	Mantenibilidad	Correctibilidad, Expandibilidad y Testabilidad.
Eficiencia	Comportamiento con respecto al tiempo a recursos.	Portabilidad	Independencia del Hardware, Independencia del Software, Instalabilidad y Reusabilidad
Mantenibilidad	Analizabilidad, Modificabilidad, Estabilidad y Testabilidad.	Confiabilidad	No-deficiencia, Tolerancia a Errores y Disponibilidad
Portabilidad	Adaptabilidad, Instalabilidad, Conformidad y Reemplazabilidad.	Usabilidad	Comprensibilidad, Facilidad de Aprender, Operabilidad y Nivel de Comunicación

Fuente: Scalone, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software

4.3.3 Fase de definición e implementación de la evaluación elemental.

4.3.3.1 Plantillas de atributos de calidad.

A continuación se muestran dos ejemplos de las plantillas de los atributos de calidad, en la cual se hace una amplia descripción. En ella se puede encontrar, el nombre del atributo, su código, su característica de más alto nivel, su súper-

característica, su definición, su tipo de criterio elemental que es la métrica a usar para su medición, su escala de valores que son los valores que puede tomar y ejemplos que menciona de donde fue que se tomó el atributo. Por otro lado, en el anexo A se pueden observar todas las plantillas de este trabajo.

Titulo : ***Menú principal de Acceso*** Código: 1.1.1 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Capacidad de compresión del sitio global***

Definición / Comentarios: Es un mecanismo que le permite al usuario ir a las principales opciones de la aplicación. Por lo general en un PMS el menú tiene las siguientes opciones que acceden a: la población de la compañía, población de proyectos, población de tareas, calendario, población de archivos, población de tickets, administrador de usuarios, y configuración del sistema.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa si tiene menú principal de acceso, entonces:=0 no disponible; 1=disponible, tiene menú de acceso principal incompleto, no están todas las opciones de la aplicación, 2= disponible, tiene todas las opciones de la aplicación.

Escala de Valores:
0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional.

Ejemplos: La aplicación dotProject en su versión demo en línea al ingreso presenta en la parte superior el menú principal.

Titulo : ***Sistema de Búsqueda de Ayuda*** Código: 1.2.1.1 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Calidad de Ayuda - Mecanismos de Ayuda y Retroalimentación***

Definición / Comentarios: Es una opción que le permite al usuario hacer búsqueda de ayudas a través de palabras claves.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa si tiene ayudas, entonces: 0= no disponible; 1=disponible, tiene pero no está organizada por categorías, 2= disponible, organizada por categorías.

Escala de Valores:

0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en su versión demo tiene ésta opción la cual se puede encontrar haciendo clic en la pestaña de ayuda.

4.3.4.2 Evaluación elemental.

La evaluación elemental comprende la recolección de datos, una vez definidos y acordados los criterios de calidad para cada atributo. Es importante tener en cuenta que para el proceso de recolección de datos se tengan en cuenta las siguientes características: (1) la medida debe ser correcta, es decir, que los datos deben ser recolectados conforme al criterio establecido y a las reglas especificas para el atributo en cuestión, (2) la medida debe ser replicable y consistente, es decir, es repetible e insensible a pequeños cambios en el entorno, herramientas y observadores, (3) la medida debe estar asociada a un periodo de tiempo. Para este trabajo los datos, fueron recolectados en un periodo que comprende entre 20 de septiembre y 26 de octubre de 2009. Por otra parte, las versiones de las herramientas evaluadas corresponden a: dotProject 2.0, ProjectNet 9.0, y ProjectPier 0.8.0.3 y (4) la medida debe ser precisa, es decir, los datos deben corresponder a las escalas, tipo de escala, rangos y niveles, precisión conforme al criterio establecido.

Los datos recolectados conciernen a 222 valores de 74 atributos que corresponden a las tres herramientas evaluadas. En este punto de la evaluación se puede hacer un análisis muy somero de los resultados a pesar que no se ha aplicado ningún mecanismo de agregación para computar valores parciales y globales, no obstante se puede realizar interesantes observaciones en consideración de los indicadores elementales.

Por ejemplo, de las tres herramientas implicadas ninguna tiene un sistema de búsqueda y de ayuda (1.2.1.1), así como de facilidad de FAQ (1.2.1.2), caso contrario es que todas tienen una permanencia de controles directos (1.3.2.1) e indirectos (1.3.2.2).

Finalmente, se puede observar que la herramienta dotProject es la que tiene mayor calificación, de igual forma, la que le sigue es ProjecNet y por último la que más ceros y menor calificación tiene es ProjectPier, ver tabla 19.

Tabla 19. Evaluación elemental

Característica y Subcaracterística	*dotProject*	*ProjectNet*	*ProjectPier*
1. Usabilidad			
1.1.1 Menú principal de Acceso	***100***	***100***	***60***
1.2.1.1 Sistema de búsqueda de ayuda	***0***	***0***	***0***
1.2.1.2 Facilidad de FAQ	***0***	***0***	***0***
1.2.1.3 Manual de Usuario	***60***	***60***	***60***
1.3.1 Cohesividad al Agrupar los Objetos de Control Principales	***100***	***100***	***100***
1.3.2.1 Permanencia de Controles Directos	***100***	***100***	***60***
1.3.2.2 Permanencia de Controles Indirectos	***100***	***100***	***60***
1.3.3.1 Uniformidad en el Color de Enlaces	***100***	***60***	***100***
1.3.3.2 Uniformidad en el Estilo Global	***80***	***100***	***50***
1.4.1 Soporte de Lenguaje Extranjero	***60***	***100***	***0***
2. Funcionalidad			
2.1.1.1 Por usuario	***100***	***0***	***0***
2.1.1.2 Por departamento	***100***	***0***	***0***
2.1.1.3 Listar por estados	***100***	***0***	***0***
2.1.2.1 Nombre	***100***	***100***	***100***
2.1.2.2 Propietario	***100***	***100***	***0***

Tabla 19. (Continuación)

2. Funcionalidad			
2.1.2.3 Compañía	***100***	***100***	***0***
2.1.2.4 Asignar contactos	***100***	***0***	***0***

2.1.2.5 Asignar departamentos	***100***	***100***	***0***
2.1.2.6 Fecha de Inicio y fecha final	***100***	***100***	***0***
2.1.2.7 Prioridad	***100***	***0***	***0***
2.1.2.8 Nombre corto	***100***	***0***	***0***
2.1.2.9 Color identificación	***100***	***100***	***0***
2.1.2.10 Tipo de proyecto	***100***	***100***	***0***
2.1.2.11 Estado	***100***	***100***	***0***
2.1.2.12 Importar tareas	***100***	***0***	***0***
2.1.2.13 Descripción	***100***	***100***	***100***
2.1.3.1 Documento	***100***	***0***	***0***
2.1.3.2 Archivos adjuntos de tareas	***100***	***0***	***0***
2.1.3.3 Contactos	***100***	***0***	***0***
2.1.3.4 Configuración	***100***	***0***	***0***
2.2.1.1 Por usuario	***100***	***0***	***0***
2.2.1.2 Por estado (terminado o sin terminar)	***100***	***0***	***0***
2.2.1.3 Por prioridad	***0***	***0***	***0***
2.2.1.4 Por compañía	***100***	***0***	***0***
2.2.1.5 Búsqueda por nombre	***0***	***100***	***0***
2.2.2.1 Descripción	***100***	***100***	***100***
2.2.2.2 Estado	***100***	***0***	***0***
2.2.2.3 Prioridad	***100***	***100***	***0***
2.2.2.4 Progreso %	***100***	***100***	***0***
2.2.2.5 Asignar a un grupo o departamento	***0***	***0***	***0***
2.2.2.6 Asignación a varios contactos o revisores	***100***	***0***	***0***
2.2.2.7 Tipo tarea	***100***	***0***	***0***
2.2.2.8 Fecha inicio y fecha final.	***100***	***100***	***0***
2.2.2.9 Tiempo estimado	***100***	***100***	***0***
2.2.2.10.1 Asignar dependencia de una tarea	***100***	***100***	***0***
2.2.2.10.2 Establecer la fecha de inicio de tareas con base de la dependencia	***100***	***100***	***0***
2.2.2.11.1 Asignación de recurso humano	***100***	***100***	***60***
2.2.2.11.2 Notificación vía email	***100***	***0***	***0***
2.2.2.12.1 Registro de versión	***100***	***0***	***0***
2.2.2.12.2 Asignar a tarea de un proyecto	***100***	***100***	***100***
2.2.2.12.3 Notificación a todos los miembros	***100***	***0***	***0***
2.2.2.13.1 Fecha Finalización Tarea	***100***	***0***	***0***
2.2.2.13.2 Progreso	***100***	***0***	***0***
2.2.2.13.3 Nombre	***100***	***0***	***0***
2.2.2.13.4 Descripción	***100***	***0***	***0***

Tabla 19. (Continuación)

2.2.2.13.5 Notificación email	***100***	***0***	***0***
2.2.3 Progreso del proyecto	***100***	***100***	***0***
3. Software Libre			

3.1.1.1 Edad	***100***	***60***	***60***
3.1.1.2 Estabilidad	***100***	***0***	***0***
3.1.1.3 Probabilidad de bifurcación, bifurcación del código fuente	***100***	***100***	***60***
3.1. 2 .1 Popularidad	***100***	***100***	***60***
3.1. 2 .2 Referencias	***100***	***100***	***60***
3.1. 2 .3 Contribución de la comunidad	***100***	***100***	***60***
3.1. 2 .4 Libros	***60***	***0***	***0***
3.1.3.1 Líder de equipo	***100***	***100***	***100***
3.2.1.1 Entrenamiento	***60***	***60***	***60***
3.2.1.2 Soporte	***60***	***100***	***0***
3.2.2.1 Herramientas	***100***	***100***	***60***
3.3.1.1 Permisividad	***0***	***0***	***0***
3.3.1.2 Protección contra bifurcaciones	***100***	***100***	***100***
3.3.2 Propietarios de los derechos autor	***60***	***60***	***60***
3.3.3 Modificación del código fuente	***100***	***100***	***100***
3.3.5 Patrocinadores	***0***	***60***	***0***
3.3.6 Estrategia independiente	***100***	***60***	***100***

4.3.4 Fase de definición e implementación de la evaluación global.

4.3.4.1 Estructura de agregación de referencias parciales usando el modelo LSP.

La implementación de la evaluación global consta de actividades, procedimientos, modelos y herramientas que permiten determinar los criterios de agregación de las preferencias de calidad elemental para luego producir las preferencias de calidad global correspondiente a cada sistema interviniente.

Para este trabajo se implementó un modelo de agregación de atributos, características y subcaracterísticas basados en LSP (Logic Scoring Preference, sus siglas en ingles), caracterizado por ser organizado, estructurado, cuantitativo y robusto. Por otra parte, para el modelo de agregación se consideraron 17 tipos de funciones lógicas de agregación (representadas en diferentes niveles de polarización y / o), modelando las diferentes relaciones entre atributos y características, entre ellas, de reemplazabilidad, simultaneidad, neutralidad, simétrica, y asimétrica para obtener valores entre 0 y 100 para así poder dar ranking a las herramientas evaluadas.

El valor obtenido es denominado el indicador de calidad global IG para el I-ésimo sistema evaluado. Por tanto, la referencia de calidad global representa

el grado de satisfacción de todos los requerimientos de calidad explícitos e implícitos.

