UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD I

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS.

Título: Guía para gestionar la calidad de software de distribuciones GNU/Linux.

Clasificación: Investigación

Autoras:

Yudit Ortega González.
Miriam Mileidys Martínez-Rico Couret.

Tutora: Ing. Yusleydi Fernández del Monte.
Co-Tutor: Ing. Miguel Albalat Aguila.

Ciudad de la Habana. Junio del 2011

Síntesis de los tutores

Síntesis de la Tutora:

Especialidad de graduación: Ingeniería en Ciencias Informáticas.

Categoría docente: Instructora.

Categoría Científica: Investigadora.

Años de graduado: 3 años.

Síntesis del Co-Tutor:

Especialidad de graduación: Ingeniería en Ciencias Informáticas.

Categoría docente: Instructor en adiestramiento

Categoría Científica: Investigador

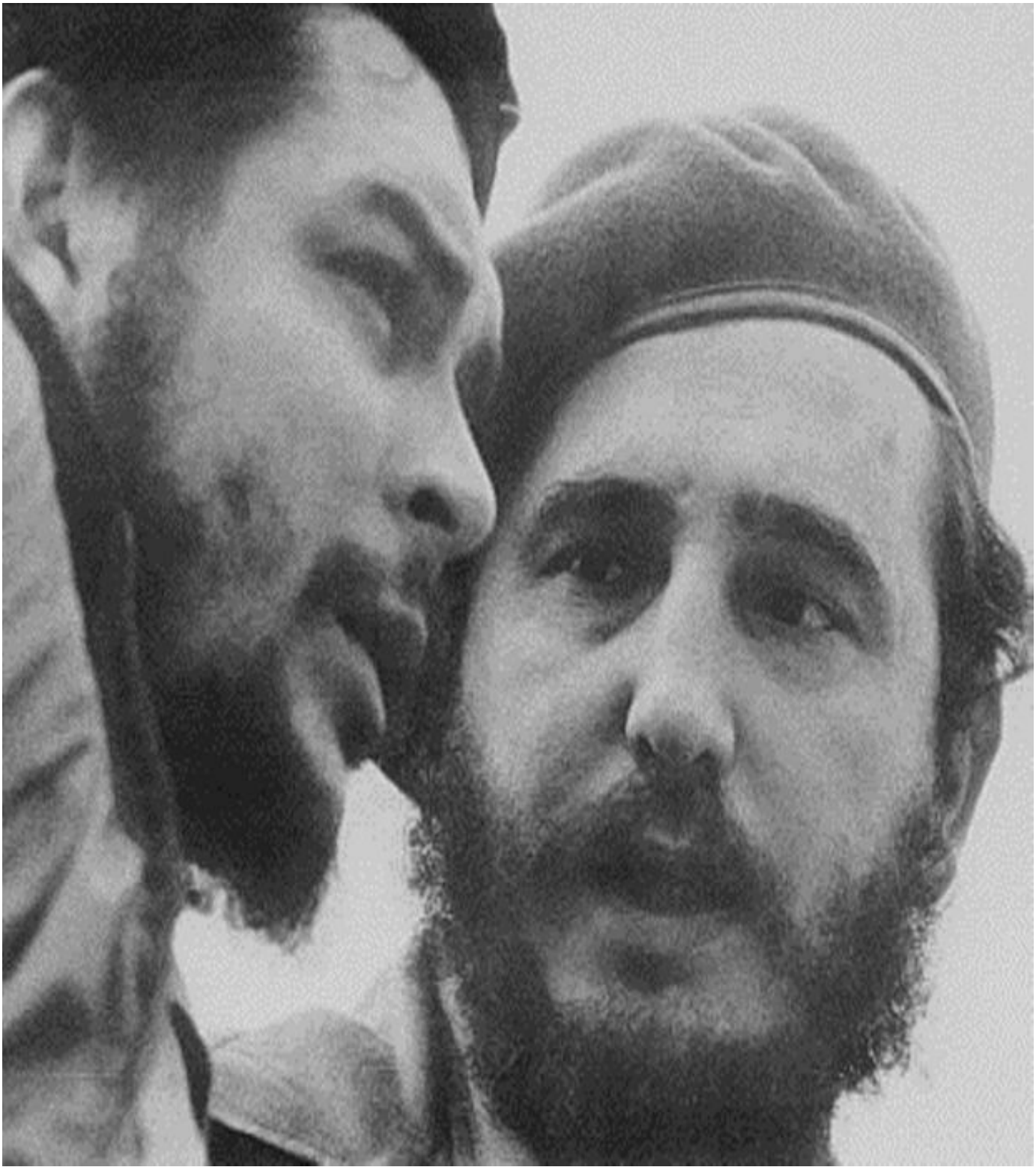
Años de graduado: 2 años.

Declaración de auditoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) y a la facultad I a que hagan uso del mismo en su beneficio.

Para que así conste firmamos la presente a los _____ días del mes de Junio del 2011.

_____	_____
Firma del Autor	Firma del Autor
Miriam Mileidys Martínez Rico Couret.	Yudit Ortega González.
_____	_____
Firma del Tutor	Firma del Co-Tutor
Ing. Yusleydi Fernández del Monte.	Ing. Miguel Albalat Aguila.



"Es responsabilidad nuestra luchar porque la calidad del producto que aquí se haga sea de las mejores y la mejor posible."

Ernesto Che Guevara.

Miriam

- *Quisiera agradecer en primer lugar a nuestra tutora Yusleydi por su apoyo, guía y esfuerzo personal, al igual que a nuestro cotutor Migue quien fue la luz en momentos de oscuridad.*

- *A Fidel y a la Revolución por darme la oportunidad de estudiar en esta universidad y cumplir mi sueño de ser una profesional*

- *A mi compañera de tesis por pensar en mi para compartir juntas esta experiencia y sobre todo por no rendirse nunca.*

- *A los profesores que me formaron, especialmente a Raydel Zumeta que más que profesor ha sido un amigo.*

- *A mi madre por ser la autora de mis más grandes alegrías, por ser amiga en todo tiempo y cómplice de mis sentimientos.*

- *A mi padre, por su persistencia en ver realizados mis sueños, su apoyo y su preocupación en todo momento*

- *A mi hermano mayor por su cariño, por estar siempre y porque en esta etapa de mi vida he sentido su apoyo más que nunca.*

- *A mis queridos abuelos, por siempre tenerme presente en sus vidas, por su ejemplo, sus consejos y porque de ellos aprendí que debo luchar por ser mejor cada día.*

- *A mi tío Adid por estar constantemente pendiente a mis necesidades, por enseñarme a nunca quedarme atrás y estar siempre a la altura de los que me rodean.*

- *A mi tía Martica, por saber comprenderme como una madre y ayudarme a ser un mejor ser humano.*

- *A mi prima Geidy que bien puedo llamar hermana, por su fidelidad y su amor incondicional.*

- *A toda mi familia querida que le debo lo que soy, porque sin su amor y su apoyo no hubiese podido hacer realidad este maravilloso sueño.*
- *A mi cosi que a pesar de la distancia está presente hoy y todos los días*

de mi vida.

- *A mis compañeros de aula, con los que he pasado los mejores momentos. A los que siguen aquí y a los que ya no están. En particular a Albert, Sergito y Santiago, por apoyarme siempre y ayudarme cuando los necesité.*

- *A mis mejores amigos en años Lissy, Cahvely, Yassel, Lorna, Yani por preocuparse por mí y brindarme cariño.*

- *A las compañeras, con las que he pasado tanto momentos alegres, Sonia, Dayana, Lia, Danne, Saily, Yoahanka, La china, Lissandra.*

- *A mi cuki por ser la mejor de las amigas en el transcurso de estos 5 años, aguantar todas mis malacrianzas y por permitirme ser parte de su familia*

- *A las niñas del apartamento*

- *En fin a todos los que de alguna manera influyeron en mi vida y posibilitaron que este día tan especial llegara. Por eso y más, muchas gracias.*

Yudit

Quisiera agradecer a mi tutora Yusleydi por su guía y esfuerzo personal en todo momento, al igual que a nuestro cotutor Miguel por apoyo incondicional. A mi compañera de tesis por estar a la altura y darme fuerzas en momentos difíciles. A mis padres que me apoyaron y me dieron fuerzas para seguir adelante y fueron la luz en la oscuridad en varios momentos, al resto de mi familia, a todos los que nos ayudaron con sus consejos y apoyo, especialmente a mis amigas Adisleydis, Diana y al mulatico de Boutique FiAle (Fidel Alejandro). A los profes que nos apoyaron y nos corrigieron errores en la realización de la guía.

- *A mi querida tía Emelina por apoyarme y darme buenos consejos.*

- *A mi primito Osbel por su apoyo y porque lo quiero muchísimo.*

- *A mi primo Amauris (mandrake como la distribución GNU/Linux), por darme fuerzas y seguridad en la vida.*

- *A mi exnobio Jorge por haberme apoyado durante cuatro años en los buenos y los malos momentos.*

- *A la compañía F.A por los momentos tan lindos y divertidos que pasamos.*

- *A todas mis compañeras de apartamento por esa linda amistad que hemos cultivado durante los 5 años, especialmente a Yixy, Dayamí, Liset, Miriam.*

- *A mi antiguo y actual grupo, por ser las personas lindas y especiales que son, por haber compartido durante estos años, por haber estado presentes en los buenos y malos momentos.*

- *A todas mis amistades durante los 5 años, finalmente a los profes que me han apoyado y me han dado el conocimiento que me ha permitido poder estar hoy aquí, convertida en toda una ingeniera.*

Dedico este trabajo y todo mi esfuerzo a las personas que más amo en el mundo mis padres y mi hermano por el amor, apoyo, confianza y dedicación que me han brindado durante toda mi vida.

Miriam

Dedico todo mi esfuerzo y empeño plasmado aquí a la mujer y el hombre que me dieron la vida: Mi mamá (Bebi) y Mi Papá (Pupi),a mi querido hermanito que tanto adoro Reinier(El Yety), a la memoria de mi abuelito que era la luz de mis ojos, a toda mi familia ,por haber sabido quererme a pesar de mis defectos, por sobrellevarme y haberme sacado adelante por encima de las miles de dificultades que hemos atravesado juntos en la vida, a ellos que son la mayor parte de mi ser, y están plasmados eternamente en mi corazón.

Yudit

Resumen

Todo proyecto en la rama de la informática generalmente, basa su desarrollo en producir software de la mejor calidad posible, que cumpla, y supere las expectativas de los usuarios. La calidad es un logro que todos buscan convirtiéndose en un atributo primordial para el producto de software.
Las dificultades afrontadas hoy en proyectos desarrolladores de software a nivel mundial se basan en la poca experiencia, el poco manejo y las pocas herramientas y documentación que tienen sobre el tema para realizar el trabajo, esto trae como consecuencia resultados ineficientes, incumplimiento de los requerimientos establecidos y la insatisfacción del cliente. Unido a estas consecuencias se integra el hecho de que la empresa productora de software pierda confiabilidad y prestigio. Este problema es evidenciado además en el desarrollo de distribuciones GNU/linux debido a que no existen métodos de calidad de software enfocados al desarrollo de las mismas, que estén disponibles para todos los interesados. Por tanto el presente trabajo se propone elaborar una guía que permita gestionar la calidad en el desarrollo de estas distribuciones, la misma ofrece procesos de aseguramiento de la calidad, además de un conjunto de buenas prácticas que facilitan el trabajo en equipo. Para desarrollar la investigación se hizo uso de métodos científicos como: estadísticos (Delphi), teóricos (Analítico-sintético, Histórico-lógico) y empíricos (Observación, Experimental), además se utilizó como técnica de recopilación de información la Encuesta. Estos métodos demostraron la necesidad de incorporar la propuesta al desarrollo de este tipo de producto y dicha guía es una nueva posibilidad para el aprendizaje de todos aquellos que desempeñen esta labor con el fin de garantizar la calidad del software.

Palabras Claves: Distribuciones GNU/Linux, calidad, aseguramiento de la calidad, guía.

Índice

Introducción

La historia ha demostrado que la ciencia, la técnica, el trabajo y la calidad resultante van unidas, en su desarrollo; que las necesidades humanas, en sus diferentes etapas históricas, requieren de productos, sistemas de producción, herramientas y técnicas adecuadas y que la calidad; como característica de esos productos es una cualidad intrínseca y un concepto intuitivo desde los inicios de la humanidad hasta la actualidad. En los medios de comunicación, la calidad tiene una presencia continuada; incluso los políticos y gobernantes incluyen el término en sus discursos y proyectos. Todo ello es debido al papel fundamental que esta juega en la competitividad de las empresas, pues se ha acabado el tiempo en que la demanda superaba a la oferta y el cliente tenía que conformarse con lo que le ofrecían. Actualmente las compañías dedicadas al suministro de productos y servicios deben adaptarse a las necesidades, gustos y exigencias de los potenciales clientes para mantenerse en el mercado. La industria del software, tiene muchas de las características de la industria tradicional, entre ellas la exigencia de que sus productos respondan a necesidades existentes. La calidad de software no es una tarea fácil, la mayoría de las características que definen al software no se pueden cuantificar fácilmente; generalmente, se establecen de forma cualitativa, lo que dificulta su medición, por lo que se requiere establecer métricas que permitan evaluar cada característica dependiendo del tipo de software que se pretende evaluar. Los factores de calidad aportan significativas ventajas tales como: mejora de la imagen de la empresa, apoyo al marketing, favorece el espíritu de equipo, genera valor añadido y crecimiento sostenido basado en la excelencia [1]. Además existen varias razones que demuestran que la calidad es crítica para la supervivencia de las empresas, ya que la misma no solo es esencial para el comercio internacional sino también que reduce las pérdidas producidas por la ausencia de esta, mantiene a los clientes e incrementa los beneficios y sobre todo es el sello distintivo de los negocios de nivel mundial. Cuba de igual forma se ha propuesto que el software que en el país se desarrolle goce de altos niveles de calidad y principalmente el software libre, en la Universidad de las Ciencias Informáticas (UCI) existe un departamento dedicado exclusivamente a la investigación y desarrollo de aplicaciones de código abierto. Actualmente se desarrolla un proyecto que lleva por nombre Nova, el mismo desarrolla una distribución cubana de GNU/Linux y se encuentra en el desarrollo de la versión 3.0. Nova aspira a proveer una línea de productos y servicios de calidad orientados a usuarios inexpertos en el área de las tecnologías de Software Libre (SWL) o que hayan experimentado un proceso de migración a las mismas. Este sistema operativo va a responder a las necesidades de las instituciones cubanas y va a respetar como valores fundamentales el logro de un estado de soberanía e independencia tecnológica. Sin embargo existen una serie de dificultades que obstaculizan su mejor desarrollo. Durante la Segunda Auditoría Interna realizada al proyecto se determinó que el 60% de las líneas de

desarrollo que lo componen, durante los procesos de calidad de software en un rango de alto, medio, bajo y muy bajo, ejecutan las actividades de calidad a un nivel muy bajo mientras que el resto las llevan a cabo al nivel bajo [2]. En el proyecto no se tienen en cuenta los factores que determinan la calidad de software de los productos que desarrollan, al mismo tiempo no se mide la calidad de los procesos que se ejecutan ni se identifican los errores cometidos desde etapas tempranas del proceso de desarrollo de software. Todos estos conflictos evidencian que las líneas de desarrollo no tienen en cuenta las actividades de garantía de la calidad de software para los diferentes procesos que llevan a cabo. Por otra parte el no hacer uso de las métricas necesarias y el no tener en cuenta desde un inicio los errores durante el proceso de desarrollo de software trae como consecuencia que en varias ocasiones los productos desarrollados no posean un nivel de calidad elevado y que no se puedan diseñar a tiempo las acciones correctivas necesarias para eliminar los problemas detectados. Una de las causas principales de la situación anterior es que no existen métodos de calidad de software enfocados al desarrollo de distribuciones GNU/Linux, que estén disponibles para los interesados en el tema, siendo esta la **situación problémica** de la investigación. De lo anteriormente planteado se llega al siguiente **problema científico:** ¿Cómo incorporar procesos de gestión de la calidad al desarrollo de distribuciones GNU/Linux?

Para dar solución al problema identificado se determina como **objetivo general** de la investigación: Elaborar una guía que permita gestionar la calidad de software de distribuciones GNU/Linux.

El **objeto de estudio** que encierra el problema a tratar es: Procesos de gestión de la calidad de Software.

Campo de acción: Procesos de gestión de la calidad de software en el desarrollo de distribuciones GNU/Linux.

Objetivos específicos:

- Sistematizar sobre las tendencias actuales de la gestión de la calidad de software.
- Elaborar una guía que brinde pautas para gestionar la calidad del software en el desarrollo de distribuciones GNU/Linux.
- Evaluar la guía propuesta para asegurar la calidad de las distribuciones GNU/Linux.

Tareas de la investigación:

- Identificación de los principales conceptos relacionados con las actividades de garantía de la calidad de software para adquirir conocimientos de los aspectos elementales de este tema.
- Sistematización de las principales tendencias que existen en la gestión de la calidad de software para determinar las principales actividades que se realizan en función de garantizar la calidad de software.
- Análisis de normas, modelos y estándares de calidad que se utilizan en el desarrollo de software para determinar cómo se llevan a cabo los procesos de gestión de la calidad en distribuciones de GNU/Linux.
- Elaboración de una guía que incluye los procesos de gestión de la calidad de software que se deben ejecutar en el desarrollo de distribuciones GNU/Linux.
- Incorporación de pautas definidas en la guía al proceso de desarrollo de software que se sigue en el proyecto Nova para determinar las ventajas y desventajas que implica la aplicación de la guía.

Se concreta la **idea a defender:**
La introducción de una guía para gestionar la calidad en el desarrollo de distribuciones GNU/Linux, puede ayudar a la obtención de procesos con calidad.

Métodos Científicos

Para el desarrollo de esta investigación se utilizan varios métodos científicos que facilitan y guían la investigación, estos son:

Métodos Estadísticos

- **Delphi:** Es un proceso repetitivo. Su funcionamiento se basa en la elaboración de un cuestionario que ha de ser contestado por los expertos. Una vez recibida la información, se vuelve a realizar otro cuestionario basado en el anterior para ser contestado nuevamente. Finalmente permite al responsable del estudio elaborar sus conclusiones a partir de la explotación estadística de los datos obtenidos. En la investigación contribuye a evaluar la guía propuesta teniendo en cuenta la opinión de especialistas en el tema.

Métodos Teóricos

- **Analítico-Sintético:** Permite estudiar de modo general el tema planteado para luego descomponerlo e interiorizar cada aspecto estudiado sintetizando lo esencial en función de lo investigado. Se utiliza en la investigación para instruirse acerca de cuáles son los procesos de gestión de la calidad de software que se persiguen en los distintos desarrollos de software, enfatizando en la manera en que esto se lleva a cabo en las distribuciones GNU/Linux.

- **Histórico-Lógico:** El objetivo principal de este método es estudiar y analizar el marco histórico y estado del arte de las características estudiadas acerca del tema de la investigación. Se utiliza para estar al tanto del comportamiento, desarrollo, y estado actual de los aspectos de la calidad de software a nivel mundial y en la UCI, aprendiendo de las nuevas experiencias y teorías que han surgido a lo largo del tiempo.

Métodos Empíricos

- **Observación:** Consiste en la percepción directa del objeto de investigación, permite conocer la realidad mediante el análisis directo con los objetos y fenómenos.
  Para ello en la investigación se establece una guía de observación. **Ver Anexo1**
- **Experimental:** El experimento es el método empírico para el estudio de un objeto en el cual el investigador crea las condiciones o adapta las existentes para el esclarecimiento de las propiedades, leyes y relaciones del objeto, para verificar una hipótesis, una teoría o un modelo. Permite poner en práctica la solución propuesta para comprobar su eficiencia y eficacia.

Técnica de recopilación de información

- **Encuesta:** Con la utilización de este método se adquiere información sobre cuestiones referidas al problema en análisis. Se hace uso de un cuestionario previamente elaborado, con la idea de conocer la opinión y valoración de la persona seleccionada. Esta técnica es de vital importancia porque entre otras cosas va a permitir llevar a cabo la evaluación de la solución propuesta en la investigación.

Población y Muestra

La población está conformada por las distribuciones GNU/Linux. De esta población se toma una muestra que la constituye la distribución GNU/Linux Nova.

Posibles resultados:

- Parámetros para obtener calidad en el desarrollo de distribuciones GNU/Linux.
- Guía que brinde pautas para gestionar la calidad del software de distribuciones GNU/Linux.

El trabajo de diploma consta de tres capítulos organizados de la siguiente manera:

Capítulo #1. Fundamentación teórica de la gestión de la calidad del software

En este capítulo se dan a conocer los conceptos fundamentales para el entendimiento de la investigación, estudiando los principales aspectos que determinan la gestión de la calidad de software, para sintetizarlos en las necesidades de distribuciones GNU/Linux.

Capítulo #2. Guía para gestionar la calidad de software de distribuciones GNU/Linux.

En este capítulo se especifican los factores y buenas prácticas a seguir para alcanzar una correcta gestión de la calidad de software en cualquier distribución GNU/Linux.

Capítulo #3. Evaluación de la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux.

En este capítulo se evalúa la propuesta de solución teniendo en cuenta el criterio de especialistas en el tema, y los resultados de poner en práctica las pautas definidas en un escenario real en que se trabaja.

Capítulo1: Fundamentación Teórica de la gestión de la calidad de software

Es necesario que la calidad de software esté presente en todo el proceso de desarrollo del mismo. Desde su etapa inicial, los desarrolladores deben considerar las actividades y acciones concretas a tomar para obtener un producto final óptimo y eficiente. Si estas acciones son dejadas para el final, o no se realizan a tiempo, el programa está destinado a fracasar, provocando pérdida de tiempo, esfuerzo y dinero de la empresa que lo desarrolle.
En el presente capítulo se le van a dar respuesta a una serie de interrogantes entre las cuales se encuentran: ¿Qué es calidad?, ¿Qué es calidad de software?, ¿Qué es software libre?, ¿Qué engloba el aseguramiento de la calidad?, ¿Cómo afrontar los procesos de calidad de software?

1.1. Software libre

Según Richard Stallman, fundador de la Free Software Foundation (Fundación del software libre) y el Proyecto GNU, la sociedad necesita información que esté realmente disponible para sus ciudadanos, por ejemplo programas que la gente pueda leer, modificar, adaptar y mejorar, no simplemente operarlos [3]. El software libre ofrece hoy en día un amplio espectro de programas que cubren varias áreas: audio, video y diseño; redes y conectividad; gerenciamiento de archivos; sistemas operativos; gerenciamiento de correo y servidores web, etc. Software libre no significa que no sea comercial. Un programa libre debe estar disponible para el uso comercial, la programación comercial y la distribución comercial. Puede haber pagado dinero para obtener copias de software libre, o puede haber obtenido copias sin costo. Pero sin tener en cuenta cómo obtuvo sus copias, siempre tiene la libertad de copiar y modificar el software, incluso de vender copias. En efecto, en los tiempos que corren y en el futuro, el dominio de los conocimientos científicos y tecnológicos básicos es y serán parte fundamental de la libertad de los ciudadanos; el software libre, por su propia naturaleza, es parte de esta base de conocimientos y es una herramienta inigualable para facilitar la educación en ciencias y tecnologías.

1.2. Concepto de calidad

Existen diferentes conceptos de calidad propiciados por relevantes figuras y organizaciones de prestigio internacional y nacional implicadas en el estudio y desarrollo del software. La norma ISO 8402 (International Standard Organization) define Calidad como: “el conjunto de características de

un elemento que le confieren la aptitud para satisfacer necesidades explicitas e implícitas" [4].

La Sociedad Americana para el Control de Calidad (ASQC, American Society for Quality Control), la define como "el conjunto de características de un producto, proceso o servicio que le confieren su aptitud para satisfacer las necesidades del usuario o cliente" [5].

Para el alcance de la investigación se arriba a la conclusión de que calidad, es un conjunto de propiedades, características, cualidades y atributos inherentes a un producto o servicio que determinan sobre este la ausencia de defectos y le confieren capacidad para satisfacer necesidades implícitas o explícitas.

1.3. Calidad de Software

Debido a la importancia que se le dedica a la calidad del software durante el proceso de desarrollo de los mismos, en estudios realizados a través de los años, varios autores la han catalogado de diferentes maneras:

Según el profesor Roger S. Pressman uno de los grandes investigadores de la Ingeniería de Software define la Calidad del Software como: "concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente" [6].

Al mismo tiempo la ISO/IEC (International Standart Organization), Organización Internacional de Estándares la define como: "la totalidad de rasgos y atributos de un producto de software que le apoyan en su capacidad de satisfacer sus necesidades explícitas e implícitas" [7].

Una vez abordadas las definiciones anteriores se llega a determinar que un software con calidad es aquel que cumpla con las necesidades y requerimientos del cliente, para lograr la satisfacción del mismo.

1.3.1. Control de Calidad de Software

Software Quality Control (Control de Calidad en Software) se define como: “Realización de pruebas para detectar defectos, bloqueando la publicación de productos defectuosos y mejorando el resultado en diferentes iteraciones del ciclo de entrega- certificación” [8].

Software Quality Assurance (garantía de calidad del software (SQA)): este medio de seguimiento de los procesos de ingeniería de software y métodos utilizados para asegurar la calidad, define el Control de Calidad de Software como: “Los mecanismos que se implican en el proceso de desarrollo, verificando que se sigue unos estándares y procedimientos, y asegurando que los problemas se encuentran y se tratan adecuadamente” [9].

1.3.1.1. Mecanismos y estrategias en el Control de Calidad de Software

Mecanismo: Calidad de desarrollo

- Estilo y buenas prácticas
- Auditorías de código
- Test unitarios
- Test de integración
- Test de regresión

Estos mecanismos ayudan a la detección temprana de defectos.

Mecanismo: ciclos de testing

Se programan "entregas" al Departamento de Calidad previas a la versión que se va a entregar al cliente. Cada entrega tiene un objetivo claro, y se puede repetir si no alcanza resultados concretos. Mejora el producto y se detectan errores antes de su salida al mercado.

Mecanismo: automatización

Un Frameworks, es un esquema como un esqueleto o patrón para el desarrollo y/o la implementación de una aplicación. Se utiliza gran cantidad de Frameworks para automatizar todo tipo de pruebas. Se realiza la mayor cantidad de pruebas en el menor tiempo posible. Con este mecanismo desaparece la línea entre desarrollo y calidad, logrando el trabajo en equipo por un objetivo en común.

Mecanismo: Bug Tracking público

Bug Tracking, es un sistema de control y seguimiento de errores que permite sanear totalmente las aplicaciones y otros proyectos, asegurando que queden libres de fallos. Cualquier versión no final debería estar abierta al público con el objetivo de integrar al usuario en el ciclo de testing. Debe ser fácil para el usuario apuntar a un defecto. El usuario puede informar que algo falla, pero sus informes no serán de calidad. División del área de calidad dedicada a los informes de usuario.

1.3.2. Modelos de Calidad de Software

En la práctica, los modelos de calidad desempeñan un papel realmente significativo en la gerencia de calidad durante el proceso de desarrollo. Al mismo tiempo resultan de gran utilidad para llevar a cabo la medición del nivel de complejidad de un sistema de software. Es interesante destacar que la organización y descomposición de los atributos de calidad ha permitido el establecimiento de modelos específicos para efectos de la evaluación de la calidad arquitectónica, antes de pasar a conocer los modelos de calidad de software es importante conocer: ¿Qué es un modelo de calidad de software? Este consiste en un conjunto de buenas prácticas para el ciclo de vida del software, enfocado en los procesos de gestión y desarrollo de proyectos. A continuación se mencionan y detallan algunos, como: CMMI, Norma McCall, ISO 15504.

1.3.2.1. Modelo CMMI

El CMM - CMMI (Capability Maturity Model) es un modelo de calidad del software que clasifica las empresas en niveles de madurez. Estos niveles sirven para conocer la madurez de los procesos que se realizan para producir software. El mismo fue desarrollado por el Carnegie Mellon SEI (Software Engineering Institute), creado por el DoD (Department of Defense) de USA [10].

Los niveles CMM - CMMI son 5:

Figura 1: Los 5 niveles de CMM-CMMI.

- **Inicial o Nivel 1 CMM – CMMI:** Aquí se encuentran las empresas que no tienen procesos. Los presupuestos se disparan, no es posible entregar el proyecto en fecha. No hay control sobre el estado del proyecto, el desarrollo del proyecto es completamente opaco.

- **Repetible o** Nivel **2 CMM - CMMI:** Quiere decir que el éxito de los resultados obtenidos se puede repetir. La principal diferencia entre este nivel y el anterior es que el proyecto es gestionado y controlado durante el desarrollo del mismo. El desarrollo no es opaco y se puede saber el estado del proyecto en todo momento. Los procesos que hay que implantar para alcanzar este nivel son: Gestión de requisitos, Planificación de proyectos, Seguimiento y control de proyectos, Gestión de proveedores, Aseguramiento de la calidad y Gestión de la configuración.

- **Definido o Nivel 3 CMM - CMMI**: Alcanzar este nivel significa que la forma de desarrollar proyectos (gestión e ingeniería) está establecida, documentada y que existen métricas (obtención de datos precisos) para la consecución de objetivos concretos. Los procesos que hay que implantar para alcanzar este nivel son: Desarrollo de requisitos, Solución Técnica, Integración del producto, Verificación, Validación, Desarrollo y mejora de los procesos de la organización, Definición de los procesos de la organización, Planificación de la formación, Gestión de riesgos, Análisis y resolución de toma de decisiones. La mayoría de las empresas que llegan al nivel 3 se detienen en el mismo, ya que es un nivel que proporciona muchos beneficios y no ven la necesidad de ir más allá porque tienen cubiertas la mayoría de sus necesidades.

- **Cuantitativamente Gestionado o Nivel 4 CMM - CMMI**. Los proyectos usan objetivos medibles para alcanzar las necesidades de los clientes y la organización. Los procesos que hay que implantar para alcanzar este nivel son: Gestión cuantitativa de proyectos, Mejora de los procesos de la organización

- **Optimizado o Nivel 5 CMM - CMMI**. Los procesos de los proyectos y de la organización están orientados a la mejora de las actividades. Los procesos que hay que implantar para alcanzar este nivel son:, Innovación organizacional, Análisis y resolución de las causas. Los niveles 4 y 5 se pueden realizar simultáneamente ya que están muy relacionados.

A grandes rasgos se ha realizado un resumen de los diferentes niveles de CMMI.

Aseguramiento de la calidad del proceso y producto en CMMI

Entre las áreas de proceso, en la categoría de soporte para el nivel 2 de CMMI se encuentra PPQA (Process and Product Quality Assurance), Aseguramiento de la calidad de producto y proceso. Esta es el área de proceso principal para el aseguramiento de la calidad del software dentro de CMMI.

Metas y prácticas

El propósito del Aseguramiento de la Calidad de Proceso y Producto es proporcionar a los diferentes equipos y a la gerencia una visión objetiva de los procesos y productos asociados. El objetivo fundamental de PPQA es garantizar que los procesos definidos están siendo respetados en la organización, así como poder detectar deficiencias en la forma de trabajar establecida.

Las metas de esta área de proceso son: 1) Evaluar objetivamente la ejecución de los procesos, los elementos de trabajo y servicios en contraste a los procesos, estándares y procedimientos definidos; 2) Identificar y documentar no conformidades; 3) Proporcionar información a las personas que están usando los procesos y a la gerencia del proyecto, de los resultados de las actividades del aseguramiento de la calidad; y 4) Asegurar de que las no conformidades son tratadas.

Las prácticas en éste área de proceso aseguran que los proceso establecidos son implementados mientras que las prácticas en el área de proceso de verificación aseguran que se cumplen los requerimientos especificados. En ciertas ocasiones estas dos áreas de proceso trabajan sobre el mismo producto pero desde dos perspectivas diferentes.

El cumplimiento de estas metas y prácticas son el camino para incrementar la madurez de la

organización en el desarrollo de software conforme a CMMI-SW, la guía para este objetivo es el equipo de QA compuesto por personal experimentado, entrenado en los procesos y actividades de QA que asegurará un nivel de calidad estándar de productos y servicios para los clientes.

Validación en CMMI

Tiene como propósito demostrar que un producto o componente de producto se ajusta a su uso previsto cuando se sitúa en su entorno previsto.

Verificación en CMMI

Tiene como propósito asegurar que los productos de trabajo seleccionados cumplen sus requerimientos especificados. Las prácticas de VER permiten identificar defectos en etapas tempranas de la creación del producto y reducir los altos costos asociados a la identificación y corrección de defectos que se pueden presentar más adelante.

1.3.2.2. ISO

La Organización Internacional para la estandarización, mejor conocida como ISO, es la agencia especializada en estandarización, conformada por representantes de los cuerpos normalizadores, fue establecida oficialmente el 23 de febrero de 1947 con el objeto de promover el uso de estándares a nivel internacional, de tal manera que se facilitara el intercambio de bienes y servicios casi como el desarrollo científico y tecnológico. Actualmente abarca los estándares nacionales de 91 países. En los Estados Unidos, la representación se llama The American National Standards Institute (ANSI) [11].

1.3.2.3. Modelo SPICE (ISO/IEC 15504)

Es un estándar internacional que se refiere a las ofertas nacionales del modelo de la madurez. Es el modelo para la mejora y evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de software, también conocido como SPICE (Software Process Improvement and Capability Determination), en español Determinación de la Capacidad y Mejora de los Procesos de Software [12].

Características del Modelo ISO/IEC 15504

Establece un marco y los requisitos para cualquier proceso de evaluación y proporciona requisitos para los modelos a ser utilizados. Proporciona guías para la definición de las competencias de un evaluador de procesos. Actualmente tiene 10 partes: 1-7 completadas y 8-10

en fase de desarrollo. Comprende: evaluación de procesos, mejora de procesos, determinación de capacidad. Proporciona en su parte 5 un Modelo de evaluación para los procesos de ciclo de vida del software definidos en el estándar ISO/IEC 12207 que define los procesos del ciclo de vida del desarrollo, mantenimiento y operación de los sistemas de software. Equivalencia y compatibilidad con CMMI. ISO forma parte del panel elaborador del modelo CMMI y SEI y viceversa, y se mantiene la compatibilidad y equivalencia de ésta última con 15504.

Niveles del Modelo SPICE ISO/IEC 15504

El modelo contempla la evaluación y mejora por niveles de madurez, es decir, diferentes estados en los que puede encontrarse una organización en función de la calidad de sus procesos. Estos niveles de madurez van desde el nivel más básico, el cero, hasta el más maduro el 5. Cada uno de los niveles tiene un conjunto de procesos asociados que están definidos en la Norma ISO/IEC 12207 y son específicos para el desarrollo del software.

- **0: no realizada:** no hay productos de trabajo identificables.
- **1: realizada informalmente:** planificación y seguimiento dependientes del conocimiento individual. Productos de trabajo identificables.
- **2: planificada:** verificada de acuerdo a los procedimientos especificados.
- **3: bien definida:** procesos bien definidos y documentados
- **4: controlada cuantitativamente:** medidas detalladas de realización, predicción, etc. Productos de trabajo evaluados cuantitativamente.
- **5: mejorada continuamente:** objetivos cuantitativos de eficiencia basados en los objetivos de negocio.

1.3.2.4. Modelo de McCall

El Modelo de McCall describe la calidad como un concepto elaborado mediante relaciones jerárquicas entre factores de calidad, en base a criterios y métricas de calidad. Este enfoque es sistemático y permite cuantificar la calidad a través de las siguientes fases: 1) Determinación de los factores que influyen sobre la calidad del software; 2) Identificación de los criterios para juzgar cada factor; 3) Definición de las métricas de los criterios y establecimiento de una función de normalización que define la relación entre las métricas de cada criterio y los factores correspondientes; 4) Evaluación de las métricas; 5) Correlación de las métricas a un conjunto de guías que cualquier equipo de desarrollo va poder seguir; y 6) Desarrollo de las recomendaciones para la colección de métricas.