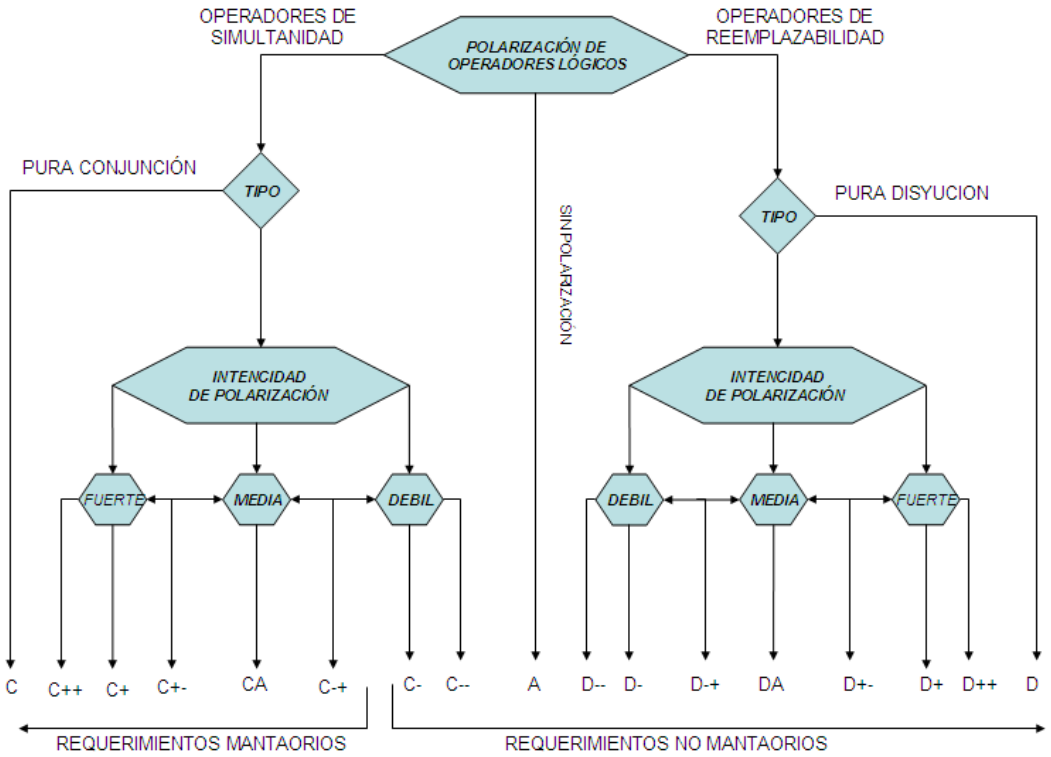
Una de las fortalezas del LSP es poder modelar diferentes relaciones lógicas entre atributos y subcaracterísticas de manera que reflejen las necesidades de los diferentes participantes en el proceso de evaluación.

Las relaciones lógicas entre atributos y subcaracterísticas del LSP reflejan las necesidades entre los diferentes participantes en el proceso de evaluación. A continuación se definen las relaciones lógicas:

- Simultaneidad o relación de conjuntividad: cuando dos entradas deben estar presentes simultáneamente.
- Reemplazabilidad o relación disyuntividad: cuando dos o más entradas pueden estar presentes alternativamente, por ejemplo, la presencia de un atributo puede reemplazar la presencia de otro.
- Neutralidad o relación ni de conjuntividad ni de disyuntividad: cuando dos o más preferencias de entrada pueden agruparse de un modo independiente.
- Relación simétrica: cuando dos o más preferencias de entrada afectan de la misma manera lógica aunque con diferentes grados de importancia.
- Relación asimétrica: cuando se requiere modelar requerimientos mandatarios combinados con requerimientos no mandatarios (atributos obligatorios se combinan con otros deseables y/u opcionales), o cuando condiciones necesarias se combinan con condiciones suficientes.

En la figura 19 se puede apreciar los 17 conectores lógicos que se tuvieron presentes para este trabajo. Por otra parte, en la tabla 20 se observa el significado de cada operador lógico. Es de mencionar que los operadores lógicos de cuasi-conjunción representan conectores “y” flexibles. Para profundizar más el tema de la definición de operadores lógicos y del modelo LSP se puede consultar al autor de la metodología WebSite-QEM.

Figura 19. Funciones operadores de las funciones lógicas



Fuente: Fuente: OLSINA, Luis Antonio. Metodología Cuantitativa para la Evaluación y Comparación de la Calidad de Sitos Web

Tabla 20. Descripción de operadores lógicos

Operador	Descripción
(A)	Modela la relación de neutralidad
(C)	Modela la conjunción pura
(C-)	Modela la cuasi-conjunción con intensidad débil.
(CA)	Modela la cuasi-conjunción con intensidad medio
(C+)	Modela la cuasi-conjunción con intensidad fuerte
(C--)	Modela la relación intermedia entre (A) y (C)

Tabla 20. (Continuación)

(C-+)	Modela la relación intermedia entre (CA) y (C-)
(C+-)	Modela la relación intermedia entre (C+) y (CA)
(C++)	Modela la relación intermedia entre (C) y (C+)
(D)	Modela la relación disyuntiva pura
(D-)	Modela la cuasi-disyuntiva con intensidad débil.
(DA)	Modela la cuasi-disyuntiva con intensidad media.
(D++)	Modela la cuasi-disyuntiva con intensidad fuerte.
(D--)	Modela la relación intermedia entre (A) y (D-)
(D-+)	Modela la relación intermedia entre (D-) y (DA)
(D+-)	Modela la relación intermedia entre (DA) y (D+)
(D++)	Modela la relación intermedia entre (D+) y (D)

Fuente: OLSINA, Luis Antonio. Metodología Cuantitativa para la Evaluación y Comparación de la Calidad de Sitos Web

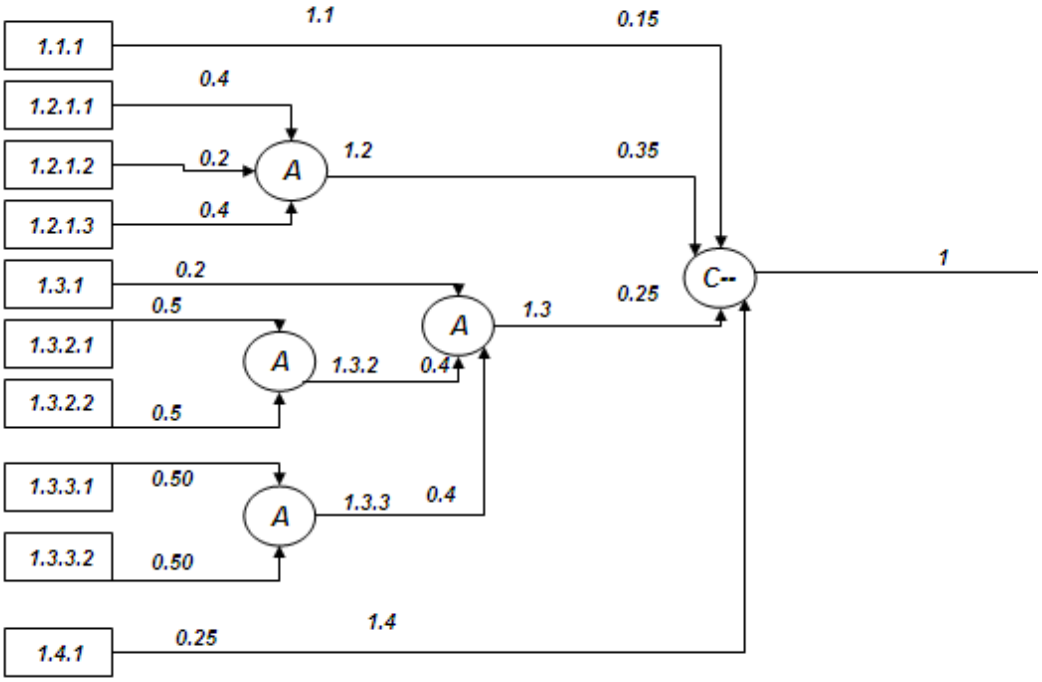
A continuación en la figura 20 que tiene tres secciones a, b, y c donde se muestra la estructura de agregación de referencias parciales usando el modelo LSP. Usabilidad, funcionalidad y software libre que corresponde a las tres características a evaluar para este caso de estudio.

La parte ***a*** corresponde a la característica, subcaracterísticas y atributos de usabilidad. Es de observar que la subcaracterística codificada 1.2 (Mecanismos de Ayuda y Retroalimentación) tiene una importancia relativa o peso de 0,35, la subcaracterística codificada 1.3 (Capacidades estéticas e inferfaz) tiene un peso de 0,25, también con el mismo peso la subcaracterística codificad 1.4 (Misceláneas), y por ultimo la subcaracterística codificada 1.1 (Capacidad de compresión del sitio global) con un peso de 0.15. Todas estas preferencias de calidad de las subcaracterísticas sirven de entrada a la función lógica *C--* la que produce como salida a la preferencia global parcial codificada 1, (Usabilidad). El operador de polarización conjuntiva de baja intensidad *C--* modela requerimientos obligatorios, es decir, un cero en una de las entradas no genera un cero en la salida aunque castiga el

resultado.

Por otro lado, las relaciones de los operadores de las subcaracterísticas 1.2 y 1.3 tienen como operador de relación de neutralidad *A* que no son ni conjuntiva ni disyuntivamente polarizada.

Figura 20. Estructura de agregación de referencias parciales usando el modelo LSP.



a)

Estructura de preferencia parcial para la característica de usabilidad.

La parte ***b*** corresponde a la característica, subcaracterísticas y atributos de funcionalidad. Es de observar que la subcaracterística codificada 2.1 (Administración múltiple de proyectos) tiene una importancia relativa o peso de 0,50 y la subcaracterística codificada 2.2 (Administración de múltiple tareas) tiene un peso de 0,50. Todas estas preferencias de calidad de las subcaracterísticas sirven de entrada a la función lógica *C*-- la que produce como salida a la preferencia global parcial codificada 2, (Funcionalidad).

Figura 20 (Continuación)

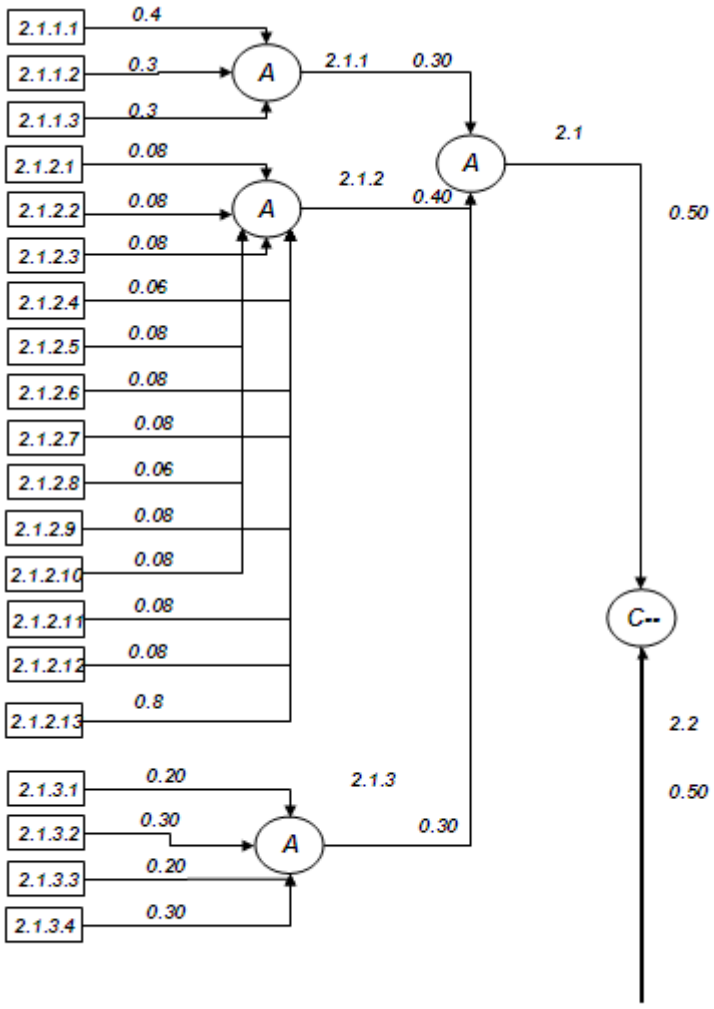
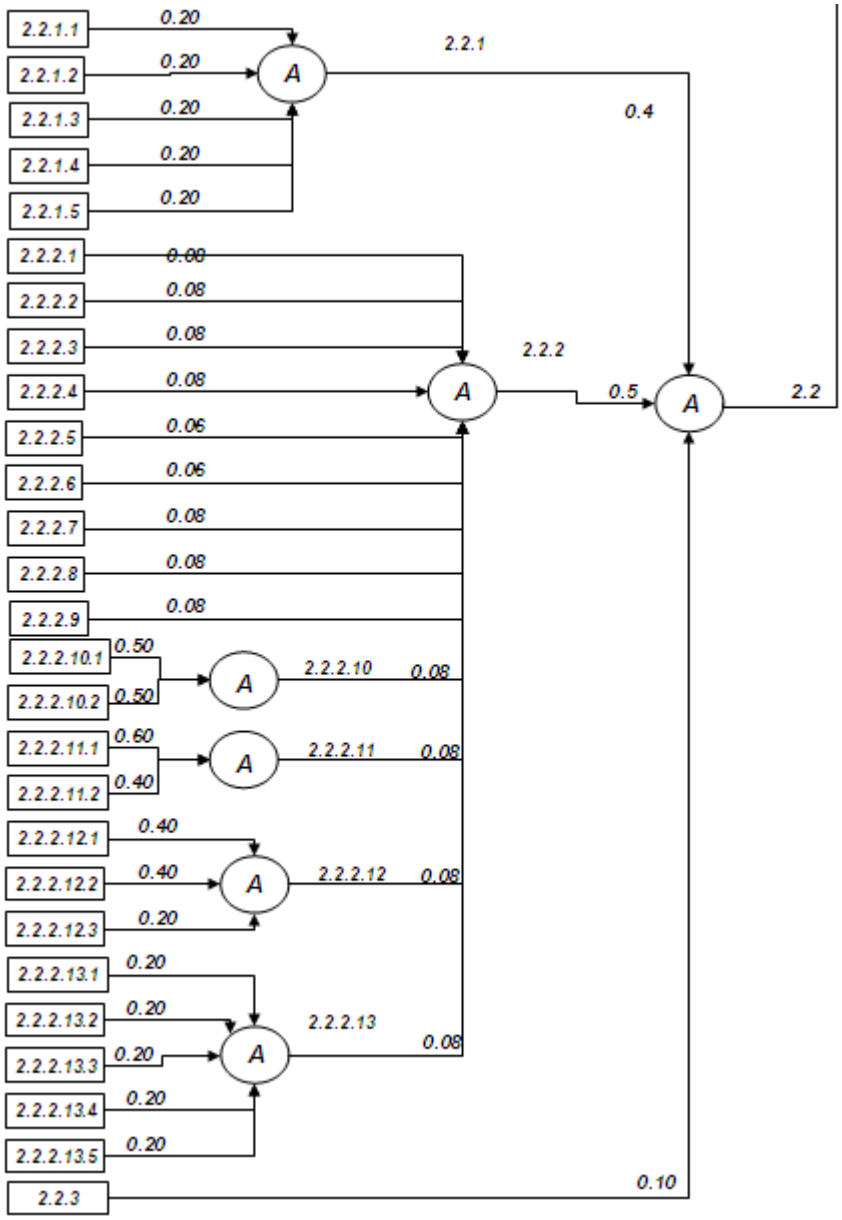