Este modelo evalúa el software desde tres puntos de vista distintos:

En este modelo, el término factor de calidad define características claves que un producto debe exhibir. Los atributos del factor de calidad que define el producto son los nombrados criterios de calidad. Las métricas de calidad denotan una medida que puede ser utilizada para cuantificar los criterios. Los mismos se clasifican según sus características en: 1) Operación del producto (utilizándolo); 2) Revisión del producto (cambiándolo); 3) Transición del producto (portándolo). La confianza en estos modelos se debe al hecho de que fueron sustraídos de las experiencias de varias empresas y de muchos proyectos exitosos que ya se han desarrollados.

1.3.3. Normas

"La norma, que surge como resultado de la actividad de normalización, es un documento que establece las condiciones mínimas que debe reunir un producto o servicio para que sirva al uso al que está destinado" [13]. A continuación se va a abordar acerca de varias normas básicas para el desarrollo de la investigación, entre las cuales podemos hacer mención de ISO 9001, ISO 1028, ISO/IEC 12207.

1.3.3.1. Norma ISO 9001

En el desarrollo de esta Norma Internacional se han tenido en cuenta los principios de gestión de la calidad enunciados en las Normas ISO 9000 e ISO 9004. Esta promueve la adopción de un enfoque basado en procesos, permite el desarrollo, implementación y la mejora la eficacia de un sistema de gestión de la calidad, para aumentar la satisfacción del cliente mediante el cumplimiento de sus requisitos.

Objetivo: Especificar los requisitos para un sistema de gestión de la calidad.

Aplicación: Todos los requisitos son genéricos y se pretende que sean aplicables a todas las organizaciones sin importar su tipo, tamaño y producto suministrado.

Requisitos generales: La organización debe establecer, documentar, implementar y mantener un sistema de gestión de la calidad y mejorar continuamente su eficacia de acuerdo con los requisitos de esta. La organización debe gestionar estos procesos de acuerdo con los requisitos de la norma. En los casos en que se opte por contratar externamente cualquier proceso que afecte la conformidad del producto con los requisitos, la organización debe asegurarse de controlar tales procesos por lo que debe estar identificado dentro del sistema de gestión de la calidad.

1.3.3.2. Norma ISO 1028

Propósito: Definir las revisiones sistemáticas aplicables a la adquisición de software, el suministro, el desarrollo, operación y mantenimiento. Esta norma describe cómo llevar a cabo una

revisión.

Ámbito de aplicación: Esta norma establece los requisitos mínimos aceptables para el análisis de programas sistemáticos, incluye los siguientes atributos: a) Equipo de la participación, b) Los resultados documentados de la revisión, y c) Los procedimientos documentados para la realización de la revisión.

¿Qué es una revisión técnica formal (RTF)?

Una RTF es una actividad de garantía de calidad del software llevada a cabo por los ingenieros del software y otros. Se enfoca principalmente en comprobar hasta qué punto el producto está bien elaborado técnicamente. La norma ISO 1028 que define cinco de este tipo de revisiones.

- **Revisiones de gestión:** sirven para controlar el progreso y detectar inconsistencias de los planes con la programación y los requisitos.
- **Revisiones técnicas:** revisan la documentación producida a lo largo del proyecto.
- **Inspecciones:** revisiones que involucran al autor de un producto.
- **Recorrido:** inspecciones conducidas únicamente por miembros del grupo de desarrollo que examinan una parte específica del producto.
- **Auditorías:** evaluaciones independientes sobre el cumplimiento de estándares, planes, procedimientos entre otros.

Esta norma no establece la necesidad de realizar exámenes específicos, pero si establece las definiciones, requisitos y procedimientos que son aplicables a los exámenes de los productos de desarrollo de software en todo el ciclo de vida del mismo. Los usuarios de esta norma deberán especificar dónde y cuándo se aplica esta norma y cualquier desviación prevista de la misma.

Aplicación de la norma: Los procedimientos y la terminología definida en esta norma se aplican a la adquisición de software, el suministro, el desarrollo, operación y mantenimiento de los procesos que requieren revisiones sistemáticas.

1.3.3.3. Norma ISO/IEC 12207

La norma ISO/IEC 12207 surge a principios de la década de los noventa, como un estándar internacional. Su principal motivación fue establecer un marco de trabajo común a la ingeniería de software, aplicable a la ingeniería y a la gestión a lo largo de todo el ciclo de vida del producto.

Ámbito de aplicación: Esta Norma Internacional no está destinada a productos de software disponibles en el mercado, a menos que se incorporen a un producto a entregar. Está dirigida a los adquirentes de los sistemas y productos de software, servicios, proveedores y desarrolladores.

Aplicación: En esta cláusula se presentan los procesos de ciclo de vida del software que se pueden emplear para la adquisición, suministro, desarrollar, operar y mantener los productos de software. El objetivo es proporcionar una guía para que los usuarios mediante esta norma internacional puedan orientarse en ella y aplicarla con prudencia.

Procesos

Los procesos se clasifican en tres tipos: principales, de soporte y de la organización. Los procesos de soporte y de organización deben existir independientemente de la organización y del proyecto ejecutado. Dentro del proceso de soporte se encuentran ocho grupos de procesos uno de ellos es el de aseguramiento de la calidad. El propósito del mismo es proveer de mecanismos para asegurar que los productos y servicios cumplan con los estándares y requerimientos establecidos, y que el desarrollo de otros procesos se apegue lo más posible a lo planificado originalmente. Las actividades son: Implementación del proceso; Aseguramiento del producto; Aseguramiento del proceso: Este proceso proporciona el marco para garantizar la independencia y objetividad de conformidad de los productos o servicios, con sus obligaciones contractuales y el cumplimiento de sus planes establecidos [14]; y Aseguramiento del sistema de calidad

1.3.4. Factores de calidad del software

Originalmente, la calidad de un programa o sistema se evaluaba de acuerdo al número de defectos por cada mil líneas de código. En 1988, un estudio realizado en los EEUU, demostró que se introducían cerca de sesenta defectos por cada mil líneas de código (60 de f/KLOC), hoy se le adicionan otros factores a la calidad del software. Los factores que determinan cualidades tanto internas como externas que debe poseer un software, están clasificados en tres grupos [15]:

1er grupo Operaciones del producto: características operativas

Corrección	Grado en que un programa satisface su especificación y logra los objetivos marcados por el usuario. (¿Hace lo que se le pide?).
Fiabilidad	Grado en que se puede esperar que un programa lleve a cabo las funciones esperadas con la precisión requerida. (¿Lo hace de forma fiable todo el tiempo?).
Eficiencia	Cantidad de recursos de computadoras y de código requeridos por el programa para realizar sus funciones con los tiempos de respuesta adecuados. (¿Qué recursos de hardware y software necesito?).

Integridad	Grado en que puede controlarse el acceso al software o a los datos por usuarios no autorizados. (¿Puedo controlar su uso?).
Facilidad de uso	Esfuerzo necesario para aprender, utilizar, preparar las entradas e interpretar las salidas de un programa. (¿Es fácil y cómodo de manejar?).

Tabla 1: Factores de calidad

2do grupo Revisión del producto: capacidad para soportar cambios.

Facilidad de mantenimiento	Esfuerzo requerido para localizar y arreglar un error en un programa. (¿Puedo localizar los fallos?).
Flexibilidad	Esfuerzo requerido para modificar un programa. (¿Puedo añadir nuevas opciones?).
Facilidad de prueba	Esfuerzo requerido para probar un programa de forma que se asegure que realiza la función requerida. (¿Puedo probar todas las opciones?).

Tabla 2: Factores de calidad

3er grupo Transición del producto: adaptabilidad a nuevos entornos.

Portabilidad	Esfuerzo requerido para transferir un programa desde un entorno HW y/o SW a otro. (¿Podré usarlo en otra máquina?).
Interoperabilidad	Esfuerzo requerido para acoplar un sistema con otras aplicaciones o sistemas. (¿Podrá comunicarse con otras aplicaciones o sistemas informáticos?).

Tabla 3: Factores de calidad

1.4. Aseguramiento de la Calidad de Software

Mundialmente es denominado por la mayoría de las empresas como QA o Quality Assurance, permite elaborar actividades sistemáticas que se necesitan para lograr la calidad en el producto, que en este caso es un software. Obviamente esta planificación debe hacerse antes del desarrollo del software. Pressman define que el aseguramiento de la calidad puede tener las siguientes actividades: evaluaciones en las etapas del desarrollo, auditorías y revisiones, estándares que se van a aplicar al proyecto, mecanismos de medida (métricas), métodos y herramientas de análisis, diseño, programación y prueba, documentación y control de software [16].

En Cuba las empresas que producen software velan por la calidad de su creación, con el objetivo de aumentar la productividad y lograr la satisfacción del cliente. La UCI, además de formar conocimientos sobre la ciencia informática en los jóvenes que en ella estudian, es una

gran desarrolladora de software. Ha logrado aumentar en pocos años su prestigio en esta área. En los proyectos productivos, cada persona involucrada en este proceso ya sea desarrollador, analista, arquitecto, jefe de proyecto, cliente o integrante del grupo de aseguramiento de la calidad son responsables de la calidad del software resultante.

Verificación y Validación

Dentro de las actividades de verificación y validación ligadas al control de la calidad se incluyen las pruebas y procesos de revisión y auditorías. Las revisiones según el estándar ANSI/IEEE Std 1028:1988 se definen como: evaluación de un elemento para determinar diferencias con los resultados planeados y recomendar mejoras [17]. Las auditorías según el estándar ISO 19011:2002 se definen como: proceso sistemático, independiente y documentado para evaluar el estado actual de manera objetiva con el fin de determinar la extensión en que se cumplen los criterios de auditoría [18].

Métricas del software

Las métricas son un buen medio para entender monitorizar, controlar y predecir el desarrollo de software. El glosario de términos de estándares de IEEE define métrica como una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado [19].

Gestión de la configuración del software

La gestión de configuración del software (GCS) es una actividad de protección que se realiza a lo largo de todo el desarrollo de software. Considerando que durante el proceso de desarrollo de software el cambio puede producirse en cualquier momento, las actividades que plantea la (GCS) permiten, identificar, controlar y garantizar que el cambio se realiza adecuadamente e informar del mismo a todos los interesados. Las **actividades de la GCS** son**:** 1) Identificación: Se trata de establecer estándares de documentación y un esquema de identificación de documento; 2) Control de cambios: Consiste en la evaluación y registro de todos los cambios que se hagan de la configuración software; 3) Auditorías de configuraciones: Sirven, junto con las revisiones técnicas formales para garantizar que el cambio se ha implementado correctamente; y 4) Generación de informes.

Plan de aseguramiento de la calidad

Para adaptar las pautas marcadas por los sistemas de calidad a cada proyecto se debe generar un plan específico a cada uno de ellos y este precisamente es el plan de

aseguramiento de la calidad. El mismo es formado por el grupo de ACS del proyecto y sirve como guía para el desarrollo de las actividades de aseguramiento de la calidad.

Estándares

La utilización de estándares como parte del proceso de aseguramiento de la calidad está presente en todo el proceso de desarrollo, y siendo más precisos, en todo el ciclo de vida. La presencia de estándares asociados directa o indirectamente al desarrollo de software es abundante. La ISO (International Standard Organization) ha aportado estándares para la industria del software. Algunos de los más importantes son: la

- ISO 9001.Modelo de sistemas de calidad para el aseguramiento de la calidad en el diseño, desarrollo, producción, instalación y mantenimiento.
- ISO 90003.Lineamientos para la aplicación de la norma ISO 9001 para el desarrollo, suministro y mantenimiento de software.
- ISO 90042.Gestión de la calidad y elementos de sistemas de calidad.

La IEEE (Institute of Electrical and Electronics Engineers). Asociación de profesionales norteamericanos que aporta criterios de estandarización de dispositivos eléctricos y electrónicos.
Su trabajo es promover la creatividad, el desarrollo y la integración, compartir y aplicar los avances en las tecnologías de la información, electrónica y ciencias en general para beneficio de la humanidad y de los mismos profesionales. Entre sus estándares se encuentra:

- IEEE-STD-830-1998: Especificaciones de los requisitos del software (SRS): Es de vital importancia que desde los inicios del proyecto se establezca un plan y un conjunto de actividades que permitan asegurar la calidad. Además es necesario controlar los cambios durante todo el proceso de desarrollo y crear un sistema de métricas para desarrollar estrategias que faciliten mejorar el proceso de software y como consecuencia mejorar el producto final.

1.5. Garantía de Calidad del Software

Es una disciplina de la ingeniería de software, se especializa en la aplicación de procesos de calidad a lo largo del proyecto de software. Su misión no se limita a actividades de verificación, sino que además asume un rol de liderazgo en la gestión de la calidad durante el proceso de creación y diseño del producto. La garantía de calidad es una especialidad compleja y abundante en metodologías, que hace necesaria la especialización de sus profesionales. Su énfasis es en

procesos y no en creación de productos. A medida que la madurez de una organización crece, se hace evidente la necesidad de asignar la gestión de la calidad a ingenieros de calidad especializados.

1.5.1. Principales actividades de garantía de la calidad del software

- Se usa la metodología de desarrollo apropiada.
- Las actividades de desarrollo han sido debidamente planeadas.
- Se han definido estándares y procedimientos para al proyecto.
- El personal ha sido debidamente entrenado en los procesos de calidad aplicables.
- Se llevan a cabo regularmente revisiones y auditorías independientes.
- El desarrollo es documentado adecuadamente para facilitar la mantención y la reutilización.
- La documentación se produce oportunamente y no después que el desarrollo ha sido completado.
- Los cambios introducidos han sido debidamente controlados.
- Las pruebas efectuadas son eficaces para detectar defectos, especialmente en aquellas áreas de mayor riesgo.
- Las actividades se llevan a cabo de acuerdo a los plazos y en los términos planeados.
- Las desviaciones a los estándares se identifican rápidamente.
- El proyecto está en condiciones para ser sometido a auditorías externas, si corresponde.
- La calidad es verificada con respecto a criterios preestablecidos.
- La gerencia es oportunamente informada de problemas y riesgos relativos a la calidad..
- Los problemas de calidad se analizan y las causas se comunican al proyecto para tomar medidas preventivas que eviten su repetición.

El beneficio principal de un programa de garantía de calidad de software es asegurar a la gerencia del proyecto que los procesos establecidos se han ejecutado cabalmente. Esta evaluación es hecha por un grupo independiente, especializado en métodos de calidad, con un criterio objetivo y con visión de contexto.

1.6. Distribuciones GNU/Linux

GNU/Linux es un sistema de libre distribución por lo que se puede encontrar todos los ficheros y programas necesarios para su funcionamiento en multitud de servidores conectados a Internet. La tarea de reunir todos los ficheros y programas necesarios, así como instalarlos en un sistema y

configurarlo, puede ser una tarea bastante complicada y no apta para muchos. Por esto mismo, nacieron las llamadas distribuciones de GNU/Linux. Una distribución es una recopilación de programas y ficheros, organizados y preparados para su instalación. La mayor parte de las distribuciones son fáciles de manejar.

El desarrollo de Linux y de las diferentes distribuciones ha tomado bastante tiempo y rumbos diferentes, por lo que el número de opciones y características creció de manera muy importante, actualmente existen más de 300 distribuciones diferentes, que en realidad solo se diferencian por el instalador utilizado, el software de manejo de paquetes, el campo de aplicación a que se enfoca y las aplicaciones incluidas. Se tiene así desde distribuciones muy pequeñas basadas sólo en modo de texto para el uso de administradores de sistemas o PCs de características muy limitadas, distribuciones comerciales y mega distribuciones que ocupan varios DVDs y que contienen una enorme cantidad de aplicaciones de todo tipo incluyendo algunas privativas.

Ejemplos de Distribuciones GNU/Linux:

Ubuntu es una distribución libre de GNU/Linux patrocinada por Canonical Ltd. Se centra en la facilidad de uso, amplio soporte de hardware y funcionalidad. Es una de las distribuciones más populares.

Debian GNU/Linux ofrece más que un S.O. puro; viene con 25113 paquetes, programas pre compilados distribuidos en un formato que hace más fácil la instalación en su computadora.

Fedora, una distribución general de buena calidad y fácil de instalar. Incluye lo último en software libre y código abierto.

Nova, distribución de GNU/Linux desarrollada por estudiantes y profesores de la Universidad de las Ciencias Informáticas, con la participación de miembros de otras instituciones, para apoyar la migración a tecnologías de software libre y código abierto que experimenta Cuba como parte del proceso de informatización de la sociedad.

1.7 Proceso de desarrollo de Software

Un proceso define quién, está haciendo, qué, cuándo y cómo para alcanzar un determinado objetivo [20]. Un proceso de desarrollo de software (PDS) es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto. A pesar de la variedad de propuestas de proceso de software, existe un conjunto de actividades fundamentales que se encuentran presentes en todos ellos, estas son:

1. Especificación de software: Se debe definir la funcionalidad y restricciones operacionales que debe cumplir el software.
2. Diseño del sistema: Se define el modelado de un sistema o un proceso, con suficientes detalles como para permitir su interpretación y realización física. Esta es la disciplina o actividad que describe más claramente a los desarrolladores lo que el sistema tiene que hacer y cómo lo debe hacer.
3. Implementación: la implementación puede ser la parte más obvia del trabajo de ingeniería de software, pero no necesariamente es la que demanda mayor trabajo ni la más complicada. La complejidad y la duración de esta actividad está íntimamente relacionada al o a los lenguajes de programación utilizados, así como al diseño previamente realizado.
4. Validación: el software debe validarse, para asegurar que trabaje como fue diseñado y que cumpla con lo que quiere el cliente.
5. Evolución: el software debe evolucionar, para adaptarse a las necesidades del cliente.

Además de estas actividades fundamentales, Pressman [21] menciona un conjunto de "actividades protectoras", que se aplican a lo largo de todo el proceso del software. Ellas se señalan a continuación: Seguimiento y control de proyecto de software, Revisiones técnicas formales, Garantía de calidad del software, Gestión de configuración del software, Preparación y producción de documentos, Gestión de reutilización, Mediciones, Gestión de riesgos.

Elementos que caracterizan un proceso de desarrollo de software.

Un marco común del proceso, definiendo un pequeño número de actividades del marco de trabajo que son aplicables a todos los proyectos de software, con independencia del tamaño o complejidad.

Un conjunto de tareas, cada uno es una colección de tareas de ingeniería del software, hitos de proyectos, entregas y productos de trabajo del software, y puntos de garantía de calidad, que

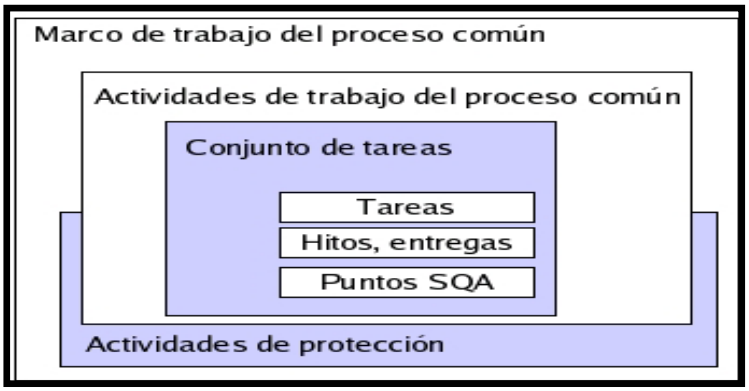


Figura 3: Procesos de desarrollo de software.

permiten que las actividades del marco de trabajo se adapten a las características del proyecto de software y a los requisitos del equipo del proyecto.

Las actividades de protección, tales como garantía de calidad del software, gestión de configuración del software y medición.

La distribución Nova actualmente se encuentra en proceso de desarrollo, los subprocesos que tienen que ver con la construcción de la misma son: Levamiento de requisitos, Empaquetamiento de software, Compilación de software, Generación de repositorios, Creación de (sistemas) instalables, Pruebas.

1.7.1. Metodología: Características generales, actividades de SQA.

En un proyecto de desarrollo de software la metodología define quién debe hacer qué, Cuándo y cómo debe hacerlo. Una metodología es un proceso. No existe una metodología de software universal. Las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigen que el proceso sea configurable. Debido a esto, en la actualidad existen varias clasificaciones para las mismas, entre las que se pueden encontrar a las Metodologías Estructuradas, Orientadas a Objetos, Tradicionales o Pesadas y Ágiles.
Estas últimas están más orientadas a la generación de código con ciclos rápidos, se dirigen a equipos de desarrollo pequeños donde los clientes y desarrolladores trabajan continuamente en constante comunicación. Además el modelo de desarrollo ágil es fácil de aprender, bien documentado y adaptable. A continuación se revisan brevemente algunas de las metodologías.

SXP

Desde septiembre del año 2006 un equipo de ingenieros de la UCI elaboró una metodología ágil compuesta por sus referentes internacionales SCRUM y XP, denominada SXP. La misma ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad

de los miembros del equipo y ayudando al líder del proyecto a tener un mejor control del mismo.
En cada una de sus fases se realizan numerosas actividades tales como el levantamiento de requisitos, la priorización de la lista de reserva del producto (LRP), definición de las historias de usuario (HU), diseño, implementación, pruebas, entre otras; de donde se generan artefactos para documentar todo el proceso. Las entregas son frecuentes. La garantía de calidad, es una parte esencial en el proceso de XP. La realización de pruebas y la publicación de los resultados debe ser lo más rápido posible, para que los desarrolladores puedan realizar con la mayor rapidez los cambios que sean necesarios. A las pruebas de aceptación también se las conoce con el nombre de pruebas de funcionalidad, y constituyen la garantía de que los requerimientos fijados por los usuarios han sido reflejados en el sistema.
SXP está especialmente indicada para proyectos de pequeños equipos de trabajo, rápido cambio de requisitos o requisitos imprecisos, muy cambiantes, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar, de forma que los jefes pueden ver día a día cómo progresa el trabajo [22].

RUP

"El Proceso Unificado de Rational (RUP), es un proceso de ingeniería de software planteado por Kruchten (1996) cuyo objetivo es producir software de alta calidad, es decir, que cumpla con los requerimientos de los usuarios dentro de una planificación y presupuesto establecido. Cubre el ciclo de vida y desarrollo de software" [23].
El Proceso Unificado es un proceso de desarrollo de software cuyo ciclo de vida se caracteriza por ser dirigido por casos de uso, centrado en arquitectura, iterativo e incremental. Consta de cuatro fases o etapas: 1) **Comienzo o Inicio:** Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema; 2) **Elaboración:** Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen; 3) **Construcción:** Se basa en la elaboración de un producto totalmente operativo y en la elaboración del manual de usuario, en esta etapa el objetivo es llevar a obtener la capacidad operacional inicial; 4) **Transición:** Se realiza la instalación del producto en el cliente y se procede al entrenamiento de los usuarios. Puede implicar reparación de errores.

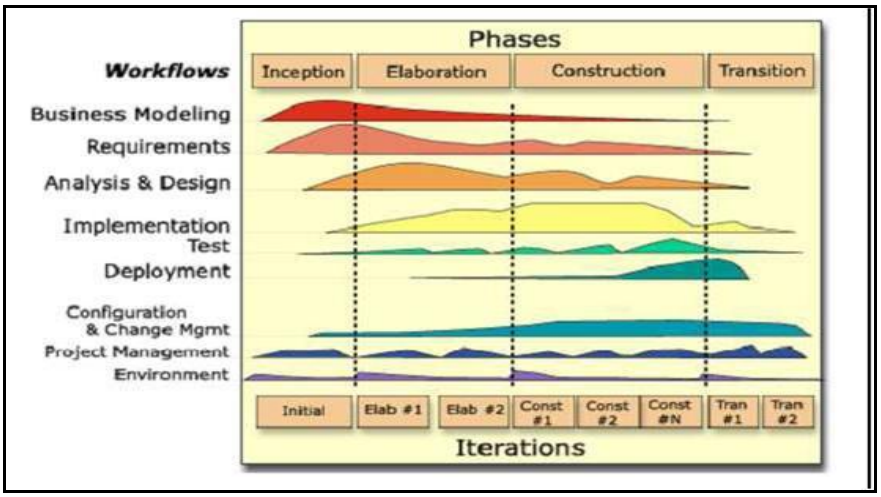


Figura 4: Las fases de la metodología RUP

Además de las fases, el ciclo de vida contiene flujos de trabajos, los cuales se dividen en flujos de trabajo de desarrollo y flujos de trabajo de soporte. Estos son: Modelamiento del negocio, Requerimientos, Análisis y diseño, Implementación, Prueba (Testeo), Despliegue, Administración del proyecto, Administración de configuración y cambios, y Ambiente.

De los 6 Flujos de trabajo de desarrollo el encargado de evaluar la calidad del producto es el de Prueba, esta evaluación no se realiza para aceptar o rechazar el producto al final del proceso de desarrollo, sino que debe ir integrado en todo el ciclo de vida. Además durante las pruebas se llevan a cabo otras actividades como: 1) Encontrar y documentar defectos en la calidad del software, 2) Generalmente asesora sobre la calidad del software percibida, 3) Provee la validación de los supuestos realizados en el diseño y especificación de requisitos por medio de demostraciones concretas, 4) Verificar las funciones del producto de software según lo diseñado y 5) Verificar que los requisitos tengan su apropiada implementación,

En cada una de las iteraciones se obtendrá una versión del producto entregable al cliente.

OpenUP

Es una metodología ágil liberada por el Eclipse Process Framework (EPF), construido sobre el Basic Unified Process, de la IBM, renombrado como OpenUP en el 2006. Como su nombre lo indica (Open – abierto, UP – Proceso Unificado o Unified Process) es una metodología que permite la realización de cambios en sus artefactos, pero que sigue la línea del proceso unificado de desarrollo de software, por lo que va a tener las mismas faces y disciplinas. También presenta las mismas características (Iterativo Incremental, Dirigido por Casos de Usos y Centrado en la Arquitectura).

OpenUP es reconocido mundialmente como uno de los procesos de desarrollo de software de mayor calidad, basándose en los principios de Adaptación, Importancia a los involucrados e interesados en los resultados del proyecto; Colaboración, Valor a la iteración; y Calidad Continua. Está pensada para pequeños y medianos proyectos.

1.8 Resumiendo

El siguiente mapa conceptual hace un resumen de los elementos más importantes abordados en este capítulo.

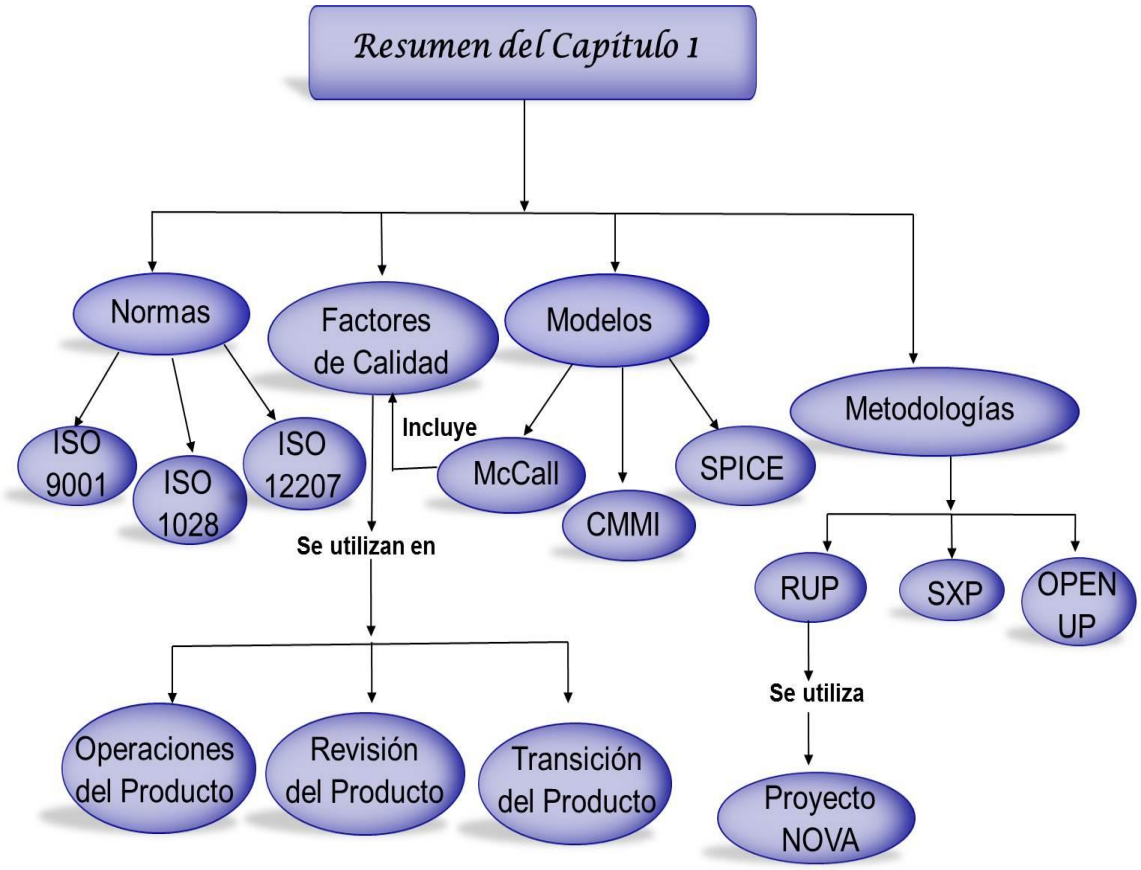


Figura 5: Resumen del capítulo 1

Capítulo2: Guía para gestionar la calidad de software de distribuciones GNU/Linux

En el presente capítulo se presenta una guía para gestionar la calidad de software en el desarrollo de distribuciones de GNU/Linux. Para esto se lleva a cabo un análisis de lo investigado en el capítulo anterior teniendo como resultado un conjunto de buenas prácticas, técnicas y políticas a tener en cuenta en el desarrollo de distribuciones de Linux.

2.1. Estructura de la guía

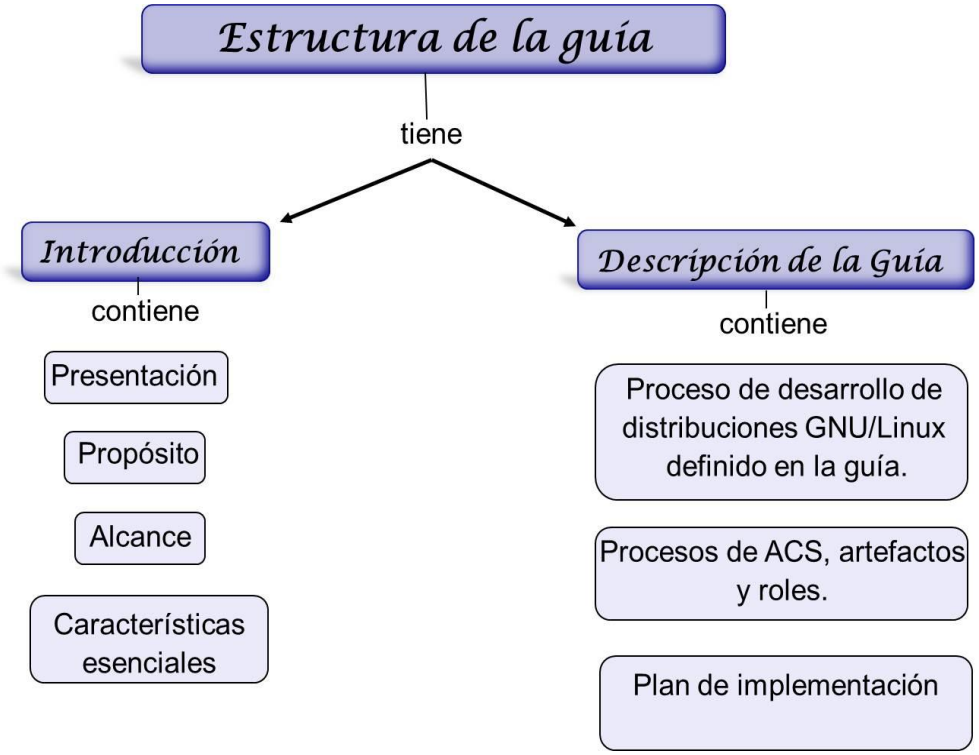


Figura 6: Estructura de la guía

2.2. Introducción de la guía

2.2.1. Presentación

La guía presentada es un documento donde se organiza el trabajo para gestionar la calidad del proceso de desarrollo de distribuciones GNU/Linux, con el objetivo de lograr un mejor nivel de productividad y mejores vías para garantizar la satisfacción del cliente o usuarios finales.

2.2.2. Propósito

La presente guía tiene como propósito orientar a una mejor gestión de la calidad de software durante el desarrollo de distribuciones de GNU/Linux.

2.2.3. Alcance

La guía puede ser utilizada en el proceso de desarrollo de cualquier distribución de GNU/Linux que contemple los subprocesos que se definen en la misma.

2.2.4. Características esenciales

La guía es orientadora, ya que brinda los pasos a seguir para lograr una correcta gestión de la calidad de software, es de gran utilidad, pues permite controlar de forma organizada el trabajo en el desarrollo de cualquier distribución GNU/Linux, mediante los procesos de calidad que son aplicados durante todo el proceso de desarrollo del software.

Los principales procesos de ACS en los que se basa la guía para garantizar la calidad distribuciones Linux se muestran en la figura 6.

Figura 7: Procesos para la gestión de la calidad de software

2.3. Descripción de la guía

2.3.1. Proceso de desarrollo de distribuciones GNU/Linux

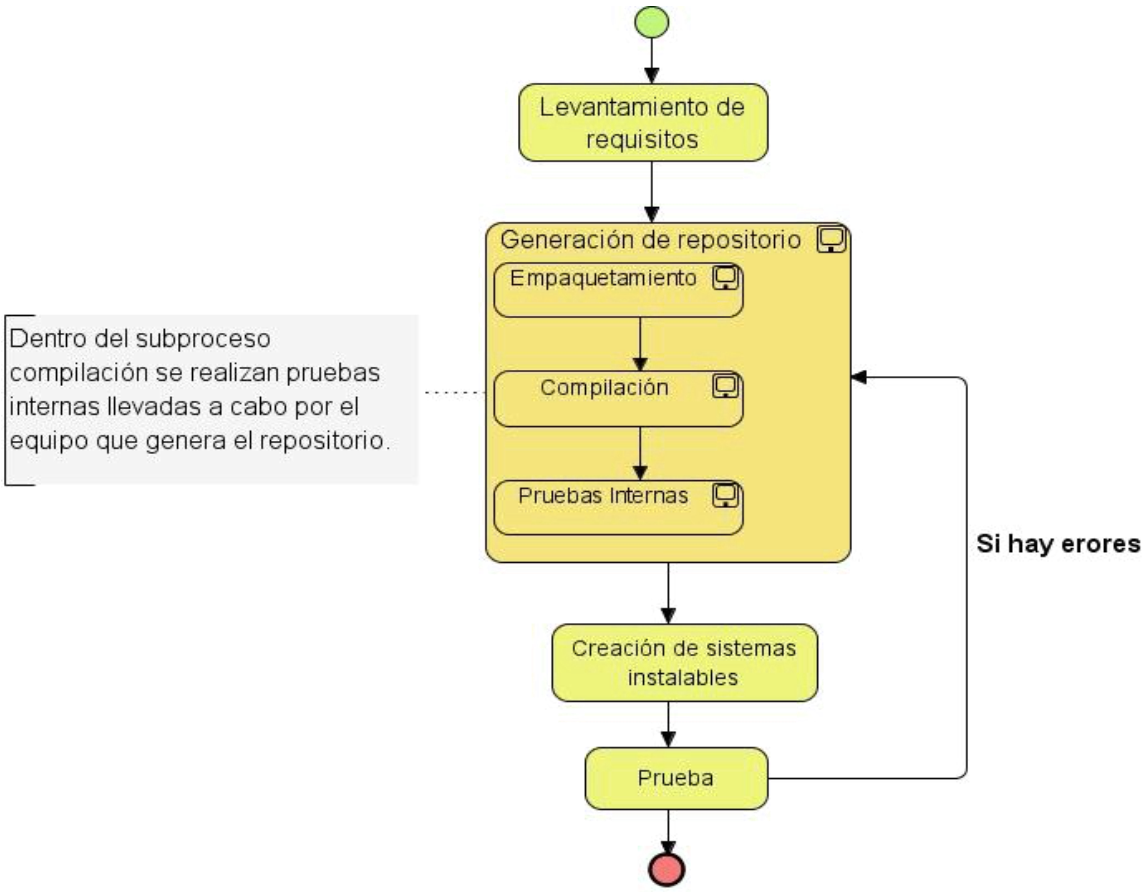


Figura 8: Proceso de desarrollo de distribuciones GNU/Linux.