2.1.1.1
0.4
2.1.1.2
0.3
2.1.1.3
0.3
A
2.1.1
0.30
2.1.2.1
0.08
2.1.2.2
0.08
2.1.2.3
0.08
2.1.2.4
0.06
2.1.2.5
0.08
2.1.2.6
0.08
2.1.2.7
0.08
2.1.2.8
0.06
2.1.2.9
0.08
2.1.2.10
0.08
2.1.2.11
0.08
2.1.2.12
0.08
2.1.2.13
0.8
A
2.1.2
0.40
2.1.3.1
0.20
2.1.3.2
0.30
2.1.3.3
0.20
2.1.3.4
0.30
A
2.1.3
0.30
A
2.1
0.50
C--
2.2
0.50

Figura 20 (Continuación)

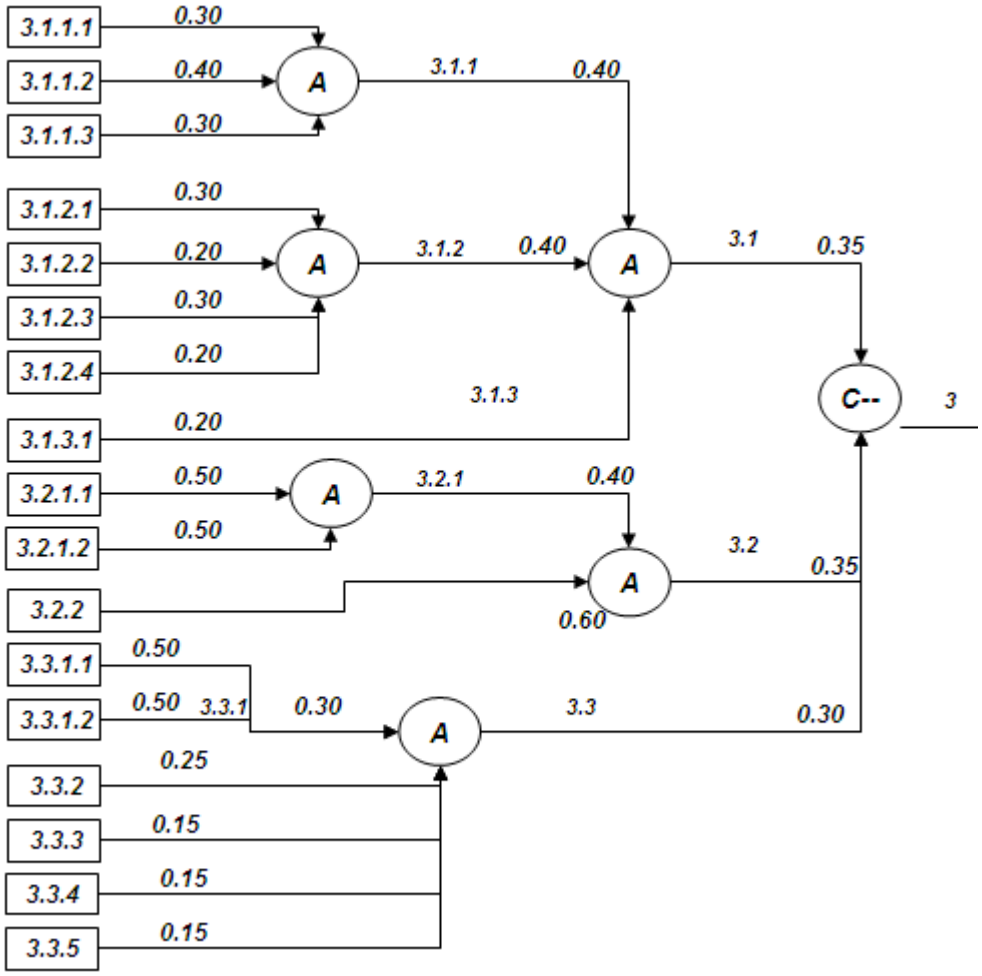


b) Estructura de preferencia parcial para la característica de funcionalidad.

La parte ***c*** corresponde a la característica, subcaracterísticas y atributos de software libre. Es de observar que la subcaracterística codificada 3.1 (Durabilidad) tiene una importancia relativa o peso de 0,35, la

subcaracterística codificada 3.2 (Solución industrializada) tiene un peso de 0,35, y la subcaracterística codificada 3.3 (Estrategia) tiene un peso de 0,30. Todas estas preferencias de calidad de las subcaracterísticas sirven de entrada a la función lógica *C--* la que produce como salida a la preferencia global parcial codificada 3, (Software libre).

Figura 20 (Continuación)

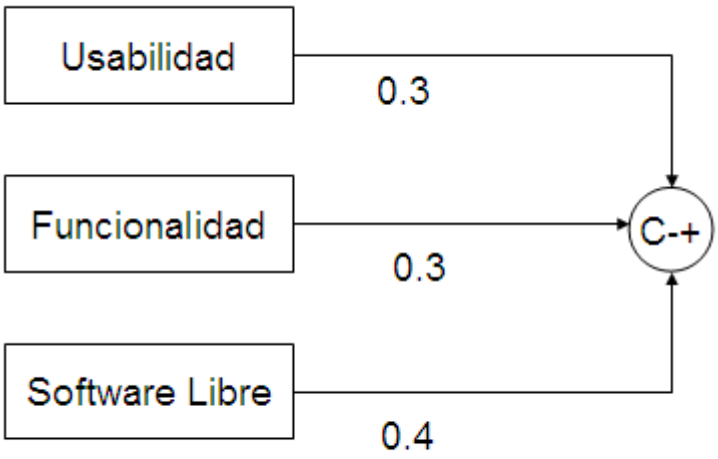


c) Estructura de preferencia parcial para la característica de software libre.

La estrategia y los mecanismos implementados en esta fase se realizaron de forma intuitiva con base en la experticia del evaluador y el nivel de criticidad del proyecto de evaluación. Es de mencionar que el estudio realizado en el capítulo dos sobre el domino de este caso de estudio fue de gran ayuda para elevar el criterio y experticia del evaluador. Por otra parte, el evaluador puede

utilizar mecanismo como encuestas y formulas de relativa importancia para computar los pesos, caso contrario de este trabajo que no fueron usados debido al costo que ello implicaba tanto en tiempo como en recursos.

Figura 21. Estructura de agregación de referencias parciales para las características de más alto nivel para computar el indicador de calidad global IG para cada sitio Web



En la figura 21 se puede apreciar la estructura de agregación de las preferencias de calidad de las características de más alto nivel, usadas para producir la preferencia de calidad para cada sitio a evaluar. Por otro lado, en consideración del perfil de usuario para el dominio de evaluación se observa que las características de usabilidad y funcionalidad no tienen el mismo nivel de importancia como lo es la característica de software libre. Las dos primeras tienen un peso de 30% en comparación a la de software libre que tiene un peso de 40%. Lo anterior se debe, a la propuesta de este trabajo de darle mayor importancia al software libre con el fin de hacer la elección de una herramienta que tenga la suficiente madurez, popularidad y sea fiel a los principios de libertades del software libre aun que sin dejar de lado la funcionalidad y usabilidad que son también de mucha importancia, por tanto, cada una tiene un peso considerable.

4.3.4.2 Resultado de los valores de las preferencias parciales y globales

de calidad, para los tres sitios PMS evaluados.

Terminada la estructura que corresponde a los criterios y funciones de agregación, obtenidos en las fases anteriores, los tomadores de decisión proceden a ejecutar el programa que calcula las preferencias de calidad parcial y global para cada sistema participante. Es de anotar que para este trabajo la operación de cómputo de valores no se realizo a través de una herramienta de cómputo, sino de forma manual, debido a que la herramienta usada por el autor de la metodología Web-site QEM no estaba disponible para su uso. Por otro lado, las razones que favorecieron el cómputo manual fueron: (1) el número de herramientas intervinientes era pequeño, (2) el número de atributos medidos era menor de 100 y (3) las funciones lógicas no eran de alta complejidad en comparación de las funciones asimétricas.

En la tabla 21 se observan los indicadores parciales de calidad para cada sitio y en la tabla 22, al final de ella el indicador de calidad global IG. El proceso ha ranqueado primero a la herramienta dotProject con 73% de la preferencia de calidad global (entrando en la barra de calidad verde), y ha ranqueado en último lugar a ProjectPier con un 31.54% de la preferencia de calidad global (entrando en la barra roja de calidad).

Tabla 21. Resultado de valores de las preferencias parciales de calidad para los tres sitios intervenidos

Características y Subcaracterísticas	dotProject	ProjectNet	ProjectPier
1. Usabilidad	**62,4**	**74,4**	**35,9**
1.1 Capacidad de compresión del sitio global	*15*	*15*	*9*
1.1.1 Menú principal de Acceso	*100*	*100*	*60*
1.2 Mecanismos de Ayuda y Retroalimentación	*8,4*	*8,4*	*8,4*
1.2.1 Calidad de Ayuda	*24*	*24*	*24*
1.3 Capacidades estéticas y de interfaz	*24*	*26*	*18,5*
1.3.1 Cohesividad al Agrupar los Objetos de Control Principales	*20*	*23*	*20*
1.3.2 Permanecía y Estabilidad en la Presentación de los Controles Principales	*40*	*40*	*24*
1.3.3 Aspectos del Estilo	*36*	*32*	*30*
1.4 Misceláneas	*15*	*25*	*0*
1.4.1 Soporte de Lenguaje Extranjero	*60*	*100*	*0*
2. Funcionalidad	***90,5***	***37***	***7.12***
2.1 Administración múltiple de proyectos	*50*	*14,4*	*3,6*
2.1.1 Filtro de proyectos	*30*	*0*	*0*

Tabla 21 (Continuación)

2.1.2 Características de un proyecto	*40*	*28,8*	*7,2*
2.1.3 Plantilla de proyecto	*30*	*0*	*0*
2.2 Administración de múltiple tareas	*40,5*	*23*	*3,52*
2.2.1 Filtro de tareas	*24*	*8*	*0*
2.2.2 Características de una tarea	*47*	*28*	*7,04*
2.2.3 Progreso del proyecto	*10*	*10*	*0*
3. Software Libre	***67,84***	***60,94***	***46***
3.1 Durabilidad	*33,88*	*24,92*	*18,76*
3.1.1 Madurez	*40*	*19,2*	*14,4*
3.1.2 Aceptación	*36,8*	*32*	*19,2*
3.1.3 Desarrollo y liderazgo	*20*	*20*	*20*
3.1.3.1 Líder de equipo	*100*	*100*	*100*
3.2 Solución industrializada	*15,96*	*17,08*	*9,24*
3.2.1 Servicios	*24*	*32*	*12*
3.2.2 Aseguramiento de Calidad	*60*	*60*	*36*
3.2.2.2 Herramientas	*100*	*100*	*60*
3.3 Estrategia	*18*	*18,9*	*18*
3.3.1 Licencia	*15*	*15*	*15*
3.3.2 Propietarios de los derechos autor	*15*	*15*	*15*
3.3.3 Modificación del código fuente	*15*	*15*	*15*
3.3.4 Patrocinadores	*0*	*9*	*0*
3.3.5 Estrategia independiente	*15*	*9*	*15*

Tabla 22. Resultados de los valores de las preferencias de calidad para las características de más alto nivel, y valores finales para las tres herramientas PMS evaluadas

Características	*dotProject*	*ProjectNet*	*ProjectPier*
1. Usabilidad	*62,4*	*74,4*	*35,9*
2. Funcionalidad	*90,5*	*37*	*7.12*
3. Software Libre	*67,84*	*60,94*	*46*
Calidad Global	*73*	*57,9*	*31.54*

4.3.4.3 Análisis de resultados.

a) Usabilidad

Según la ISO/IEC 9126 medir la usabilidad permite responder a la pregunta. ¿El software, es fácil de usar y de aprender?. Al observar los resultados se

aprecia que entre las herramientas evaluadas la que mayor peso obtuvo en usabilidad fue ProjectNet con 74,4% seguida de dotProject con 62,4% y en último lugar ProjectPier con 35,9%. Al profundizar se observa que ProjectNet en la subcaracterística codificada 1.3 (Capacidades estéticas y de interfaz) obtuvo dos puntos por encima de dotProject, la primera con un peso de 26% y la segunda con un peso de 24%, caso contrario entre ProjectNet y ProjectPier que tiene una diferencia de 8,5%. Por otro lado, la subcaracteristica donde hay una gran distancia entre la primera y la segunda corresponde a la codificada 1.4 (Misceláneas), ProjectNet obtuvo un peso de 25% y dotProject de 15% con una diferencia de 10 puntos. Finalmente, se puede concluir que ProjectNet tuvo una calificación mayor en usabilidad debido a que tiene un mayor soporte en lengua extranjera y sus capacidades de estética y de interfaz son de mayor claridad y elegancia para la compresión por parte del usuario final. El caso extremo se presenta entre ProjectNet y ProjectPier donde la diferencia de la característica de usabilidad es abismal correspondiente en 38,5 puntos teniendo fuertes debilidades en la capacidad de compresión del sitio global, misceláneas, y capacidades estéticas y de interfaz.

b) Funcionalidad

La funcionalidad mide que las funciones y propiedades satisfagan todas las necesidades explicitas e implícitas de los usuarios, en otras palabras, significa asegurar que el producto funciona tal como estaba especificado. El puntaje máximo de esta característica fue para dotProject con un peso de 90,5%, y el más bajo fue para ProjectPier con un peso de 7,2% quedando en el medio ProjectNet con un peso de 37%. Al observar en la subcaracterística codificada 2.1 (Administración múltiple de proyectos), la diferencia entre la primera herramienta y la última herramienta es 46,4 puntos. Dicha diferencia es bastante grande y se debe a que en ProjectPier la gran mayoría de las mediciones obtenidas en los atributos de la subcaracterística 2.1 es 0, sólo los atributos codificados: 2.1.2.13 (Descripción) y 2.1.2.1 (Nombre) obtuvieron un valor de 100. Caso contrario ocurre entre dotProject y ProjectNet, en donde la primera todos sus atributos tuvieron una calificación de 100, y la segunda sólo nueve de sus atributos obtuvieron una calificación máxima y el resto de 0. Por otra parte, la subcaracterística codificada 2.2 (Administración de múltiples tareas), se obtuvieron las siguientes calificaciones. Primer puesto, lo obtuvo dotProject con un peso de 40,5%, el segundo puesto fue para ProjectNet con un peso de 23% y el último le correspondió a ProjectPier con un peso de 3.52%. La diferencia entre primer y el último es 36,98 puntos, menor en relación a la subcaracterística 2.1 pero aun así es bastante grande. Se aprecia que en ProjectPier las subcaracterísticas codificadas: 2.2.1 (Filtro de tareas) y 2.2.3 (Progreso del proyecto) obtuvieron un peso de 0. Dada la calificación

anterior es la razón del castigo en el peso final de la subcaracterística 2.2. Para concluir, se puede decir que doProject obtuvo una altísima calificación casi cumpliendo con todos los requisitos parciales de calidad. Por tanto, doProject permite hacer una gran gestión para proyectos y sus respectivas tareas, diferente de ProjectNet que aun por ser un proyecto relativamente joven aun no cuenta con la madurez para prestar todos los servicios que debe tener un PMS. Por otro lado, el ProjectNet tiene una mejor calificación en relación a ProjectPier, pero aun así está dentro de la barra de color rojo de insatisfactorio. Es decir, que por 3 puntos no alcanza a estar entre la barra de color gris de aceptabilidad.

c) Software libre

Uno de los objetivos de este trabajo es evaluar las características correspondientes al software libre en aplicaciones de entorno Web. Por tanto, una de las razones de darle el mayor peso a esta característica es la de medir entre las tres herramientas involucradas cuál de ellas tiene más afinidad a las libertades y bondades del movimiento de software libre.

La herramienta que mejor evaluación tuvo en esta característica fue para dotProject con un peso de 67,84%, la de menor calificación fue para ProjectPier con un peso de 46%, y la calificación media le correspondió a ProjectNet con un peso de 60,94%. La diferencia entre la mejor calificación y la peor es de 21,84 puntos. Dicha diferencia no es tan significativa en comparación a las dos anteriores características. Al detallar la subcaracterística codificada 3.1 (Durabilidad) se observa que la de mayor calificación le correspondió a dotProject con un peso de 33,88%, seguida por ProjectNet con un peso de 24,92%, y por último a ProjectPier con un peso de 18,76%. La diferencia entre la primera y la última es de 15,12 puntos, lo que indica que es menor en relación a las otras subcaracterística, por ejemplo, la codificada 2.1 (Administración múltiple de proyectos), y 2.2 (Administración de múltiples tareas). Por otra parte, la subcaracterística codificada 3.2 (Solución industrializada), se observa que la herramienta que mayor calificación obtuvo fue para ProjectNet con un peso de 17,08%, seguida por dotProject con un peso de 15,69%, y por último para ProjectPier con un peso de 9,24%. La diferencian entre el primero y último es de 7,84 puntos, menor que la anterior subcaracterística. Finalmente, la última subcaracterística codificada 3.3 (Estrategia), se observa que la evaluación con mayor calificación correspondió a la herramienta ProjectNet con un peso de 18,9%, seguida por dotProject con un peso de 18% y con el mismo puntaje para ProjectPier. Es decir, en esta subcaracterística la diferencia entre el primero y el último es de tan sólo 0,09%, la menor en relación a todas las demás subcaracterística.

Para terminar, se puede decir que las tres herramientas evaluadas todas tuvieron una mejor calificación en la característica de software libre en relación a las características de usabilidad y funcionalidad. Entonces, se puede concluir que cada herramienta sigue los lineamientos del software libre. Es por eso que dotProject es un proyecto que tiene mayor durabilidad ofreciendo mayor grado de madurez, aceptación, desarrollo y liderazgo, en comparación a las otras dos herramientas. Por otra parte, es de anotar que la herramienta ProjectNet tiene mejor calificación sobre dotProject en las subcaracterísticas codificada 3.2 (Solución industrializada) y 3.3 (Estrategia), y aun así la mayor calificación final de la característica la tiene dotProject. Esto se debe a que en la subcaracterísticas codificadas 3.2 y 3.3 son muy bajitas en relación a la subcaracterística codificada 3.1, por consiguiente, todas están en un nivel muy similar en relación a la solución industrializada y de estrategia.

4.4 CONCLUSIONES

La administración de proyectos es el dominio al que pertenecen las tres herramientas software evaluadas en éste caso de estudio. Las categorías y criterios propuestos en la sección 2.3.1.1 en especial la categoría administración de proyectos, y sus criterios: recursos, tareas, ruta crítica, administración de múltiples escenarios y el diagrama reducido de clases de la figura 18 permitió establecer las subcaracterísticas y atributos de la característica de funcionalidad del árbol de requerimiento de calidad propuesto en la sección 4.3.2.6. Todo lo anterior es producto del estudio del dominio al que pertenecen los productos evaluados. En dicho estudio se pensaría que todos los atributos esenciales que hacen parte para la administración de proyectos deberían estar presentes en todo software especializado para este fin como son los casos de los atributos: 2.2.3 progreso del proyecto y 2.2.2.2 estado de la tarea. Para el primer atributo sólo está presente en las herramientas dotProject y ProjectNet, caso contrario de la herramienta ProjectPier que no tiene ésta funcionalidad y para el segundo atributo sólo la herramienta dotProject tiene ésta funcionalidad las otras dos no la tienen. Por tanto, se puede determinar que a pesar de lo básico que puedan ser las subcaracterísticas y los atributos de requerimientos de calidad en relación a la a la característica de funcionalidad no siempre los productos software especializados en el dominio evaluado garantiza que cumplan con que tengan en un 100% la funcionalidad básica del domino. De igual forma ocurre con la característica de software libre, por ejemplo, los atributos: 3.1.1.2 estabilidad, 3.1.2.4 libros y 3.3.1.1 permisividad. En el primer atributo sólo está presente en dotProject y en las otras dos herramientas no lo está, para el segundo atributo también está sólo presente en dotProject y en las otras dos tampoco está y

para el tercer atributo curiosamente en ninguna de las tres herramientas tiene presente este atributo. Por consiguiente, por básico que sean los atributos de requerimiento de calidad de la característica de software libre no es garantía que todo software de este tipo tenga en un 100% todos los atributos correspondientes a ésta característica. Por otra parte, lo que si es de esperar que otras características evaluadas como la de usabilidad se presente que en algunas herramientas evaluadas uno o varios atributos estén presentes o no lo estén, no es de obligación que todos los productos los tenga en un 100%. Por ejemplo, el atributo 1.4.1 soporte de lenguaje extranjera sólo está presente en dotProject y ProjectNet y en ProjectPier no se presenta, pero esto no quiere decir que ProjectPier se una mala herramienta para la gestión de proyecto y no sigua la filosofía del movimiento de software libre que serían las características esenciales para los productos que se están evaluando en este caso de estudio.

5. CONCLUSIONES

Las conclusiones para este trabajo se hacen en relación a dos perspectivas. La primera corresponde a la guía metodológica propuesta y la segunda al caso de uso realizado.

a) conclusiones de la guía metodológica

La guía metodología para la evaluación de aplicaciones Web adaptada al software libre, es, una guía sencilla y practica de usar, diseñada especialmente para ser usada por empresas o profesionales del área de TI, con el objetivo de permitir evaluar la calidad en aplicaciones Web de software libre. A continuación se mencionan las conclusiones que se percibieron de esta propuesta al ser experimentada en el caso de estudio.

- Estudio del dominio. Se determinó que es un aspecto clave para conseguir una buena evaluación y selección de un producto de calidad a través de esta propuesta. Esto se debe a que es la base fundamental para crear un criterio sólido de conocimiento, permitiendo definir los elementos a evaluar y las métricas de evaluación, siendo independiente en relación al modelo o estándar de calidad del producto seleccionado. Por lo tanto, se puede concluir que entre más completo sea este estudio se obtendrá una mejor definición y alcance de la evaluación.