La figura 8 muestra el proceso de desarrollo de distribuciones GNU/Linux propuesto por la guía, para conformar el mismo se realizó un estudio de distribuciones como: Debian, Ubuntu Nova entre otras, se llegó a la conclusión que los subprocesos que las conforman son bastantes similares, independientemente de que cada proyecto los lleven a cabo cómo estime conveniente.

Es válido aclarar que este proceso desarrollo se trata de una especulación, pues no existe un proceso detallado para el desarrollo de este tipo de software que se encuentre accesible para las personas interesadas.

Cuando se desea realizar una gestión adecuada, eficaz y eficiente en proyectos de software, es necesario que se ponga en funcionamiento cuatro elementos muy importantes: personal, proyecto, proceso y producto.

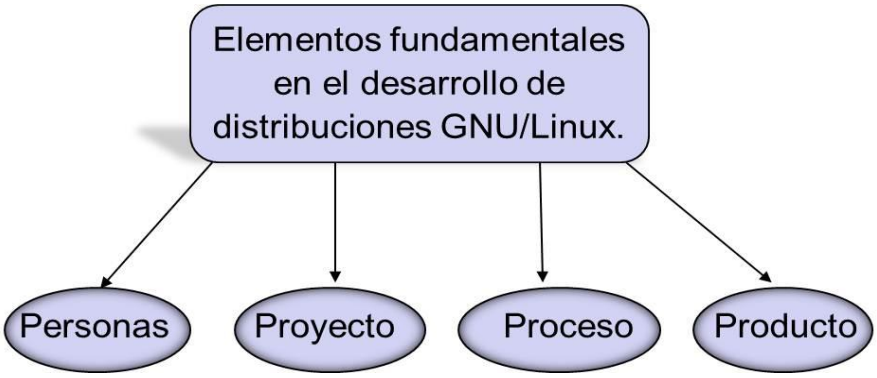


Figura 9: Las cuatro P del proceso de desarrollo de software.

Personas: Los principales autores de un proyecto de software son los arquitectos, desarrolladores, ingenieros de prueba, y el personal de gestión que les da soporte, además de los usuarios, clientes, y otros interesados. Las personas son realmente seres humanos, a diferencia del término abstracto trabajadores.

Proyecto: Elemento organizativo a través del cual se gestiona el desarrollo de software. El resultado de un proyecto es una versión de un producto.

Proceso: Un proceso de ingeniería de software es una definición del conjunto de actividades necesarias para transformar los requisitos de usuario en un producto. Es una plantilla para crear proyectos.

Producto: Artefactos que se crean durante la vida del proyecto, como los modelos, código fuente, ejecutables, y documentación [24].

Los procesos de ACS que define la guía van en función de detectar posibles desviaciones que puedan existir principalmente en los subprocesos que se ejecutan.

2.3.2. Procesos de ACS, artefactos y roles.

2.3.2.1. Procesos ACS aplicadas al personal

Personal	
Nombre del proceso	Formación al personal
Meta	Verificar que se capacite al personal involucrado en el proyecto.
Actividades que componen el proceso	➢ Diagnóstico de las características de las personas con que se cuenta, esto para saber su nivel de conocimiento, motivación y trabajar en función de esto para la asignación de tareas.30% ➢ Comprobación de la gestión de cursos de capacitación, conferencias para asegurar que cada persona involucrada en el proceso este lo suficientemente capacitada para desempeñar su rol.40% ➢ Comprobación de los métodos de formación en función de la entrada de personal al proyecto. 10% ➢ Verificar que estén bien definidas las competencias por roles.20%
Criterios de evaluación	➢ Comprobar que cada persona está capacitada para el rol que desempeña en el proyecto. ➢ Comprobar que se capacite al personal del proyecto.
¿Cómo evaluar el resultado de la aplicación del proceso de ACS?	**Ver anexo 2**
Roles	➢ Líder del proyecto ➢ Asegurador de la calidad ➢ Administrador de la calidad

2.3.2.2. Procesos de ACS aplicadas a los elementos de configuración del proyecto

Elementos de configuración del proyecto	
Nombre del proceso	Auditoria de la gestión de la configuración en el proyecto
Meta	➢ Comprobar si los cambios se han hecho de acuerdo a las políticas definidas en el plan de gestión de la configuración de software. ➢ Comprobar si los cambios se han documentado debidamente.
Actividades que componen el proceso	➢ Auditorías a los elementos de configuración de software.50% ➢ Revisión de los cambios realizados y el procedimiento para estos.50%
Criterios de evaluación	Los criterios de evaluación están reflejados en la siguiente lista de chequeo. ➢ ¿Se ha hecho el cambio especificado en la orden de cambio de ingeniería? ¿Se han incorporado modificaciones adicionales? ➢ ¿Se ha realizado una revisión técnica formal para comprobar la corrección de los ECS? ➢ ¿Se han seguido adecuadamente los estándares de ingeniería del software? ➢ ¿Se han seguido los procedimientos del GCS para señalar el cambio, registrarlo y divulgarlo? ¿Se han actualizado adecuadamente todos los ECS relacionados?
¿Cómo evaluar el resultado de la aplicación del proceso de ACS?	Ver anexo 2

Artefactos de entrada	➢ Elementos de la configuración de software ➢ Plan de GCS ➢ Repositorios ➢ Plantilla donde se documentan las solicitudes de cambio.
Artefactos de Salida	➢ No conformidades detectadas ➢ Acciones correctivas ➢ Evaluación
Roles	➢ Asegurador de la calidad ➢ Administrador de la calidad ➢ Gestor de la configuración de software ➢ Líder de proyecto

2.3.2.3. Procesos de ACS aplicadas al proceso

Levantamiento de Requisitos	
Nombre del proceso	Verificación en el levantamiento de requisitos
Meta	Verificar que el levantamiento de requisitos se haya llevado a cabo según los estándares definidos para este subproceso.
Actividades que componen el proceso	Las actividades de revisiones técnicas formales que se van aplicar son: ➢ Verificar que exista un plan para llevar a cabo el levantamiento de requisitos en el proyecto.20%

	➢ Verificar que el levantamiento de requisitos se lleve a cabo de acuerdo a las políticas definidas en la Guía de migración al software libre. 30% ➢ Garantizar que el levantamiento de requisitos se realice de acuerdo a la norma IEEE 830. 40% ➢ Garantizar que se tienen en cuenta las necesidades de los usuarios finales recogidas en sitios que le dan soporte a la aplicación. 10%
Criterios de evaluación	➢ Comprobar que el levantamiento de requisitos se ajusta a sus especificaciones. ➢ Comprobar que los requisitos cumplan con las características definidas en el documento IEEE-STD-830-1998: Especificaciones de los requisitos del software (SRS)
¿Cómo evaluar el resultado de la aplicación del proceso de ACS?	Ver anexo 2
Artefactos de entrada	➢ Guía de migración ➢ SRS ➢ IEEE-STD-830-1998: Especificaciones de los requisitos del software (SRS) ➢ Reportes de sitios que guardan las necesidades de los usuarios finales
Artefactos de Salida	➢ No conformidades detectadas. ➢ Acciones Correctivas. ➢ Evaluación.
Roles	➢ Asegurador de la calidad. ➢ Administrador de la calidad. ➢ Analista.

Empaquetamiento de software

Nombre del proceso	Verificación del subproceso empaquetamiento de software.
Meta	Verificar que el empaquetamiento se haya llevado a cabo según los estándares definidos para este subproceso. Anexo 4
Actividades que componen el proceso	Las actividades de revisiones técnicas formales que se van aplicar son: ➢ Verificar que exista un plan para llevar a cabo el empaquetamiento en el proyecto.5% ➢ Descubrir errores.20% ➢ Verificar que el software alcanza sus requisitos. 25% Ver Anexo 3 ➢ Garantizar que el software se desarrolla de acuerdo a ciertos estándares predefinidos: Estándar .deb.30%.Ver anexo 3 ➢ Optimización del proceso de compilación.10% ➢ Conseguir un paquete de manera uniforme. 10%
Criterios de evaluación	➢ Comprobar que el software empaquetado se ajusta a sus especificaciones. ➢ Comprobar que el software empaquetado respeta los criterios o estándares definidos en el proyecto. ➢ Verificar que los cambios en el producto de software se implementan adecuadamente.
¿Cómo evaluar el resultado de la aplicación del proceso de ACS?	Ver anexo 2
Artefactos de entrada	➢ Código ➢ Políticas de empaquetamiento de Debian o Ubuntu.
	➢ No conformidades detectadas.

Artefactos de Salida	➢ Acciones correctivas. ➢ Evaluación.
Roles	➢ Asegurador de la calidad. ➢ Administrador de la calidad. ➢ Mantenedor de paquetes.

Compilación	
Nombre del proceso	Verificación del subproceso de compilación.
Meta	Verificar que la compilación se haya llevado a cabo según los estándares definidos para este subproceso. Ver Anexo 4
Actividades que componen el proceso	En función de lo definido en las revisiones del tipo "Recorrido", las actividades a ejecutar son: ➢ Verificar que exista un plan para llevar a cabo la compilación en el proyecto.20% ➢ Verificar que el subproceso se lleve a cabo según las políticas establecidas por el proyecto. 30% Ver Anexo 4 ➢ Observación del proceso de compilación.20% Verificar si produce algún error en la compilación e informar al proceso de empaquetamiento.30%
Criterios de evaluación	➢ Adhesión del proceso a lo establecido por el proyecto para llevar a cabo el mismo.
¿Cómo evaluar el resultado de la aplicación del proceso de ACS?	Ver anexo 2
Artefactos de entrada	➢ Políticas de empaquetamiento de Debian y Ubuntu. ➢ Paquete

Artefactos de Salida	➢ No conformidades detectadas. ➢ Acciones correctivas. ➢ Evaluación.
Roles	➢ Asegurador de la calidad. ➢ Administrador de la calidad. ➢ Mantenedor de paquetes: nadie compila, la compilación se realiza de manera automática, existe un observador que es administrador del servidor de compilación.

Generación de repositorios	
Nombre del proceso	Verificación en la generación de repositorios.
Meta	Verificar que la generación de repositorios se haya llevado a cabo según los estándares definidos para este subproceso. Anexo 4
Actividades que componen el proceso	Las actividades de revisiones técnicas que se van aplicar son: ➢ Verificar que exista un plan para llevar a cabo la generación de repositorio en el proyecto.20% ➢ Descubrir errores.20% ➢ Verificar que el software alcanza sus requisitos.20% Ver anexo 4 ➢ Tener en cuenta los resultados de los procesos de calidad que fueron aplicados a los subprocesos de empaquetamiento y compilación.20% ➢ Garantizar que el software se desarrolla de acuerdo a ciertos estándares predefinidos.10% Ver anexo 4 ➢ Revisar que el paquete que se suba al repositorio está libre de errores.10% ➢ Verificar que se realicen las actualizaciones de acuerdo

	a los estándares definidos.10% Ver anexo 4
Criterios de evaluación	➢ Adhesión del producto de software a sus especificaciones. ➢ Adhesión del proceso que se sigue a los criterios o estándares aplicados en el proyecto. ➢ Verificar que los cambios en el producto de software se implementan adecuadamente. ➢ Verificar que los paquetes binarios sean agrupados en el repositorio.
¿Cómo evaluar el resultado de la aplicación del proceso de ACS?	Ver anexo 2
Artefactos de entrada	➢ Políticas de empaquetamiento de Debian y Ubuntu. ➢ Paquete binario ➢ Repositorios
Artefactos de Salida	➢ No conformidades detectadas. ➢ Acciones correctivas. ➢ Evaluación.
Roles	➢ Asegurador de la calidad. ➢ Administrador de la calidad. ➢ Mantenedor de paquetes.

Pruebas	
Nombre del proceso	Verificación del subproceso pruebas.
Meta	Comprobar que las pruebas realizadas al software se lleven a cabo según el plan de pruebas definido por el proyecto.
Actividades que componen el proceso	➢ Comprobar que exista un plan para llevar a cabo las pruebas en el proyecto.10% ➢ Verificar que esté creado el ambiente y las condiciones para llevar a cabo las pruebas.5% ➢ Verificar que en las pruebas se tuvieron en cuenta los siguientes factores de calidad, usabilidad, funcionalidad, seguridad del software.15% ➢ Revisar si en los escenarios de prueba se tuvieron en cuenta el cumplimiento de los requisitos funcionales y no funcionales.20% ➢ Verificar que se prueben la interfaz y sus componentes .15% ➢ Comprobar que se realicen interacciones con diferentes partes del sistema, como: Sistema Operativo, Sistema de archivo, Hardware, Interfaces entre sistemas 20%.

	➢ Verificar que se elabore un Informe de Pruebas, en el que se registrarán los resultados de las mismas.5% ➢ Verificar que se corrijan los fallos y errores detectados.10%
Criterios de evaluación	➢ Verificar que las pruebas se realicen en cada iteración del software. ➢ Comprobar que los defectos se corrijan tan pronto como sean encontrados. ➢ Comprobar que para cada actividad de desarrollo haya una actividad correspondiente de pruebas. ➢ Comprobar que el producto software respeta los criterios o estándares aplicados en el proyecto y cumpla con los factores de calidad requeridos. Ver anexo 4 ➢ Verificar que se analicen las lecciones aprendidas para futuras liberaciones del proyecto y la mejora de la madurez de prueba. ➢ Correspondencias de las pruebas con el tipo de producto. ➢ Adherencia al proceso de pruebas definido. ➢ Comprobación de los factores de calidad a cumplir en distribuciones de GNU/Linux.
¿Cómo evaluar el resultado de la aplicación del proceso de ACS?	Ver anexo 2
Artefactos de entrada	➢ Producto ➢ Plan de prueba ➢ Casos de prueba ➢ Especificación de requisitos del proyecto

Artefactos de Salida	➢ No conformidades detectadas. ➢ Acciones correctivas. ➢ Evaluación.
Roles	➢ Asegurador de la calidad. ➢ Administrador de la calidad. ➢ Probador. ➢ Desarrolladores del producto.

Los subprocesos creación de sistemas instalables y pruebas, van a estar relacionados con los procesos de ACS que se le aplicarán al producto.

2.1.1.1. Procesos de ACS aplicadas al producto

Producto	
Nombre del proceso	Control de calidad al producto.
Meta	Controlar que el producto cumpla con los requerimientos especificados en el proyecto.
Actividades que componen el proceso	Las actividades de bug tracking público que se van aplicar son: ➢ Control y seguimiento de errores.40% ➢ Controlar que cualquier versión no final esté abierta al público con el objetivo de integrar al usuario en el ciclo de testing.60%
Criterios de evaluación	➢ Controlar que los defectos se corrijan tan pronto como sean encontrados.

	➢ Comprobar que el producto software respeta los criterios o estándares aplicados en el proyecto y cumpla con los factores de calidad requeridos.
¿Cómo evaluar la actividad?	Ver anexo 2
Artefactos de entrada	➢ Producto ➢ Especificación de requisitos de software
Artefactos de Salida	➢ No conformidades detectadas. ➢ Acciones correctivas. ➢ Evaluación.
Roles	➢ Asegurador de la calidad. ➢ Administrador de la calidad. ➢ Probador

Producto	
Nombre del proceso	Validación del producto.
Meta	Validar que el producto cumpla con los requerimientos especificados en el proyecto.
Actividades que componen el proceso	Algunas de las actividades de validación aplicadas al producto fueron: ➢ Validar que se realicen técnicas de evaluación dinámicas.40%

	➢ Validar que se realicen técnicas de evaluación estáticas.40% ➢ Validar que se realicen pruebas de aceptación.20%
Criterios de evaluación	➢ Controlar que los defectos se corrijan tan pronto como sean encontrados. ➢ Comprobar que el producto software respeta los criterios o estándares aplicados en el proyecto y cumpla con los factores de calidad requeridos. ➢ Comprobar que el producto software cumple con los requisitos establecidos por el cliente.
¿Cómo evaluar el resultado de la aplicación del proceso de ACS?	Ver anexo 2
Artefactos de entrada	➢ Producto. ➢ Plan de Pruebas. ➢ Pruebas de aceptación por parte del cliente o usuarios finales. ➢ Especificación de requisitos.
Artefactos de Salida	➢ No conformidades detectadas ➢ Acciones correctivas. ➢ Acta de aceptación
Roles	➢ Asegurador de la calidad. ➢ Administrador de la calidad. ➢ Probador.

2.1.1. Plan de implementación de la guía.

La siguiente figura muestra la aplicación de los procesos de aseguramiento de la calidad que propone la guía al proceso de desarrollo de las distribuciones GNU/Linux. Donde se evidencia que los más aplicados fueron verificación y control, esto se debe a que estos dos procesos permiten identificar errores desde etapas tempranas de la creación del producto, reduciendo así los elevados costos que pudieran ocasionar la identificación y corrección de estos defectos si se presentan más adelante.

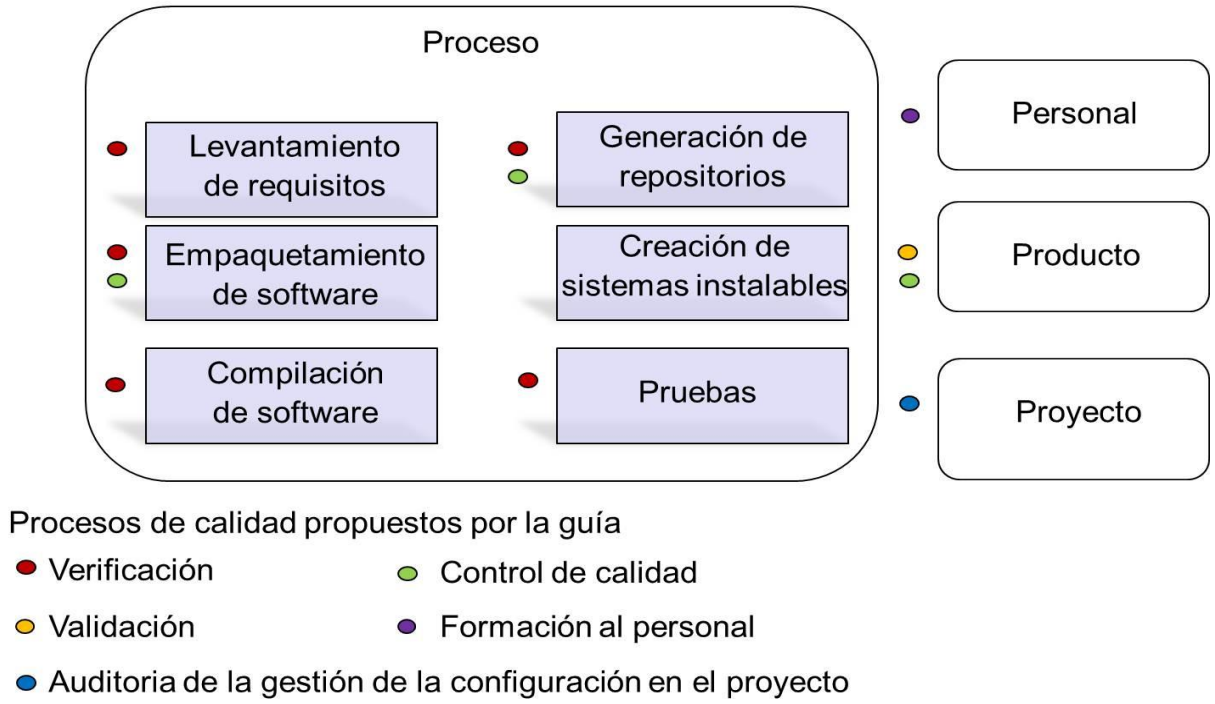


Figura 10: Plan de implementación de la guía

Capítulo3: Evaluación de la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux

Terminada la propuesta de la guía para gestionar la calidad de software en el desarrollo de distribuciones GNU/Linux, se hace necesario someterla a un proceso de evaluación y refinamiento para evaluar la completitud y valor de la misma. Los métodos de predicción posibilitan pronosticar el comportamiento de la propuesta de solución, para lo que se va a utilizar el método Delphi mediante el criterio de expertos.

Por otra parte también se va a llevar a cabo el método experimental poniendo en práctica parte de la solución para comprobar si la misma responde a los objetivos de la investigación.

1.1 Evaluación por el método Delphi

El método Delphi, cuyo nombre se inspira en el antiguo oráculo de Delphos, fue ideado originalmente a comienzos de los años 50 en el seno del centro de investigación estadounidense RAND Corporation por Olaf Helmer y Theodore J. Gordon, como un instrumento para realizar predicciones sobre un caso de catástrofe nuclear. Desde entonces, ha sido utilizado frecuentemente como sistema para obtener información sobre el futuro [26].

Consiste en la selección de un grupo de expertos a los que se les aplican cuestionarios sobre cuestiones referidas al problema en análisis y a la solución propuesta. Mediante las respuestas estos expresan su punto de vista y estas se tabulan para obtener los resultados de forma estadística. El método suele dividirse en tres etapas o fases fundamentales: Fase inicial, Fase exploratoria y Fase final

Las principales características son las siguientes:

- **Anonimato:** se expresa a través del no-conocimiento de las respuestas, puesto que los miembros del grupo contestan las preguntas sin confrontarse, incluso sin conocerse entre sí.

- **Iteración:** se pueden hacer tantas rondas como sean necesarias para obtener datos concisos.
- **Retroalimentación controlada:** después de cada ronda de preguntas se tabulan las respuestas y se procesan de forma tal, que antes de la siguiente ronda los participantes pueden evaluar los resultados de la ronda anterior, así como las razones dadas para cada respuesta y su dispersión del promedio.
- **Respuesta estadística del grupo**: entre cada ronda de preguntas, la información obtenida se procesa por medio de técnicas estadístico-matemáticas, las que dotan al investigador de un instrumento objetivo y concreto en el cual pueden apoyarse para tomar una decisión final.

1.2 Fase inicial

Durante esta primera fase se delimita el contexto, el objetivo a alcanzar con la aplicación del método seleccionado, el diseño, los elementos básicos del trabajo y por último se produce la selección de los expertos que protagonizarán la evaluación de los resultados.

3.2.1. Formulación del problema

Primeramente se definen los atributos **(C)** a evaluar por los especialistas, estos deben ser precisos, medibles e independientes. Los atributos que se tuvieron en cuenta fueron los siguientes:

C1. Importancia de la creación y utilización de la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux.
C2. Alcance de la guía.
C3. Necesidad de recurrir a la guía para el desarrollo de distribuciones GNU/Linux
C4. Complejidad en la aplicación de la guía.
C5. Efectividad de la guía.

C6. Adaptabilidad a los proyectos que desarrollan distribuciones GNU/Linux.
C7. Determinar el nivel de importancia de los procesos de aseguramiento de calidad que ofrece la guía
C8. Influencia de la guía en la obtención de distribuciones GNU/Linux con calidad.
C9. Cumplimiento del objetivo de creación de la guía.

Estos criterios constituyen la base para la confección del cuestionario que se les aplica a los expertos.

3.2.2. Selección del grupo de expertos

Se entiende por experto, a personas que poseen un alto grado de conocimientos sobre el tema de estudio, que sean capaces de ofrecer valoraciones definitivas del problema en cuestión y hacer, además, las sugerencias que considere provechosas para su enriquecimiento, debe estar dispuesto a participar en la evaluación. La correcta elección de estas personas propicia la obtención de resultados exitosos.
Teniendo en cuenta lo anteriormente planteado, se realiza la selección de especialistas bajo las siguientes condiciones: a) Debe ser graduado de nivel superior, b) Debe estar vinculado a la investigación y al desarrollo de distribuciones GNU/Linux, c) Debe poseer conocimiento acerca de la gestión de la calidad de software, d) Debe poseer conocimientos sobre el aseguramiento de la calidad de software, e) Debe contar con un año de experiencia como mínimo en el tema, f) Debe tener noción y habilidades sobre los procesos de calidad del software, g) Práctica con el trabajo relacionado con la gestión de calidad de software en proyectos desarrolladores de distribuciones Linux, y h) Un mismo especialista no tiene que dominar todos los temas

Luego de estar definidos los temas que deben dominar los expertos, se seleccionan un total de 8 especialistas vinculados a la producción y a la docencia.

Experto 1: Sonia Guerrero Lambert
Experto 2: Mailen Proenza Guerra
Experto 3: Yunier Soler Franco
Experto 4: Nilet María Soto López
Experto 5: Jorge Luis Machín Castillo
Experto 6: Daniel Hernández Bahr
Experto 7: Mónica María Albo Castro
Experto8: Sergio Jesús García De La Puente

Para seleccionar cuáles de los expertos participará en el proceso de validación de la propuesta, es necesario determinar las competencias de cada uno de ellos, la cual consiste en el nivel de calificación en la esfera de conocimiento tratada; en algunas ocasiones se tiende a pensar que la competencia está dada por el nivel científico o el cargo que ocupan, sin embargo no siempre esto determina el grado de conocimiento. La competencia de los especialistas se determina por el coeficiente K, que se calcula haciendo uso de la siguiente fórmula matemática:

$$K = \frac{1}{2}(Kc + Ka)$$

Donde:

Kc: es el coeficiente de conocimiento.

Ka: es el coeficiente de argumentación.

Kc, es el coeficiente de conocimiento o información que tiene el posible experto acerca del problema, calculado sobre la valoración del propio experto en una escala del 0 al 10, de esta forma, la evaluación "0" indica que el experto no tiene absolutamente ningún conocimiento de la problemática correspondiente, mientras que la evaluación "10" significa que el experto tiene pleno conocimiento de la misma. El experto debe marcar con una cruz (X) en la casilla que estime pertinente. Posteriormente este valor obtenido se multiplica por 0.1 para obtener el coeficiente en un rango de 0 a 1.

$$Kc = criterio * 0.1$$

La siguiente tabla muestra el nivel de conocimiento de los especialistas seleccionados.

Grado de conocimiento o información del tema tratado											
Expertos	0	1	2	3	4	5	6	7	8	9	10
1										**X**	
2								**X**			
3						**X**					
4							**X**				
5					**X**						
6						**X**					
7							**X**				
8									**X**		

Tabla 4: Autovaloración de conocimiento de los especialistas con respecto al tema.

Expertos	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Kc	0,9	0,7	0,5	0,6	0,4	0,5	0,6	0,8

Tabla 5: Resultado del cálculo del coeficiente de conocimiento

Ka es el coeficiente de argumentación o fundamentación de los criterios del experto. Para calcular este coeficiente, el experto debe clasificar en alto, medio o bajo según su consideración, cuáles fueron sus fuentes para la obtención del conocimiento. Cada nivel de clasificación posee un valor y la suma de los valores marcados por cada criterio será el coeficiente de argumentación (Ka) del candidato a experto, a partir de la tabla patrón definido en el informe titulado Criterio de expertos: Método Delphi extraído del sitio: http://eva.uci.cu/file.php/69/Bibliografia_Especilizada_Tema_4/Metodo_DELPHY.pdf.

$$Ka = \Sigma\ valoresTP$$

Valores TP: son los valores de la tabla patrón que se corresponden con lo marcado por los especialistas.

Tabla patrón mediante la cual los posibles expertos clasifican las fuentes de obtención de sus conocimientos.

FUENTES DE ARGUMENTACIÓN	Grado de influencia de cada una de las fuentes en sus criterios		
	Alto	Medio	Bajo
Análisis teóricos realizados por usted	0.3	0.2	0.1
Su experiencia obtenida	0.5	0.4	0.2
Trabajos de autores nacionales	0.05	0.05	0.05
Trabajos de autores extranjeros	0.05	0.05	0.05
Su propio conocimiento del estado del problema en el extranjero	0.05	0.05	0.05
Su intuición	0.05	0.05	0.05

Tabla 6: Tabla patrón para calcular el coeficiente de argumentación

FUENTES DE ARGUMENTACIÓN	Grado de influencia de cada una de las fuentes en sus criterios		
	Alto	Medio	Bajo
Análisis teóricos realizados por usted	E1,E3, E5, E7,	E4,E8	E2,E6,
Su experiencia obtenida	E1, E2,E3, E5, E7,E8	E2,E6,E4	
Trabajos de autores nacionales	E8	E1,E2,E4	E3, E5, E7,E8
Trabajos de autores extranjeros	E1,E7,E8	E2 ,E5,	E3, E6,E4
Su propio conocimiento del estado del problema en el extranjero	E1,E5,	E7,E8	E2,E3, E6,E4
Su intuición	E5,E4	E2,E3, E6,E7,E8	E1,

Tabla 7: Resultado de la tabla patrón para el coeficiente de argumentación.

Expertos	1	2	3	4	5	6	7	8
Ka	1	0,7	1	1	1	0,7	1	0.9

Tabla 8: Valores del coeficiente de argumentación para cada especialista.

Después de haber calculado el Ka, se evalúa cuáles de los candidatos pueden pasar a ser expertos.

El código de interpretación de tales coeficientes de competencias es:

Si 0.8 < K <1.0, el coeficiente de competencia es alto.

Si 0.5 < K < 0.8, el coeficiente de competencia es medio.

Si K < 0.5 el coeficiente de competencia es bajo.

Es recomendable incluir en el grupo de expertos a aquellos que posean un coeficiente de competencia medio o alto.

Experto	Coeficiente de conocimiento **Kc**	Coeficiente de argumentación **Ka**	Coeficiente de competencia **K**	Grado de influencia del coeficiente de competencia.
1	0,9	1	0,95	Alto
2	0,7	0,7	0,7	Medio
3	0,5	1	0,75	Medio
4	0.6	1	0.8	Alto
5	0,4	1	0,7	Medio
6	0,5	0,7	0,6	Medio
7	0,6	1	0,8	Alto
8	0,8	0,9	0.85	Alto

Tabla 9: Resultados generales para el panel de especialistas.

Teniendo en cuenta que los especialistas están de acuerdo en responder las preguntas del cuestionario y que todos tienen un coeficiente entre medio y alto, se decide que los ochos están aptos para conforman el panel de expertos.

3.3. Fase exploratoria

En esta etapa las principales actividades a realizar son la elaboración y aplicación de los cuestionarios, el método puede tener tantas iteraciones como sean necesarias, pero en este caso, debido al alto ritmo de trabajo de la UCI se hace difícil y extenso el proceso de hacer más de una ronda de preguntas. Para la realización de estas actividades se deben tener presente la longitud y el tipo de pregunta, pues un planteamiento demasiado conciso provoca una excesiva variedad de interpretaciones y uno muy largo requiere administrar demasiados elementos de una sola vez, por lo que la forma en que se realiza el cuestionario es con planteamientos de mediana longitud.

3.3.1. Elaboración del cuestionario

Inicialmente se hace entrega de un resumen de la propuesta que se desea validar a todos los expertos, lo cual permite la comprensión de la solución por parte del panel posibilitando la resolución de la encuesta.
El cuestionario se centra en metas o indicadores que debe cumplir la propuesta para que sea válida. Se definen un conjunto de preguntas abiertas y cerradas que contribuyen a la obtención de evaluaciones y criterios de los especialistas. Las preguntas abiertas contienen un espacio donde se les permite exponer su opinión personal de la propuesta realizada, lo cual es importante pues permite conocer sugerencias que pueden influir en el comportamiento de una determinada cuestión. Por otra parte las preguntas cerradas admiten registrar ciertos aspectos con el objetivo de mostrar gráficamente los resultados obtenidos en el cuestionario. La respuesta para este tipo de preguntas se representa en una escala dividida en 5, donde 1 constituye el nivel más bajo y el 5 el máximo nivel, sin importar si esta división está propuesta de forma cuantitativa (%) o cualitativa (**muy alto**, **alto**, **medio**, **bajo**, **muy bajo**). El cuestionario está disponible en el anexo 2.

Para orientar el proceso de evaluación se definen un conjunto de criterios de evaluación para hacer un balance de equivalencias, con el fin de hacer un análisis general que ofrezca un resultado integrador.

Criterios Cualitativos	Criterio expresado en %	Criterio Numérico
Muy alta	100%	5
Alta	75%	4
Media	50%	3
Baja	25%	2
Muy baja	0%	1

3.4. Fase final

Luego de haber procesado y estudiado la información obtenida de los cuestionarios, se realiza un análisis estadístico de la misma. Es fundamental calcular el porcentaje de las respuestas creadas por los especialistas, pues este valor va a indicar cuán positivas o negativas fueron referidas a cada uno de los atributos sometidos a evaluación.

3.4.1. Cálculo de concordancia entre los expertos

Definido ya el grupo de especialistas y enviado a estos la encuesta, se buscaron sus criterios sobre la guía de buenas prácticas para gestionar la calidad de software en el desarrollo de distribuciones GNU/Linux. Previo a realizar la explotación de los resultados y con el objetivo de aportarle un mayor peso a la validación, se decidió calcular el coeficiente de concordancia Kendall (W) entre los especialistas, para medir en qué magnitud estos están de acuerdo en sus respuestas. Este coeficiente fue posible calcularlo utilizando el software estadístico Statistical Product and Service Solutions (SPSS).

Los valores del W deben oscilar entre 0 y 1 (0 < W < 1), mientras más próximos se encuentren los valores a 1, mayor será el nivel de concordancia entre los criterios de los expertos.

Test Statistics

N	8
Kendall's W(a)	,529
Chi-Square	33,858
df	8
Asymp. Sig.	,000

Figura 11: Coeficiente Kendall obtenido

El resultado obtenido del W se encuentra en el rango de (0.5 < W < 1), por lo que se considera aceptable y el cálculo de Chi Cuadrado es de 33,8 > 0, 05, rechazándose la hipótesis nula, por lo que se concluye que existe una concordancia significativa entre los criterios de los expertos. Evidenciando que la guía es correcta.

3.4.2. Desarrollo práctico y explotación de los resultados

A continuación se muestran los gráficos y porcentajes de los resultados obtenidos por cada uno de los criterios establecidos.

Importancia de la creación y utilización de la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux.

Para la valoración de este objetivo se tiene en cuenta la respuesta de los especialistas a la pregunta 1.El siguiente gráfico muestra el resultado obtenido:

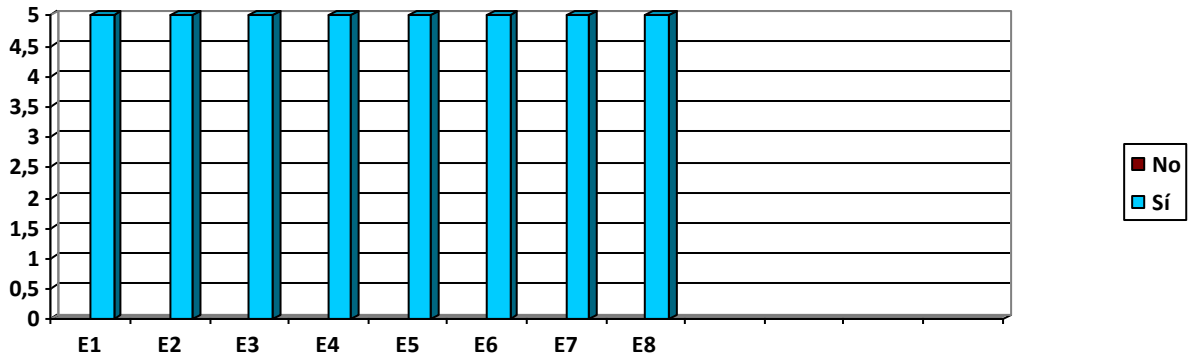


Gráfico 1: Importancia de la guía

Como se puede observar en el gráfico, todos los especialistas coinciden en la importancia de la creación y utilización da la guía de buenas prácticas para ayudar a los desarrolladores de distribuciones GNU/Linux a elevar la calidad de las mismas, por lo que se obtiene un 100% de aceptación, opinando lo siguiente:

- Una guía que ayude a definir y organizar el proceso de gestión de la calidad de software es muy importante para garantizar el éxito de cualquier distribución de GNU/Linux.

- Es importante debido a que el proceso de desarrollo que se lleva a cabo es diferente a lo normalmente conocido, por lo que se hace necesario definir los procesos, entre ellos la gestión total de la calidad, garantizando que el proceso pueda continuarse por otros integrantes.
- Garantiza que los procesos de desarrollo se realicen de forma ordenada y en pos de alcanzar los objetivos propuestos en el menor tiempo posible.

- La guía cuenta con varios procesos que van abarcando todas las partes a tener en cuenta para el desarrollo de un producto, garantizando así, una mejora en el aseguramiento de la calidad. Se puede decir que es una buena práctica a seguir en los futuros productos desarrollados en la universidad

- Es un tipo de software que no posee una metodología definida y aceptada que incluya actividades de aseguramiento de la calidad, como lo hace por ejemplo RUP; por tanto se necesita un esfuerzo intencionado en cómo se garantizará la calidad de estos sistemas; ya que un software que no demuestre calidad y satisfaga las expectativas de los usuarios, fracasará sin importar la utilidad que tenga.

- Cualquier obra ingenieril requiere de manera obligatoria se tenga en cuenta la gestión de la calidad, el desarrollo de software no es la excepción de las ingenierías, por tanto ello requiere un control de la calidad minucioso y serio, más aún en aplicaciones que tienen un marcado impacto social como lo son los sistemas operativos, puesto que son utilizados por más personas que cualquier otro sistema dado por sus características de ser sistema base para la ejecución de las aplicaciones. Considero atinada e inteligente la decisión de la confección de una guía para orientar la gestión de la calidad de software en el desarrollo de distribuciones GNU/Linux.

Alcance de la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux.

El alcance de la guía se responde en la pregunta 2 donde los especialistas podían seleccionar en un rango del 1 al 5, 1 representaba que la guía no era abarcadora y 5 que lo era en su totalidad, el resultado se muestra en la gráfica siguiente:

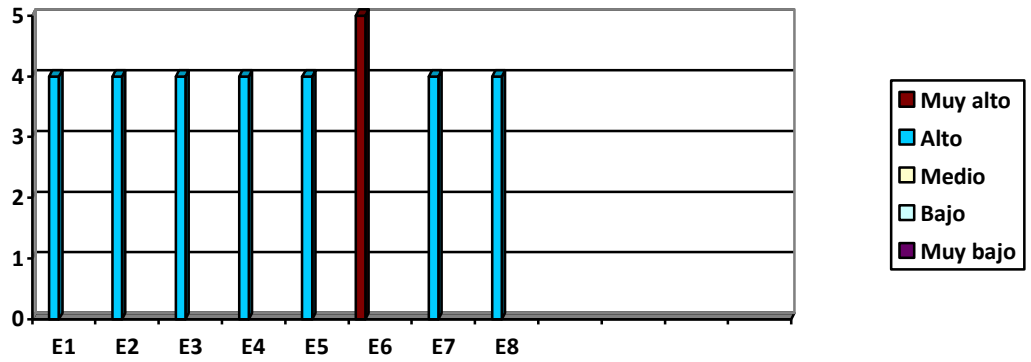


Gráfico 2: Alcance de la guía

Como se puede observar 7 especialistas (E1, E2, E3, E4, E5, E7, E8) coinciden en que el alcance de la guía para gestionar la calidad de software de distribuciones GNU/Linux es alto, y uno (E6) coinciden en que es muy alto, lo que constituye un 78,1% de aceptación de que la guía es lo suficientemente abarcadora, emitiéndose criterios como:

- Es bastante abarcadora debido a que incluye la verificación, la validación y el control de la calidad en los procesos que son necesarios.

- La guía presentada es correcta y entendible y lo que pone tiene sentido, pero mi conocimiento en gestión de la calidad no me permite estar segura de que se ha abarcado todo lo que podía ser realizado.

Necesidad de recurrir a la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux

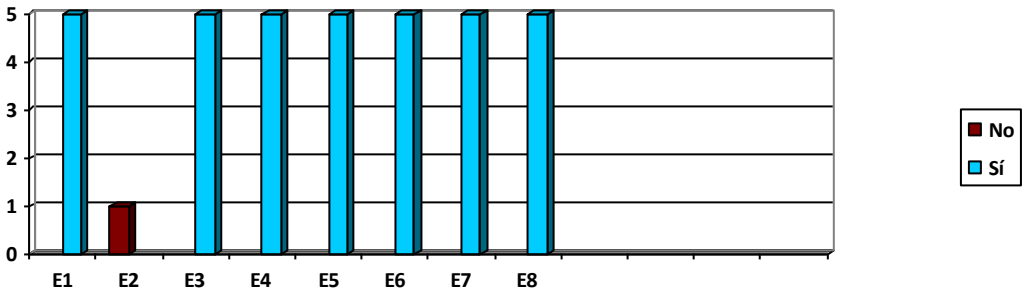


Gráfico 3: Necesidad de uso de la guía

Las pregunta 3 responde el criterio que plantea la necesidad del utilizar la propuesta, el cual está encaminado a prever si se hace necesario aplicar o no la guía. Las posibles respuestas eran: _Si, _No. El gráfico anterior muestra el resultado obtenido.

La gráfica muestra las respuestas de los especialistas donde se puede apreciar que un 87.5 % está de acuerdo con que se debe utilizar la guía. Se exponen los siguientes criterios:

- Actualmente han sido liberadas varias versiones de Nova con problemas en algunas de las aplicaciones que incluye, algo que se hubiese podido evitar contando con una guía de gestión de calidad.
- La calidad de software es la que le dice al resto de los desarrolladores que lo que están realizando va por el camino correcto. No contar con una guía que establezca cómo hacer esto hace más difícil poder validar el trabajo que se está realizando.
- Hasta el momento, esto se realizaba de forma difusa y sin criterios que guiaran el proceso.
- Es un producto novedoso en Cuba, que tiene que hacerse un lugar en el mundo donde ya existen cientos de distribuciones linux, por lo que las comparaciones y las críticas de los usuarios son fuertes hasta que estos productos no demuestren con su calidad su superioridad en un mercado de usuarios de software libre.
- Esta guía se debía haber materializado mucho antes, dado que el desarrollo de distribuciones GNU/Linux en nuestra institución es bastante joven, la realidad es que alrededor de seis a siete años se ha estado desarrollando un sistema operativo propio con metas muy ambiciosas que obligan a un desarrollo mucho más serio, minucioso y responsable, donde no se dejen cabos sueltos y se tengan en cuenta los estándares y necesidades de los clientes; esto y más le da cumplimiento un correcto aseguramiento de la calidad.

Complejidad en la aplicación de la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux

La complejidad de poner en práctica la propuesta está enfocada en prever la dificultad de empleo o entendimiento de la guía; este criterio se responde en las pregunta 4 representada en el siguiente gráfico.

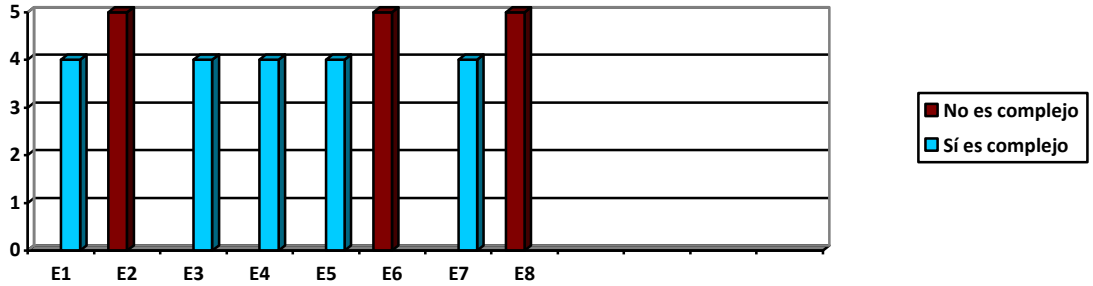


Gráfico 4: Complejidad de uso

El 37.5% coincide en que la aplicación de la guía de gestión de la calidad de software no representa ninguna complejidad. Los especialistas exponen los siguientes criterios:

- La mayor dificultad podría ser adaptar a los equipos de desarrollo a utilizar esta guía.
- Puede ser difícil de llevar a la práctica teniendo en cuenta lo ágil que se desarrolla el proyecto.
- Puede existir alguna dificultad en el proceso de aplicación de la guía porque siempre que se intenta aplicar algún nuevo proceso al desarrollo, es necesario un período inicial para que las personas involucradas se adapten a los cambios.
- Si no se realiza una intencionada capacitación de las personas que deben cumplir el rol de aseguramiento de la calidad y hacer cumplir esta guía.
- La intensión y voluntad de aplicar la guía existe, además hoy más que nunca nuestro sistema operativo en desarrollo Nova, necesita de una herramienta así para estar a la altura de sus tiempos y los retos que le aguarda los nuevos compromisos productivos de la universidad y la migración a software libre que lleva a cabo nuestro país.

Efectividad de la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux

La pregunta cinco se centra en comprobar la efectividad de la utilización de la guía de buenas prácticas para gestionar la calidad de software en el desarrollo de distribuciones GNU/Linux, los especialistas deben seleccionar su respuesta lo que va a pemitir afirmar o negar el potencial de la solución, en el siguiente gráfico se puede observar los resultados logrados:

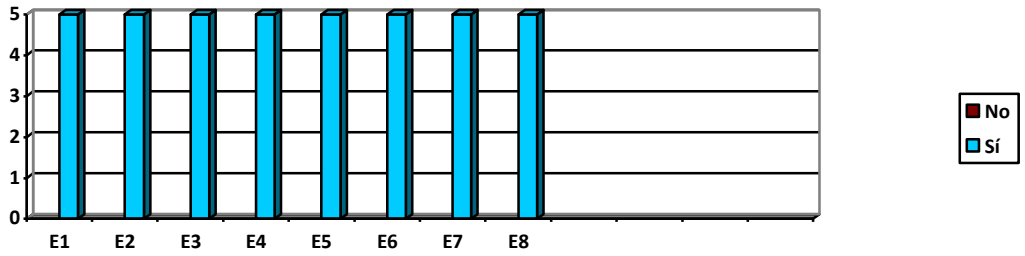


Gráfico 5: Efectividad de la guía

El resultado muestra la concordancia de 8 especialista de 8 encuestados donde los resultados positivos marcan un 100% exponiéndose los siguientes criterios:

- Abarca en su mayoría a los procesos y subprocesos de construcción de Nova e incluye la preparación del personal que es muy importante.
- Es un documento que puede servir de guía para todo el que lo necesite, estandarizando el proceso.
- Si se aplica fielmente puede ser muy efectiva.
- Pudiera ser extensiva al desarrollo de las aplicaciones basadas en la línea de comandos que se ejecutan en el sistema operativo y las personalizaciones de éstas para los entornos de escritorios, donde en ambos casos deben respetarse estándares y normas de usabilidad ya definidas por sus creadores originales. Pero viendo desde la perspectiva de en lo que hoy está inmerso el desarrollo de Nova, creo que es bien efectiva ésta guía.

Adaptabilidad a los proyectos que desarrollan distribuciones GNU/Linux.

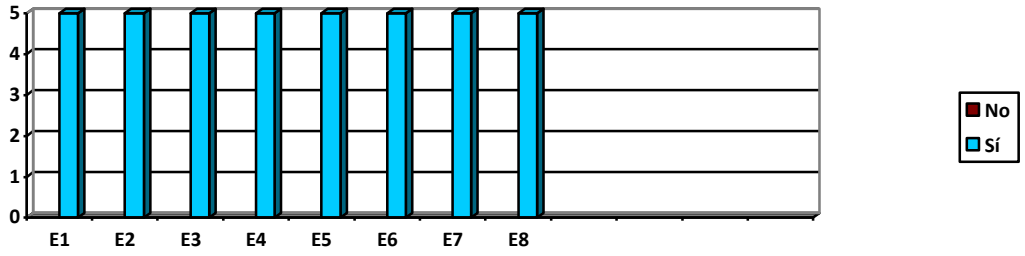


Gráfico 6: Adaptabilidad a otros proyectos productivos

La adaptabilidad de la guía a los proyectos productivos se responde en la pregunta seis, donde se desea conocer el grado de adaptación que se pronostica para la propuesta de solución, observándose un 100% de concordancia por parte de los especialistas en la adaptabilidad de la Guía de indicadores, representado en la gráfica que aparece anteriormente. Se valora que los procesos de construcción de las distribuciones de GNU/Linux son bastante similares por lo tanto esta guía puede ser utilizada en cualquiera de estos.

Nivel de importancia de los procesos de aseguramiento de la calidad que ofrece la guía.

La pregunta siete responde al criterio, comprobándose el grado de efectividad de los diferentes procesos de calidad que ofrece la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux. Los especialistas deben seleccionar su respuesta en un rango del 1 al 5, teniendo en cuenta de que 1 es el nivel más bajo y 5 el máximo nivel. En el siguiente gráfico se puede observar los resultados logrados.

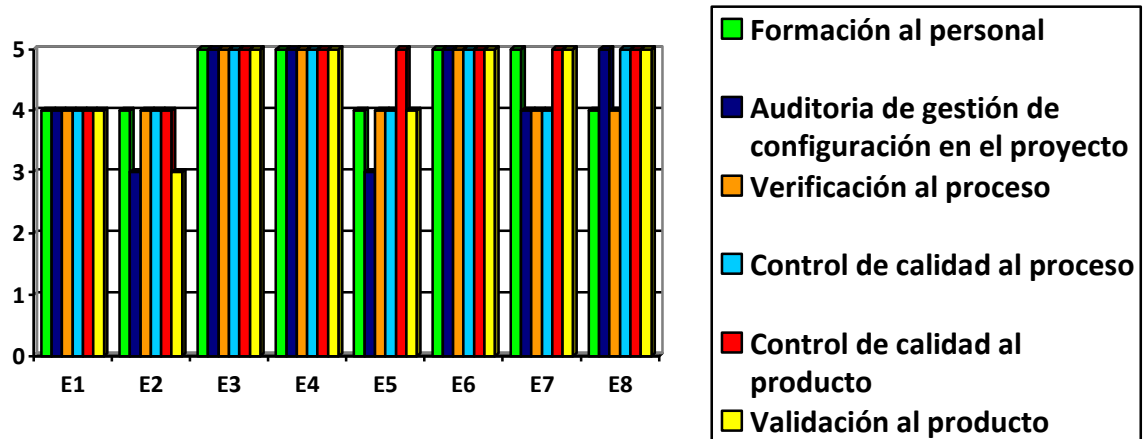


Gráfico 7: Nivel de importancia de los procesos ACS

Influencia de la guía en la obtención de distribuciones GNU/Linux con calidad.

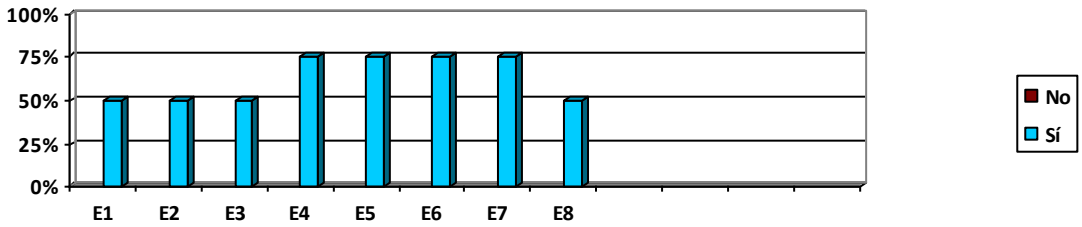


Gráfico 8: Influencia de la guía

La pregunta 8 del cuestionario está orientada a conocer en qué medida la guía contribuirá a que se desarrollen distribuciones GNU/Linux con la calidad requerida.

Cumplimiento del objetivo de creación de la guía de gestión de la calidad de software para el desarrollo de distribuciones GNU/Linux

Este objetivo describe si realmente la solución resuelve la problemática de elevar la calidad de las distribuciones GNU/Linux, aportando métodos de calidad de software enfocados en el proceso de desarrollo de este tipo de producto. Se encuestan a los especialistas y estos son los resultados:

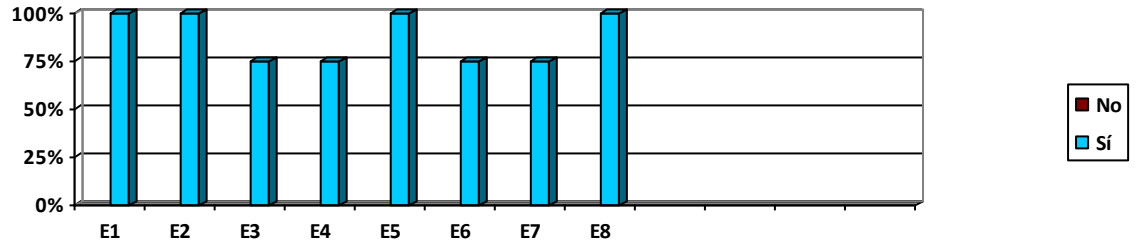


Gráfico 9: Cumplimiento del objetivo de la guía

3.4.3. Resultado Final

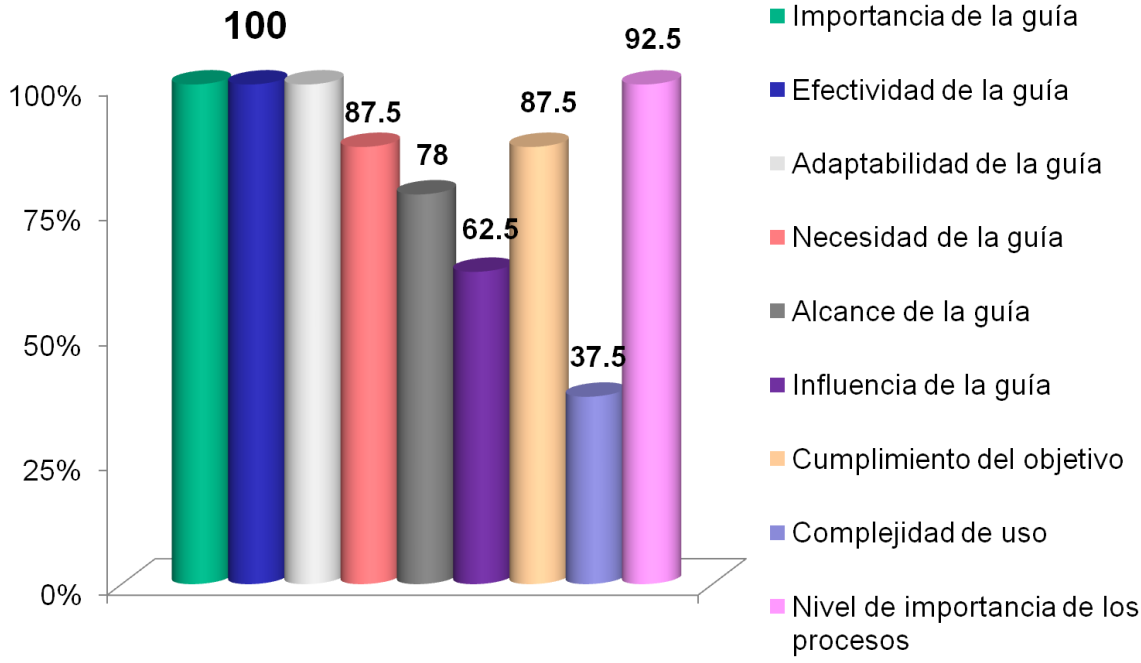


Gráfico 10. Porciento de cumplimiento por criterio.

El porcentaje de respuestas de los especialistas a cada uno de los objetivos propuestos fue positivo evaluando así la confección de la guía, los resultados generales se pueden observar en el siguiente gráfico.

La solución propuesta es calificada por los especialistas como un procedimiento adecuado que tiene en cuenta aspectos estudiados en importantes fuentes bibliográficas y que proporciona un cierto orden a la gestión de la calidad de software para el desarrollo de las distribuciones GNU/Linux.

3.5. Evaluación por el método Experimental

El experimento es una actividad para obtener conocimiento sobre el objeto de estudio, por lo que presenta las características siguientes: aísla el fenómeno y las propiedades que se estudian, provoca la aparición del fenómeno que se quiere estudiar en condiciones controladas, modifica las condiciones en que ocurre el fenómeno y se realiza a través de métodos empíricos.[27]

Para aplicar el método experimental se va a tomar como muestra al proyecto Nova, donde se pondrá en práctica parte de la solución propuesta.

Se establecieron una serie de indicadores para observar su comportamiento antes y después de poner en práctica el proceso ACS Verificación al subproceso prueba, a los mismos se les va a otorgar una evaluación entre 0-5, donde 0 significa que no se tenían en cuenta y 5 el máximo valor de la escala.

Indicadores

1. Verificar que esté creado el ambiente y las condiciones para llevar a cabo las pruebas.
2. Verificar que las pruebas se realicen en cada iteración del desarrollo software.
3. Comprobar que los defectos se corrijan tan pronto como sean encontrados.
4. Comprobar que para cada actividad de desarrollo haya una actividad correspondiente de pruebas.
5. Comprobar que el producto software respeta los criterios o estándares aplicados en el proyecto y cumpla con los factores de calidad requeridos.
6. Verificar que se analicen las lecciones aprendidas para futuras liberaciones del proyecto y la mejora de la madurez de prueba.
7. Verificar la correspondencia de las pruebas con el tipo de producto.
8. Comprobar la adherencia al proceso de pruebas definido.

La siguiente gráfica muestra el comportamiento de los mismos antes y después de poner en práctica el proceso de ACS donde se observa que anteriormente los indicadores representados con color rojo, tienen un nivel bajo en la escala, esto significa que no se verifican los mismos durante el proceso de pruebas en el proyecto Nova.

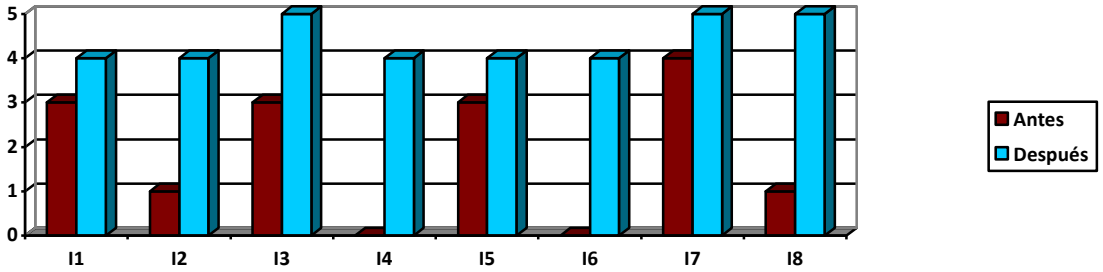


Gráfico 11. Comportamiento de los indicadores.

Después de haber aplicado el proceso de ACS se llegó a la conclusión de que los indicadores antes mencionados aumentaron su nivel considerablemente debido a que ya existe un proceso de ACS que verifica que se tengan en cuenta durante el proceso de pruebas.

Conclusiones

Deseando darle solución con la aplicación de la guía a los problemas presentes en los proyectos desarrolladores de distribuciones GNU/Linux en cuanto a la gestión de la calidad del proceso de desarrollo de software, concluye este trabajo investigativo, en el cual se llegan a las siguientes conclusiones:

- La propuesta presentada brinda un conjunto de procesos de ACS que se deben tener en cuenta durante todo el ciclo de vida del proceso de desarrollo de distribuciones GNU/Linux.

- Los procesos de ACS propuestos están correctamente ubicados dentro del proceso de desarrollo de distribuciones GNU/Linux, están bien definidos y son poco complejos.

- La guía facilita conceptos y definiciones en correspondencia a las tareas que se deben desarrollar, propone normas y estándares, así de igual forma modelos y metodologías.

- La evaluación de la guía arrojó excelentes resultados, permitiendo afirmar que está bien estructurada, es entendible, eficiente y abarcadora.

- La guía tiene como máxima intención hacer el trabajo más fácil para el equipo de desarrollo si se emplea correctamente y se tienen en cuenta las recomendaciones que brinda.

- Aunque está enfocada a la gestión de la calidad del proceso, garantiza que el producto final cumpla con las necesidades del cliente.

Recomendaciones

Debido a la importancia que tiene tener en cuenta el aseguramiento de la calidad desde el inicio del desarrollo del software, se ha considerado realizar algunas recomendaciones con el objetivo de mejorar este aspecto:

- Se recomienda aplicar la propuesta realizada en los proyectos que se dediquen al desarrollo de distribuciones GNU/Linux.
- Dar seguimiento al desarrollo de la guía enfocándola al aseguramiento de calidad del software en el producto.
- Cumplir con lo planteado durante todo el ciclo de desarrollo de una distribución GNU/Linux.
- Realizar estudios sobre los estándares existentes a nivel internacional para el aseguramiento de la calidad del proceso de desarrollo de software.

Referencias Bibliográficas

1. **Lovelle, Juan Manuel Cueva.** Calidad del Software. España : s.n., 1999.
2. **Colectivo de autores, Fernández del Monte,Yusleydi , Albo, Mónica M.** Segunda Auditoría Interna proyecto Nova. Ciudad Habana : s.n., 2010.
3. **Pressman, Roger S.** Ingenieria de Software Un Enfoque Práctico. 5ta Edición. s.l. : McGraw Hill, 2002.
4. **Pillou, Jean.** ISO 8404 Calidad. [En línea] 2008. [Citado el: 3 de noviembre de 2010.] http://es.kioskea.net/contents/qualite/qualite-introduction.php3...
5. **ASQ.** American Society for Quality Control (ASQC). [En línea] 2010. [Citado el: 19 de febrero de 2011.] http://www.asq.org/.
6. **Pressman, Roger S.** Ingenieria de Software Un Enfoque Práctico. 5ta Edición. s.l. : McGraw Hil, 2002.
7. **ISO/IEC.** ISO - International Organization for Standardization. [En línea] 2010. [Citado el: 20 de octubre de 2010.] http://www.iso.org/iso/home.htm.
8. **Martínez, Juan J.** Control de Calidad en Software Libre. España : s.n., noviembre 2008.
9. **Martínez, Juan J.** Control de Calidad en Software Libre. . España : s.n., noviembre 2008.
10. Que significa CMMI: Mejora continuaa y cambio. [En línea] [Citado el: 8 de diciembre de 2010.] http://asprotech.blogspot.com/2010/11/mejora-continua-y-cambio.html.
11. **ISO/IEC.** ISO - International Organization for Standardization. [En línea] [Citado el: 16 de octubre de 2010.] http://www.iso.org/iso/home.html.
12. Determinación de la Capacidad de Mejora del Proceso de Software. [En línea] 2005. [Citado el: 9 de enero de 2011.] http://www.sispyme.com/SISPYME/noticias/noticia57/noticia57.html.
13. **Dergarabedian, César.** Normas de calidad en software, una llave que abre mercados – iProfesional.com. [En línea] 7 de marzo de 2006. [Citado el: 4 de diciembre de 2010.] http://tecnologia.iprofesional.com/notas/25157-Normas-de-calidad-en-software-una-llave-que-abre-mercados.
14. **Singh, Raghu.** Una introducción a la Norma Internacional IEC/ISO 12207 Ciclo de vida los precesos de software. Washington, DC, USA, 1999.
15. Calidad en Ingeniería del Software. [En línea] 2002. [Citado el: 20 de enero de 2011]. http://dmi.uib.es/~bbuades/calidad/sld001.htm
16. **Colectivo de autores, Vidal,Dalvis Matos, Giro,Edilberto Alcolea.** Guía para la obtención de la reemplazabilidad y la conformidad de la portabilidad en Generadores de Reportes Dinámicos. Habana : s.n., 2010.

17. **IEEE.** “ANSI/IEEE Std 1028:1988 IEEE standard for software reviews and audits”. [En línea] 1989 de junio. [Citado el: 15 de noviembre de 2010.] http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel1/2366/1231/00029123.pdf.

18. **ISO.** “ISO 19011:2002 Auditorias de gestión de calidad y/o ambiental”. [En línea] octubre de 2002. [Citado el: 7 de noviembre de 2010.] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?.
19. **IEEE.** “IEEE Standard Glossary of Software Engineering Terminology”. [En línea] septiembre de 1990. [Citado el: 11 de noviembre de 2010.] http://ieeexplore.ieee.org/xpl/freeabs_all.jsp.
20. **Informáticas, Universidad de las Ciencias.** Introducción al proceso de desarrollo de software. Habana : s.n., 2008.
21. **Pressman, Roger S.** Ingeniería del Software (Un enfoque práctico),. s.l. : MacGrawHill, 2006.
22. **Oduardo, Irina González.** Arquitectura del sistema de clonación y distribución de imágenes de sistemas operativos. Habana : s.n., 2009.
23. Modelos de desarrollo de MDCs. [En línea] 2008. [Citado el: 28 de noviembre de 2010.] http://modelosdesarrollomdc.blogspot.com/2008/10/propuesta-de-una-metodologia-de.html.
24. **Daniel.** Gestión de Proyectos de Software. [En línea] 25 de abril de 2008. [Citado el: 27 de febrero de 2011.] http://danielvn7.wordpress.com/2008/04/25/gestion-de-proyectos-de-software/.
25. **Colectivo de autores, Saez,Daniel, Peris,Martin, Anes,David**. Migración al software libre. Guía de buenas prácticas. Valencia : s.n., 13 de noviembre del 2007.

26. **Cabezas Trujillo, Daniel.** Aplicaciones del método Delphi. Bogotá : s.n., 15 de marzo del 2004.
27. **Colectivo de autores, León,Rolando Alfredo Hernández, González,Sayda Coello**.El proceso de investigación científica. Ciudad Habana : Editorial Universitaria del Ministerio de Educación Superior, 2011, 2011. ISBN 978-959-16-1307-3.

Bibliografía

1. **Colectivo de autores, Saez,Daniel, Peris,Martin, Anes,David**. Migración al software libre. Guía de buenas prácticas. Valencia : s.n., 13 de noviembre del 2007.
2. Control de Calidad en Software Libre. Valencian : III Congrés de Programari Lliure - Comunitat Valenciana, Noviembre 2008.
3. **Hernandez,Andres**. ¿Qué es software libre? [En línea] 24 de Mayo de 2009 . [Citado el: 12 de octubre de 2010.] http: http://www.apesol.org/softwarelibre.php.
4. **Colectivo de autores, Ginestà,Marc Gibert, Matías,Martín Hernández, González, Álvaro Peña**. Ingeniería del Software en entornos de SL. Barcelona : Fundació per a la Universitat Oberta de Catalunya, marzo 2005. 39-43, 08035.
5. **Beekmans,Gerard**. Linux From Scratch. s.l. : Proyecto LFS-ES., 1999–2007. 20070830 .
6. **Cabezas,Raúl Trujillo**. Aplicaciones del método Delphi. Bogotá : s.n., 15 de marzo del 2004.
7. **Documentación, Instituto Nacional de Tecnologías de la**. GUÍA DE VALIDACIÓN Y VERIFICACIÓN. España : Ministerio de industria, turismo y comercio, Noviembre 2009.
8. **Colectivo de autores, McCall,J.A, Richards,P.K, Walters,G.F**. “Factors in Software Quality”, RADC. s.l. : RAD CTR-77-369.
9. **Colectivo de autores, Dolado,J.J, Fernández,L.** Medición para la Gestión en la Ingeniería del Software. s.l. : Ra-ma, 2000.
10. **Colectivo de autores, León,Rolando Alfredo Hernández, González,Sayda Coello**.El proceso de investigación científica. Ciudad Habana : Editorial Universitaria del Ministerio de Educación Superior, 2011, 2011. ISBN 978-959-16-1307-3.
11. **Colectivo de autores, Juristo,Natalia, Moreno,Ana M, Vegas,Sira**. TÉCNICAS DE EVALUACIÓN DE SOFTWARE. Ciudad Habana : s.n., 17 de octubre de 2006.
12. TÉCNICAS DE EVALUACIÓN DE SOFTWARE. 17 de octubre de 2006.
13. **Colectivo de autores, Navarro,José Manuel, Garzás,Javier.** Revista Española de Innovación, Calidad e Ingeniería del Software (REICIS). Barelona, España : Luis Fernández-Sanz, Juan J. Cuadrado-Gallego, 1 de abril del 2010. Vol .6, No 1, 2010.
14. **Durán,Miguel Udaondo**. Gestión de calidad. Madrid, España : Diaz de Santos, SA, 1992. ISBN: 84-7978-013-4.
15. **Galloway,Dianne**. Mapping Work Processes. Wisconsin-USA : Ediciones Gestibn 2000. S.A., 1994. 84-8088-292- I.
16. **Godoy,Rudy**. Aseguramiento de calidad en software libre. Arequipa- Perú : XIV CONEIS, 2005.
17. **Lovelle, Juan Manuel Cueva**. Calidad del Software. España : s.n., 21 de Octubre de 1999.

18. **Sanchez, María A. Mendoza**. Metodologías De Desarrollo De Software. Perú : s.n., 7 Junio del 2004.
19. **Cabezas Trujillo,Daniel.** Aplicaciones del método Delphi. Bogotá : s.n., 15 de marzo del 2004.
20. **Informáticas,Universidad de las Ciencias**. Introducción al proceso de desarrollo de software. Habana : s.n., 2008.
21. **Pressman, Roger S**. Ingenieria de Software Un Enfoque Práctico. 5ta Edición. s.l. : McGraw Hil, 2002.
22. **Martínez, Juan J**. Control de Calidad en Software Libre. . España : s.n., noviembre 2008.
23. **Pressman,Roger S**. Ingenieria de Software Un Enfoque Práctico. 5ta Edición. s.l. : McGraw Hil, 2002.
24. Determinación de la Capacidad de Mejora del Proceso de Software. [En línea] 2005. [Citado el: 9 de enero de 2011.
25. **Singh, Raghu**. Una introducción a la Norma Internacional IEC/ISO 12207 Ciclo de vida los precesos de software. Washington, DC, USA, 1999.
26. **Martínez, Juan J**. Control de Calidad en Software Libre. España : s.n., noviembre 2008.
27. **Monferrer A., Raúl.** Especificación de Requisitos Software según el estándar IEEE 830. Castellón de la Plana, España: Universitat Jaume I, 2000.