- Experticia del evaluador o evaluadores. Se estableció que si tienen una alta experticia en el conocimiento del dominio, este define la profundidad del estudio del mismo. Por otro lado, si tienen una alta experticia en el uso de modelos o estándares de calidad puede asumir responsabilidades en la toma de decisiones como la asignación de recursos y tiempos, necesarios para la realización de encuestas y estudios adicionales.

- Selección del modelo o estándar de calidad más adecuado para evaluar la calidad de un producto. Se estableció que depende significativamente del trabajo realizado en la identificación requerimientos de la evaluación (corresponde a la actividad 1.1 de la etapa de evaluación de la guía propuesta) y el estudio del dominio del producto avaluar. Por tanto, una correcta definición de requerimientos y un amplio conocimiento del dominio del producto, da claridad y criterio en la selección.

- Complejidad que presenta la realización de la evaluación de calidad de un producto aplicando está propuesta. Se identificó que se relaciona con los siguiente tres aspectos: 1) la complejidad del modelo o estándar

seleccionado, 2) número de aplicaciones software a evaluar, entre mayor sea el número será mayor la complejidad, y 3) número de características, subcaracterísticas y atributos de calidad a evaluar, entre mayor sea el número de características, subcaracterísticas y atributos será mayor la complejidad.

- Durabilidad del software. Se evidenció que medir la durabilidad del software es un aspecto muy importante para determinar su estabilidad, especialmente en el software libre, por tanto, queda claro que será mayor su estabilidad y madurez si estos tres aspectos son altos: el tiempo del proyecto, número de personas que trabaja en el equipo base, número de liberaciones de nuevas versiones y corrección de errores.

b) conclusiones de caso de estudio

El caso de estudio se uso una técnica de investigación con el fin de corroborar o refutar a la hipótesis nula. En este caso se parte de una hipótesis que declara: “en aplicaciones Web de software libre, de un dominio determinado, como PMS, LMS, CMS, entre otros, la calidad de los artefactos (las aplicaciones) satisfacen en general los requerimientos de calidad en relación a un perfil de usuario. Particularmente, que cada aplicación Web de software libre satisface al menos el punto crítico de aceptabilidad del 60% de la preferencia global, conforme a los requerimientos de calidad acordado”. La anterior es una técnica de investigación denominada Caso de Estudio. Dicha hipótesis no se cumplió, sólo una de las herramientas evaluadas supero el punto crítico de aceptabilidad: dotProject con un puntaje de 73% en su referencia de calidad global. Las otras dos están por debajo del punto crítico de aceptabilidad, es el caso de ProjectNet que tiene una distancia de 2,1 puntos y el caso de ProjectPier que tiene una distancia mucho mayor de 28,46 puntos. Por tanto, se puedo determinar que la única razón para que no se alcanzaran el punto crítico de calidad por parte de las dos herramientas, se debió a la falta de madurez debido a que dichas herramientas pertenecen a proyectos relativamente jóvenes y que hasta ahora se están consolidando en la comunidad de software libre. Lo anterior quiere decir que si una herramienta de software libre proviene de un proyecto con tradición específicamente en la característica de software libre de durabilidad del software que se menciona en las conclusiones de la guía, existe una alta probabilidad de que logre superar el punto crítico de calidad.

6. RECOMENDACIONES Y TRABAJOS FUTUROS

Recomendaciones

Con respecto a todo lo realizado para este trabajo se relacionan las siguientes recomendaciones, producto de la propuesta e implementación de esta guía metodológica.

- Los requerimientos de calidad por parte del cliente deben ser claros y precisos.
- El evaluador o los evaluadores deben tener una contextualización general de la organización.
- El estudio de dominio debe ser profundo en relación a la experticia de los evaluadores.
- La complejidad depende del estándar o modelo seleccionado. Sin embargo, un aspecto para que no sea tan complejo su uso se debe al conocimiento y experiencia que tiene los evaluadores o el líder del equipo de evaluación. Por tanto, se aconseja que los evaluadores deben tener un mínimo de experiencia en el uso de modelos y/o estándares de calidad habiendo realizado al menos una evaluación.

Trabajos futuros

A continuación se menciona algunos aspectos que puedan ser tenidos en cuenta para la elaboración de trabajos futuros en relación a este trabajo.

- El desarrollo de una herramienta software en ambiente Web para la evaluación de calidad de aplicaciones. Dicha herramienta sería modular. Un modulo para parametrizar métricas. Otra modulo de captura de datos. Y otro módulo para procesamiento y resultados. El valor agregado de esta solución es su flexibilidad para la parametrización de métricas correspondientes a cada caso de evaluación. Además, de permitir capturar una calificación por parte de varios evaluadores y sacar un valor promedio para aplicar a la métrica correspondiente.
- Complementar el estudio realizado en el caso de estudio se podría a través de una evaluación de calidad en uso.

BIBLIOGRAFÍA

[1]. Atos Origin. [en línea]. [citado 28 Noviembre de 2009]. <aviable for Internet: http://www.atosorigin.com/en-us/about_us/Company_Profile/default.htm >.

[2] BEDINI G. Alejandro. Extracto del libro en formato digital “calidad tradicional y de software”. Universidad Técnica Federico Santa María.

[3] Blog CMMI Ingeniería de Software de Verdade. [en línea]. [citado el 7 Agosto de 2009]. <aviable for Internet: http://www.blogcmmi.com.br/avaliacao/lista-de-empresas-cmmi-no-brasil>.

[4]. CABALLERO CERVANTES, Omar Higinio. Tecnología de Información y Herramientas para la Administración de Proyectos de Software. Revista Digital Universitaria, Volumen 7 Numero 6, 2006.

[5] CALVO MANZANO, José Antonio y CABALLERO RÚA, Edgar Henry. Mejora de la calidad del software en el entorno de microempresas de TI. Universidad Politécnica de Madrid. Departamento de lenguajes y sistemas informaticos en ingeniería de software. 2007

[6]. Capgemini. [en línea]. [citado 28 Noviembre de 2009]. <aviable for Internet: http://www.es.capgemini.com/>.

[7] CERVERA PAZ, Ángel. El modelo de mcCall como aplicación de la calidad a la revisión del software de gestión empresarial. [citado el 5 de Junio 2009]. <aviable for Internet: http://www.monografias.com/trabajos5/call/call.shtml#mc>.

[8] COLLEVA, Juan Guillermo. Medición y Evaluación de la Calidad en Uso de Aplicaciones Web. Grupo de Investigación y Desarrollo en Ingeniería de Software e Ingeniería. Facultad de Ingeniería de la UNLPam. 2005

[9] DÁVILA, Abraham, MELENDEZ, Karina y FLORES, Luis. Determinación de los Requerimientos de Calidad del Producto Software Basados en Normas Internacionales. Lima

[10]. Documentos Delta. Administración de Proyectos. 2007. [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://www.deltaasesores.com/docum/txadmproy2007.pdf >.
.

[11] FANTINI, Adriana Claudia. Métricas: Pautas para un adecuado sistema de evaluación de la calidad en e-learning. Universidad Nacional de la Patagonia San Juan Bosco. Facultad de Ciencias Económicas.

[12] FUENTES CONTRERAS, Matías, TEUBER HERIQUEZ, Pablo. Modelo de Calidad CMMI. <aviable for Internet: http://www.monografias.com/trabajos57/modelo-calidad-cmmi/modelo-calidad-cmmi.shtml >.

[13] HERRARA, Alejandra. Las Empresas Mexicanas de Desarrollo de Software a la Medida – Un Modelo de Competitividad .Software Guru Conocimiento en práctica, Nro 21 2008. [en línea]. [citado el 7 Agosto de 2009]. <aviable for Internet: http://www.sg.com.mx/ >.

[14] Ingenieros Software. Calidad. [en línea]. [citado el 7 Agosto de 2009]. <aviable for Internet: http://www.ingenierosoftware.com/calidad/cmm-cmmi.php >.

[15]. List of project management software. (2005). [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://en.wikipedia.org/wiki/List_of_project_management_software >.

[16] MARCHANT RAMÍREZ, Loreto. Hacia un Modelo de Implementación del Alineamiento Estratégico. Actualizaciones para el Desarrollo Organizacional Primer Seminario. 2005. [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://www.eumed.net/libros/2005/lmr/4.htm >.

[17]. Microsoft Guía de licenciamiento & Administración de Software. Beneficios para la organización [en línea]. [citado el enero de 2009]. <aviable for Internet: http://www.microsoft.com/argentina/public/kit_base/licenciamiento/licsemgt/investmt/organize.htm >.

[18]. Microsoft. Resumen de Enterprise Project Management (EPM) Solution. [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://www.microsoft.com/latam/office/project/prodinfo/epm/overview.mspx#E4C >.

[19]. MORALES, Oscar Alberto. Fundamentos de la Investigación Documental y la Monografía. [citado el 27 de Enero 2008]. <aviable for Internet: http://webdelprofesor.ula.ve/odontologia/oscarula/publicaciones/articulo18.pdf>.

[20]. OLSINA, Luis Antonio. Metodología Cuantitativa para la Evaluación y

Comparación de la Calidad de Sitos Web. Tesis Doctoral. Noviembre de 1999. Universidad de la Plata, Facultad de Ciencias Exactas, Argentina.

[21]. MUÑOS ESPIN, Galo Fabián, MARTINES RODRÍGUEZ, Fernando, y RANGEL CABALLERO, Jaime Alfredo. MPLEMENTACIÓN DE UNA APLICACIÓN WEB PARA LA MEDICIÓN DE LA CALIDAD DE WEB. Tesis de Maestría, Universidad Autónoma de Bucaramanga, Facultad de Ingeniería de Sistemas.

[22] OLSINA, Luis y ROSSI, Gustavo. A Quantitative Method for Quality Evaluation of Web Sites and Applications. GIDIS. Department of Informatics. Faculty of Engineering. UNLPam.

[23] OOS Watch open source software advisory service [citado el 20 de Noviembre 2009]. <aviable for Internet: http://www.oss-watch.ac.uk/resources/osmm.xml >.

[24] OSS Partner. [citado el 20 de Noviembre 2009]. <aviable for Internet: http://www.osspartner.com/portail/sections/accueil-public/evaluation-osmm>.

[25]. REYNOSO, Álvaro. Alineamiento Estratégico – la eliminación de la teoría de la conspiración. Grupo Kaizen S.A. [en línea]. [citado el 1 julio de 2008]. <aviable for Internet: http://www.grupokaizen.com/bsce/Alineamiento_estrategico_la_eliminacion_de_la_teoria.pdf >.

[26]. Project Management Institute. Guía de los Fundamentos de la Dirección de Proyectos Tercera Edición

[27]. SCALONE, Fernanda. Estudio Comparativo de los Modelos y Estándares de Calidad del Software. Tesis de Maestría. Junio de 2006. Universidad Tecnológica Nacional, Facultad Regional Buenos Aires, Argentina

[28] Slprog. [en línea]. [citado el 7 Agosto de 2009]. <aviable for Internet: http://slprog.wikispaces.com/>.

[29] Spin Latinoamérica. [en línea]. [citado el 7 Agosto de 2009]. <aviable for Internet: http://www.latinspin.org/articulos.htm >.

[30]. Universidad Autónoma de Bucaramanga. [en línea]. [citado 3 Junio de 2009]. <aviable for Internet: http://caribdis.unab.edu.co/portal/page?_pageid=233,151929&_dad=portal&_schema=PORTAL >.

[31]. VEGA DÍAZ, José Alberto. Estado y tendencia de la administración de proyectos en México. Tesis de Maestría. Diciembre de 2004. Universidad de las Américas Puebla, México.

ANEXOS

ANEXO A

A continuación se relacionan todas las plantillas de especificación de los atributos correspondientes a los requerimientos de calidad para el caso de estudio.

Titulo : ***Menú principal de Acceso*** Código: 1.1.1 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Capacidad de compresión del sitio global***

Definición / Comentarios: Es un mecanismo que le permite al usuario ir a las principales opciones de la aplicación. Por lo general en un PMS el menú tiene las siguientes opciones que acceden a: la población de la compañía, población de proyectos, población de tareas, calendario, población de archivos, población de tickets, administrador de usuarios, y configuración del sistema.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa si tiene menú principal de acceso, entonces:=0 no disponible; 1=disponible, tiene menú de acceso principal incompleto, no están todas las opciones de la aplicación, 2= disponible, tiene todas las opciones de la aplicación.

Escala de Valores:
0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional.

Ejemplos: La aplicación dotProject en su versión demo en línea al ingreso presenta en la parte superior el menú principal.

Titulo : ***Sistema de Búsqueda de Ayuda*** Código: 1.2.1.1 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Calidad de Ayuda - Mecanismos de Ayuda y Retroalimentación***

Definición / Comentarios: Es una opción que le permite al usuario hacer búsqueda de ayudas a través de palabras claves.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa si tiene ayudas, entonces: 0= no disponible; 1=disponible, tiene pero no está organizada por categorías, 2= disponible, organizada por categorías.

Escala de Valores:

0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en su versión demo tiene ésta opción la cual se puede encontrar haciendo clic en la pestaña de ayuda.

Titulo : ***Facilidad de FQA*** Código: 1.2.1.2 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Calidad de Ayuda - Mecanismos de Ayuda y Retroalimentación***

Definición / Comentarios: Es un mecanismo que el usuario tiene para buscar respuesta a problemas mas frecuentes.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Escala de Valores:
0=0%
1=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación AceProject en su versión demo tiene ésta opción la cual se puede encontrar haciendo clic en la pestaña de ayuda.

Titulo : ***Manual de uso de la aplicación*** Código:1.2.1.3 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Calidad de Ayuda - Mecanismos de Ayuda y Retroalimentación***

Definición / Comentarios: Es una opción que le permite consultar el manual o manuales del usuario. Esta documentación es de vital importancia para el uso funcional de la aplicación que es utilizada para resolver inquietudes y para la capacitación de nuevos usuarios.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa si tiene manuales de usuario, entonces: 0= no disponible; 1=disponible, tiene manuales en documentos pero sin versión en multimedia; video, 2= disponible, tiene manuales en documentos y en versión multimedia.

Escala de Valores:
0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en su versión demo tiene ésta opción la cual se puede encontrar haciendo clic en la pestaña de ayuda.

Titulo : ***Cohesividad al Agrupar los Objetos de Controles Principales*** Código: 1.3.1 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Capacidades estéticas y de interfaz***

Definición / Comentarios: Es un mecanismo que permite agrupar los objetos de controles principales, es decir, que por cada control principal u opción del menú principal se agrupan los otros controles relacionados a cada uno.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

0=0%
1=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en su versión demo.

Titulo : ***Permanencia de Controles Directos*** Código: 1.3.2.1 Tipo: Atributo
Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Permanecía y Estabilidad en la Presentación de los Controles Principales*** - ***Capacidades estéticas y de interfaz***

Definición / Comentarios: Es un mecanismo permite la permanencia directa de los controles del menú principal del sitio que permiten su navegación. Es decir, si se ingresa a una opción del menú principal y se navega por esta, la permanencia de los controles principales de mantenerse.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa la permanencia de los controles directos, entonces: 0= no disponible; 1=disponible, no tienen permanencia de los controles directos en su totalidad, 2= disponible, tiene permanencia de los controles directos en su totalidad.

Escala de Valores:
0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación doProject en su versión demo.

Titulo : ***Permanencia de Controles Indirectos*** Código: 1.3.2.2 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***
Súper-característica: ***Permanecía y Estabilidad en la Presentación de los Controles Principales -Capacidades estéticas y de interfaz***

Definición / Comentarios: Es un mecanismo permite la permanencia indirecta, es decir, diferente a los controles del menú principal (en donde se encuentran los controles a los subsitios) del sitio que permiten su navegación.
Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa la permanencia de los controles indirectos, entonces: 0= no disponible; 1=disponible, no tienen permanencia de los controles indirectos en su totalidad, 2= disponible, tiene permanencia de los controles directos en su totalidad.

Escala de Valores:
0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación doProject en su versión demo.

Titulo : ***Uniformidad en el Color de Enlaces*** Código: 1.3.3.1 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Capacidades estéticas y de interfaz*** - ***Capacidades estéticas y de interfaz***

Definición / Comentarios: Este atributo modela la uniformidad en el color de los enlaces que tiene la aplicación

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa si el color de sus enlaces es uniforme, entonces: 0= no uniforme; 1=uniforme, no todos los enlaces son uniforme, 2=disponible, todos sus enlaces son uniforme.

Escala de Valores:

0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación openGoo en su versión demo.

Titulo : ***Uniformidad en el Estilo Global*** Código: 1.3.3.2 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Capacidades estéticas y de interfaz*** - ***Capacidades estéticas y de interfaz***

Definición / Comentarios: Este atributo modela la uniformidad del sitio global, en una apreciación subjetiva la uniformidad global de la aplicación.

Tipo de Criterio Elemental: Valor dado en porcentaje de acuerdo a la apreciación de la uniformidad global de la aplicación, puede tomar un valor desde 0% hasta 100% correspondiente a la uniformidad de estilo observada.

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación openGoo en su versión demo.

Titulo : ***Soporte al leguaje extranjero*** Código: 1.4.1 Tipo: Atributo

Característica de más Alto Nivel: ***Usabilidad***

Súper-característica: ***Misceláneas*** - ***Capacidades estéticas y de interfaz***

Definición / Comentarios: Este atributo modela la disponibilidad parcial o total de lenguajes extranjeros soportados por el sitio Web. No se computa el lenguaje nativo como lenguaje extranjero.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; en donde se evalúa si soporta leguaje extranjero, entonces: 0= no disponible; 1=disponible, solo soporta un lenguaje extranjero, 2=disponible, soporta más de un leguaje extranjero.

Escala de Valores:

0=0%
1=60%
2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación openGoo en su versión demo.

Titulo : **Filtro por usuario** Código: 2.1.1.1 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Filtro de proyectos**

Definición / Comentarios: Esta opción permite filtrar los proyectos a través del el usuario propietario. Dicho usuario debe ser asignado al momento de crear el proyecto tiene como rol de ser el principal responsable del proyecto.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: El modulo de proyectos de la aplicación dotProject de la versión demo en línea, se listan los proyectos en relación al usuario propietario.

Titulo : **Filtro por compañía y/o departamento** Código: 2.1.1.2 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Filtro de proyectos**

Definición / Comentarios: Esta opción permite filtrar los proyectos a través del de la compañía o departamento al que pertenece el proyecto. La compañía y departamento son asignados al momento de crear el proyecto.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: El módulo de proyectos de la aplicación dotProject de la versión demo en línea, se listan los proyectos en relación al usuario propietario.

Titulo : ***Listar por estado*** Código: 2.1.1.3 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Filtro de proyectos**

Definición / Comentarios: Es una opción que le permite al usuario listar los proyectos según su estado, un proyecto puede tener varios estados, los más frecuentes son: no definido, en propuesta, en diseño, en progreso, en suspenso, completado, en modo de plantilla y archivado. Lo anterior facilita clasificar los proyectos a través su estado, además de permitir listarlos. El estado se define en el momento de la creación del proyecto y puede ser actualizado cada vez que el proyecto cambie de estado.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0). En este caso solo se va a evaluar la existencia de esta opción y no la existencia y los diferentes tipos de estados, ya que para cada aplicación pueden variar.
Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos se puede listar los proyectos a través del estado del mismo.

Titulo : ***Nombre*** Código: 2.1.2.1 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta opción permite colocarle un nombre al proyecto, dicho nombre será visualizado en el listado de proyectos.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia el campo Project Name.

Titulo : ***Propietario (Usuario) del proyecto*** Código: 2.1.2.2 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta característica permite definir en cada proyecto un usuario con el rol de propietario, es decir, el que mayor responsabilidad tiene en el proyecto o director de proyecto, adicionalmente esto le da unos privilegios sobre el mismo.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia el campo Project Owner.

Titulo : ***Compañía*** Código: 2.1.2.3 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta opción permite especificar la compañía al que pertenece un proyecto. Una aplicación para la administración de proyectos puede ser usada para llevar la gestión de proyectos de varias compañías, por lo cual, es de suma importancia está característica.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia el campo Company.

Titulo : ***Asignar contactos*** Código: 2.1.2.4 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta característica permite asignar contactos al proyecto, dichos contactos sólo se podrán asignar por compañía. Es de mencionar que los contactos o revisores son personas que no tienen acceso a la aplicación, pero son notificados del desarrollo del mismo a través de correo electrónico.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia la opción Select contacts.

Titulo : ***Departamento*** Código: 2.1.2.5 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta opción permite especificar el departamento al que pertenece un proyecto según su compañía. De esta forma, se puede especificar a un mayor nivel la unidad a la que pertenece un proyecto en una compañía.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia la opción Select departament.

Titulo : **Fecha de inicio y fecha finalización propuesta** Código: 2.1.2.6 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**
Definición / Comentarios: Esta opción permite definir cual es la fecha de inicio y de finalización de un proyecto. Por otro lado, la fecha de finalización puede variar según las diferentes novedades que se presenten durante el desarrollo de un proyecto, es por eso que muchas veces se le da el nombre de fecha sugerida o fecha objetivo.
Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: Esta opción la tiene todas las herramientas que fueron objeto de estudio para este trabajo.

Titulo : **Prioridad** Código: 2.1.2.7 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**
Definición / Comentarios: Esta característica permite definir la prioridad del proyecto. En la administración de proyecto se manejan por lo general tres tipos de prioridades para los proyectos: baja, media y alta. Esto ayuda a los usuarios a identificar de una forma rápida la criticidad del proyecto.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: Esta opción la tiene todas las herramientas que fueron objeto de estudio para este trabajo.

Titulo : ***Nombre corto*** Código: 2.1.2.8 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta opción permite darle un nombre corto al proyecto.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia la opción Short name.

Titulo : ***Color de identificación*** Código: 2.1.2.9 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta utilidad permite que cada proyecto tenga un color, el cual, sirve para identificar al proyecto de una forma rápida en el listado de proyectos.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia la opción Color identifier.

Titulo : ***Tipo de proyecto*** Código: 2.1.2.10 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta opción puede especificar el tipo de proyecto. En la administración de proyectos se usan varios tipos de proyectos, entre los más comunes se encuentran, de tipo administrativo o de tipo operativo.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia la opción Project type.

Titulo : ***Estado*** Código: 2.1.2.11 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta opción permite definir el estado del proyecto. En las aplicaciones administración de proyectos se usan varios tipos de estados, como por ejemplo, no definido, en propuesta, en diseño, en progreso, en suspenso, completado, en modo de plantilla y archivado

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia la opción Status.

Titulo : ***Importar tareas*** Código: 2.1.2.12 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta características permite importar las tareas definidas en otro proyecto. Dicha opción es de gran utilidad ya que permite de una forma rápida importar tareas que sean similares de otros proyectos.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia la opción Import task.

Titulo : ***Descripción*** Código: 2.1.2.13 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Características de un proyecto.**

Definición / Comentarios: Esta característica permite hacer una breve descripción del proyecto, en la cual se puede especificar el objetivo general y sus alcances. Es de mucha utilidad está opción ya que permite a los usuario tener un conocimiento más a fondo del proyecto.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de proyectos, en New Project se aprecia la opción Status.

Titulo : ***Documentos*** Código: 2.1.3.1 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Plantilla de proyecto.**

Definición / Comentarios: Esta característica permite importar o heredar de un proyecto de tipo plantilla los documentos o archivos adjuntos.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en la versión demo en línea en el módulo de proyectos, en New Project.

Titulo : ***Importar archivos de tareas*** Código: 2.1.3.2 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Plantilla de proyecto.**

Definición / Comentarios: Esta característica permite importar los archivos adjunto que tienen las tareas que sea importando de un proyecto.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en la versión demo en línea en el módulo de proyectos, en New Project.

Titulo : ***Contactos*** Código: 2.1.3.3 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Plantilla de proyecto.**

Definición / Comentarios: Esta característica permite importar o heredar los contactos de un proyecto de tipo plantilla.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en la versión demo en línea en el módulo de proyectos, en New Project.

Titulo : ***Configuración*** Código: 2.1.3.4 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de proyectos – Plantilla de proyecto.**

Definición / Comentarios: Esta característica permite importar o heredar la configuración básica de un proyecto, dicha configuración corresponde al propietario del proyecto, la compañía, el departamento, la fecha de inicio, la fecha final, la prioridad, el color de identificación, entre otros.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0). Los datos básicos de un proyecto puede variar entre aplicaciones para este caso sólo se evaluara la disponibilidad de heredar los datos básicos del proyecto padre.

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en la versión demo en línea en el módulo de proyectos, en New Project.

Titulo :***Filtro por usuario*** Código: 2.21.1 Tipo: Atributo

Característica de más Alto Nivel: ***Funcionalidad***

Súper-característica: ***Administración de múltiple de tareas-Filtro por usuario***

Definición / Comentarios: Es un mecanismo que le permite al usuario listar las tareas que están asignadas a un usuario en particular. Por lo general, casi siempre se presenta en la sección de tareas.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en su versión demo en línea en la sección de tareas, existen varios filtros, entre ellos, el filtro por usuario. Las tareas serán filtradas en relación al usuario seleccionado, es decir, se mostraran todas las tareas en las que esta asignado dicho usuario.

Titulo :***Filtro por estado*** Código: 2.2.1.2 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: ***Administración de múltiple de tareas - Filtro por proyectos***

Definición / Comentarios: Es una opción que le permite al usuario listar las tareas según su estado, para este caso son dos los estados a evaluar, terminado y sin terminar. Esto facilita a tener de una manera rápida y organizadas las tareas que falta por concluir y las que fueron ya finalizadas.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en modulo de tareas existe un litro de tares con varias opciones entre las cuales se encuentra filtrar las tareas que están terminadas y las sin terminar.

Titulo :***Filtro por prioridad*** Código: 2.2.1.3 Tipo: Atributo

Característica de más Alto Nivel: ***Funcionalidad***

Súper-característica: ***Administración de múltiple de tareas - Filtro por proyectos***

Definición / Comentarios: Es un mecanismo que le permite al usuario listar las tareas según su prioridad. Por lo general, casi siempre se presenta en la sección de tareas.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en su versión demo en línea en la sección de tareas, existen varios filtros, entre ellos, el filtro por prioridad. Las tareas serán filtradas en relación a la prioridad del usuario.

Titulo :***Filtro por compañía*** Código: 2.2.1.4 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - *Filtro por proyectos***

Definición / Comentarios: Las aplicaciones de administración proyectos pueden ser usadas para administrar varios proyectos que estén siendo gestionados por diferentes compañías, para tal caso, un filtro por compañía permite listar todas las tareas que están asociados a los proyectos asignados a una compañía.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en modulo de tareas existe un filtro de tares con varias opciones entre las cuales se encuentra filtrar las tareas por compañía.

Titulo : ***Búsqueda*** Código: 2.2.1.5 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas**

Definición / Comentarios: La búsqueda es una herramienta que le permitir al usuario hacer una búsqueda rápida de tareas a través del nombre, por medio de su nombre completo o parte del mismo. Por ejemplo, levantamiento de requisitos del sistema, nombre de la tarea, para obtener mayor precisión en el resultado se debería colocar el nombre completo, en caso que el usuario no recuerde el nombre completo puede usar parte del mismo, por ejemplo sistemas, obteniendo como resultado las tareas que tienen la palabra sistemas.
Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).
Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en su versión demo en línea tiene la opción para hacer búsqueda de tareas a través del nombre de la tarea.

Titulo : ***Descripción de tarea*** Código: 2.2.2.1 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas – Características de tarea**

Definición / Comentarios: Esta opción debe permitir el registro de la descripción de la tarea. La descripción ayuda a los usuarios a determinar cual es el objetivo general de la tarea, además de ser cumplimentada con información adicional.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en la versión demo en línea en la opción de crear nueva tarea se observa un campo Details que sería usado para registrar la descripción de la tarea. Es de mencionar que este campo tiene opciones para la edición de texto, como negrita, cursiva, numeración, viñetas, hipervínculos, remover link, deshacer y rehacer.

Titulo : **Estado *de tarea*** Código: 2.2.2.2 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas – Características de tarea**

Definición / Comentarios: El estado de la tarea permite determinar si la tarea está activa o inactiva.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en la versión demo en línea en la opción de crear nueva tarea se observa un campo Status que sería usado para registrar el estado de la tarea.

Titulo :**Prioridad de tarea** Código: 2.2.2.3 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas – Características de tarea**

Definición / Comentarios: Las tareas se caracterizan por tener prioridad, para esto en la administración de proyecto las tareas manejan varias prioridades, las cuales le indica de cierta forma a los miembros involucrados la importancia que representa la tarea en el desarrollo del proyecto. Las prioridades más usadas en la administración de proyectos son: crítico, urgente, alta, media y baja.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea en la opción de crear nueva tarea se observa el campo Priority que sería usado para registrar la prioridad de la tarea.

Titulo :**Progreso de tarea** Código: 2.2.2.4 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas – Características de tarea**

Definición / Comentarios: Esta opción permite llevar el progreso de la tarea, es decir, cuanto ha sido su avance, esto se da en porcentaje. Por ejemplo, la tarea, levantamiento de requerimientos, tiene un progreso del 20%. En algunos casos el progreso de las tarea esta relacionado al progreso de las actividades que la tarea tiene asociadas, por decir, la anterior tarea tiene asociada la actividad, visita a usuarios funcionales, la cual, su progreso es del 20%, supóngase que es la única actividad, el progreso de la tarea será del 20%. En otros casos el progreso se coloca directamente en la tarea.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0= no tiene opción de progreso, 1= tiene opción progreso directamente sobre la tarea, 2= tiene opción de progreso directamente sobre la tarea, además de calcular el progreso según el progreso de las actividades asociadas que tiene la tarea.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea en la opción de crear nueva tarea se observa el campo progreso, además que permite crear actividades asociadas al proyecto y estas tienen un control de progreso de la actividad, el cual está relacionado con la tarea, se le conoce como New log.

Titulo :**Asignar a departamento o grupo** Código: 2.2.2.5 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas – Características de tarea**

Definición / Comentarios: La asignación de la tarea a un departamento o grupo es de gran utilidad, de esta forma se clasifica las tareas en relación a la estructura organizacional al que pertenece el proyecto, un proyecto está asociado a una compañía, y ésta puede tener departamentos o grupos de trabajo, los cuales se asocian a las tareas.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación AceProject en la versión demo en línea en la opción de crear nueva tarea se observa el campo de grupo.

Titulo :Asignar contactos o revisores a la tarea Código: 2.2.2.6 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de tareas – Características de tarea**
Definición / Comentarios: Esta opción permite tener asociados a una tarea a contactos o revisores, su función es de estar enterados de cómo se esta desarrollando la tarea, de igual forma, ellos no podrán hacer ninguna modificación sobre el proyecto o tarea, estos privilegios están reservados para los usuarios asignados al proyecto. Por otra parte, los contactos o revisores serán notificados a través de correo electrónico, debido que por lo general no tienen cuenta de usuario sobre la aplicación de administración de proyectos.
Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).
Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en la opción de crear nueva tarea se observa la opción de Select contacts, la cual permite asignar contactos a la tarea, dichos contactos debe ser previamente creados.

Titulo : **Tipo de tarea** Código: 2.2.2.7 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de tareas – Características de tarea**
Definición / Comentarios: Esta opción permite clasificar la tarea según los criterios que se tenga definidos para el proyecto, por ejemplo, si la tarea es operativa o administrativa.
Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea en la opción de crear nueva tarea se observa la opción de Task type.

Titulo : **Fecha de inicio y fecha finalización** Código: 2.2.2.8 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de tareas – Características de tarea**
Definición / Comentarios: La fecha de inicio y de finalización de una tarea permite definir el periodo de tiempo sobre el cual se designo para la realización de la tarea.
Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: Esta opción la tiene todas las herramientas que fueron objeto de estudio para este trabajo.

Titulo : **Tiempo estimado** Código: 2.2.2.9 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de tareas – Características de tarea**
Definición / Comentarios: Esta opción permite registrar el número de horas que se estima que va tomar el desarrollo de la tarea. Dicha información es de mucha utilidad para la planificación del proyecto. Es de anotar que en varias aplicaciones trae herramientas adicionales que permite ayudar a calcular el tiempo estimado, ya sea en días o en horas. Por ejemplo, el dotProject, se especifica la hora de inicio y final de la jornada de trabajo sobre los cinco días de la semana, con está información más la fecha de inicio y finalización de la tarea se puede calcular el tiempo estimado ya sea en horas o días.
Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no tiene la opción de registro del tiempo estimado, 1= tiene la opción de registro del tiempo y 2= tiene la opción del tiempo estimado más la herramienta que ayuda a calcular el tiempo estimado. • 0=0% • 1=60% • 2=100%
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea en la opción de crear nueva tarea se observa la opción Expected Duration.

Titulo : **Asignar dependencia de tarea** Código: 2.2.2.10.1 tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de tareas – Características de tarea – Dependencia de tareas**
Definición / Comentarios: Una de las principales características en la administración de proyectos tiene que ver con la dependencia de tareas. Es muy común que se presente este tipo de situaciones, que una tarea dependa de otra tarea, es decir, que no puede iniciar hasta que una tarea predecesora se haya completado exitosamente. Las aplicaciones de administración de proyecto tienen mecanismos que permiten administrar la dependencia de tareas, de tal forma que una tarea puede depender de no sólo una tarea si no de varias tareas para dar inicio.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no tiene la opción de asignar dependencia de tarea, 1= tiene la opción de asignar la dependencia de una sola tarea, 2= tiene la opción de asignar la dependencia a más de una tarea. • 0=0% • 1=60% • 2=100%
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación AceProject en la versión demo en línea, seleccionada una tarea en la pestaña Dependencias.

Titulo : Establecer la fecha de inicio de tareas con base de la dependencia Código: 2.2.2.10.2 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de tareas – Características de tarea – Dependencia de tareas**
Definición / Comentarios: Esta opción permite que la fecha de inicio de la tarea corresponda a la última fecha de finalización de las tareas que depende. Esta opción es de gran ayuda para el usuario ya que automáticamente define la fecha de inicio en relación a la fecha final de la última tarea dependiente.
Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea, seleccionada una tarea en la pestaña Dependencias hay un campo de chequeo llamado Set task date based on dependencia.

Titulo : Asignación de recurso humano Código: 2.2.2.11.1 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de tareas – Características de tarea- Asignación de recurso humano**

Definición / Comentarios: Esta opción permite asignar el recurso humano que va trabajar en cada tarea. El recurso humano son usuarios, los cuales tiene asignado un perfil, y dicho perfil tiene designado varios permisos. Por otra parte, es importante asignar la dedicación que va tener el usuario para cada tarea, puede ser en porcentaje o en horas.
Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0= no tiene opción de asignación de recurso humano, 1= tiene opción de asignación de recurso humano sin dedicación, 2= tiene opción de asignación de recurso con dedicación. • 0=0% • 1=60% • 2=100%
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea en la opción de crear nueva tarea se observa la opción Human Resources.

Titulo : **Notificación por email de asignación recurso humano** Código: 2.2.2.11.2 Tipo: Atributo
Característica de más Alto Nivel: **Funcionalidad**
Súper-característica: **Administración de múltiple de tareas – Características de tarea- Asignación de recurso humano**
Definición / Comentarios: Esta opción es de gran ayuda por que permite notificar por vía email a las personas asignadas a la tarea.
Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).
Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea en la opción de crear nueva tarea en la opción Human Resources se observa el campo de chequeo Notify Asignes of Task by Email.

Titulo : **Registro de versión** Código: 2.2.2.12.1 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - Características de tarea - Adjuntar archivo**
Definición / Comentarios: Esta opción permite llevar el registro de la versión de un documento o archivo adjunto. Es de gran ayuda para llevar el control de versiones de archivos.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: En la aplicación AceProject en la versión demo en línea en la opción archivos adjuntos correspondiente a una tarea se encuentra el campo versión.

Titulo : **Asignar a una tarea** Código: 2.2.2.13.2 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - Características de tarea – Adjuntar archivo**

Definición / Comentarios: Un archivo adjunto debe poder ser asignado a una tarea de un proyecto.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: En la aplicación dotProject en la versión demo en línea en la sección de archivos en la opción nuevo archivo se encuentra el campo de selección de proyecto, el cual, despliega todas las tareas que están asociadas al proyecto, y el campo tarea se puede seleccionar la tarea a que va pertenecer el archivo.

Titulo : **Notificación vía email** Código: 2.2.2.12.3 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - Características de tarea - Adjuntar archivo**

Definición / Comentarios: Una vez sea adjuntado un archivo es importante tener la opción de notificación por email para informar a los miembros asignados a la tarea que se adjunto un nuevo archivo o versión.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: En la aplicación dotProject en la versión demo en línea en la sección de archivos en la opción nuevo archivo se encuentra un campo de chequeo para habilitar la opción de notificación por email.

Titulo : **Fecha de finalización** Código: 2.2.2.13.1 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - Características de tarea - Actividad de tarea**

Definición / Comentarios: Esta opción permite modificar la fecha de finalización de una tarea desde una actividad de tarea. Una actividad de tarea es un mecanismo que permite registrar actividades o problemas que surgen en el desarrollo de una tarea, la cual, puede retrazar el desarrollo normal de la misma, viéndose afectado la fecha de finalización de la tarea. Por tal motivo, es de gran utilidad para el usuario poder hacer ésta modificación desde la actividad de tarea y no tener que ir hacerla en la opción de modificar tarea.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de tareas una vez seleccionada una tarea aparece la opción New log en ella se puede ver le campo Task en date

Titulo : **Progreso de tarea** Código: 2.2.2.13.2 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - Características de tarea - Actividad de tarea**

Definición / Comentarios: Esta opción permite modificar el progreso de la tarea desde una actividad de tarea. Una actividad de tarea es un mecanismo que permite registrar actividades o problemas que surgen en el desarrollo de una tarea, la cual, puede retrazar el desarrollo normal de la misma, viéndose afectado el progreso de la tarea. Por tal motivo, es de gran utilidad para el usuario poder hacer ésta modificación desde la actividad de tarea y no tener que ir hacerla en la opción de modificar tarea.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de tareas una vez seleccionada una tarea aparece la opción New log en ella se puede ver le campo Progress.

Titulo : **Nombre** Código: 2.2.2.13.3 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - Características de tarea - Actividad de tarea**

Definición / Comentarios: Esta opción permite colocarle un nombre corto a cada actividad o problema de tarea, permitiendo ser visualizado en el listado de actividades de tareas.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de tareas una vez seleccionada una tarea aparece la opción New log en ella se puede ver el campo Summary.

Titulo : **Descripción** Código: 2.2.2.13.4 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - Características de tarea - Actividad de tarea**

Definición / Comentarios: Esta opción permite hacer una descripción completa de la actividad o problema de tarea, siendo de gran ayuda para los demás usuarios, permitiéndoles tener una información al por menor de los acontecimientos durante el desarrollo de cada tarea.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de tareas una vez seleccionada una tarea aparece la opción New log en ella se puede ver el campo Descripción.

Titulo : **Notificación por email** Código: 2.2.2.13.5 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas - Características de tarea - Actividad de tarea**

Definición / Comentarios: Esta opción permite que cada vez que se cree una nueva actividad o problema de tarea esta sea notificada a través de email a las personas implicadas en el desarrollo de la misma. Esto es de gran utilidad ya que cualquier miembro asignado a la tarea puede crea una actividad o problema de tarea, y ésta pueda notificarlo al responsable de la tarea, a lo demás miembros asignados a la tarea, a los contactos u otros.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de tareas una vez seleccionada una tarea aparece la opción New log en ella se puede ver los campos de notificación a través email.

Titulo : ***Progreso proyecto*** Código: 2.2.3 Tipo: Atributo

Característica de más Alto Nivel: **Funcionalidad**

Súper-característica: **Administración de múltiple de tareas**

Definición / Comentarios: Está opción permite que el usuario sepa cual es el progreso del proyecto según el avance de las tareas que tiene asignadas.

Tipo de Criterio Elemental: Es un criterio binario, discreto y absoluto: sólo se pregunta si está disponible (1) o sino está disponible (0).

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: La aplicación dotProject en la versión demo en línea en el módulo de tareas se puede observar que las tareas están agrupadas en relación al proyecto que pertenece, y al costado del nombre del proyecto se encuentra el progreso del mismo dado en porcentaje.

Titulo : ***Edad*** Código: 3.1.1.1 Tipo: Atributo
Característica de más Alto Nivel: ***Software libre***

Súper-característica: ***Durabilidad – Madurez***

Definición / Comentarios: Este atributo modela el tiempo que lleva la aplicación desde la liberación de su primera versión.
Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0= tiene una edad menor a 3 meses, 1= tiene una edad entre 3 meses y 3 años, 2= tiene una edad entre mayor de 3 años.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Estabilidad*** Código: 3.1.1.2 Tipo: Atributo
Característica de más Alto Nivel: ***Software libre***

Súper-característica: ***Durabilidad - Madurez***

Definición / Comentarios: Este atributo modela que tan estable se encuentra la aplicación software.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=el software es inestable debido al gran numero de versiones o parches generados por defectos secundarios, 1= estable la producción de las versiones existentes debido a la edad, pero se presentan dificultades para estabilizar las próximas liberaciones, 2= el software es estable. Las liberaciones presentan correcciones de errores (bugs, su nombre en ingles), pero principalmente corresponde a funcionalidades nuevas.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Probabilidad de bifurcación, bifurcación del código fuente*** Código: 3.1.1.3 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: ***Durabilidad - Madurez***

Definición / Comentarios: Este atributo modela si una aplicación de software libre tiene la posibilidad de bifurcarse.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=el software es muy probable que se bifurque en el futuro, 1= el software viene de una bifurcación y tiene muy pocas probabilidades de que se bifurque, 2=el software tiene pocas probabilidades de que se bifurque, y no proviene de una bifurcación.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Popularidad*** Código: 3.1. 2 .1 Tipo: Atributo
Característica de más Alto Nivel: ***Software libre***

Súper-característica: ***Durabilidad – Aceptación***

Definición / Comentarios: Este atributo modela la popularidad que puede tener una aplicación de software libre.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0= pocos usuarios identificados, 1= se detecta la presencia de la aplicación software en los repositorios de proyectos de código abierto, 2= se encuentra numerosos usuarios y referencias.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos:

Titulo : ***Referencias*** Código: 3.1. 2 .2 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: ***Durabilidad - Aceptación***

Definición / Comentarios: Este atributo modela el nivel referencia que puede tener una aplicación de software libre.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no tiene ninguna referencia, 1=pocas referencias, no es muy usada para cosas criticas, 2= frecuentes referencias para implementaciones criticas.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos:

Titulo : ***Contribución de la comunidad*** Código: 3.1. 2 .3 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: ***Durabilidad - Aceptación***

Definición / Comentarios: Este atributo modela el nivel de contribución de la comunidad de software libre hacia la aplicación software.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no tiene aportes de ninguna comunidad o sin actividad real en foros, correos o listas, 1= tiene una comunidad con una notable actividad, 2= tiene una comunidad fuerte: gran actividad en foros, numerosos colaboradores y defensores.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos:

Titulo : ***Libros*** Código: 3.1. 2 .4 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: ***Durabilidad - Aceptación***

Definición / Comentarios: Este atributo modela el nivel publicaciones en libros que tiene la aplicación de software libre

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no tiene libros, 1=tiene menos de 5 libros, 2= más de 5 libros.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Líder de equipo*** Código: 3.1.3.1 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: ***Durabilidad - Desarrollo y liderazgo***

Definición / Comentarios: Este atributo modela el liderazgo que tienen los equipos de desarrollo de la aplicación de software libre.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0= 1 a 2 personas implicadas, no claramente identificadas, 1= entre 2 y 5 personas independientes, 2= mas de 5 personas.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos:

Titulo : ***Entrenamiento*** Código: 3.2.1.1 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: ***Solución industrializada – Servicios***

Definición / Comentarios: Este atributo modela la disponibilidad de la documentación para manejo de la aplicación de software libre.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no ofrece información de capacitación, 1=ofrece una restringida información de capacitación y en un sólo idioma, 2= ofrece una gran variedad, modular y progresiva en varios idiomas.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos:

Titulo : ***Soporte*** Código: 3.2.1.2 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***

Súper-característica: ***Solución industrializada – Servicios***

Definición / Comentarios: Este atributo modela el nivel soporte que tiene una aplicación de software libre.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=No ofrece ningún apoyo, excepto a través de foros públicos y lista de correos, 1=ofrece apoyo por parte de un solo contratante sin un fuerte compromiso en la calidad del servicio, 2= múltiples contratantes ofrecen servicios con una fuerte garantía de compromiso (por ejemplo, garantía en tiempo de resolución)

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Herramientas*** Código: 3.2.2.2 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: ***Solución industrializada - Aseguramiento de Calidad***

Definición / Comentarios: Este atributo modela la disponibilidad de herramientas para el manejo de errores (bugs, su nombre ingles).

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no dispone de herramienta para gestión de errores, 1=dispone de una herramienta estándar pero mal utilizadas, 2= dispones de herramientas con una alta actividad

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos:

Titulo : ***Permisividad*** Código: 3.3.1.1 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***

Súper-característica: ***Estrategia - Licencia***

Definición / Comentarios: Este atributo modela la permisividad de la licencia cuando el usuario quiere convertirse el propietario del código fuente de una aplicación de software libre.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=muy estricta como la GPL, 1= moderadamente estricta se encuentra entre ambos extremos (GPL y BSD) de doble dependencia en función del tipo de usuario (persona, empresa) o sus actividades, 2= muy permisiva como las licencias BSD y Apache Licenses.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Protección contra bifurcaciones*** Código: 3.3.1.2 Tipo: Atributo
Característica de más Alto Nivel: ***Software libre***

Súper-característica: ***Estrategia - Licencia***

Definición / Comentarios: Este atributo modela la protección que tiene una aplicación de software libre contra las bifurcaciones.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=muy permisivas como las licencias BSD y Apache Licenses, 1= moderadamente permisiva situado entre ambos extremos(GPL y BSD) de doble dependencia en función del tipo de usuario (persona, empresa) o sus actividades, 2= muy estricta como la licencia GPL.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos:

Titulo : ***Propietarios de los derechos autor*** Código: 3.3.2 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: Estrategia

Definición / Comentarios: Este atributo modela la protección de los derechos de autor de una aplicación de software libre.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=derechos por unos pocos individuos o entidades, por lo que es más fácil cambiar su licencia, 1=derechos de numerosas personas que poseen el código fuente en forma homogénea, 2= derechos en una persona jurídica en que confía la comunidad.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Modificación del código fuente*** Código: 3.3.3 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: Estrategia

Definición / Comentarios: Este atributo modela el control de modificación del código fuente de una aplicación de software libre.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no tiene una propuesta practica para hacer modificación del código fuente, 1=proporciona herramientas para acceder y modificar el código fuente (como CVS y SVN) pero no son usadas para desarrollar el software, 2= proceso de modificación código es bien definido, expuesto y respetado con base en los diferente roles asignados.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Patrocinadores*** Código: 3.3.5 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: Estrategia

Definición / Comentarios: Este atributo modela los patrocinadores que tiene el proyecto.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no tiene patrocinadores, el equipo central no se paga, 1=tiene un único patrocinado que puede determinar su estrategia, 2=patrocinado por la industria.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional
Ejemplos:

Titulo : ***Estrategia de independencia*** Código: 3.3.6 Tipo: Atributo

Característica de más Alto Nivel: ***Software libre***
Súper-característica: Estrategia

Definición / Comentarios: Este atributo modela la estrategia de independencia definida para el proyecto.

Tipo de Criterio Elemental: Es un criterio multi-nivel, discreto y absoluto; donde 0=no detectables o estrategia de fuerte dependencia de un único agente (persona, empresa, patrocinador), 1=visión estratégica compartida con otros proyectos de software libre, pero si un fuerte compromiso de propietarios de derechos de autor, 2=fuerte independencia del equipo central, jurídicos entidad que posee los derechos, la fuerte participación en el proceso de estandarización.

- 0=0%
- 1=60%
- 2=100%

Tipo de Recolección de Datos: Manual, Observacional

Ejemplos: