Universidad de las Ciencias Informáticas
Facultad 6

Título: Sistema de Métricas para Requisitos Funcionales.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas.

Autores

ANNIA DURÁN VIENES

YUDIEL E. HURTADO GONZÁLEZ

Tutor

MSC. KARINA PÉREZ TERUEL

ING. ASNIER ENRIQUE GÓNGORA RODRÍGUEZ

Ciudad de La Habana, Junio 2007

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Autores: Annia Durán Vienes.
Yudiel E. Hurtado González.

Tutores: Msc. Karina Pérez Teruel.
Ing. Asnier Enrique Góngora Rodríguez.

_____ Firma del primer Autor

_____ Firma del primer Tutor

_____ Firma del segundo Autor

_____ Firma del segundo Tutor

Datos de Contacto

Ing. Asnier Góngora Rodríguez: Ingeniero en Ciencias Informáticas graduado de la Universidad de las Ciencias Informáticas (2007). Profesor de la Universidad de las Ciencias Informáticas, en la Disciplina de Ingeniería y Gestión de Software desde el año 2007. Cuenta con 9 meses de trabajo en la Educación Superior. Cursa la Maestría de Calidad de Software. Se desempeña laboralmente como Asegurador de la calidad del proyecto Mapeo Cerebral Humano Cubano de la Facultad 6.

Correo: agongora@uci.cu

"La calidad nunca es un accidente; siempre es el resultado de un esfuerzo de la inteligencia".

John Ruskin

Dedicatoria

A mis padres por darme mucho más que la vida

Annia

A mi hermano Frank Anthony

Yudiel

Agradecimientos

A mi flaquito Yudiel, por ser más que mi compañero de tesis, por ser amigo y aguantarme perretas y peleas. Por todos los consejos y siempre conmigo, gracias. Te quiero mucho.

A mis padres, Ricardo e Isabel Cristina, por apoyarme incondicionalmente. Por hacerme crecer ante las dificultades y creer que todo puede ser y hacerse mejor. Por ser la mejor guía, gracias. Los amo infinitamente.

A mi niña Ada Isabel y mi hermano Jackson, por ser la sal de mi vida.

A mis abuelos Jesús, Mirta, Angel Luis y Naida, por todo su amor.

A mis hermanas Lornita y Yaya, por estar a mi lado. Por ser personitas tan especiales y darme la inocencia o la locura que a veces me falta, saben que las adoro a las dos.

A Mayda y Agustín por quererme como a una hija y a mi abuelita Adelaida por su cariño. Siempre los llevo en mi corazón

A mis tutores Asnier y Karina por apoyarme siempre y luchar por la tesis tanto como yo.

A Magdelis, Marbelis, Liana, Yandy y Vladimir por marcar la diferencia en mi camino.

A Yanet, Yaimí, Yusmary, Martica, Anabel, Ana Isabel y Milenys, por apoyarme en los momentos difíciles.

A Anays, por su preocupación y ayuda incondicional.

A todos aquellos que se preocuparon, me apoyaron y ayudaron durante todos estos años, muchas gracias, todo hubiese sido muy diferente sin ustedes.

Annia

Agradecimientos

A mi compañera de tesis Annia, por confiar en mí y abrirme las puertas cuando todas se cerraban.

A mis padres Ovidio y Oilda que siempre me han apoyado en todos mis años de estudio.

A mi familia de Santa Cruz, tía Daisy, Dainel, Daisel y Kevin por atenderme tan bien todo este tiempo.

A mi abuela Yolanda por todo su cariño y preocupación.

A Guido y Yunior por no dejarme morir y estar a mi lado cuando más lo necesitaba.

A mis amigos Leoder (el enano), a súper Yordy, la súper Yury, la súper Tita, Yanelis (Mami de la compota), Yiselis (la Yise) David, Kenia (la Keku) por tan buenos momentos vividos.

A las que se graduaron Yaritza, Lisdanis y Dayani.

A las chicas del apartamento 130106 (la Yanella, la Roxana, La Leydis y la Ailias), por brindarme su casa durante todo este tiempo.

A Tahimi la mejor tía del comedor, por toda su ayuda.

A mis tutores por guiarme tan bien durante esta investigación.

A Anays Más y Onay Mercader por su colaboración incondicional.

A todos los que de alguna u otra forma estuvieron a mi lado en estos años de estudios compartiendo alegrías y tristezas, los que aun están aquí y continúan, a los que ya no están, a todos los que colaboraron para que este trabajo cumpliese su objetivo, y que no se han mencionado.

A todos muchas gracias.

Yudiel Emilio

Resumen

La experiencia adquirida por los desarrolladores de *software* tras años de trabajo ha demostrado que en sus inicios los proyectos se ven afectados por la pérdida o modificación de requisitos; incidiendo notablemente en la calidad del producto que se construye. Las métricas de *software* permiten la evaluación objetiva del desarrollo del mismo, evitando errores mayores en fases posteriores que ocasionan muchas veces el retraso e incluso el abandono del proyecto.

En la presente investigación se expone la situación actual de la disciplina de Ingeniería de Requisitos en la UCI mediante el procesamiento de los resultados obtenidos de la aplicación de encuestas y entrevistas. Se presentan los resultados obtenidos del estudio realizado a los Modelos de Calidad, explicando además la metodología mediante la cual de confeccionó el Sistema de Métricas y se propone a su vez el mismo. Se valida el resultado de la investigación mediante el criterio de expertos en cuanto a importancia, eficiencia y aplicabilidad y se explica cómo evaluar la calidad de los proyectos productivos empleando el Sistema de Métricas propuesto.

Palabras claves: Ingeniería de Requisitos, métricas, métricas de software, requisitos funcionales.

Índice

Índice de Figuras

Índice de tablas

Introducción

La industria del *software* en el mundo se ha desarrollado vertiginosamente en estos últimos años. Cada día se obtienen nuevas versiones de productos ya existentes o salen al mercado nuevas propuestas. Sin embargo aún éstos no cumplen con las expectativas de los clientes, no teniendo para ellos la calidad deseada.

La experiencia adquirida por los desarrolladores de *software* tras años de trabajo ha demostrado que en sus inicios los proyectos se ven afectados por la pérdida o modificación de requisitos así como la redacción ambigua de los mismos. En muchos casos se realizan entrevistas a clientes que no son los más indicados por no tener conocimiento amplio del negocio, razón por la cual no proveen la información requerida, además de hacer peticiones inadecuadas. Estos factores afectan notablemente la calidad del *software* que se construye que no es más que la "concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos (…) y con los requerimientos implícitos no establecidos formalmente, que desea el usuario". **(PRESSMAN 2005)**

Pero, ¿cómo saber si un *software* tiene calidad? La medición es fundamental en el ciclo de vida del *software* pues permite la evaluación objetiva del desarrollo del mismo. Además posibilita la identificación de tendencias, que pueden ser positivas o negativas, así como la realización de mejores estimaciones de tiempo. Para medir la calidad de *software* existen diversos modelos, como por ejemplo: Modelo de Bohem, Modelo FCM (Factors/Criteria/Metrics), Modelo CMM (Capability Maturity Model) y SPICE (*Software Process Improvement and Capability determination*). Algunos de estos modelos, como el FCM incluye también métricas que evalúan diferentes atributos de calidad del producto en el nivel de diseño o código.

Las métricas de *software* están orientadas a diferentes aspectos que determinan la calidad del producto, como por ejemplo el tamaño y la función. Como muestras de éstas se tienen: la medición de errores por KLDC (miles de líneas de código), defectos por KDLC y LDC (líneas de código) por persona-mes. En el segundo caso se encuentran la medida punto de función y su ampliación que incluye elementos para evaluar aplicaciones donde la complejidad de algoritmos es alta. Además se aplican métricas a los requisitos que los valoran y modelan verificando aspectos como la especificidad, corrección, capacidad de verificación, concisión, entre otros, por ejemplo las Métricas de Bang.

La aplicación de métricas permite la confección de un historial con los resultados obtenidos y su reutilización en proyectos venideros viabilizando el proceso de desarrollo. También se emplean para señalar los problemas existentes en determinadas áreas de forma que puedan solucionarse, mejorando así la calidad del proceso del *software*. Además conlleva a una toma de decisiones mucho más fundamentada. Cada equipo de proyecto puede recopilar diferentes medidas y convertirlas en métricas para emplearlas en el proyecto que esté desarrollando y además transmitirlas a otros desarrolladores.

La Universidad de las Ciencias Informáticas (UCI) surge en el año 2001 para potenciar el desarrollo de *software* en el país. Su principal objetivo es llevar el proceso de informatización a cada rincón del territorio nacional. Para ello los estudiantes y profesores se vinculan a proyectos productivos que desarrolla el centro dirigidos al proceso antes mencionado, para empresas nacionales y exportación que son comercializados por Albet, empresa creada en el 2005 con este fin. Actualmente se produce para más de 20 empresas nacionales y para los proyectos del ALBA. Cuenta con una Dirección de Calidad integrada por un Laboratorio de Certificación, Grupo de Auditoría y Control, Grupo de Revisiones, Aseguramiento de la Calidad y Métricas, siguiendo los modelos de calidad CMMI e ISO 15504.

Aún con éstas condiciones, debido a la juventud del centro se presentan problemas con la producción. El proceso de levantamiento de requisitos se desarrolla de manera poco rigurosa. Las entrevistas con los clientes no son regulares en la mayoría de los casos y los requisitos sufren modificaciones frecuentes, aspecto que influye negativamente en la entrega en tiempo del producto. No existe ningún archivo donde puedan consultarse datos históricos de proyectos anteriores que faciliten la toma de decisiones o la previsión de fallos futuros. De esta forma se cometen errores que más tarde tributan a la inconformidad del usuario final. Para vencer estas dificultades puede aplicarse un sistema de métricas que determine desde el inicio del ciclo de vida del *software* la calidad de los requisitos funcionales. A medida que aumente la calidad del *software* se constatará un decremento en el número de defectos y la cantidad de trabajo a realizar. Factores que influirán directamente sobre el coste global del proyecto que lógicamente disminuirá.

La situación antes planteada conlleva al siguiente **problema de la investigación**:

¿Cómo evaluar la calidad de los requisitos funcionales de *software* en la UCI?

Como **objetivo de la investigación** se plantea:

Proponer un Sistema de Métricas para Requisitos Funcionales de *Software*.

A partir de este objetivo se derivan los siguientes **objetivos específicos**:

- Identificar métricas para requisitos funcionales.
- Definir métricas para requisitos funcionales.
- Validar el sistema de métricas propuesto.

Se determina como **objeto de estudio** la Ingeniería de requisitos en el desarrollo de un proyecto y como **campo de acción** las métricas para requisitos funcionales.

Para dar cumplimiento a estos objetivos se acometieron las siguientes tareas:

- Análisis de la bibliografía existente sobre el tema.
- Confección del marco teórico de la investigación.
- Realización de un estudio del estado actual de las métricas para requisitos funcionales en el mundo.
- Análisis de las métricas para requisitos funcionales que plantean los modelos de calidad.
- Estudio de las métricas para requisitos funcionales utilizadas en proyectos.
- Realización de encuestas a analistas de requerimientos para conocer las métricas para requisitos funcionales utilizadas en la producción de *software* en la UCI.
- Realización de entrevistas a analistas de requerimientos, líderes de proyecto y especialistas de calidad de la UCI.
- Selección de una metodología para la definición de métricas.

- Análisis de la propuesta de sistemas de métricas para requisitos funcionales acorde a los lineamientos de calidad de *software* en la UCI.
- Selección de la población inicial de expertos.
- Determinación del grado de experiencia.
- Selección de expertos para el comité de evaluación.
- Evaluación de la validez y objetividad del sistema de métricas propuesto.

Los métodos científicos de trabajo constituyen el conjunto de acciones reglamentadas que posibilitan el avance en el conocimiento del objeto de estudio hasta lograr el objetivo de la investigación. En esta investigación se emplearon como **métodos empíricos**:

La entrevista a profesores con experiencia en temas relacionados a la calidad de *software* con el objetivo de intercambiar criterios para la elaboración y aplicación de métricas.

La encuesta a estudiantes y profesores que trabajan actualmente en proyectos para determinar los problemas que se presentan con respecto al levantamiento de requisitos y la gestión de los mismos, donde puede incidir la aplicación de las métricas y el conocimiento que tienen sobre el tema.

Los **métodos teóricos** permiten la interpretación de datos empíricos y se emplean en la construcción y desarrollo de teorías. En esta tesis se emplea el **método analítico-sintético** para extraer de la información encontrada los elementos más importantes relacionados al objeto de estudio.

El **método inductivo-deductivo** como forma de razonamiento que permite el análisis de elementos generalizadores a otros más específicos. Finalmente el **análisis histórico-lógico** para estudiar la trayectoria histórica real de los fenómenos así como su evolución y desarrollo en un período de tiempo. Este método es de vital importancia en la investigación pues permite la comparación de resultados y la elaboración de conclusiones de acuerdo a los mismos.

La tesis consta de tres capítulos:

Capítulo I: se abordan temas relacionados al estado actual de la Industria del *Software* en el mundo, así como su desarrollo y perspectivas futuras en Cuba. Se explican conceptos relacionados con la Ingeniería de Requisitos, la Calidad del *Software*, la medición de la misma y las Métricas de *Software*. Se exponen los Modelos de Calidad y las Métricas que fueron analizadas en el transcurso de esta investigación, así como la metodología en la que se basa el Sistema de Métricas que se propone.

Capítulo II: se expone la situación actual de la disciplina de Ingeniería de Requisitos en la UCI mediante el procesamiento de los resultados obtenidos de la aplicación de encuestas y entrevistas. Se presentan los resultados obtenidos del estudio realizado a los Modelos de Calidad durante la investigación. Se explica la metodología mediante la cual se confeccionó el Sistema de Métricas y se propone el mismo.

Capítulo III: se valida el resultado de la investigación mediante el criterio de expertos en cuanto a importancia, eficiencia y aplicabilidad. Se explica cómo evaluar la calidad de los proyectos productivos empleando el Sistema de Métricas propuesto.

Capítulo 1
Fundamentación Teórica

1.1. Introducción

En este capítulo se abordan temas relacionados a la Industria del *Software* su estado actual y perspectivas futuras en Cuba. Se explican conceptos relacionados con la Ingeniería de Requisitos, la Calidad del *Software* y la medición de la misma. Además se exponen los Modelos de Calidad, las Métricas y las Metodologías para definir métricas, estudiadas en el transcurso de esta investigación.

1.2. Estado del arte en la industria del *software* en el mundo

La necesidad de aplicar los principios de otras ingenierías al desarrollo de *software* está plenamente justificada por el alto grado de fracasos en el mismo. Desde 1994 el Grupo Standish realiza estudios en los que se encuesta a directores de proyectos sobre la situación de éstos y sus principales problemas en Estados Unidos. La experiencia adquirida durante años de desarrollo en la industria del *software* ha demostrado que al menos un tercio de los proyectos se cancelan durante su desarrollo y que la gran mayoría presenta graves desviaciones respecto a plazos y presupuestos iniciales. Aún así en la actualidad el mercado mundial de *software* es de 196 200 mil millones de dólares y es precisamente Estados Unidos quien lo domina con una importación del 72%, seguido por España y Canadá.

La India con sus más de $1.060 millones de habitantes se muestra como el modelo más representativo en la promoción de la industria del *software* y servicios informáticos en el mundo, cualidad ganada gracias al esfuerzo del Estado en conjunto con el sector privado. En el año 2005 este sector presentó el 3.15% Producto Interno Bruto (PIB) del país con $16.500 millones y se espera para el 2014 ascender al 10%. Además es el país con mayor número de ingenieros altamente calificados, generando anualmente alrededor de 200 000 técnicos informáticos de primer nivel. Se sitúa en el tercer lugar a nivel mundial en cuanto a mano de obra altamente competitiva, condición por la cual

posee 42 de las 52 empresas que en todo el mundo han obtenido el nivel 5 del *Capability Maturity Model* (CMM). En este nivel la organización entera se enfoca en la mejora continua de procesos. La organización tiene los medios para identificar debilidades y fortalecer los procesos proactivamente, con la meta de prevenir la ocurrencia de defectos. Los datos en la efectividad del proceso de *software* son usados para realizar análisis de beneficio de costo de nuevas tecnologías y cambios propuestos a los procesos de *software* de la organización. Se identifican las innovaciones que aprovechan las mejores prácticas de ingeniería de *software* y se transfieren a lo largo de la organización.

El 62% de las exportaciones de este gigante del *software*, en cuanto a servicios y productos, tienen como destino final los Estados Unidos. Le sigue el mercado Europeo con aproximadamente el 24%.Ver Figura 1.

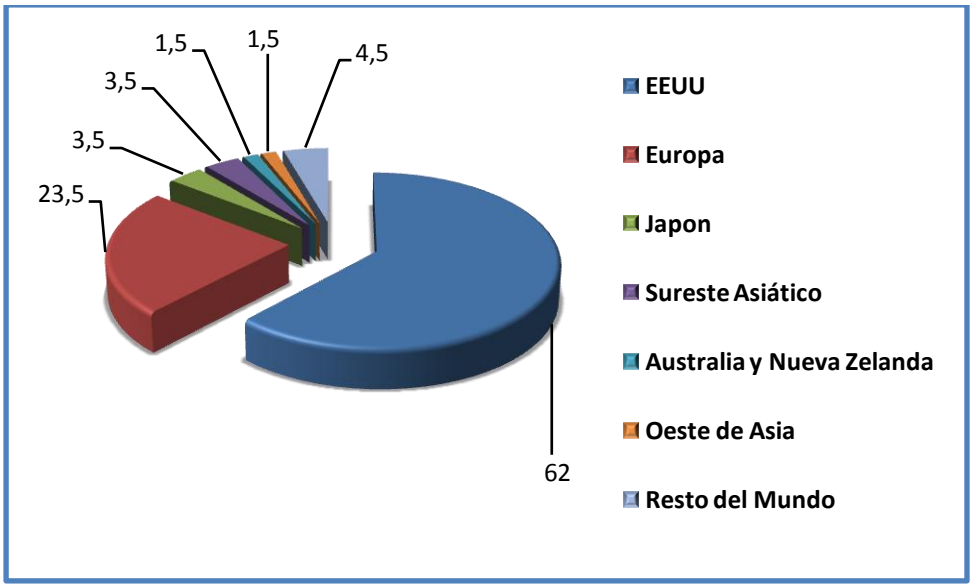


Figura 1 Destino de exportaciones a nivel mundial

1.2.1. Industria de *software* en Latinoamérica

En América Latina la producción de *software* ha adquirido un papel protagónico en la economía como fuente notable de ingresos. El desarrollo de este sector en cada país está determinado por los esfuerzos que realice su gobierno, así como las condiciones que tiene el mismo para invertir en las nuevas tecnologías y formar personal calificado.

Los países latinoamericanos tienen una participación del 1.8% con 3572 mil millones de dólares que sobre el gasto total de las Tecnologías de la Información y las Comunicaciones (TIC´s) representa el 8%, cifra notablemente baja considerando que Estados Unidos tiene una participación del 18%. Por otro lado la relación de inversión en las TIC´s sobre el PIB latinoamericano es sólo el 2%, mientras que en los países desarrollados es 7.5%.Como países importantes en este sector podemos destacar a Brasil, México y Argentina con una participación del 52%, 17% y 11% respectivamente. Ver Figura 2.

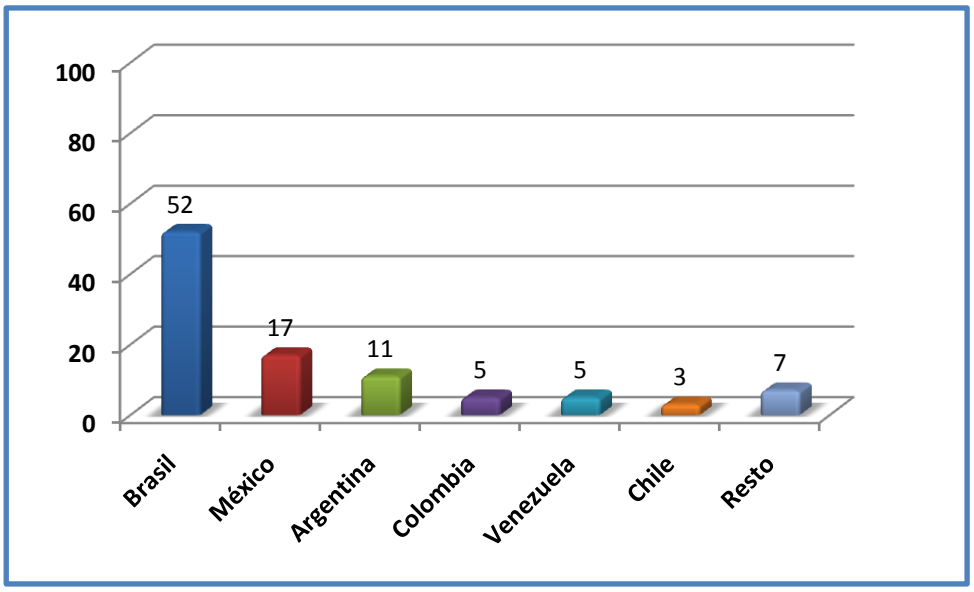


Figura 2 Mercado de *Software* en Latinoamérica

Brasil tiene el mayor mercado de la región, con alto desarrollo en *Enterprise Resourse Planning* (ERP), *Costumer Relation Management* (CRM) y comercio electrónico. Se encuentra asociado a otros países como Estados Unidos, Alemania, Reino Unido e Israel. Consta con una buena participación de los proveedores locales.

México le sigue muy de cerca con una relación inversión TIC/PIB de 1.5%, más baja que el promedio de la región (2%), aún cuando su competencia externa está dominada por los Estados Unidos. Sus productos son altamente demandados en los sectores económicos de mayor crecimiento y concentración. Tiene un mercado de *software* de aproximadamente $7013 millones y se estima que su crecimiento anual es de un 3%. Se espera que para el 2013 alcance una producción anual de $5 000 millones, hecho que lo convertiría en el líder de la región en cuanto a productos de *software* y contenidos digitales en español.

Argentina por su parte es el mayor exportador de la región con una industria madura en comparación con el resto con una buena infraestructura de soporte además de una oferta diversa y de buena calidad. Costa Rica ha sido uno de los países que más beneficios ha obtenido por concepto de prestación de servicios informáticos. Su sector de los servicios para exportación es el más desarrollado en toda Centroamérica con más de 70 empresas incluyendo centros corporativos y desarrolladoras de *software*. Su producción se ha concentrado en soluciones empresariales ERP. Como se muestra en la siguiente tabla.

Segmento	**Empresa más representativa**	**Operaciones**
Centros de contacto	*Sykes*	*14*
Servicios compartidos	*GBS-P&G, HP, IBM*	*19*
Diseño e ingeniería	*Align Tech, Intel*	*6*
Desarrollo de software	*Intel*	*12*
Centros corporativos regionales - HQ	*BAT, Coca Cola, Kraft, Merck, Pfizer, Roche*	*17*
Reparación	*Teradyne*	*3*

Tabla 1 Sector de Servicios Informáticos en Costa Rica

También se destaca en el desarrollo de productos de *software* estandarizado que se comercializa más de una vez, como por ejemplo: *software* empotrado y aplicaciones especializadas. En la actualidad existen más de 300 compañías productoras de *software* de alta calidad instaladas en el país, algunas de ellas asociadas a Microsoft y Oracle.

Costa Rica exportó en el año 2003 $70 millones y ya en el 2005 ésta cifra había aumentado a $90 millones. De forma general puede decirse que se exporta el 20% de la producción total a Centro América, América del Sur, Estados Unidos y Europa. Uruguay cuenta con una industria de *software* orientada al mercado exterior con una participación del 60% en el mercado de aplicativos de *software* y un 35% en licencias. Sus mayores importaciones vienen de Estados Unidos. Los resultados obtenidos en la rama han sido palpables con una exportación del 34% de su producción, que aporta $240 millones.

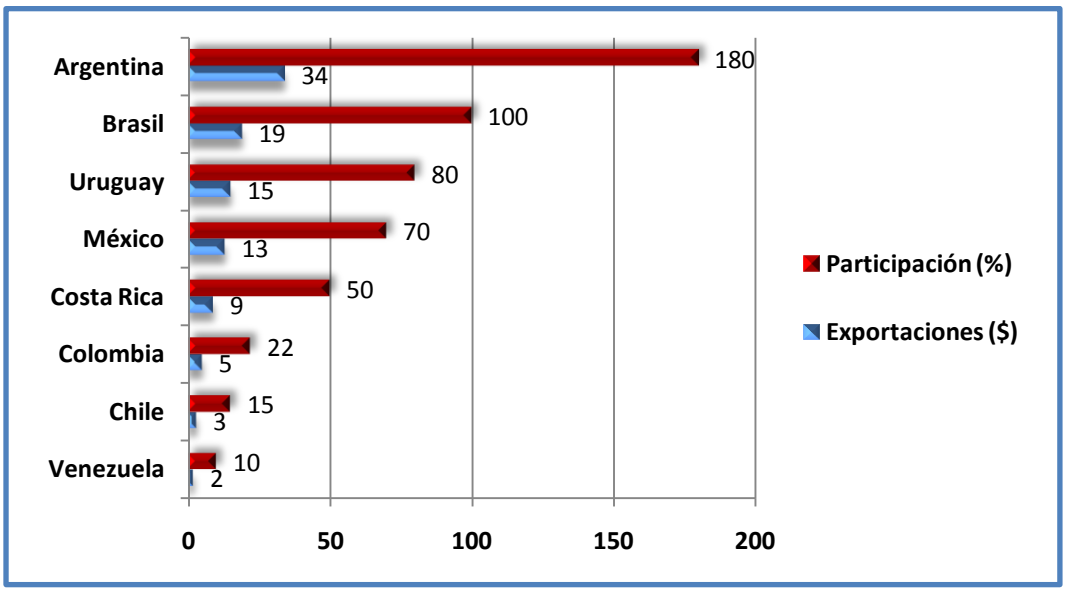


Figura 3 Principales productores de *Software* en Latinoamérica

1.3. Estado actual de la Industria del *Software* en Cuba

En Cuba, a pesar de no tenerse la tecnología de punta requerida para un desarrollo acelerado se realizan grandes esfuerzos por insertarse en esta nueva industria como fuente proveedora de grandes riquezas. Desde 1997 se apuesta por el desarrollo de la Informática como renglón económico generador de importantes ingresos para el país y como medio para mejorar la calidad de vida del pueblo cubano.

En ese año la Resolución Económica del V Congreso del Partido Comunista de Cuba contenía orientaciones para el desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC) y se aprobaron los Lineamientos Generales para la Informatización de la Sociedad hasta el año 2000. Con el inicio del nuevo milenio se creó el Ministerio de la Informática y las Comunicaciones (MIC) con el objetivo de fomentar el uso de las TIC en la economía, la sociedad y los servicios a la población aumentando la calidad de vida de la misma. Luego se formula el Programa Rector de la Informatización de la Sociedad donde se detalla la estrategia a seguir en el período 2000 – 2002.
La informatización de la sociedad cubana no es más que el proceso de una amplia utilización en forma ordenada de las Tecnologías de la Información y las Comunicaciones (TIC), con el objetivo de satisfacer las necesidades cognitivas de todas las esferas de la sociedad. Esta estrategia está

contenida en el Programa Rector de la Informatización de la Sociedad en Cuba. En el mismo se definen 7 áreas de acción:

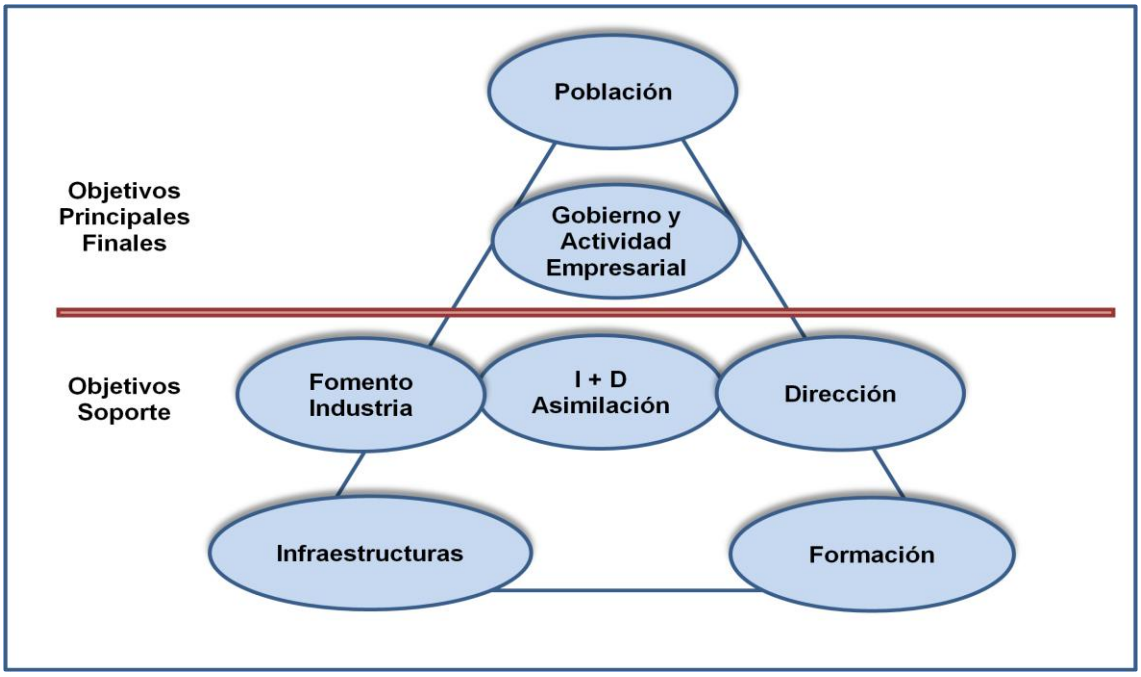


Figura 4 Estrategia cubana de informatización de la sociedad en Cuba

Este programa promueve el uso intensivo de las TIC en todo el país, siguiendo la estrategia prevista que contempla al ciudadano como el objeto principal de sus acciones, apostando por un mejor desempeño del mismo como ente social. Esta estrategia no solo será importante en el incremento del nivel cultural de la población sino también decisiva para una mejora sustancial de la situación económica del país.

"La Industria Cubana del *Software* (ICSW) está llamada a convertirse en una significativa fuente de ingresos nacional, como resultado del correcto aprovechamiento de las ventajas del considerable capital humano disponible.". **(MINREX 2004)**

Por tal motivo se han creado empresas que cumplen con diferentes funciones relacionadas con la producción, mantenimiento, venta y prestación de servicios de *software*. Algunas de ellas son:

Centro Nacional de Superación y Adiestramiento en Informática (CENSAI): Se encarga de la capacitación y adiestramiento de especialistas, desarrolladores informáticos y trabajadores en general. Lleva 10 años produciendo *software* educativo.

Empresa Productora de *Software* para la Técnica Electrónica (SOFTEL): Desarrolla, produce y comercializa aplicaciones informáticas y servicios asociados. Desarrolla proyectos a la medida.

Grupo de Tecnologías de la Información: produce y comercializa de forma mayorista y minorista en Cuba y en el extranjero, proyectos integrales o no asociados a la Informática, Automática y Telemática, así como los productos asociados a éstas tecnologías.

Empresa de Tecnologías de la Información y Servicios Telemáticos Avanzados CITMATEL: se destaca en la introducción de nuevas tecnologías de primera línea en el país, proveedora de Internet en Cuba, pionera en el desarrollo de redes de transmisión de datos. Desarrolla productos de multimedia, DVDs y sistemas para la automatización empresarial.

Centersoft: creada en 1999 en Tarará es el primer centro cubano dedicado exclusivamente a la producción de *software*. Hasta el momento ha desarrollado proyectos con Brasil, España, Italia, México y Suiza. Además ha vendido productos de *software* a Italia, México y República Dominicana y a la Organización Mundial de la Salud (OMS).

NOVASOFT LTDA: creada en 1988. Compañía líder en el desarrollo y comercialización de *Software* de Gestión Empresarial para todo tipo de empresas del mercado. Su objetivo principal es ofrecer soluciones de *Software* que satisfagan las necesidades y expectativas de manejo de información de sus clientes, empleando siempre los últimos avances tecnológicos. Asegura la calidad en la creación de productos, soporte y servicio postventa.

En Cuba se produce *software* para diferentes renglones nacionales como por ejemplo salud, educación y economía. Estos productos son exportables y constituyen actualmente una fuente de ingresos en aumento aunque vale aclarar que aún existe un alto porciento de soluciones artesanales que frenan el avance en el sector. Además aún el índice de fallos en los proyectos de *software* es notable, se retrasan las entregas, se sobregiran presupuestos y se presentan problemas con la calidad del producto. Aún muchos procesos son informales y en algunos casos no confiables.

En la actualidad las empresas que producen *software* necesariamente deben velar por la calidad de los productos y servicios debido a la alta competitividad que existe en esta industria. Siendo sumamente importante la reducción del tiempo de desarrollo sin descuidar la calidad antes mencionada para de esta forma aumentar la productividad así como la satisfacción del cliente. El proceso de aseguramiento de la calidad de *software* es el factor clave en el desarrollo de productos informáticos. Softcal, empresa asociada al MIC creada en 1977 se encarga de ésta importante actividad en el territorio nacional. Especializada en ofrecer soluciones de gestión empresarial tiene como principal objetivo elevar la eficacia y la eficiencia de las empresas mediante el empleo de modernas técnicas informáticas ofreciendo además servicios de consultoría organizacional, aumentando la calidad y competitividad en las mismas. También ofrece servicios a empresas extranjeras radicadas en el país y está asociada a otros países como Canadá, México, España e Italia.

1.4. Ingeniería de Requisitos. Definiciones

Según la IEEE Std. 610.12-1990, un requisito es:

a) Capacidad o condición que debe reunir un sistema para satisfacer un contrato, estándar, especificación u otro documento formalmente impuesto.
b) Representación documentada de una condición o capacidad.

Los requisitos pueden catalogarse de dos formas: funcionales (RF), que representan las condiciones o cualidades que debe cumplir el sistema que se pretende desarrollar, especificando la interacción que tendrá con los usuarios o no funcionales (RNF) si describen atributos o restricciones del propio sistema o del proceso de desarrollo (por ejemplo: características de la interfaz, prestaciones, atributos de calidad, condiciones de hardware).

La Ingeniería de Requisitos (IR) es una rama de la Ingeniería de *Software* (IS) que determina el proceso de establecimiento de los principios, técnicas, herramientas y métodos que permiten a los desarrolladores descubrir, documentar y mantener los requisitos de un producto informático, así como las restricciones sobre las cuales éste deberá operar, de forma sistemática y repetible. Como término fue mencionada por primera vez por Fritz Bauer en la primera conferencia sobre desarrollo de *software*

patrocinada por el Comité de Ciencias de la OTAN que se celebró en Garmisch, Alemania en Octubre del año 1968.

La necesidad de aplicar los principios de otras ingenierías al desarrollo de *software* está plenamente justificada por el alto grado de fracasos en los proyectos de este tipo. Sus principales causas están íntimamente relacionadas con la correcta identificación y gestión de los requisitos que deben cumplir las aplicaciones que se desarrollan con el objetivo de satisfacer las necesidades de los usuarios.
Si se garantiza un proceso de gestión de requisitos con calidad se evitarán futuros errores en el *software* y los atrasos que éstos traen consigo. La IR facilita la comprensión de los deseos del cliente, analiza necesidades confirmando la viabilidad de las mismas, negocia soluciones razonables especificándolas sin ambigüedad. Finalmente valida los requisitos transformándolos en un sistema operacional.

Según Roger S. Pressman la Ingeniería de Requisitos puede definirse en 5 pasos fundamentales:

1. **Identificación de requisitos:** se realiza la entrevista a los clientes y demás implicados para conocer el ajuste del producto a las necesidades del negocio y el uso diario que tendrá. Este proceso se afecta cuando no se definen correctamente los límites del sistema, los clientes o usuarios no están seguros de lo que realmente necesitan. En muchos casos no hay una comprensión real del dominio del problema y se omite información que se considera obvia, que puede ser de vital importancia para especificar requisitos ambiguos o inestables.

2. **Análisis y negociación de requisitos:** generalmente los clientes solicitan más funcionalidades de las que realmente son necesarias o pueden realizarse. Por tal motivo, después de la recopilación, éstos se agrupan en categorías y se crean subconjuntos dentro de ellas viabilizando el estudio de la relación de cada uno con el resto.

3. **Especificación de requisitos:** para la realización de este paso puede definirse una plantilla estándar, indistintamente del tamaño del proyecto que se desarrolle. De ésta forma se garantiza que los requisitos se presenten de forma más comprensible.

4. **Modelado del Sistema:** en esta etapa se desarrolla un anteproyecto que permita valorar todos los componentes del sistema, sus relaciones entre sí y cómo se han reflejado los requisitos, mostrando la estética del sistema.

5. **Gestión de Requisitos:** se define como el conjunto de actividades que permiten identificar, controlar y seguir la evolución de los requisitos a lo largo del ciclo de vida del sistema. Estas actividades son análogas a las de la Gestión de Configuración de *Software*. Desde el inicio a cada requisito ha de asignársele un identificador y se agrupa según su tipo (funcional, de datos, de comportamiento, de interfaz, de salida). Luego se desarrollan diferentes matrices que posibilitan su seguimiento, por ejemplo: matriz de seguimiento de características, de orígenes, de dependencias, de subsistemas y de interfaces.

1.4.1. Estado actual de la Ingeniería de Requisitos en la UCI

En la Universidad de las Ciencias Informáticas se avanza en el estudio y aplicación de la disciplina de IR aunque aún su desarrollo es incipiente. Así lo demuestran diferentes problemas que afectan la calidad del proceso de desarrollo. En algunos casos se observan atrasos en la entrega de los productos e inconformidad en los clientes. Durante la fase de inicio del proyecto en el flujo de trabajo de levantamiento de requisitos se producen retrasos en la planificación por variaciones continuas en los requisitos. Al culminar el pasado curso académico algunos estudiantes trataron el tema de la IR en sus trabajos de diploma, reflejando en ellos la necesidad de desarrollo de la misma en el centro.

Actualmente un grupo importante de estudiantes y profesores de la UCI desarrollan investigaciones relacionadas a la Ingeniería de Requisitos y aúnan sus esfuerzos por un mejor conocimiento y aplicación práctica de la misma.

1.5. Especificación de Requisitos de *Software*

El documento de Especificación de Requisitos de *Software* (ERS) es una descripción completa y detallada del comportamiento que tendrá el sistema que se desarrollará. En él se listan todos los Requisitos Funcionales (RF) y los Requisitos No Funcionales (RNF).

1.5.1. Calidad de un ERS

Un ERS debe ser: correcto, inequívoco, completo, consistente, comprobable, modificable e identificable. Si se logra que cumpla con estas condiciones entonces puede afirmarse que tiene calidad. Para lograr este objetivo es de suma importancia la comunicación con el cliente y el entendimiento entre ambas partes. La redacción del documento debe ser clara, precisa y comprensible para todos los involucrados, evitando así malentendidos que puedan atrasar la planificación del trabajo ya realizada.

1.6. Calidad de *Software*

Según el diccionario de la Real Academia Española la calidad no es más que “la propiedad o conjunto de propiedades inherentes a algo, que permiten juzgar su valor.” La calidad está definida por las características medibles de los elementos, dígase tamaño, textura, color, atributos comparables con estándares previamente definidos.

En el caso del *software* el primer elemento que se tiene en cuenta cuando se evalúa la calidad es la concordancia de éste con los requisitos que el cliente especificó en sus inicios. También es importante la calidad de los modelos resultantes de los flujos de Análisis, Diseño e Implementación. Si ésta es alta es muy poco probable que el producto final no la posea. Influye también la aplicación de estándares y criterios de desarrollo que guían la IS. Por último se encuentran los requisitos no funcionales, sin el cumplimiento de éstos la calidad no será del todo fiable aunque el producto sea explotable en sus principales funcionalidades. La Calidad del *Software* se diseña conjuntamente con él, nunca al final.

1.7. Modelos y Estándares de Calidad estudiados para la investigación

Los Modelos de Calidad no son más que documentos que integran en sí las mejores prácticas para un desarrollo de *software* eficiente. Proponen temas de administración en los que cada organización que los aplique debe hacer énfasis e integran diferentes prácticas dirigidas especialmente a procesos clave. De esta forma permite medir la evolución de la calidad durante todo el ciclo de vida del sistema informático.

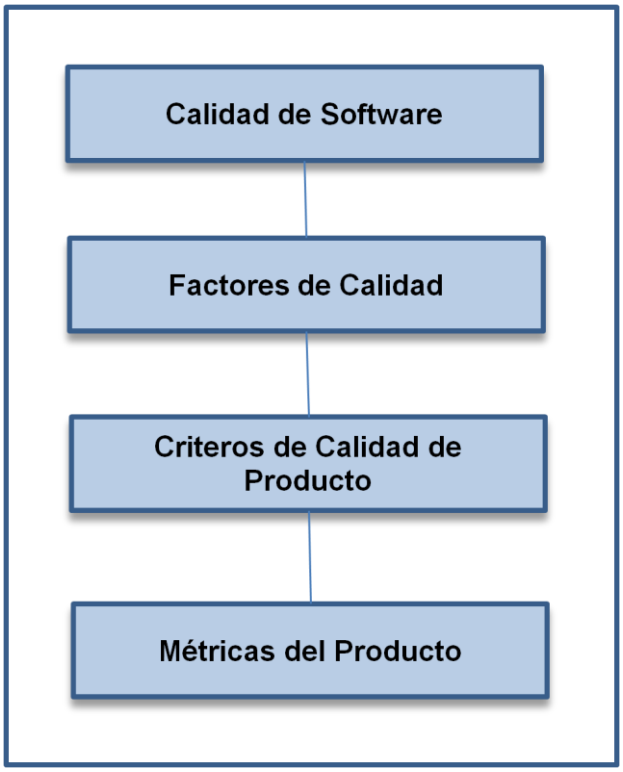


Figura 5 Modelo de Calidad de *Software*

Los Estándares de Calidad por su parte, permiten la definición de un conjunto de criterios de desarrollo que serán las pautas a seguir para la aplicación de la Ingeniería de *Software*. Suministran lo necesario para que todos los procesos se realicen de igual modo siendo de ésta forma, una guía para el logro de la calidad y la productividad.

Ambos posibilitan que en las entidades donde se empleen se realicen las actividades teniendo en cuenta la calidad de las mismas. Factor de vital importancia en la actualidad especialmente en el mercado de *software* donde la competencia es cada día más alta y los clientes más exigentes con respecto a la eficiencia de los servicios y productos que reciben. Además permiten la medición y planificación de esta condición del sistema aumentando la experiencia de los desarrolladores en general. Para la realización de ésta investigación se han analizado los siguientes modelos y estándares:

1.7.1. Estándar de Gestión de la Calidad ISO-9001:2000

Especifica los requisitos para un sistema de gestión de la calidad que pueden utilizarse para su aplicación interna por las organizaciones. Está estructurada en cuatro grandes bloques, completamente lógicos, por lo que con éste modelo del sistema de gestión de calidad basado en ISO se puede desarrollar cualquier actividad. Su estructura es válida para diseñar e implantar cualquier sistema de gestión, no solo el de calidad.

1.7.2. Estándar ISO 15504 *(SPICE: Software Process Improvement and Capability dEtermination)*

Fue creado en enero de 1993. Es un modelo de madurez para la evaluación de procesos de *software* y la base para el comercio internacional. Su objetivo es evaluar los procesos que utiliza la organización de acuerdo a las necesidades que presenta, señalar fortalezas, debilidades y determinar los riesgos inherentes a los mismos. Los resultados de éste análisis se utilizan para mejorar el proceso o para determinar la capacidad de la organización. Además generan un procedimiento de valoración de procesos que se caracterice por ser repetible, comparable y verificable.

Ejecuta, planifica, gestiona, controla y mejora las fases de adquisición, suministro, desarrollo, operación y soporte. Proporciona una base común para los diferentes modelos y métodos de evaluación de procesos de *software*, asegurando que los resultados de la valoración sean iguales en un contexto semejante. Básicamente esta actividad examina los procesos seleccionados en cuanto a dónde son efectivos en el logro de las metas, lo cual es realizado a partir de la determinación de la capacidad del mismo.

Tiene alcances diferentes, pues puede aplicarse tanto a nivel de dirección como a nivel de usuarios, de forma tal que se asegura que el proceso está alineado con las necesidades del negocio. Sus principales características son:

- Marco funcional que contempla una dimensión funcional de los procesos.
- Evidencia para la evaluación.
- Recurrencia dada por la selección de instancias de proyectos o productos.

Integra una serie de niveles, asociados a prácticas genéricas, por los que deben pasar los procesos obteniéndose finalmente la madurez. Estos niveles son:

Nivel 0: Incompleto.
Nivel 1: Fabricado informalmente.
Nivel 2: Planeado.
Nivel 3: Bien definido.
Nivel 4: Controlado cuantitativamente.
Nivel 5: Mejora continua.

También se definen Procesos Generales: Cliente – Proveedor, Ingeniería, Administración, Apoyo o Soporte, Proyecto y Organización. Estos, a diferencia de los niveles antes mencionados, presentan prácticas específicas que deben cumplirse para lograr el paso de uno a otro.

1.7.3. Modelo CMMI

Es un Modelo para la mejora o evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de *software*. Actúa sobre las áreas de proceso y es capaz de mejorar 22 de éstas, en la versión que integra desarrollo de *software* e ingeniería de sistemas (CMMI-SE/SW) y 25 áreas de proceso en la que cubre también integración del producto (CMMI-SE/SW/IPPD). En su representación continua permite a la organización seleccionar un área de procesos (o grupo de áreas) y mejorar el proceso relacionado a ella(s). Esta representación emplea los niveles de capacidad para caracterizar la mejora relativa a un área de procesos individual. Estos niveles se agrupan en 4 categorías según su finalidad: Gestión de proyectos, Ingeniería, Gestión de procesos y Soporte. En la escalonada emplea grupos de áreas de procesos predeterminados para definir un camino de mejora para la organización. Este camino está caracterizado por 5 niveles de madurez: Inicial, Gestionado, Definido, Gestionado Cuantitativamente, Optimización. El empleo de este modelo genera beneficios como:

- Mejora del tiempo de entrega.
- Incremento de la productividad.
- Mejora de la calidad como medida de defectos.
- Incremento de la satisfacción del cliente.
- Reducción del coste de la calidad.

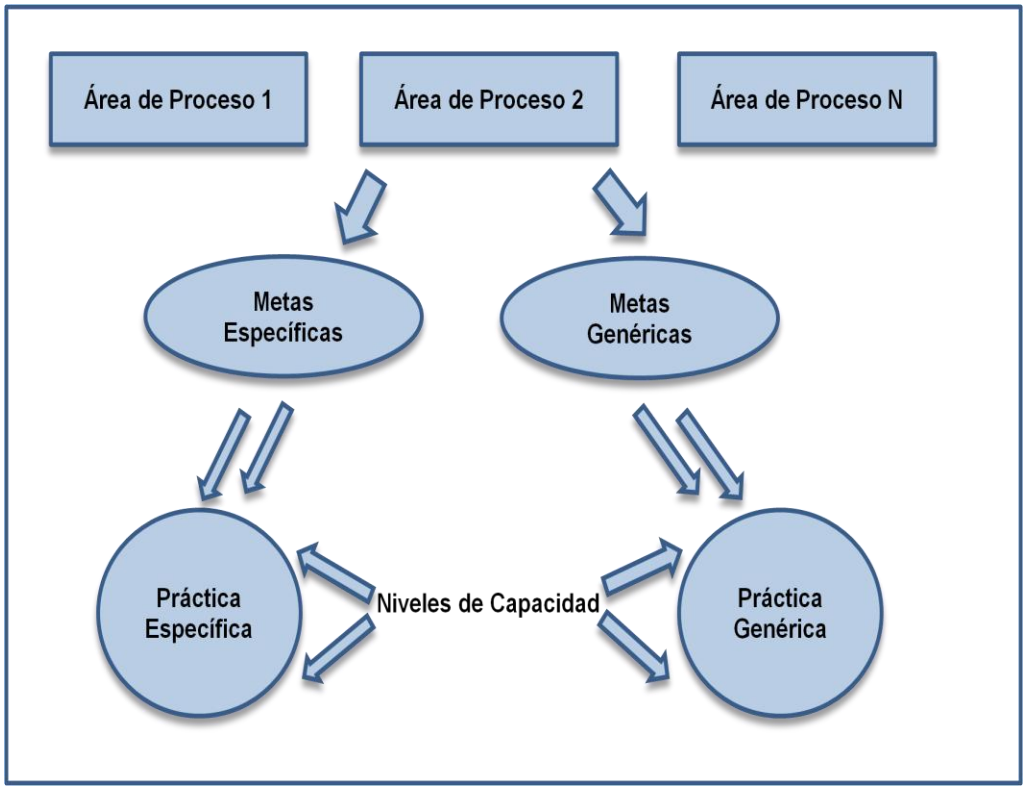


Figura 6 CMMI. Enfoque continuo

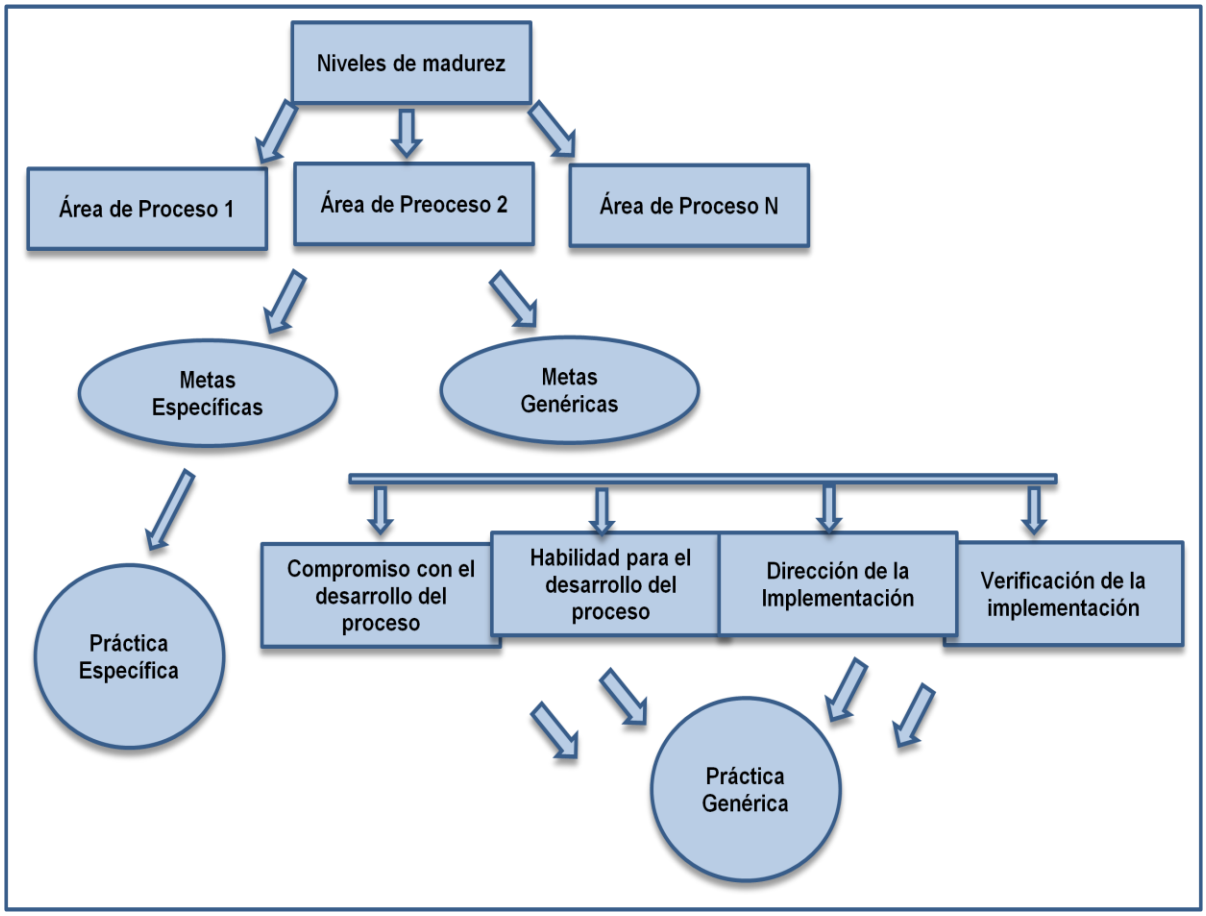


Figura 7 CMMI. Enfoque escalonado

1.7.4. Estándar *IEEE Std. 830-1998 Recommended Practice for Software Requirements Specifications*

Es un estándar que define la calidad de un documento de ERS (Especificación de Requisitos de *Software*) en cuanto a las características que han de presentar los requerimientos de *software* que la organización debe poseer. Fue aprobado el 25 de junio de 1998. Establece que un buen ERS debe ser correcto, inequívoco, completo, consistente, organizado de acuerdo a su importancia y/o estabilidad, comprobable, modificable e identificable.

Correcto: todo requisito que se declare en él tiene que aparecer en el producto.
Inequívoco: cada requisito tiene una única interpretación posible.
Completo: reconoce todos los requisitos, ya sean funcionales o no, define la respuesta del sistema ante cualquier situación posible, tiene claras y correctas todas las referencias.
Consistente: no debe haber incongruencia entre requisitos.
Organizado de acuerdo a su importancia y/o estabilidad: debe identificar mediante un valor numérico la importancia y estabilidad del requisito en particular.
Comprobable: cada requisito ha de ser comprobable.
Modificable: debe ser posible de modificar fácilmente y su estructura ha mantenerse inalterable ante cualquier cambio.
Identificable: debe estar claro el origen de cada requisito, así como las referencias desde y hacia él.

1.7.5. *NC ISO/IEC 9126-1: 2005 Ingeniería de Software – Calidad del Producto*

Esta Norma Cubana consta de 4 partes:

- Modelo de Calidad.
- Métricas Externas.
- Métricas Internas.
- Calidad en el uso.

Está basada en el estándar ISO/IEC 9126 (1991) *"Software product evaluation – Quality characteristics and guidelines for their use."* Su propósito es especificar y evaluar todas las características de calidad relevantes del producto de *software* a través de métricas ya validadas o comúnmente aceptadas.

Es aplicable a cualquier aplicación informática. Provee una terminología consistente sobre la calidad de *software* y un marco para la especificación de los requisitos de calidad. También ofrece recomendaciones y requisitos para las métricas de los productos y de calidad durante su uso. Brinda diversos ejemplos de éstas métricas aplicables al producto una vez especificados los requisitos de calidad y los objetivos del diseño. Puede ser empleado por clientes, programadores, revisores y personal de aseguramiento de la calidad para:

- Validar la integridad de la definición de los requisitos.
- Identificar los requisitos del *software*.
- Identificar los objetivos del diseño del *software*.
- Identificar los objetivos de ensayo del *software*.
- Identificar los criterios de aseguramiento de la calidad.
- Identificar los criterios de aceptación para un producto de *software* terminado.

Define la calidad del producto y el ciclo de vida de la siguiente forma:

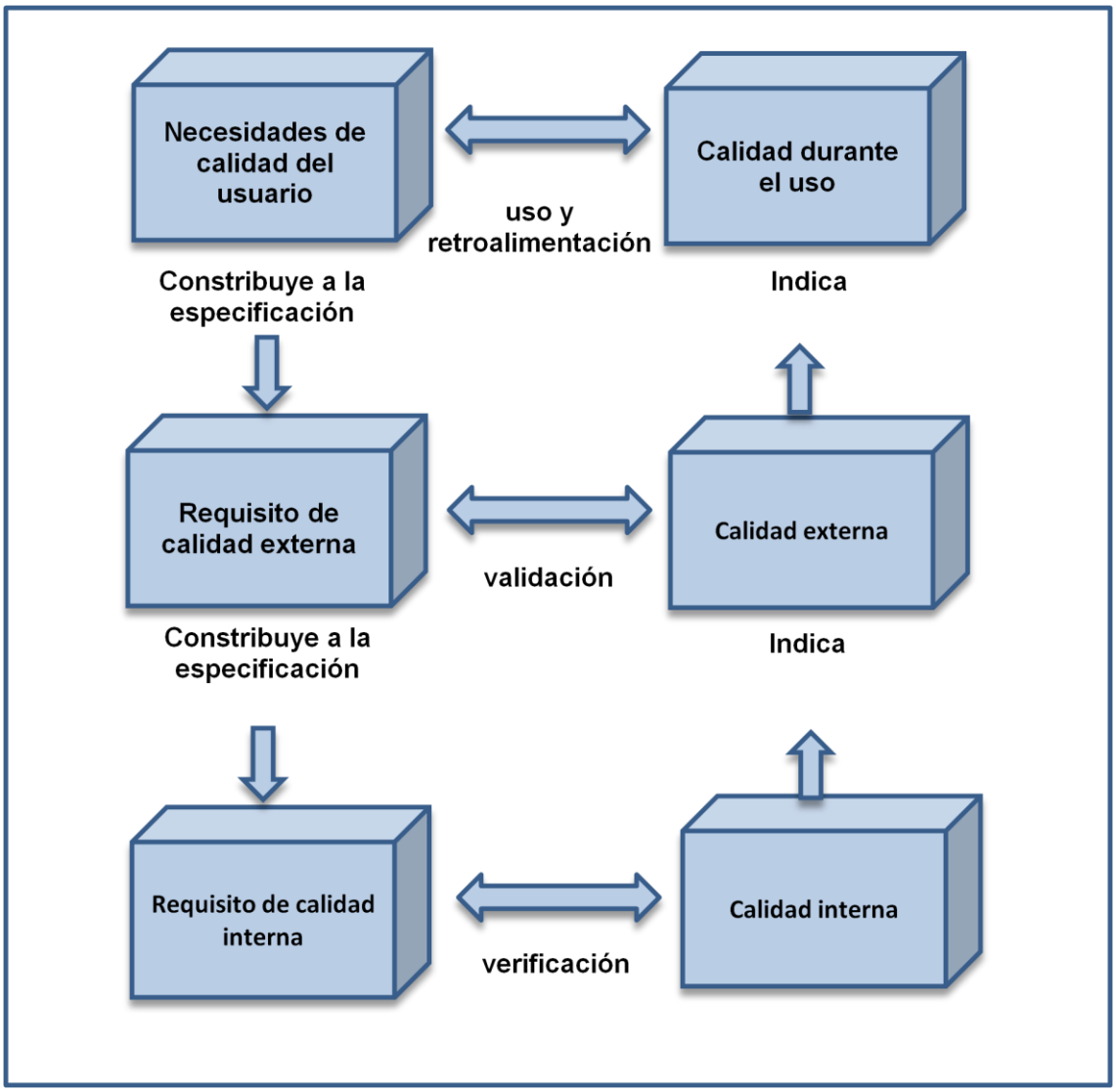


Figura 8 Calidad en el ciclo de vida del *software*

Describe un modelo de calidad para los productos de *software* que se divide en dos partes y especifica cómo ha de ser empleado el mismo:

- Calidad externa e interna: especifica 6 características para la calidad externa e interna, divididas a su vez en subcaracterísticas como resultado de cualidades y atributos internos del *software*. Ver Figura 9.
- Calidad durante su uso: especifica 4 características de calidad durante el uso del producto. Ver Figura 10.

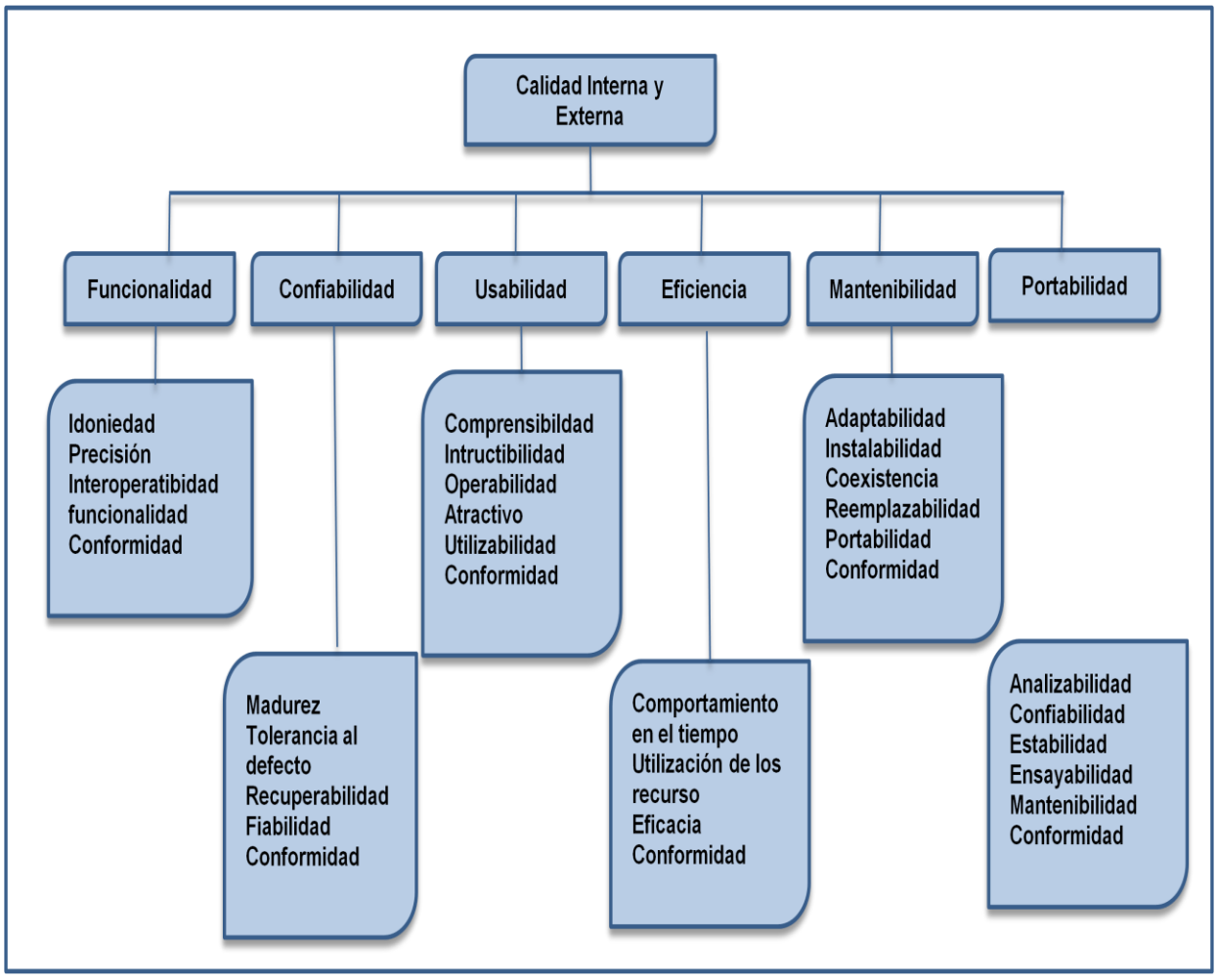


Figura 9 Modelo para la calidad externa e interna

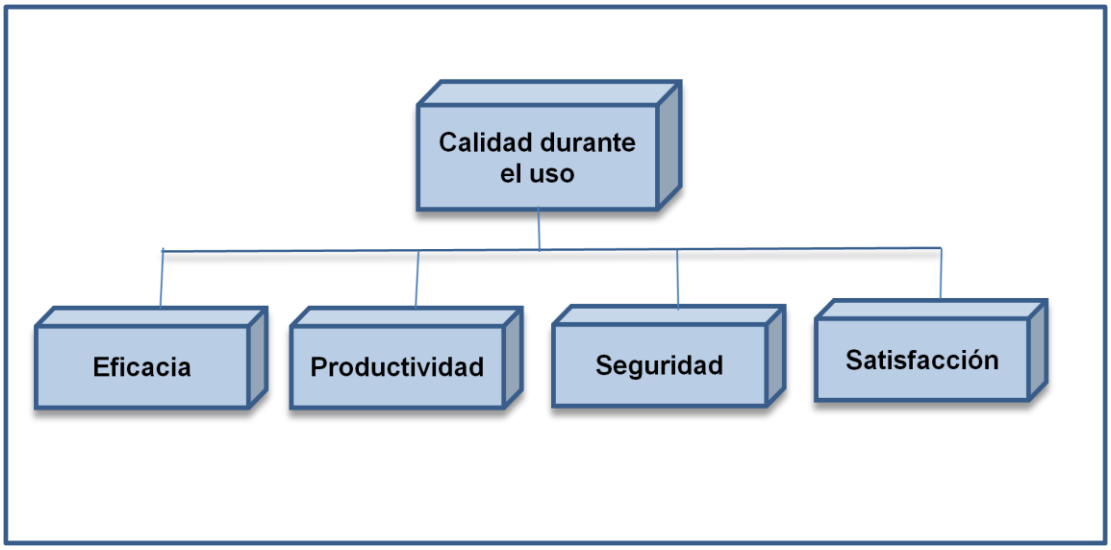


Figura 10 Modelo para la calidad durante su uso

1.8. Métricas de *Software*. Definiciones

La acción de medir es de vital importancia en el desarrollo de cualquier proyecto en toda ingeniería. Permite conocer el estado de avance o retraso del trabajo que se realiza además de la evaluación objetiva de factores como dimensión, capacidad y calidad del mismo**.**

En la Ingeniería de *Software* (IS), disciplina que guía el desarrollo y mantenimiento de sistemas informáticos, como en el resto de las ingenierías, la medición es sumamente necesaria. A través de ella se caracteriza, evalúa, predice y mejora el ciclo de vida del *software* desde fases tempranas evitando muchos de los errores que generalmente se detectan una vez iniciado su despliegue y cuya corrección cuesta 100 veces más que el desarrollo del mismo. El objetivo final de cada proyecto es entregar al cliente un producto de máxima calidad.

Con el fin de determinar la calidad de un sistema informático se emplean las métricas que proporcionan a los desarrolladores una forma cuantitativa de valorar la calidad de los atributos internos de la aplicación en construcción, permitiéndoles predecir la que tendrá al estar terminada completamente. En este caso se tienen en cuenta factores como la cohesión, complejidad ciclomática, cantidad de líneas de código, entre otras.

"Una métrica del *software* relata de alguna forma las medidas individuales sobre algún aspecto (…). Un ingeniero del *software* desarrolla métricas para obtener indicadores. Un indicador es una métrica o una combinación de métricas que proporcionan una visión profunda del proceso que permite al gestor de proyectos (…) ajustar el producto, el proyecto o el proceso (…). La única forma racional de mejorar cualquier proceso es (…) utilizar las métricas para proporcionar indicadores que conducirán a una estrategia de mejora". **(PRESSMAN 2005)**

Para mejorar realmente el proceso es necesario recopilar y medir diferentes atributos en el proceso, desarrollar un juego de métricas y a partir de ellas obtener indicadores que permitan la toma de decisiones. Siendo el proceso el único factor controlable. Las medidas que se toman pueden ser directas o indirectas. En el primer grupo se encuentran el coste y el esfuerzo del proceso de ingeniería, las LDC (líneas de código) del producto, así como los defectos informados en determinado período de tiempo. Como indirectas se tienen la funcionalidad, eficiencia, facilidad de mantenimiento, fiabilidad, entre otras. Factores que no son más que las capacidades que fueron definidas desde un inicio como requisitos no funcionales del sistema. **(PRESSMAN 2005)**

Las métricas pueden ser privadas o públicas ya que éstas son definidas por el propio grupo de desarrollo y por sus integrantes de forma individual. Un ejemplo de métrica privada es el índice de defectos que se le han detectado a un implementador durante su trabajo. Cuando ésta se analiza como información para todo el equipo junto a otros datos significativos, se convierte en pública. **(PRESSMAN 2005)**

Ejemplos de métricas:

Orientadas al tamaño: consideran el tamaño del *software* que se ha construido por medio de la normalización de medidas de calidad o productividad. Por ejemplo:

- Errores por KLDC (miles de líneas de código)
- Páginas de documentación por KLDC

Orientadas a la función: emplean como valor de normalización una medida de la funcionalidad de la aplicación entregada.

- Puntos de Función: se calcula al llenar una tabla que indique campos como el número de entradas y salidas de usuario, a los que se asocia un nivel de complejidad. Luego se emplea la fórmula:

  PF = cuenta-total * [0.65 + 0.01 * 6 * (Fi)] **(PRESSMAN 2005)**

  Donde:

  PF: puntos de función.

  cuenta-total: suma de todas las entradas de la tabla.

  F_i (i = 1 a 14) valores de ajuste de la complejidad.

Orientadas a la calidad de la especificación: su objetivo es medir la calidad del modelo de análisis y la especificación de requisitos correspondiente.

- Especificidad de requisitos.
- Consistencia
- Completitud de los requisitos funcionales
- Grado de validación de requisitos
- Chequeo de realismo
- Verificabilidad

1.8.1. Empleo de métricas en la Industria cubana del *Software*

Para tener éxito en el mercado del *software* en la actualidad es imprescindible el incremento de la calidad de los productos dado a la competencia existente. Por tal motivo es necesaria la aplicación de modelos de calidad y métricas que verifiquen si se ha alcanzado el nivel deseado.

Como resultado de la investigación de la Dr. Aylin Febles Estrada, Licenciada en Ciencias de la Computación, Máster en Informática Aplicada y Sofía Álvarez Cárdenas, realizada en 31 empresas de la Industria Cubana de *Software* se supo que un 61% de los desarrolladores y los líderes de proyectos no conocen que es GCS y en un 79% no se aplica ningún procedimiento asociado a este proceso.

En el Centro de Referencia de Ingeniería de *Software* (CRIS) de la CUJAE se creó un Modelo de Referencia para la Gestión de Configuración en la Pequeña y Mediana Empresa (PYME), MConfig.pm y una Guía Práctica para la Gestión de Configuración en la PYME. Al aplicarlo en el grupo de Gestión Universitaria Digital (GUD) y la fábrica de *software* Temaukel pertenecientes a la CUJAE se obtuvieron resultados satisfactorios.

Otro ejemplo del control de la calidad en la ICSW es CITMATEL que posee certificado su Sistema de Gestión de la Calidad desde el año 2005 bajo la norma NC ISO 9001:2001. Este sistema está conformado por 15 procesos. Uno de ellos es el proceso de producción de *software*. Este abarca: multimedia, gestión económica y web, que son los 3 ambientes de trabajo que se desarrollan en la empresa. Se inicia al realizarse la solicitud por parte del cliente, ya sea externo o interno y finaliza con la entrega del producto. Un paso importante de este sistema de calidad es la evaluación obligatoria de todo proyecto. De esta forma se detectan las fallas y pueden ser eliminadas antes de que el producto salga al mercado.

Durante un estudio realizado por el Centro de Estudios de Ingeniería de Sistemas (CEIS) en 18 empresas de desarrollo de *software* se detectó que a pesar de que en ellas existían direcciones de calidad que incluían revisiones y de la existencia de los recursos necesarios para esta actividad, no se elaboraban o aprobaban planes de revisiones y auditorías para los proyectos. Solo se realizaban pruebas al final del ciclo de vida del producto y no se registraban datos sobre los defectos que se encontraban para de esta forma tener una medida de la calidad que poseía el producto.

Es por eso que investigadores del centro propusieron la aplicación de un Modelo de Proceso de Revisiones que incluye un Sistema de Procedimientos, Métricas y Herramientas que se adecúa a las características propias de la empresa.

Por otro lado en Cuba se ha combinado el empleo de herramientas CASE que almacenan datos históricos, facilitando la predicción y eficiencia del trabajo, con métricas que permiten planificar, monitorear y controlar el proceso de desarrollo de *software*.

Algunos ejemplos de métricas que se emplean para medir el control de configuración son:
Proceso:

- Seguimiento del avance del proceso.
- Calidad de la planificación.
- Error de la planificación.
- Razón de costo de la planificación.
- Pedidos de cambio:
- Estado de los pedidos de cambio.
- Esfuerzo del personal.
- Defectos:
- Resumen de defectos.
- Defectos por hora.
- Eficiencia de la eliminación de defectos.
- Frecuencia de defectos.

1.9. Metodología para la definición de métricas

De forma general para definir una métrica el primer paso a seguir es la documentación del proceso de desarrollo mediante la recopilación de datos previamente identificados. Debe definirse además un procedimiento para la recolección de los mismos. Deben establecerse metas y las métricas necesarias para alcanzarlas. Han de implantarse las herramientas necesarias para el análisis de las métricas. Se crea una base de datos para archivar toda la información recolectada además de las propias métricas. Finalmente se define un mecanismo de retroalimentación. Para efectuar todo este proceso debe ser conformado un equipo que se encargue únicamente de él.

1.9.1. PSM: *Practical Software and Systems Measurement*

Desarrollada para conocer los desafíos en la gestión de *software*. Describe un proceso guiado por la información que dirige las metas técnicas y comerciales de la organización. PSM está definido por un conjunto de 9 prácticas denominadas principios de medición que ayudan en la medición a nivel del

proyecto proporcionando los datos necesarios para dirigirlo con el objetivo de completar costos, cronogramas y objetivos técnicos exitosamente. Estas definen un enfoque de análisis dirigido cuyo fin es brindar la información cuantitativa necesaria que ayude al Gerente del Proyecto en la toma de decisiones relacionadas al *software* y al sistema. Además da soporte a los requisitos de medición a nivel de organización y provee la información requerida para la identificación de diversos aspectos fuera del alcance de los proyectos. Las funciones de administración de *software* en sí se agrupan de la siguiente forma:

Administración de proyecto: permite la planificación y control de los productos.
Administración del proceso: asegura que los procesos se desarrollan de forma correcta y son mejorados.
Ingeniería del producto: asegura la satisfacción del cliente y por ende la aceptación del producto.

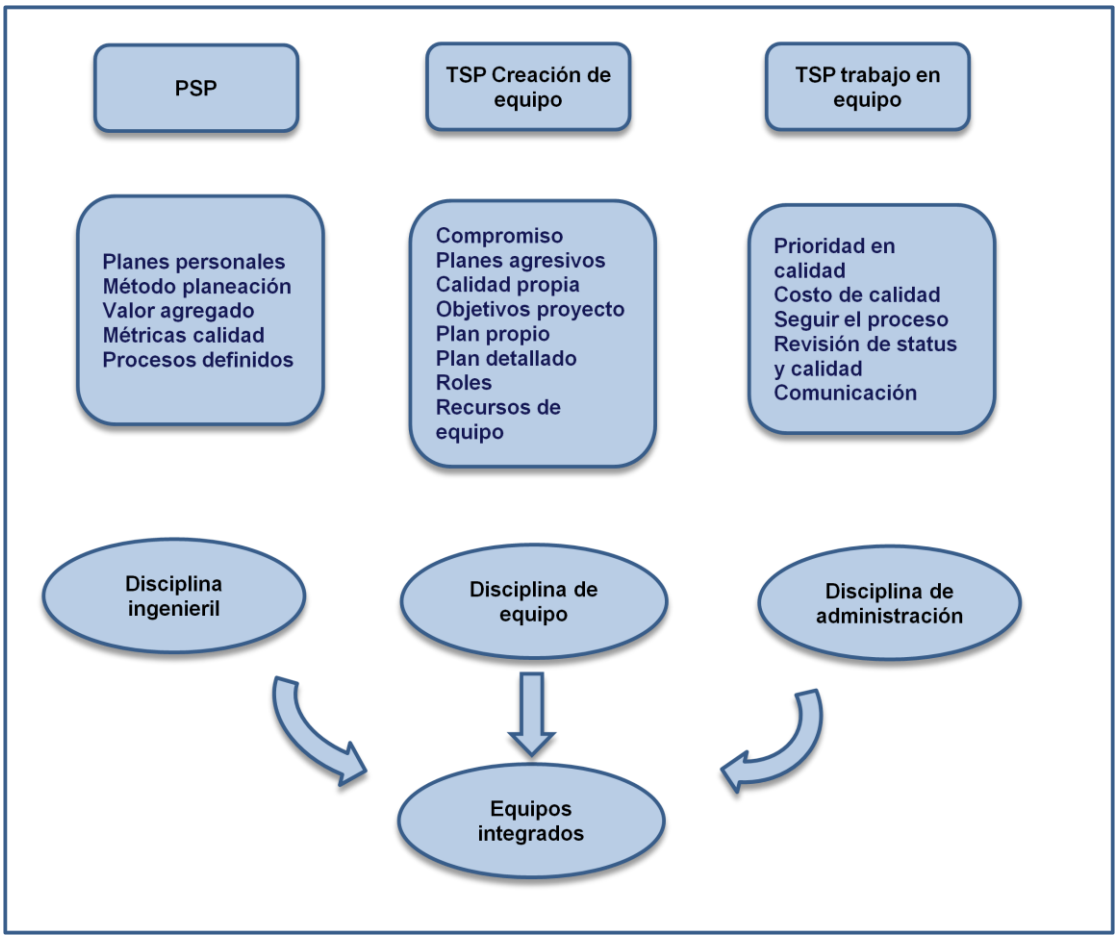


Figura 11 Estructura de PSM

Esta metodología representa las mejores prácticas empleadas por los profesionales en la medición del *software*. El proceso es flexible adaptándose a los objetivos y necesidades específicas de cada programa.

1.9.2. GQM: *Goal Question Metric*

Desarrollada originalmente por Baili y Weiss en 1984 y extendida en 1990 por Rombach como resultado de muchos años de experiencia e investigación. Se basa en que la medición debe ser realizada siempre orientada a un objetivo. Con GQM este objetivo es definido y refinado mediante preguntas potencialmente medibles que a su vez son respondidas por métricas que aportan toda la información necesaria. Presenta 4 fases: Planificación, Definición, Recopilación de Datos e Interpretación y 3 niveles: Conceptual, Operacional y Cuantitativo. En éstas fases primero se emplean las metas del negocio para llegar a las métricas, luego se recopilan los datos y se analizan con el objetivo de mejorar el proceso de la toma de decisiones.

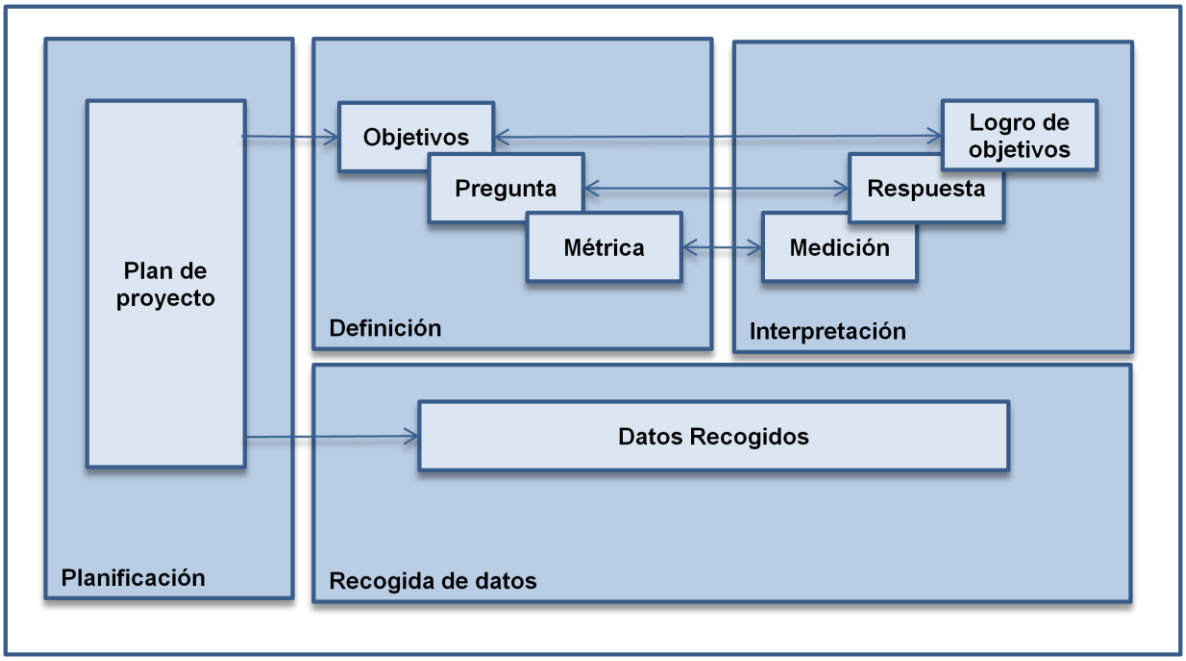


Figura 12 Fases de GQM

Para su aplicación se crea un equipo independiente al equipo que desarrolla el proyecto y se selecciona el área que se desea mejorar, ya sea del producto o el proceso. En éste el Gerente de Proyecto es el responsable de la continuidad del proceso de medición.

Esta metodología es aplicable a todo el ciclo de vida de *software*, procesos y recursos se alinean fácilmente con el ambiente organizacional y puede ser empleada individualmente por los miembros del equipo de desarrollo.

1.9.3. IEEE 1061-1998 Standard for a *Software* Quality Metrics Methodology

Estándar que provee una metodología para el establecimiento de requerimientos de calidad e identificar, implementar, analizar y validar métricas de calidad de *software* para procesos y productos. Es aplicable a cualquier *software* en todas las fases de su ciclo de vida. Este estándar esta dirigido a aquellos que se encuentran asociados a la adquisición, desarrollo, uso, soporte, mantenimiento y auditoria de *software*. Puede ser empleado por: gerentes de proyectos, desarrolladores, auditores y usuarios.

Específicamente el uso de ésta metodología estándar para medir la calidad permite a la organización:

- Evaluar el logro de las metas de calidad.
- Establecer los requerimientos de calidad para un sistema a su salida.
- Establecer estándares y criterios de aceptación
- Evaluar el nivel de calidad alcanzado contra los requerimientos requeridos
- Detectar anomalías o puntos de problemas potenciales en el sistema
- Predecir el nivel de calidad que será alcanzado en el futuro
- Monitorear los cambios en la calidad del *software* si este es modificado
- Evaluar la facilidad de cambio en el sistema durante la evolución del producto

Esta metodología es un acercamiento sistemático al establecimiento de requerimientos de calidad e identificación, implementación, análisis y validación de métricas de procesos y productos de *software*. Comprende 5 pasos:

- Establecer los requerimientos de calidad de *software*.
- Identificar métricas de *software*.
- Implementar el sistema de métricas de *software*.

- Analizar los resultados de las métricas de *software*.
- Validar las métricas de *software*.

Estos pasos pueden ser aplicados iterativamente por la visión ganada de la aplicación de un paso que puede mostrar la necesidad de evaluación de los resultados de pasos previos. Cada paso establece las actividades necesarias para completar los resultados indicados.

A consideración de los autores se utiliza en la presente investigación la *IEEE 1061-1998 Standard for a Software Quality Metrics Methodology* como metodología para la definición de métricas de *software* por ser una de las más generales, sencillas y explícitas. Además es fácil de implementar y la forma de aplicación que propone involucra un mayor número de personas, ya que no es específica para determinados roles del equipo de desarrollo del proyecto.

1.10. Conclusiones del capítulo

En este capítulo se dio una panorámica general del estado actual de la Industria del *Software* en el mundo y en Cuba. Se especificaron conceptos relacionados a la Ingeniería de Requisitos, Calidad y Métricas de *Software* con el objetivo de facilitar la comprensión del contenido de los capítulos posteriores. Además se establece la metodología a emplear para la definición de las métricas que conformarán el sistema que se propondrá.

Como se pudo constatar es absolutamente necesario elevar el nivel de calidad de nuestros productos dado el crecimiento vertiginoso de la producción de *software* a nivel mundial. La mejor forma de lograr esta meta es mediante la aplicación de métricas para controlar la calidad de los productos desde fases tempranas del ciclo de vida del *software*.

Capítulo 2
Propuesta del Sistema de Métricas

2.1. Introducción

Este capítulo tiene como objetivo realizar un diagnóstico para conocer el empleo o no de métricas, por parte de los proyectos productivos de la universidad, específicamente en requisitos funcionales. Así como proponer un Sistema de Métricas para Requisitos Funcionales capaz de dar una medida de la calidad de los mismos, mediante la utilización de la metodología que propone la *IEEE Standard for a Software Quality Metrics Methodology* para definir y validar métricas. Para realizar este diagnóstico se aplicaron diversas técnicas, entre ellas se encuentran, entrevistas, encuestas, revisión bibliográfica, análisis de datos, entre otros.

2.2. Técnicas aplicadas

2.2.1 La entrevista

Es un método de recopilación de datos empíricos, un proceso de comunicación entre dos o más personas, generalmente de forma oral donde las preguntas al entrevistado se realizan por vía directa. Estas interrogantes deben desarrollar se de manera tal que permitan respuestas precisas.

2.2.2. La revisión bibliográfica

Se utiliza para recoger la información que se encuentra registrada en un documento establecido. Se utilizaron documentos en formato digital de los diferentes modelos de calidad analizados en el Capítulo I, libros de la disciplina Ingeniería de *Software* y la documentación perteneciente a los proyectos encuestados.

2.2.3. La encuesta

Es una técnica que se aplica con el objetivo de obtener determinada información de un sujeto preseleccionado de antemano a través de pautas que se indican en un guión o cuestionario previamente diseñado. Esta técnica permite recabar una serie de datos en un amplio número de casos. En efecto, constituye una de las formas más utilizadas para indagar sobre las características principales de un segmento poblacional y/o para conocer las opiniones, experiencias y expectativas de grupos de personas.

2.3. Proceso de elaboración de la encuesta

La aplicación de la encuesta se realiza mediante un procedimiento que se inicia con el diseño (formulación de los objetivos e interrogantes, elección de opciones metodológicas) y continúa con la construcción de muestras, en el caso en que se decida guardar representatividad estadística. Al mismo tiempo que se realiza el diseño se operacionalizan las dimensiones y variables bajo indagación, a partir de su traducción en un cuestionario o guía de preguntas. Por último, se realiza la recolección de datos por diversas vías (depende del tipo de metodología escogida), el procesamiento, el análisis de la información y la presentación final de los resultados.

2.3.1. Formulación de objetivos

El presente estudio se efectúa con la finalidad de conocer el nivel de información que tienen los desarrolladores de *software* de la universidad sobre las métricas relacionadas a la actividad que realizan así como el empleo de las mismas en su trabajo. Además se presta atención especial a las métricas relacionadas con la calidad de los Requisitos Funcionales y su aplicación en la fase de Inicio del ciclo de vida del *software*.

Como producto de esta investigación se espera obtener un Sistema de Métricas para Requisitos Funcionales que guíe a los proyectos productivos por un camino exitoso cuyo final sea un *software* con la calidad deseada. Con el objetivo de darle cumplimiento a lo antes expuesto se desarrollan los siguientes tópicos que serán la base para las encuestas que se aplicarán:

- ¿Qué conocimiento tienen los Analistas de Requisitos, Líderes de Proyectos y demás roles principales acerca de las métricas en general?
- ¿Se considera importante la aplicación de métricas en los proyectos productivos como garantía para obtener un producto de calidad?
- ¿Los miembros de los equipos de desarrollo han aplicado métricas en algún proyecto?
- ¿Cómo y cuáles métricas se aplican en los diferentes proyectos productivos de la universidad?
- ¿Qué conocimiento tienen los Analistas de Requisitos, Líderes de Proyectos y demás roles principales acerca de las métricas para requisitos funcionales?
- ¿En los proyectos productivos existe el rol de Analista de Medición?
- ¿Qué tiempo demora regularmente la obtención de requisitos en los proyectos productivos de la Universidad?
- ¿Cuál es la longitud promedio de los requisitos funcionales de los proyectos productivos de la Universidad?

2.3.2. Diseño de la Muestra

Una vez definidos los objetivos de estudio se trazó una estrategia para la aplicación de las encuestas. Debido a que la investigación no cuenta con el tiempo necesario para el estudio de los 137 proyectos productivos vigentes en la Universidad en el presente año 2008, en la pesquisa se tienen en cuenta aquellos que presentan un mayor nivel de madurez, es decir los que ya tienen todos sus requisitos definidos y documentados. Este inconveniente de tiempo, así como accesibilidad de los individuos en algunos casos, provoca que se defina como universo de estudio 12 proyectos elegidos a consideración de los autores de la presente investigación.

Para la selección de la muestra de la población que será analizada una vez definido el universo de estudio se realizó un Muestreo Probabilístico, seleccionando dentro del mismo el Muestreo Aleatorio Estratificado (MAE).

A) Muestreo Aleatorio Estratificado

El Muestreo Aleatorio Estratificado consiste en subdividir la población en subpoblaciones de tal forma que la unión de éstas será la población, y la intersección de las mismas tenga como resultado el conjunto vacío, es decir, no tendrán elementos comunes. A las

subpoblaciones se les llamará estratos, se tratará de conformar éstos de modo que los elementos dentro de ellos sean homogéneos. El tamaño de la muestra se distribuirá entre los estratos en función de distintos criterios. La característica que distingue al MAE es que la selección de la muestra de cada estrato se desarrolla bajo el procedimiento de muestreo irrestricto aleatorio y se realizará independientemente de los diferentes estratos. Es recomendable estratificar en función del tamaño de las unidades y distribuir la muestra proporcionalmente al número de unidades de los estratos.

Notación

Se supone que la población consta de n unidades que se encuentran distribuidas en L estratos. Las unidades antes mencionadas constituyen una partición de la población; se representará por N_h en el n_i de u en el estado h-ésimo, de aquí: $N = N_1 + N_2 + ... + N_h + ... + N_l$

Total de la población $x_h = \sum_{1}^{N_h} x_{hi}$ media $\bar{x}_h = \frac{1}{N_h} \sum_{i}^{N_h} x_{hi}$

$f_h = \frac{n_h}{N_h}$ Fracción de muestreo del estrato.

Si el tamaño de la muestra de los estratos se distribuye proporcionalmente al número de unidades en el estrato, es decir, si se cumple que:

$n_h = n \frac{N_h}{N}$ para todo h, entonces se concluye que la distribución de la muestra se ha hecho con asignación proporcional.

Para la determinación del tamaño de muestra de una población con S^2 desconocida puede emplearse la expresión:

$$n = \frac{\left(\frac{Z1 - \frac{\propto}{2}}{d}\right)^2 P(1-P)}{1 + \frac{1}{N}\left(\frac{Z1 - \frac{\propto}{2}}{d}\right)^2 P(1-P) - \frac{1}{N}}$$

Donde:

1 - α: Nivel de confianza

d : Error absoluto.

Z: Percentil de la distribución normal

P: Proporción de la población

N: Tamaño de la muestra.

Realizando el cálculo del tamaño de muestra para un nivel de confianza del 90%, con un error absoluto de 0.10, se obtiene el percentil de la distribución normal de 1.64 y se asume como proporción de la población $P = 0.5$, de donde se obtiene:

$$n = \frac{\left(\frac{1.64}{0.10}\right)^2 0.5(1-0.5)}{1 + \frac{1}{383}\left(\frac{1.64}{0.10}\right)^2 0.5(1-0.5) - \frac{1}{383}} = \frac{67.24}{1.18} = 56.98 \approx 57$$

Conociendo que n = X, N_h es la cantidad de integrantes en cada proyecto y N = Y como el total de integrantes de los diferentes proyectos.

Al calcular el tamaño de muestra por estrato (se considera estrato a cada proyecto) aplicando

$n_h = n\frac{N_h}{N}$; resultan los tamaños de muestra por estrato que se muestran en la siguiente tabla:

Estratos	Cantidad	Tamaño de la muestra
MINPAL (Facultad 6)	37	6
LIMSCC (Facultad 6)	64	10
UCID – FAR (Contabilidad y Finanzas)	13	2
SIMPRO (Facultad 5)	17	3
PDVSA – SCADA (Facultad 5)	27	4
ONE (Facultad 3)	31	5
APS (Facultad 7)	9	1
BioSyS (Facultad 7)	42	6
PMF – MINFAR	34	5
Registros y Notaría (Facultad 3)	58	9
Equipos Médicos (Facultad 6)	12	2
Ensayos Clínicos (Facultad 6)	39	6
TOTAL	**383**	**59**

Tabla 2 Tamaños de muestra por estratos de la población

2.3.3. Clasificación de variables

Una variable es una función que asocia a cada elemento de la población la medición de una característica que se desea observar. De acuerdo a la característica que se estudia en este caso, se muestran en la Figura 2.1 los valores que toma la variable.

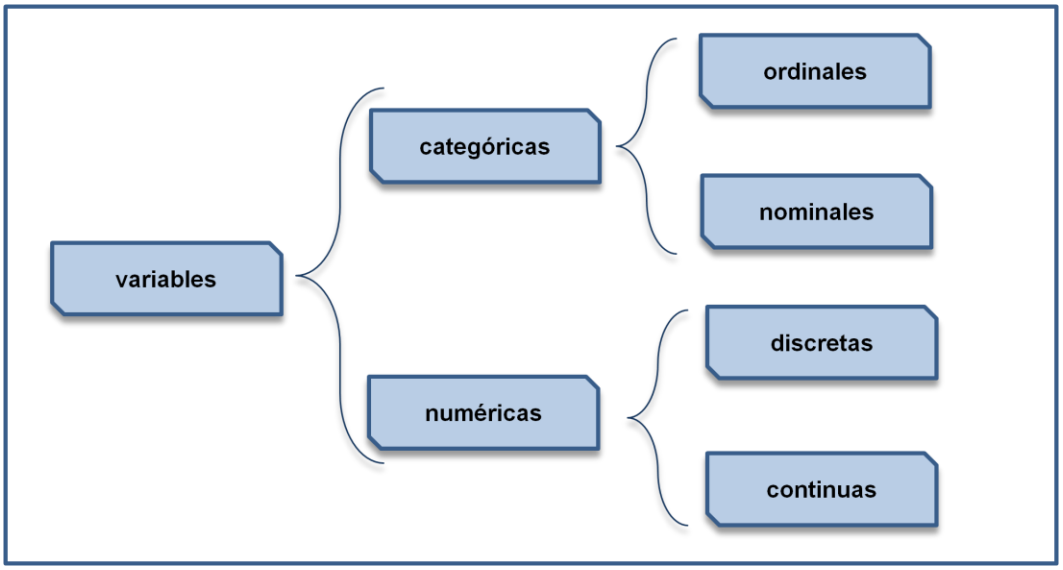


Figura 13 Clasificación de variables

Para este diseño en específico se utilizaron las variables categóricas nominales y las numéricas discretas. A continuación se explica brevemente en qué consiste cada una de ellas.
Las variables **categóricas** son aquellas cuyos valores son del tipo categórico, es decir, que indican categorías, son etiquetas alfanuméricas o "nombres".

- Variables **categóricas nominales**: son las variables categóricas que, además de que sus posibles valores son mutuamente excluyentes entre sí, no tienen alguna forma "natural" de ordenación. Por ejemplo, cuando sus posibles valore son: "sí" y "no". A este tipo de variable le corresponde las escalas de medición nominal.

Las variables **numéricas** toman valores numéricos. A estas variables le corresponde las escalas de medición de intervalo.

- Variables **numéricas discretas**: son las variables que únicamente toman valores enteros o numéricamente fijos. Por ejemplo: las ocasiones en que ocurre un suceso, la cantidad de pesos que se gastan en una semana, los barriles de petróleo producidos por un determinado país, los puntos con que cierra diariamente una bolsa de valores, etcétera.

2.3.4. Trabajo de campo (recolección de los datos)

Se denomina trabajo de campo al conjunto de las acciones que se realizan para obtener los datos que se necesitan, dígase localizar a las personas que deben contestar las preguntas, la gestión y la administración de los cuestionarios o métodos alternativos de recolección de la información deseada.
Se utilizó como metodología para la recolección de los datos a través de encuestas las encuestas personales.

- **Encuestas personales:** implican formular preguntas directas cara a cara a un grupo de personas. Este tipo de encuestas otorga mayor flexibilidad en la obtención de la información, ya que permite gran libertad en cuanto al formato y la longitud del instrumento. Por otro lado, la interacción directa facilita la comprensión de las preguntas, permite aclararlas e introducir variantes (mostrar objetos o fotografías) que otro tipo de encuestas no permite.

2.4. Resultados de la encuesta aplicada a los integrantes de los proyectos productivos de la UCI

Una vez de aplicados los cuestionarios, recolectados y procesados los datos, se realiza el análisis de los resultados. Se observa primeramente que un gran número miembros de los equipos de desarrollo de los diferentes proyectos conocen que es una métrica, sin embargo existe un gran nivel de desinformación en cuanto a su uso en los proyectos y en muy pocos casos las han aplicado en los mismos. Los resultados obtenidos se muestran en los gráficos y tablas que aparecen a continuación.

Estratos	Tamaño de la muestra	%
MINPAL (Facultad 6)	6	16.2
LIMSCC (Facultad 6)	10	15.6
UCID – FAR (Contabilidad y Finanzas)	2	15.4
SIMPRO (Facultad 5)	3	17.6
PDVSA – SCADA (Facultad 5)	4	14.8
ONE (Facultad 3)	5	16.1
APS (Facultad 7)	1	11.1
BioSyS (Facultad 7)	6	14.3
PMF – MINFAR	5	14.7
Registros y Notaría (Facultad 3)	9	15.5
Equipos Médicos (Facultad 6)	2	16.5
Ensayos Clínicos (Facultad 6)	6	15.4

Tabla 3 Por ciento de estudiantes encuestados por proyectos productivos

Figura 14 Grado de conocimiento de los desarrolladores de *software* sobre métricas

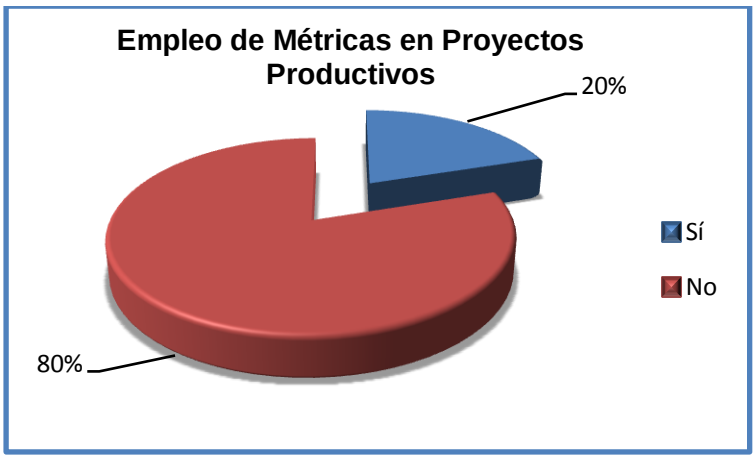


Figura 15 Proporción del empleo de Métricas en los Proyectos Productivos de la UCI

En los resultados obtenidos del análisis de la información recopilada a través de las encuestas queda demostrado el bajo índice de aplicación de las métricas en los Proyectos Productivos de la Universidad. De la muestra seleccionada menos del 16% de los individuos han aplicado métricas alguna vez, aún cuando han participado en el desarrollo de diversos productos de *software*. Sin embargo más del 84% coincide en que es importante el empleo de las mismas en el desarrollo de aplicaciones informáticas como premisa fundamental para garantizar la calidad.

Las preguntas dirigidas a los analistas del sistema aportaron la información correspondiente al tiempo de la captura de los requisitos funcionales en los diferentes proyectos encuestados. Empleando estos datos los proyectos fueron clasificados en dos grupos, pequeños y extensos, los primeros son aquellos que cuentan con un número menor o igual a 30 requisitos funcionales y los extensos sobrepasan la cantidad antes mencionada. Además los requisitos funcionales fueron previamente clasificados en dos grupos: medios, con una longitud menor o igual a 6 palabras y extensos con una longitud mayor a 6 palabras. Partiendo de las clasificaciones anteriores y utilizando la *IEEE Standard for a Software Quality Metrics Methodology* se definen a continuación una serie de métricas de *software* para requisitos funcionales.

2.5. Descripción general de la metodología *IEEE Standard for a Software Quality Metrics Methodology*

Este estándar esta dirigido a aquellos que se encuentran asociados a la adquisición, desarrollo, uso, soporte, mantenimiento y auditoria de *software*. Puede ser empleado por:

- Un líder de proyecto para identificar, definir y priorizar los requerimientos de calidad de un sistema.
- Un desarrollador del sistema para identificar los rasgos específicos que debe tener el *software* en orden de cumplir con los requerimientos de calidad.
- Una organización auditora o aseguradora de la calidad y un sistema desarrollado para evaluar si los requerimientos de calidad han sido cumplidos.
- Un encargado del mantenimiento del sistema para asistir en las modificaciones de implementación durante la evolución del producto.
- Un usuario para asistir en la especificación de los requisitos de calidad para el sistema.

Específicamente el uso de esta metodología estándar para medir la calidad permite a la organización:

- Evaluar el logro de las metas de calidad.
- Establecer los requerimientos de calidad para un sistema a su salida.
- Establecer estándares y criterios de aceptación.
- Evaluar el nivel de calidad alcanzado contra los requerimientos requeridos.
- Detectar anomalías o puntos de problemas potenciales en el sistema.
- Predecir el nivel de calidad que será alcanzado en el futuro.
- Monitorear los cambios en la calidad del *software* si este es modificado.
- Evaluar la facilidad de cambio en el sistema durante la evolución del producto.

2.5.1. Metodología para definir métricas de calidad de *software*

La metodología para métricas de calidad de *software* es un acercamiento sistemático al establecimiento de requerimientos de calidad e identificación, implementación, análisis y validación de métricas de procesos y productos de *software* para un sistema de *software*. Comprende cinco pasos.

Estos pueden ser aplicados iterativamente por la visión ganada de la aplicación de un paso que puede mostrar la necesidad de evaluación de los resultados de pasos previos. Cada paso establece las actividades necesarias para completar los resultados indicados.

1. Establecer los requerimientos de calidad de *software*

El resultado de este paso es una lista de los requerimientos de calidad. Las actividades para completar este resultado se muestran desde **a** hasta **c.**

a. Identificar una lista de posibles requerimientos de calidad

Identificar los requerimientos de calidad que son aplicables al sistema de *software*. Se emplea la experiencia organizacional, estándares, regulaciones o leyes para crear esta lista. Adicionalmente, se listan otros requerimientos del sistema que puedan afectar la viabilidad de los requerimientos de calidad. Se consideran requerimientos contractuales e implicaciones, como costo o planificación, garantías, requerimientos de métricas del cliente e intereses propios de la organización.

No deben regularse mutuamente requerimientos exclusivos en esta etapa. Es mejor enfocarse en la calidad de combinaciones factores/métricas directas en vez de métricas predictivas. Se debe asegurar que todos los involucrados en la creación y uso del sistema participen en el proceso de identificación de requerimientos de calidad.

b. Determinar la lista de requerimientos de calidad

Cada uno de los requisitos de calidad listados por la importancia debe ser tasado. La lista de posibles requerimientos de calidad se determina según los aspectos siguientes:

Inspeccione todas las partes involucradas:

Las prioridades relativas de los requerimientos se discuten con todos los involucrados. Se obtiene el peso de cada grupo de requisitos de calidad contra los requisitos del sistema. Se asegura que todos los puntos de vista son considerados.

Cree la lista de requerimientos de calidad:

Los resultados del estudio son resumidos en una lista simple de requerimientos de calidad. Los factores de calidad propuestos por esta lista deben tener relaciones conflictivas o cooperativas. Cualquier conflicto relacionado con los requerimientos ha de ser resuelto. Adicionalmente, si la opción de los requerimientos de calidad entra en conflicto con el costo, planificación o funcionalidad del sistema, debe alterarse uno u otro. Hay que tener cuidado al elegir la lista deseada para asegurar que los requerimientos son técnicamente viables, razonables, complementarios, alcanzables y verificables. Es importante la obtención del acuerdo de todas las partes con esta lista final.

c. Cuantifique cada factor de calidad

Para cada factor de calidad se asigna una o más métricas directas que lo representen y valores de métricas directas que sirvan de requerimientos cuantitativos a dicho factor de calidad.

Por ejemplo si “eficiencia alta” es uno de los requerimientos de calidad debe emplearse una métrica directa como por ejemplo: uso actual de recursos/uso de recursos asignados con un valor del 90%. Emplear métricas directas para verificar el logro de los requerimientos de calidad. La lista cuantificada de requerimientos de calidad y sus definiciones es aprobada nuevamente por todos los involucrados.

2. Identificar métricas de *software*

El resultado de este paso es un grupo de métricas aprobadas. Las actividades para lograr este resultado se explican desde **a** hasta **b**.

a. Aplicar el marco de trabajo de las métricas de calidad

La lista cuantitativa de los requerimientos cualitativos y sus definiciones es aprobada nuevamente por todos. Crear un mapa de los requerimientos de calidad basado en la estructura de árbol jerárquico mostrado en el Anexo 1.

En este punto se completa sólo el nivel de factores cualitativos. Luego descomponer cada factor de calidad en subfactores. La descomposición en subfactores continúa cuantos niveles sean necesarios hasta que el nivel de subfactores de calidad esté completo.

Empleando el marco de trabajo de las métricas de *software*, descomponer los subfactores de calidad en métricas. Por cada métrica validada en el nivel de métricas, asignar un valor objetivo, un valor crítico y un rango que debe ser alcanzado durante el desarrollo. El marco de trabajo debe ser revisado por todas las partes involucradas. Se emplean solo las métricas validadas, métricas directas o métricas validadas respecto a métricas directas para evaluar la calidad del producto actual y futuro. Se retienen las métricas no validadas como candidatas a análisis futuros. Además, se emplean solo métricas que están asociadas con los requerimientos de calidad del proyecto de *software*. Cada métrica debe documentarse de acuerdo al formato de la tabla del Anexo 2.

b. Realice un análisis costo-beneficio

Identificar el costo de aplicar la métrica

Identificar y documentar todos los costos asociados a cada métrica. Para cada métrica estimar y documentar los costos e impactos siguientes:

- Costos de utilización de la métrica en los que se incurre durante la recolección de datos, cálculos automáticos de métricas, aplicación, interpretación y reporte de resultados.
- Costos de cambios en el *software* provocados por cambios en el proceso de desarrollo.
- Costos de los cambios en la estructura organizacional provocada por la producción de *software*.
- Equipamiento especial para implementar el plan de métricas.
- Entrenamiento para implementar el plan de métricas.

En la presente investigación no se hace análisis del costo, pues este paso se hace cuando se trabaja con un proyecto en específico, no para casos generales como el Sistema de Métricas que se propone.

Identificar los beneficios de la aplicación de las métricas

Identificar y documentar los beneficios asociados a cada métrica. Algunos beneficios pueden ser los siguientes:

- Identificar las metas de calidad e incrementar el conocimiento sobre la misma.
- Proveer una retroalimentación sistemática de problemas de calidad al proceso de desarrollo.
- Incrementar la satisfacción del cliente cuantificando la calidad del *software* antes de su entrega.
- Proveer una base cuantitativa para la toma de decisiones sobre la calidad del *software*.
- Reducir los costos del ciclo de vida del *software* incrementando la eficiencia del proceso.

c. Ganar conocimiento sobre las métricas

Todos los involucrados revisan las métricas ajustadas. Las métricas se adoptan y sustentan por este grupo.

3. Implementar el sistema de métricas de *software*

La salida de este paso es una descripción de los datos, un plan de entrenamiento y una planificación. Las actividades para lograr este resultado se muestran desde **a** hasta **b**.

a. Definir procesos de recolección de datos

Para cada métrica en el grupo de métricas, se determinan los datos que serán recolectados y las asunciones que se harán acerca de los mismos (por ejemplo: muestra aleatoria y medidas objetivas o subjetivas). Mostrar el flujo de datos del punto de recolección para la evaluación de las métricas. Se identifican las herramientas y se describe cómo deben ser utilizadas. Describir los procedimientos de almacenamiento de datos.

Establecer una matriz de traceabilidad entre métricas y datos. Identificar las entidades organizacionales que participaran en la recolección de datos, incluyendo aquellos responsables del monitoreo de la recolección de datos. Escribir el entrenamiento y la experiencia requerida para la recolección de datos y el proceso de entrenamiento para el personal involucrado.

b. Prototipo del proceso de medición

Probar la recolección de datos y el procedimiento de cálculo de métricas en el *software* seleccionado que actuará como prototipo. Seleccionar muestras similares al proyecto en el que las métricas serán empleadas. Hacer un análisis para determinar si los datos son recolectados uniformemente y si las instrucciones han sido interpretadas consistentemente. En particular verificar qué requiere un juicio subjetivo para determinar si las descripciones e instrucciones son lo suficientemente claras para asegurar resultados uniformes.

2.6. Definición de métricas de calidad de *software*. Propuesta de sistema de métricas para requisitos funcionales

1. Requerimientos de calidad de *software*

Los requerimientos de calidad necesarios para la creación de un sistema de métricas para requisitos funcionales están asociados a la etapa de levantamiento de requisitos del *software*, específicamente aplicables a la confección de la documentación correspondiente a esta etapa del proyecto. A continuación se listan los mismos.

2. Lista de los requerimientos de calidad

RC1. Tiempo de duración del levantamiento de requisitos.
RC2. Redacción de los requisitos funcionales en cuanto a longitud.
RC3. Redacción de los requisitos funcionales en cuanto a composición de los mismos.
RC4. Requisitos funcionales correctos e inequívocos.

RC5. Requisitos funcionales consistentes y comprobables.

RC6. Requisitos funcionales modificables e identificables.

RC7. Ambigüedad de los requisitos funcionales.

3. Métricas de *software*. Marco de trabajo de las métricas de calidad

A continuación se muestra un mapa de los requerimientos de calidad basado en la estructura de árbol jerárquico mostrando los requerimientos de calidad, los subfactores de calidad correspondiente a cada requerimiento y las métricas resultantes.

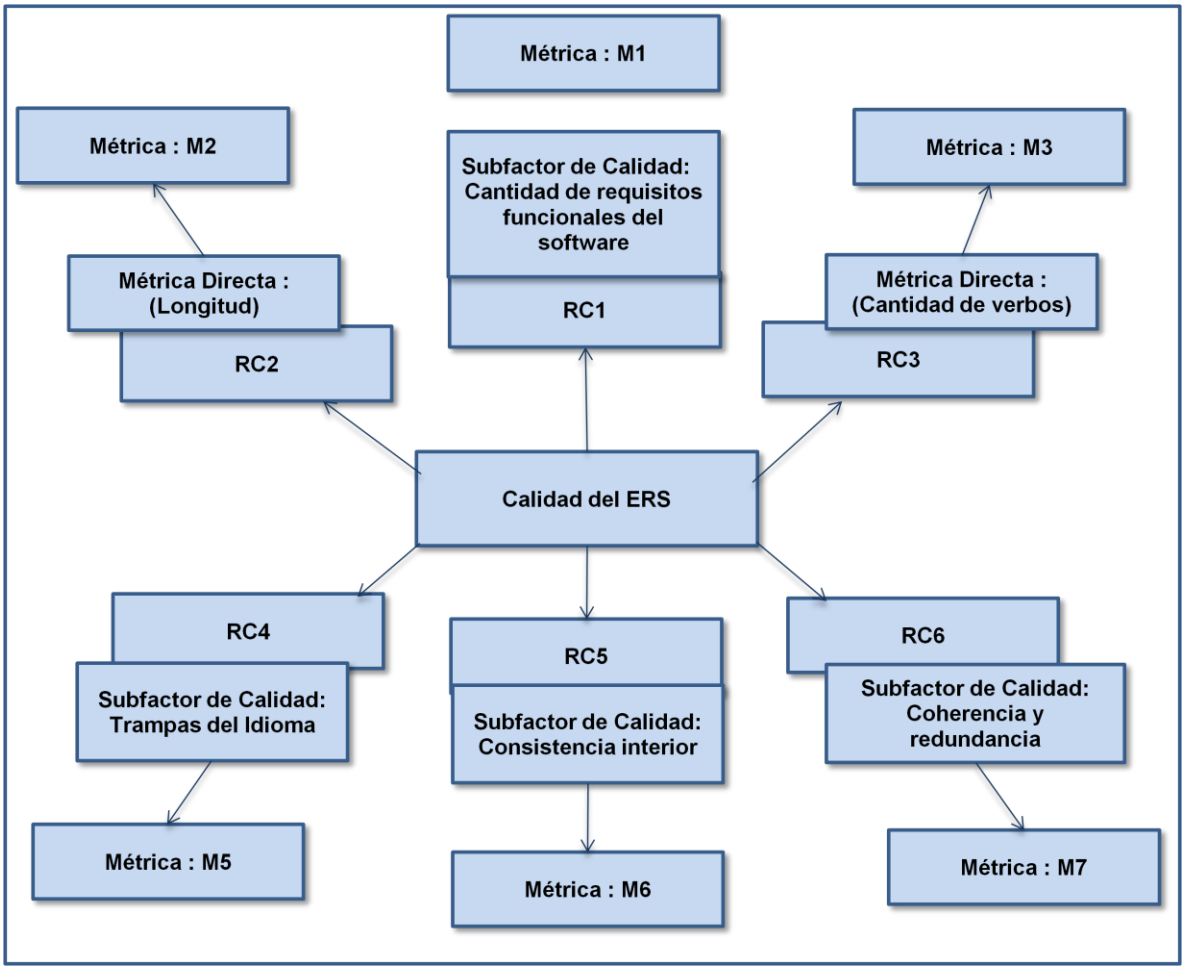


Figura 16 Marco de trabajo de los requerimientos de calidad

Listado de métricas obtenidas

M1. Estimación del tiempo de duración del levantamiento de requisitos.
M2. Longitud de los requisitos funcionales.
M3. Composición de los requisitos funcionales.
M4. El ERS tiene que ser correcto e inequívoco.
M5. El ERS tiene que ser consistente y comprobable.
M6. El ERS tiene que ser modificable e identificable.

En el mapa de trabajo de las métricas de *software* se han descompuesto los subfactores de calidad en métricas. Por cada métrica validada en el nivel de métricas, se asignó un valor objetivo, un valor crítico y un rango que debe ser alcanzado durante el desarrollo. (Estos datos se muestran en el siguiente epígrafe).

Solo se emplearon las métricas validadas (métricas directas o métricas validadas respecto a métricas directas) para conformar métricas cuantitativas; como es el caso de M1 y M2. Las métricas desde M3 a M7 no fueron validadas quedando como candidatas a análisis futuros.

4. Documentación de las métricas cuantitativas obtenidas

En este epígrafe cada métrica es documentada utilizando el formato de las tablas que aparecen a continuación. Se documentaron las métricas “Estimación del tiempo de duración del levantamiento de requisitos” y “Longitud de los requisitos funcionales”. En las tablas se muestra toda la información necesaria e indispensable que caracteriza a cada métrica, su beneficio, valor objetivo, interpretación, entre otros.

´

Elemento	Descripción
Nombre	Estimación del tiempo de duración del levantamiento de requisitos.
Beneficio	Poder estimar aproximadamente un tiempo de duración de la etapa de levantamiento de requisitos.
Impacto	No define la calidad final del producto.
Valor Objetivo	1 < T < 5, 5 < T < 12 semanas.
Factores de Calidad	Tiempo de duración del levantamiento de requisitos.
Herramientas	No existe ninguna herramienta.
Datos	Cantidad de Requisitos Funcionales.
Interpretación	Compara el resultado de la aplicación de la métrica con el valor objetivo.
Entrenamiento Requerido	No se requiere de entrenamiento previo.

Tabla 4 Descripción textual de la métrica "Estimación del tiempo de duración del levantamiento de requisitos"

Elemento	Descripción
Nombre	Longitud de los requisitos funcionales.
Beneficio	Poder corregir la redacción de los requisitos funcionales de acuerdo a su longitud.
Impacto	Evita errores de redacción que pueden afectar el desarrollo de fases posteriores.
Valor Objetivo	L<10
Factores de Calidad	Redacción de los requisitos funcionales en cuanto a longitud.
Herramientas	No existe ninguna herramienta.
Datos	Cantidad de palabras del requisito funcional.
Interpretación	Compara el resultado de la aplicación de la métrica con el valor objetivo.
Entrenamiento Requerido	No se requiere de entrenamiento previo.

Tabla 5 Descripción textual de la métrica "Longitud de los requisitos funcionales"

5. Implementar el sistema de métricas de *software*

❖ Tiempo de duración del levantamiento de requisitos

Esta métrica orientada a estimar el tiempo que deberá durar el levantamiento de requisitos de un proyecto determinado, se emplea utilizando la cantidad de requisitos funcionales del proyecto para determinar si es pequeño o extenso y luego según el tipo de proyecto se escoge el rango de tiempo correspondiente.

Sea:

TE

Donde:

TE: Tiempo que demora el levantamiento de requisitos.

P: Un proyecto determinado.

PP: Proyectos pequeños. (Menores de 30 requisitos funcionales)

PE: Proyectos extensos. (Mayores de 30 requisitos funcionales)

Para determinar TE

Si **P Є PP** entonces 2 < **TE** < 4 semanas.

Si **P Є PE** entonces 4 < **TE** < 8 semanas.

❖ Longitud de los requisitos funcionales

Sea:

L: Longitud del requisito funcional.

Para determinar L:

Se cuentan las palabras del requisito funcional sin tener en cuenta artículos, preposiciones o conjunciones.

Para determinar si el requisito es correcto:

Si **L < 6** es correcto y es un requisito medio.

Si **L > 6** es correcto y es un requisito extenso.

Si **L > 10** es incorrecto.

Si el requisito es incorrecto debe reconsiderarse su redacción. Debe tenerse en cuenta como posible solución la división del requisito en partes, conformando otros más pequeños que detallen la funcionalidad en general. De esta forma puede hacerse más comprensible para todos los involucrados en el proyecto.

6. Resultados del estudio realizado a los modelos de calidad

Después de un estudio profundo de los modelos de calidad existentes se seleccionó de éstos información para definir las métricas que podían ser útiles en la confección del sistema que se

propone. Estas fueron en algunos casos adaptadas a las condiciones específicas de la producción en la UCI. De este proceso se obtienen las métricas siguientes:

- Composición del requisito funcional.
- El ERS tiene que ser correcto e inequívoco.
- El ERS tiene que ser consistente y comprobable.
- El ERS tiene que ser modificable e identificable.
- Los requisitos deben ser específicos.

❖ Composición del requisito funcional

Los requisitos funcionales no pueden estar compuestos por más de un verbo y este debe de estar en infinitivo.

❖ El ERS tiene que ser correcto e inequívoco

Correcto

Un ERS es correcto si, y sólo si, todos los requisitos se encuentran declarados en el *software*. No hay herramienta o procedimiento que asegure la exactitud, por lo que alternativamente el cliente o el usuario pueden determinar si el requisito refleja la necesidad real correctamente. Identificar los requerimientos hace este procedimiento más fácil y hay menos probabilidad de error.

Inequívoco

Un SRS es inequívoco si, y sólo si, cada requisito declarado tiene sólo una interpretación. Como mínimo, se requiere que cada característica de la última versión del producto se describa usando un único término. En casos dónde un término en un contexto particular tenga significados múltiples, el término debe ser incluido en un glosario donde su significado es hecho más específico. Un SRS es una parte importante del proceso de requisitos del ciclo de vida de *software* y se usa en el diseño, aplicación, supervisión, comprobación, aprobación y

pruebas. El ERS debe ser inequívoco para aquéllos que lo crean y para aquéllos que lo usan. Sin embargo, estos grupos no tienen a menudo el mismo fondo y por consiguiente no tienden a describir los requisitos del *software* de la misma manera.

- ***Trampas del Idioma***

Los requisitos son a menudo escritos en el idioma natural (por ejemplo, inglés) que resulta inherentemente ambiguo. El ERS podría ser revisado por una parte independiente para identificar el uso ambiguo del idioma para que pueda corregirse.

- ***Representación hecha con herramientas***

En general, los métodos de requisitos e idiomas y las herramientas que los apoyan entran en tres categorías generales: objeto, procesos y conductual. El término objetos-orientados organiza los requisitos en lo que se refiere a los objetos en el mundo real, sus atributos, y los servicios realizados por esos objetos. El término procesos organiza los requisitos en las jerarquías de funciones que comunican el flujo de datos. Los términos conductuales describen la conducta externa del sistema por lo que se refiere a alguna noción de lo abstracto, las funciones matemáticas o el estado de las máquinas. El grado en que se usan estas herramientas y los métodos puede ser útil preparando un ERS pero depende del tamaño y complejidad del programa. Aún usando cualquiera de estos términos es mejor retener las descripciones en el idioma natural. Así el cliente poco familiar con las anotaciones del ERS podrá entenderlo.

❖ El ERS tiene que ser consistente y comprobable

Consistente

La consistencia se refiere a la consistencia interior

- ***Consistencia interior***

Un ERS es internamente consistente si, y sólo si, ningún subconjunto de requisitos individuales generó conflicto en él.
Los tres tipos de conflictos probables en un ERS son:
a) Las características especificadas en el mundo real de los objetos pueden chocar. Por ejemplo:

- El formato de un informe del rendimiento puede describirse en un requisito como tabular pero en otro como textual.
- Un requisito puede declarar que todas las luces serán verdes mientras otro puede declarar que todas las luces sean azules.

b) Puede haber conflicto lógico o temporal entre dos acciones especificadas. Por ejemplo:

- Un requisito puede especificar que el programa sumará dos entradas y otro puede especificar que el programa los multiplicará.
- Un requisito puede declarar que "A" siempre debe seguir "B", mientras otro puede requerir que "A" y B" ocurran simultáneamente.

c) Dos o más requisitos pueden describir el mismo mundo real del objeto pero con diferente uso de las condiciones para ese objeto. Por ejemplo, una demanda del programa para una entrada del usuario puede llamarse una "sugerencia" en un requisito y una "señal" en otro. El uso de terminología normal y definiciones promueve la consistencia.

Comprobable

Un ERS es comprobable si, y sólo si, cada requisito declarado es comprobable. Un requisito es comprobable si, y sólo si, allí existe algún proceso rentable finito con que una persona o máquina puede verificar que el producto del *software* reúne el requisito. En general cualquier requisito ambiguo no es comprobable. Los requisitos de No-verificable incluyen las declaraciones como "trabaja bien", "interface humana buena" y "normalmente pasará" no pueden verificarse los requisitos de esos porque es imposible de definir las condiciones "bueno," "bien" o "normalmente". La declaración que "el programa nunca entrará en una

vuelta infinita" es el no-verificable porque la comprobación de esta calidad es teóricamente imposible.

Un ejemplo de una declaración comprobable es: El rendimiento del programa se producirá dentro de 20 segundos de evento 60% del tiempo; y se producirá dentro de 30 segundos de evento 100% del tiempo. Esta declaración puede verificarse porque usa condiciones concretas y las cantidades mensurables. Si un método no puede inventarse para determinar si el *software* reúne un requisito particular, entonces ese requisito debe quitarse o debe revisarse.

- **El ERS tiene que ser modificable e identificable**

Modificable

Un ERS es modificable si, y sólo si, su estructura y estilo son tales que puede hacerse cualquier cambio a los requisitos fácilmente, completamente y de forma consistente mientras conserva la estructura y estilo. Para que sea modificable se requiere un ERS que contenga:

a) Una estructura coherente y fácil de usar en la organización de volúmenes de información, un índice y las referencias cruzadas explícitas.

b) No es redundante (es decir, el mismo requisito no debe aparecer en más de un lugar en el ERS).

c) Cada requisito expresado separadamente, en lugar de intercalado con otros requisitos. La redundancia no es un error, pero puede llevar fácilmente a los errores. La redundancia puede ayudar hacer un ERS más legible de vez en cuando, pero un problema puede generarse cuando el documento redundante se actualiza. Por ejemplo, un requisito puede alterarse en un solo lugar donde aparece. El ERS se vuelve incoherente entonces. Siempre que la redundancia sea necesaria, el ERS debe incluir la cruz explícita - las referencias para hacerlo modificable.

Identificable

Un ERS es identificable si el origen de cada uno de sus requisitos está claro y si facilita las referencias de cada requisito en el desarrollo futuro o documentación del mismo. Se recomiendan dos tipos de identificabilidad:

- El identificable dirigido hacia atrás (es decir, a las fases anteriores de desarrollo). Esto depende explícitamente en cada requisito de las referencias de su fuente en los documentos más antiguos.
-

a) El identificable delantero (es decir, a todos los documentos provenientes del ERS). Esto depende de que cada requisito en el ERS tenga un único nombre o número de la referencia.

El identificable delantero del ERS es especialmente importante cuando el producto del *software* entra en funcionamiento y en la fase de mantenimiento. Como el código y los documentos del plan se modifican, es esencial poder determinar el juego completo de requisitos que pueden afectarse por esas modificaciones.

❖ Los requisitos deben ser específicos. (Métrica de Davis)

Los requisitos deben ser específicos, es decir, no deben estar redactados de forma ambigua. Para conocer si el requisito tiene esta característica o no es necesario que sea interpretado por diferentes revisores.

Estos revisores deben ser aquellos que desempeñan dicho rol en el proyecto, otros miembros del equipo de desarrollo aunque no hayan participado en la fase de levantamiento de requisitos y además el o algunos de los clientes que solicitan el *software* formando así un equipo de revisión. Cada miembro del equipo analizará y dará su interpretación del mismo. Si cada revisor llega a la misma conclusión entonces el requisito no es ambiguo, en caso contrario debe modificarse su redacción cuantas veces sea necesario.

Sea:

Nr = Nf + Nnf

Donde:

Nr: Cantidad de requisitos.

Nf: Cantidad de requisitos funcionales.

Nnf: Cantidad de requisitos no funcionales.

Para determinar la especificidad de los requisitos:

Sea:

E = Nui/Nr

Donde:

E: Grado de especificidad.

Nui: Cantidad de requisitos que fueron interpretados de igual forma por los revisores.

Nr: Cantidad de requisitos.

La variable **E** tomará valores en el rango **0 – 1**.

El ERS tienen menor especificidad si **E** es **0** o se aproxima a él.

El ERS tiene mayor especificidad si **E** es **1** o se aproxima a él.

2.7. Tabla Resumen del Sistema de Métricas para Requisitos Funcionales Propuestos

Métrica	Descripción
Tiempo de duración del levantamiento de requisitos.	Métrica para estimar el tiempo que debe durar un proyecto en el levantamiento de requisitos.
Longitud de los requisitos funcionales.	Métrica para determinar si el requisito esta bien redactado en cuanto a la longitud de los mismos.
Composición del requisito funcional.	Métrica para determinar si el requisito esta bien redactado en cuanto a la composición de los mismos.
ERS correcto e inequívoco.	Métrica para determinar si el ERS es correcto e inequívoco.
ERS consistente y comprobable.	Métrica para determinar si el ERS es consistente y comprobable.
ERS modificable e identificable.	Métrica para determinar si el ERS es modificable e identificable.
Especificidad del ERS (Métrica de Davis).	Métrica definida por Davis utilizada para medir especificidad.

Tabla 6 Resumen del Sistema de Métricas para Requisitos Funcionales propuesto

2.8. Conclusiones del capítulo

En este capitulo se hace un análisis de los proyectos productivos en cuanto al uso de métricas y en específico métricas orientadas a requisitos funcionales. Se describe la metodología utilizada para la confección de métricas y se definen a su vez 7 métricas para conformar el Sistema de Métricas para Requisitos Funcionales propuesto.

Capítulo 3
Validación del Sistema de Métricas

3.1. Introducción

Este capítulo tiene como objetivo realizar la validación del sistema de métricas propuesto en el Capítulo II. Para realizar esta actividad se aplicó el Sistema de Métricas a un proyecto real y se apoyó la validación utilizando el método Criterios de expertos para obtener información de especialistas acerca de la utilidad y posible eficiencia del Sistema de Métricas para Requisitos Funcionales propuesto en la presente investigación.

Para la evaluación futura de un proyecto mediante el sistema que se propone se describe la IEEE *Standard for a Software Quality Metrics Methodology* como metodología para definir y validar las métricas que lo componen.

3.2. Criterio de expertos: Método Delphi

El método *Delphi* consiste en la selección de un grupo de expertos a los que se les pregunta su opinión sobre cuestiones referidas a acontecimientos del futuro. Las estimaciones de los expertos se realizan en sucesivas rondas, anónimas, con el objeto de tratar de conseguir un consenso, pero con la máxima autonomía por parte de los participantes.

Las preguntas se refieren, por ejemplo, a las probabilidades de realización de hipótesis o de acontecimientos con relación al tema de estudio (que en este caso sería el desarrollo futuro del Sistema de Métricas para Requisitos Funcionales). La calidad de los resultados depende, sobre todo, del cuidado que se ponga en la elaboración del cuestionario y en la elección de los expertos consultados.

Se considera la utilización de este método dada la existencia de las siguientes condiciones:

- No existen datos históricos con los que trabajar.
- El impacto de los factores externos tiene más influencia en la evolución que el de los internos.
- De acuerdo al problema no se facilita el uso de una técnica analítica precisa.
- Se desea mantener la heterogeneidad de los participantes a fin de asegurar la validez de los resultados.

3.2.1. Fases del método

Previo a la aplicación del método:

- Delimitar el contexto y el horizonte temporal, en el que se desea realizar la previsión sobre el tema de estudio.
- Seleccionar el panel de expertos y conseguir su compromiso de colaboración.
- Explicar a los expertos en qué consiste el método.

Fase 1. Definición de objetivos. En esta primera fase se plantea la formulación del problema y un objetivo general. El objetivo general que se persigue es conocer si está correcto el Sistema de Métricas para Requisitos Funcionales propuesto.

Fase 2. Selección de expertos. Esta fase presenta dos dimensiones:

- **Dimensión cualitativa:** Se seleccionan en función del objetivo prefijado y atendiendo a criterios de experiencia, posición, responsabilidad, acceso a la información y disponibilidad. Dadas las características del sistema que se propone, se seleccionaron especialistas de calidad que se desempeñan en diferentes responsabilidades en los proyectos productivos de la universidad y dominen las siguientes temáticas: Procesos de desarrollo de *Software*, Ingeniería de *Software*, Ingeniería de Requisitos y Estándares Internacionales de Calidad.

- **Dimensión Cuantitativa:** Elección del tamaño de la muestra en función de los recursos medios y tiempo disponible. Se tomó como muestra 7 expertos de calidad, que es el mínimo requerido por el método, en función del tiempo limitado para el desarrollo de la investigación.

Para comprobar si la selección de los expertos es correcta, se emplea la valoración por competencias. Este método consiste en la obtención del coeficiente de competencia (k) del experto a partir del resultado de la autovaloración del mismo, sobre su conocimiento o información sobre el tema (kc) y el coeficiente de argumentación o valoración (ka) mediante la siguiente ecuación.

$k = (kc + ka)/2$

El código de interpretación de los coeficientes de competencias es como sigue:

Si $0,8 < k < 1,0$ coeficiente de competencia alto.
Si $0,5 < k < 0,8$ coeficiente de competencia medio
Si $k < 0,5$ coeficiente de competencia bajo

Experto	Kc	Ka	k	Clasificación
E1	0.6	0.9	0.75	Medio
E2	0.6	1.0	0.80	Alto
E3	0.6	0.7	0.65	Medio
E4	0.4	0.6	0.50	Medio
E5	0.6	0.9	0.75	Medio
E6	0.8	0.9	0.85	Alto
E7	0.8	0.8	0.80	Alto

Tabla 7 Resultados de valoración por competencia

Fase 3. Elaboración y lanzamiento de los cuestionarios. Estos se elaboran de manera que faciliten la respuesta por parte de los encuestados. Las respuestas habrán de ser cuantificadas y ponderadas. Ver Anexo 4.

Fase 4. Explotación de resultados. El objetivo de los cuestionarios sucesivos es disminuir la dispersión y precisar la opinión media consensuada. En este caso no fue necesario realizar más de una iteración

en la aplicación de los cuestionarios porque las respuestas coincidieron en más de un 95 % a favor de la aprobación del sistema de métricas propuesto.

3.3. Análisis de las encuestas realizadas a los expertos de calidad

Con el objetivo de validar el Sistema de Métricas para Requisitos Funcionales propuesto, los expertos encuestados realizaron un estudio previo del mismo. Para poder emitir los diferentes criterios con respecto a las interrogantes que se muestran en el cuestionario del Anexo 4. A partir del análisis de los datos recolectados se obtuvieron los resultados siguientes:

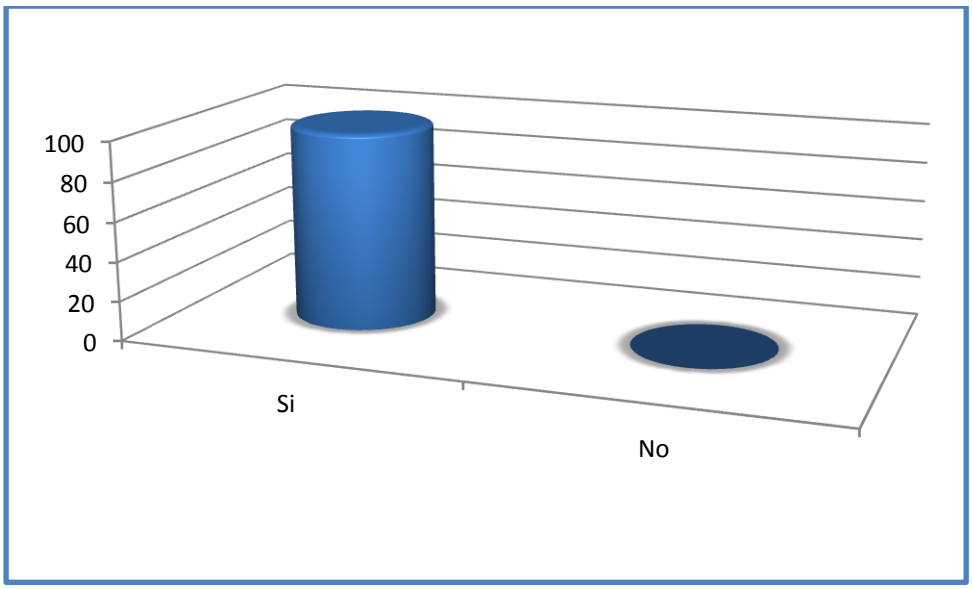


Figura 17 Opinión sobre importancia de la aplicación del Sistema de Métricas

Como se muestra en el gráfico todos los expertos coinciden en que el sistema de métricas es importante para optimizar el proceso de levantamiento de requisitos, lográndose una mejor planificación en los proyectos y aumentando la calidad en esta etapa de desarrollo del *software*; posibilitando además un mejor entendimiento entre los desarrolladores y los clientes.

A continuación aparece el grado de eficacia de la aplicación del sistema de métricas en los proyectos productivos de la UCI de acuerdo a la valoración realizada por los especialistas.

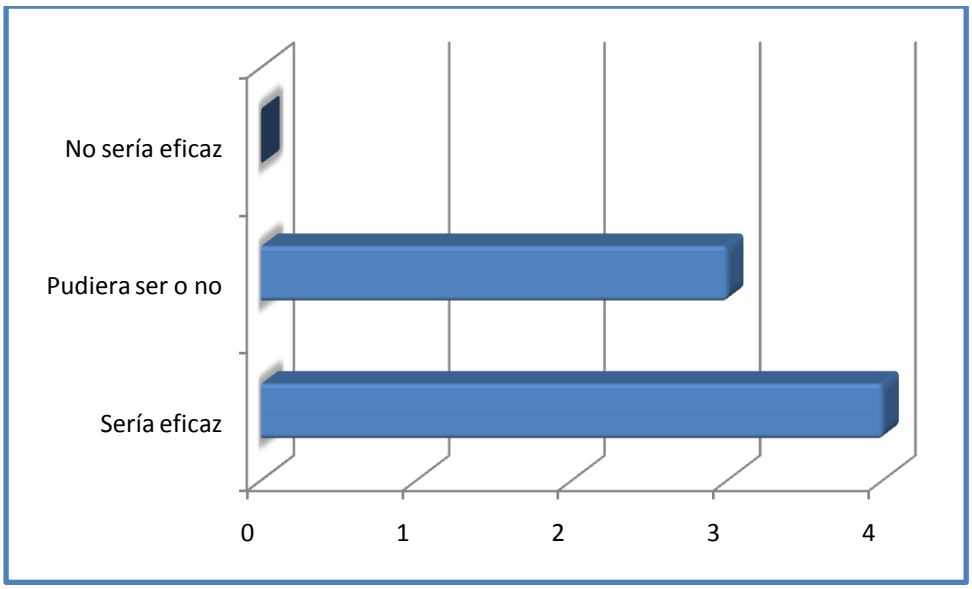


Figura 18 Opinión sobre posible éxito del Sistema de Métricas

Finalmente todos coinciden en que la aplicación del sistema de métricas propuesto podría ser exitosa debido a la necesidad que existe en estos momentos de mejorar la calidad de los proyectos productivos desde las primeras fases del ciclo de vida del *software*.

3.4. Opiniones de los Especialistas

Ing. Martha Nieves Borrero.
Calidad Facultad 6.
Profesor (a) Adiestrado (a)
1 año de experiencia

Considero importante la aplicación del Sistema de Métricas para Requisitos Funcionales porque es la única forma de obtener datos históricos reales y poder mejorar el proceso de desarrollo. Pienso que puede aplicarse exitosamente pues todos los proyectos productivos lo necesitan. Además las métricas son completamente reales.

Ing. Heney Díaz Pérez.
Especialista de Calidad de *Software*.
Profesor (a) Adiestrado (a)

Creo que es importante la aplicación de un Sistema de Métricas para Requisitos Funcionales pues tributa a la optimización y mejor ejecución de un levantamiento de requisitos. Su aplicación puede ser exitosa pues el tiempo y esfuerzos requeridos son mínimos.

Ing. Maikel Castro Pérez.
Especialista de Calidad de *Software*.
Profesor (a) Adiestrado (a)

Es importante la aplicación del Sistema de Métricas para Requisitos Funcionales ya que contribuiría a una mejor captura, definición y refinamiento de los requisitos funcionales antes de comenzar a desarrollar el sistema. Puede aplicarse exitosamente porque se estaría realizando la captura de requisitos de forma estadística y de acuerdo con técnicas que permitirían asegurar la calidad desde edades tempranas del desarrollo de *software*. Las métricas seleccionadas son correctas y ayudarían a obtener requisitos concretos que ayudarían al desarrollador a comprender mejor cómo el cliente desea que se desarrolle su aplicación.

Ing. Saumel Tejeda Diaz.
Líder de Proyecto.
Profesor(a) Asistente
2 años de experiencia

Es importante la aplicación del Sistema de Métricas para Requisitos Funcionales porque se lograría una mejor planificación en el proyecto y calidad en la formulación de los requisitos. Puede aplicarse exitosamente porque sería bueno que los proyectos que empiezan realizaran una estimación del tiempo de levantamiento de requisitos para lograr una mejor planificación.

Ing. Delvis Echevarría Pérez.
Aseguramiento de la Calidad de *Software*.
Profesor (a) Adiestrado (a)

La aplicación de un Sistema de Métricas para Requisitos Funcionales es de gran importancia para la exactitud de los mismos. Pienso que se puede aplicar exitosamente porque para los requisitos no existen métricas, y podrían ayudar a una mayor eficiencia de los mismos.

3.5. Aplicación del Sistema de Métricas a un proyecto productivo

El Sistema de Métricas fue aplicado a un proyecto productivo de importancia en la facultad con el objetivo de validar la fiabilidad del mismo e identificar posibles errores en su funcionamiento. De este proceso se obtuvieron los resultados que se muestran a continuación.

Métrica	Valor
1. Tiempo de duración del levantamiento de requisitos.	5 semanas
2. Longitud de los requisitos funcionales.	Todos medios
3. Composición del requisito funcional.	Todos correctos
4. ERS correcto e inequívoco.	Sí
5. ERS consistente y comprobable.	Sí
6. ERS identificable y modificable.	Sí
7. Especificidad del ERS (Métrica de Davis).	0.96

Tabla 8 Resultados de validación

El proyecto ya había sido evaluado por calidad anteriormente por lo que se conocía que se encontraba correcta la documentación. El mismo era extenso contando con una cantidad de 137 requisitos funcionales, en la documentación correspondiente al levantamiento de requisitos, que demoró 5 semanas aproximadamente según los datos aportados por los analistas del sistema. Se puede

observar como este tiempo se ajusta al rango que se especifica en la métrica "Tiempo de duración del levantamiento de requisitos". Los requisitos funcionales cuentan además con una correcta redacción en cuanto a longitud y composición de los mismos. Todos los requisitos se encuentran en la categoría de "medios" por lo que se aprecia la importancia de redactar requisitos con longitudes pequeñas para garantizar su precisión.

El ERS resultó correcto, inequívoco, consistente, comprobable, identificable y modificable, y la especificidad calculada según la métrica de Davis es de un 0.96, muy próximo a 1, por lo que se considera un alto grado de especificidad. Los resultados positivos obtenidos apoyan la comprobación de la validez del Sistema de Métricas propuesto.

3.6. Descripción general de la metodología para validar métricas *IEEE Standard for a Software Quality Metrics Methodology*

En este epígrafe se describe la metodología *IEEE Standard for a Software Quality Metrics Methodology* para validar las métricas propuestas en el Capítulo II.

3.6.1. Recolectar datos y calcular la métrica

Mediante el formato que se brinda en las tablas de los Anexos 2 y 3, recolecte y almacene datos en la base de datos de métricas del proyecto en el tiempo adecuado en el ciclo de vida. Verifique los datos en cuanto a exactitud y la unidad de medida apropiada.

Monitorear la recolección de datos. Si una muestra de datos es utilizada, verificar requerimientos como aleatoriedad, tamaño mínimo de la muestra, muestras homogéneas conocidas. Verifique la uniformidad de los datos si más de una persona los recolecta. Calcule los valores de la métrica a partir de los datos obtenidos.

1. **Analizar los resultados de las métricas de *software***

   Los resultados de este paso son cambios en la organización y en el proceso de desarrollo que son indicados por la interpretación y el empleo de los datos de medición. Las actividades para lograr este resultado se dan desde **a** hasta **b**.

a. Interpretar los resultados

Interpretar y recolectar los resultados. Analizar las diferencias entre los datos recolectados de las métricas y los valores objetivos. Investigar las diferencias significativas.

b. Identificar la calidad del *software*

Identificar y revisar los valores de las métricas de calidad para los componentes de *software*. Identificar valores de las métricas que están fuera de los intervalos anticipados de tolerancia (baja o inesperadamente alta calidad) para estudios posteriores.

La calidad inaceptable puede ser manifestada como una complejidad excesiva, documentación inadecuada, poca traceabilidad, u otros atributos indeseables. La existencia de condiciones como éstas, indican que el *software* puede no satisfacer los requerimientos de calidad cuando se vuelva operacional. Muchas de las métricas directas que usualmente son de interés, no pueden ser recolectadas durante el desarrollo del *software* (métricas de confiabilidad). Las métricas de validación se emplean cuando las métricas directas no estén disponibles. Se emplean métricas directas o validadas para pasos de componentes y procesos de *software*.

Comparar los valores de las métricas con los valores críticos de las métricas. Analizar en detalle los componentes de *software* cuyos valores son diferentes a los críticos. En dependencia del resultado del análisis, rediseñar (la calidad aceptable es alcanzada mediante el rediseño), cancelar (la calidad es tan pobre que rediseñar no es factible), o no hacer cambios (desviaciones de valores críticos de métricas resultan insignificantes) los componentes de *software*.

2. Hacer predicciones de calidad de *software*

Utilizar las métricas validadas durante el proceso de desarrollo para predecir los valores de las métricas directas. Luego comparar los valores predichos de las métricas directas con los valores objetivos para determinar valores banderas del *software* para un futuro análisis. Hacer

predicciones para los componentes de *software* y pasos del proceso. Se deben analizar en detalle los componentes de *software* y los pasos del proceso donde esos valores de las métricas directas se desvían de los valores objetivos.

3. **Asegurar la complacencia con los requerimientos**

   Utilizar métricas directas para asegurar la complacencia de los productos de *software* con los requerimientos de calidad durante las pruebas de sistema y aceptación. Se usan las métricas directas para los componentes y pasos del proceso de *software*. Se comparan estos valores de las métricas con los valores objetivos de las métricas directas. Luego se clasifican los componentes de *software* y los pasos del proceso cuyas medidas se desvían de los valores objetivos como no complacientes.

3.6.2. Validar las métricas de *software*

El resultado de este paso es un grupo de métricas validadas para hacer predicciones de factores de calidad. Las actividades para lograr este resultado se muestran desde **1** hasta **3**.

1. **Aplicación de la metodología de validación**

   El propósito de la validación de las métricas es identificar métricas de proceso y producto que pueden predecir valores de factores de calidad específicos, que son representaciones cuantitativas de los requerimientos de calidad. Las métricas deben indicar si los requerimientos de calidad han sido alcanzados o cuales serán igualmente alcanzados en el futuro.

   Cuando es posible medir los valores de un factor de calidad en el punto deseado de su ciclo de vida, verdaderos valores de calidad (como por ejemplo la confiabilidad) no están disponibles. Se obtienen después de la entrega del producto o en las últimas fases del proyecto. En estos casos las métricas validadas deben ser empleadas en fases tempranas del desarrollo del *software* para poder predecir valores de los factores de calidad.

La validación no implica una validación universal de las métricas para cualquier aplicación. Más bien se refiere a la validación de la relación entre un grupo de métricas y un factor de calidad para una aplicación determinada.

El historial de la aplicación de métricas indica que las métricas predictivas eran raramente validadas (ha sido demostrado mediante análisis estadístico que las métricas miden características del *software* propuestas a medir). De cualquier forma, es importante que las métricas predictivas sean validadas antes de que se empleen para evaluar la calidad del *software*. De otra manera, las métricas serán mal empleadas (las métricas que serán empleadas pueden tener poca o ninguna relación con las características deseadas de calidad).

Mientras que los subfactores de calidad son útiles para identificar y establecer factores de calidad y métricas, estos no son empleados en la validación de métricas, porque el enfoque de la validación se centra en determinar donde existe una relación estadística significativa entre los valores de una métrica predictiva y los valores del factor de calidad.

Los factores de calidad pueden ser afectados por múltiples variables. Una única métrica, por consiguiente, no puede ser suficientemente representativa para un factor de calidad que ignore estas otras variables.

2. Aplicar criterio de validación

Para ser considerada válida, una métrica predictiva debe tener un alto grado de asociación con los factores de calidad que representa en conformidad con los umbrales listados debajo. Una métrica puede ser válida respecto a determinado criterio de validez e inválida para otro. Para el propósito de evaluar la validez de la métrica, se deben definir los siguientes umbrales:

V – Cuadrado del coeficiente de correlación linear.
B – Rango del coeficiente de correlación.
A – Error de predicción.
α – Nivel de confianza.
P – Tasa de éxito.

La descripción de cada criterio de validación se brinda debajo.

a) **Correlación:** la variación del valor del factor de calidad explicado en la variación de los valores de las métricas que esta dado por el cuadrado del coeficiente de correlación linear (R_2) entre la métrica y el factor de calidad correspondiente, debe exceder V.

   Este criterio evalúa si hay suficiente asociación linear entre el factor de calidad y la métrica para garantizar el uso de la métrica como sustituta para el factor de calidad, cuando no es factible usarlo después.

b) **Traceo:** si una métrica M esta directamente relacionada con un factor de calidad F, para un producto o proceso dado, entonces un cambio en el factor de calidad de FT1 a FT2, debe estar acompañado por un cambio en el valor de la métrica de MT1 a MT2. Este cambio debe estar en la misma dirección (si F se incrementa, M se decrementa). Para realizar esta prueba, calcule el coeficiente del rango de correlación (r) de n pares de valores del factor de calidad y de la métrica. Cada par factor/métrica debe ser medido en el mismo punto en el espacio de tiempo y los n pares de valores son medidos n veces en un periodo de tiempo. El valor absoluto debe exceder B.
   Este criterio evalúa si la métrica es capaz de tracear cambios en la calidad de un producto o un proceso durante su ciclo de vida.

c) **Consistencia:** si los valores del factor de calidad F1, F2, Fn, correspondientes a productos o procesos 1, 2, n tienen la relación F1 > F2 > Fn entonces los valores correspondientes a la métrica deben tener la relación M1 > M2 > Mn. Para desarrollar esta prueba calcule los coeficientes del rango de correlación (r) entre pares de valores (del mismo componente de *software*) del factor de calidad y la métrica. El valor absoluto de r debe exceder B. Este criterio evalúa si hay consistencia entre los rangos de los valores de factores de calidad de un grupo de componentes de *software* y los rangos de los valores de las métricas para el mismo grupo de componentes. Este criterio debe ser empleado para determinar si una métrica puede medir de forma precisa el rango, dada la calidad, de un grupo de productos o procesos.

d) **Predictibilidad:** si una métrica se usa en un tiempo T1 para predecir un factor de calidad para un producto o proceso dado, debe predecir un factor de calidad relacionado Fp_{T2} con una exactitud de:

$$\left|\frac{Fa_{T2} - Fp_{T2}}{Fa_{T2}}\right| < A$$

Donde:

Fa_{T2} : valor actual de F en el tiempo T2

Este criterio evalúa si la métrica es capaz de predecir el valor del factor de calidad con la precisión requerida.

e) **Poder discriminativo:** una métrica debe ser capaz de discriminar entre componentes de *software* de alta y baja calidad. El grupo de valores de métricas asociados al anterior deben ser significativamente más alta (o baja) que aquellos asociados con el posterior. Este criterio evalúa si la métrica es capaz de separar los componentes de alta calidad de los de baja calidad. Esta capacidad identifica valores críticos para métricas que deben ser usadas para identificar los componentes de *software* con una calidad inaceptable. Para desarrollar esta prueba, ponga el factor de calidad y los datos de la métrica en una tabla de contingencia y calcule el valor estadístico χ^2. Este valor debe ser mayor que el χ^2 de α.

f) **Confiabilidad:** una métrica debe demostrar la correlación, traceabilidad, consistencia, predictibilidad y poder discriminativo de al menos P% de las aplicaciones de la misma. Este criterio es empleado para asegurar que la métrica ha pasado una prueba de validación en un porciento suficiente de aplicaciones por lo que se puede confiar en que la métrica puede desarrollar su función consistentemente.

3. Procedimiento de validación

a. Identificar la muestra de factores de calidad

Una muestra de factores de calidad debe ser dibujada de la base de datos de las métricas.

b. Identificar la muestra de factores de calidad

Una muestra del mismo dominio (los mismos componentes de *software*), que se uso en el paso 1 debe ser dibujada de la base de datos de las métricas.

c. Realizar análisis estadístico

El análisis descrito en el paso 2 debe ser realizado. Antes de que se use la métrica para evaluar la calidad del producto o proceso, esta debe ser validada contra el criterio descrito en el paso 2. Si una métrica no pasa la prueba de validación, debe ser solo empleada de acuerdo al criterio prescrito en esas pruebas (si pasa solo la prueba de traceo debe ser empleada solo para tracear la calidad del producto o proceso).

d. Documentar resultados

Documentar los resultados debe incluir las métricas directas, las predictivas, el criterio de validación y los resultados numéricos, como mínimo.

e. Revalidar las métricas

Una métrica validada no tiene por qué ser válida en otros ambientes o aplicaciones futuras. Por consiguiente, una métrica predictiva debe ser revalidada antes de ser empleada por otra aplicación o en un ambiente diferente.

f. **Evaluar la estabilidad del ambiente**

La validación de las métricas debe ser desarrollada en un ambiente de desarrollo estable (donde el lenguaje de diseño e implementación o las herramientas de desarrollo del proyecto no varíen durante la etapa de desarrollo del proyecto en que se realiza la validación).

Una organización inicia este proceso de acuerdo a la validación de los requerimientos de este estándar con la recolección de datos y validando métricas en un proyecto (validación del proyecto). Este proyecto debe ser similar a aquel en que las métricas son aplicadas (proyecto de aplicación) con respecto a las habilidades de ingeniería de *software*, aplicación, tamaño y ambiente de ingeniería de *software*.

La validación y aplicación de métricas debe ser realizada durante la misma fase del ciclo de vida en diferentes proyectos. Por ejemplo si la métrica X es recolectada durante la fase de diseño del proyecto A y después es validada respecto al factor de calidad Y, que es recolectado durante la fase de operación del proyecto A, la métrica X puede ser usada durante la fase de diseño del proyecto B para predecir el factor de calidad Y respecto a la fase de operaciones del proyecto B.

3.7. Conclusiones del capítulo

En este capítulo se obtienen resultados satisfactorios de la validación del sistema de métricas mediante el método Criterio de Expertos. Además se describe la metodología a emplear por los proyectos productivos para evaluar la calidad del levantamiento de requisitos de los mismos mediante el sistema propuesto.

Conclusiones

- En la UCI existe un empleo deficiente de métricas de calidad en la producción de *software*.
- El levantamiento de requisitos no se realiza con la calidad necesaria siendo esta una etapa de vital importancia para el desarrollo futuro del producto de *software*.
- Se propuso un Sistema de Métricas para Requisitos Funcionales para medir la calidad de los proyectos productivos desde el proceso de levantamiento de requisitos.
- Se propuso una metodología para la definición de métricas y aplicación de las mismas en los proyectos productivos.
- Se validó el sistema propuesto mostrando un grado aceptable de eficiencia.
- El Sistema de Métricas para Requisitos Funcionales propuesto contribuye al incremento de la calidad de la confección de la documentación relacionada a la especificación del *software*.
- Dada la metodología seleccionada éste sistema es flexible, pues puede ajustarse a las condiciones reales del proyecto en el que se emplee.

Recomendaciones

- Considerar la modificación de la métrica “Estimación del tiempo de duración del levantamiento de requisitos” en cuanto a factores externos como: complejidad de los requisitos funcionales y disponibilidad del cliente.
- Implementar las métricas de calidad que no tienen métricas directas asociadas.
- Realizar la medición de la calidad de los proyectos productivos empleando el Sistema de Métricas para Requisitos Funcionales de acuerdo a la metodología *IEEE 1061 – 1998 Standard for a Software Quality Metrics Methodology.*

Bibliografía

ALEXIS GARCIA- PÉREZ, D. A. M., ALFREDO SOMOZA-MORENO. Imperatives of Free and Open Source *Software* in Cuban Development. MIT Press Journals, 2006. 3.

CAELUM. Calidaddel*Software*.com, 2006. [2008]. Disponible en: http://www.calidaddel*software*.com/

CARMEN PAGES, L. D.-M., JOSÉ-JAVIER MARTÍNEZ, JOSÉ-ANTONIO GUTIÉRREZ DEFINICIÓN DE MÉTRICAS DE CALIDAD EN EL PROCESO DE PARAMETRIZACIÓN DE SISTEMAS ERP. Revista de Procesos y Métricas de las tecnologías de la Información, 2007, 4(3): 82.

CASAÑOLA, Y. T. PROPUESTA DE MODELO DE PRODUCCIÓN DE *SOFTWARE* PARA LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS.

CITMATEL. Biblioteca Virtual de las Ciencias en Cuba, 2005. [2008]. Disponible en: http://www.bibliociencias.cu/

GET. Grupo de Electrónica para el Turismo, 2006. [2008]. Disponible en: http://www.get.tur.cu/index.php

HERNÁNDEZ, D. D. L. N. Modelo de gestión de la calidad para empresas de proyectos cubanas.: Avances. CITMA, 2005. 7.

ING. JENNY RUIZ DE LA PEÑA, I. O. A. C. Importancia de la Ingeniería de *Software* en la producción de *software*. Ciencias Holguín, 2007.

LÓPEZ, F. La informatización como mina de la eficiencia. Granma, 2003.

MIC. MInisterio de la Informática y las Comunicaciones, 2008]. Disponible en: http://www.informaticahabana.com/

---. Oficina para la Informatización, 2008]. Disponible en: http://www.infosoc.cu/default.php

MUÑOZ, D. C. C. Métricas de *Software*. Conceptos básicos y formalización., 2006.

ARAUJO, I. Consultores en Negocios Internacionales, 1999. [2008]. Disponible en: http://portal.araujoibarra.com/

AYLIN FEBLES ESTRADA, S. A. C. LA GESTIÓN DE CONFIGURACIÓN Y EL DESARROLLO DE *SOFTWARE* EN LAS UNIVERSIDADES. UNA EXPERIENCIA PRÁCTICA, 2005.

DURÁN, M. R. MEDICIONES PRÁCTICAS DE *SOFTWARE* Y SISTEMAS (PSM): UNA PROPUESTA PARA LA PRODUCCIÓN DE *SOFTWARE* EN LA UCI.: Informática 2007, 2007.

ESPINOSA, Y. M. Comercialización de *Software*. Estrategia y necesidad, 2006.

LLERENA, M. G. M. G. Experiencias en la certificación del proceso de calidad del *software* bajo la norma NC ISO 9001:2001, 2006.

MARTHA DUNIA DELGADO DAPENA, S. A. C., ALEJANDRO ROSETE SUÁREZ. Una propuesta de introducción de las revisiones en el proceso de desarrollo de *software*. Revista Investigación Operacional, 2005. 26.

SEVILLA, U. D. Departamento de Lenguajes y Sistemas Informáticos, 2006. [2008]. Disponible en: http://www.lsi.us.es/docencia/pagina_asignatura.php?id=48

Astigarraga, Eneko. Code Syntax. Internet Solutions. [En línea] http://www.codesyntax.com/prospectiva/Metodo_delphi.pdf.

GTIC. 2005. Grupo de Tecnologia de la informacion y las Comunicaciones. [En línea] GTIC, 2005. http://www.gtic.ssr.upm.es/encuestas/delphi.htm.

Izquierdo, Paula, y otros. UAM. [En línea] www.uam.es/personal_pdi/economicas/rmc/prevision/pdf/DELPHI.ppt.

Mojena, Bedsy Guevara. 2007. [En línea] 2007. http://bibliodoc.uci.cu/TD/TD_0765_07.pdf .

Montes, Yehilit Gil. 2007. GUÍA PARA LA APLICACIÓN DE TÉCNICAS PARA EL DESARROLLO DE REQUISITOS EN LOS PROYECTOS *SOFTWARE* DE LA FACULTAD No 3. [En línea] 2007. http://bibliodoc.uci.cu/TD/TD_0210_07.pdf

IEEE. 1998. IEEE-STD-830-1998 : ESPECIFICACIONES DE LOS REQUISITOS DEL. 1998.

1998. IEEE Standard for a *Software* Quality. s.l. : *Software* Engineering Standards Committee, 1998. ISBN 0-7381-1510-X SS94706.

Referencias Bibliográficas

PRESSMAN, R. S. *Ingeniería de Software. Un enfoque práctico*. Ciudad de La Habana, 2005. p.

MINREX. CubaMinrex, 2004. [2008]. Disponible en:

http://www.cubaminrex.cu/Sociedad_Informacion/Cuba_SI/Informatizacion.htm

Glosario de Términos

Analista de Medición: rol que se dedica a definir mediciones de los procesos para los proyectos y la recolección de las mismas. Además es el encargado de definir las métricas.

Analista de Requisitos: rol que se encarga de la gestión de los requisitos en un proyecto.

Calidad: cualidades que caracterizan un producto o servicio y que determinan su utilidad y existencia. Esta implica eficiencia, confiabilidad, seguridad, integridad, etc.

Ciclo de vida del *software*: ciclo que cubre las 4 fases por las que transita el *software* durante su desarrollo: inicio, elaboración, construcción, transición.

Cliente: persona, grupo de personas u organización que solicita la construcción de un sistema de *software* o la mejora de uno ya existente.

Código: conjunto de líneas que conforman un bloque de instrucciones escritas según las reglas sintácticas de un lenguaje de programación específico entendible para los programadores. Este código para ser ejecutado es interpretado por un compilador.

CRM: hace referencia a una estrategia de negocio basada principalmente en la satisfacción de los clientes, pero también a los sistemas informáticos que dan soporte a esta estrategia.

Defecto: anomalía del sistema. Fallo cometido en un producto durante su proceso de desarrollo que se detecta después de entregado al cliente. Falta de conformidad con los requisitos.

Equipo de desarrollo: grupo de personas que se encargan del desarrollo del *software* durante todo su ciclo de vida.

Error: fallo cometido en un producto durante el proceso de la Ingeniería de *Software* que se detecta antes de la entrega al cliente.

ERP (Enterprise Resource Planning): Sistema o Software administrativo que integra todas las áreas de una empresa (Como contabilidad, compras, o inventarios), mediante procesos transparentes y en tiempo real en bases de datos relacionales y centralizadas.

Estándar: sinónimo de norma, regulación. Modelo o guía que se sigue para realizar determinado proceso.

Estándar de Calidad: permiten definir un conjunto de criterios de desarrollo que guían la forma en que se aplica la Ingeniería de *Software*. Es una base sobre la cual todos los procesos se efectúan de la misma forma como guía para lograr la productividad y la calidad.

Estrato: Conjunto de elementos que, con determinados caracteres comunes, se ha integrado con otros conjuntos previos o posteriores para la formación de una entidad o producto históricos, de una lengua, etc.

Grupo Standish: creado en 1985 y radicado en Boston. Recolecta información sobre casos de fallas reales de las TI y sus entornos. Es el grupo líder en proyectos de Información Tecnológica, con años de experiencia práctica en asesoramiento de riesgo, costos y valor de retorno de las Inversiones en TI.

IEEE: *Institute for Electrical and Electronic Engineers* (Instituto de Ingenieros Eléctricos y Electrónicos). Organización profesional que establece los estándares para telecomunicaciones y computadoras.

IEEE Std. 610.12-1990: (Diccionario Estándar de Computación): Recopilación de Glosarios Estándares de la IEEE que cubren los campos de la matemática computacional, aplicaciones computacionales, modelamiento y simulación, procesamiento de imágenes e ingeniería de *software*, entre otros.

IEEE Std.830 – 1998: establece la estructura del documento de especificación de requisitos de *software* y las características que determinan la calidad de los requisitos.

Ingeniería de Requisitos: disciplina de la ingeniería de *software* que define un conjunto de actividades implicadas en determinar, documentar y gestionar los requisitos de un sistema de *software*.

Ingeniería de *Software*: disciplina de la ingeniería que concierne todos los aspectos de la producción de *software*. Para ello involucra personas, herramientas y técnicas.

ISO: International Organization for Standarization (Organización Internacional para la Estandarización). Organización no gubernamental compuesta por representantes de varios países, fue fundada el 23 de febrero de 1947. Promueve el empleo de estándares industriales y comerciales a nivel mundial.

ISO/IEC 15939: define un proceso de medición aplicable a la ingeniería de sistemas y *software* y a las disciplinas de gestión. El proceso se describe a través de un modelo que define las actividades del proceso de medición requeridas para especificar adecuadamente la información que se requiere.

KLDC: miles de líneas de código.

LDC: líneas de código

Líder de proyecto: Jefe Principal del equipo de desarrollo de *software*.

Medición: acción de determinar una medida.

Medida: cantidad que resulta de medir una magnitud.

Medidas directas: aspectos medibles de determinado objeto. (Ejemplo: longitud, tamaño).

Medidas indirectas: no son tan fáciles de determinar como las directas y generalmente dependen de la combinación de éstas. (Ejemplo: eficiencia, facilidad de mantenimiento).

Métrica de *Software*: medida cuantitativa que determina el grado en que un sistema, componente o proceso de *Software*, posee un atributo dado.

Métricas de Bang: métrica que mide el tamaño del *software* a implementar como consecuencia del modelo de análisis.

Métricas privadas: métricas recopiladas por los integrantes de los proyectos de *software* que indican la efectividad de las actividades de control individuales. Estos indicadores son asimilados por el proyecto.

Métricas públicas: las medidas privadas se combinan y dan paso a nuevas métricas públicas para el proyecto, este a su vez puede hacerlas públicas para el proceso.

Modelo: arquetipo que se toma como pauta a seguir.

Modelo de Calidad: contienen las mejores prácticas, proponen temas relacionados a la administración en los que la organización debe hacer énfasis, integra prácticas dirigidas a los procesos clave y permite la medición de la evolución de la calidad.

Modelos de Calidad de *Software*: modelos que establecen pautas para obtener un *software* de calidad o medir ésta en el mismo. Pueden aplicarse a fases específicas del proceso, al proceso en general o al producto final.

Modelo de Bohem: determina el orden de las etapas involucradas en el desarrollo de *software* estableciendo el criterio de transición de una a otra.

Modelo FCM (Factors/Criteria/Metrics):

Modelo CMMI (Capability Maturity Model Integrated): (Modelo de Capacidad y Madurez Integrado) modelo de evaluación de los procesos de una organización. Establece un conjunto de prácticas o procesos clave agrupados en Áreas Clave de Procesos (KPA), para cada una de las cuales se definen un conjunto de buenas prácticas que definen la madurez de la organización.

Modelo ISO 9001-2000: conjunto de normas internacionales que forman un modelo para la implementación de Sistemas de Gestión de la Calidad en las organizaciones.

Modelo ISO 15504 (SPICE: *Software* Process Improvement and Capability Determination): estándar internacional para el aseguramiento de procesos desarrollado bajo el auspicio de la ISO y la IEC (International Electrotechnical Commission). Ha evolucionado de un modelo de referencia para buenas prácticas de *software* a un marco de trabajo para la apreciación de procesos abarcando el campo de las tecnologías de la información. Es aplicable a múltiples disciplinas y permite a las comunidades establecer sus propios modelos de referencia.

Muestro Aleatorio Estratificado: MAE. Tipo de muestreo estadístico que divide una población en estratos o subpoblaciones de acuerdo a criterios que sean importantes en el estudio.

Muestreo Irrestricto Aleatorio: tipo de muestreo donde la muestra de la población es seleccionada de forma tal que cada muestra posible tiene igual probabilidad de ser seleccionada.

Muestreo Probabilístico: muestreo compuesto por todos aquellos métodos para los que puede calcularse la posibilidad de extracción de cualquiera de las muestras posibles.

Normas: regulaciones de obligatorio cumplimiento.

Proceso: secuencia de actividades que se realizan en un orden predefinido y con un fin determinado. Plantilla para crear proyectos.

Proceso de Desarrollo de *Software*: conjunto de todas las actividades necesarias para transformar los requisitos del cliente en el producto de *software*. Explica los pasos necesarios para terminar un proyecto.

Producto: resultado final del proceso de desarrollo de *software* que es entregado al cliente.

Proyecto: elemento organizativo a través del cual se gestiona el desarrollo de *software*. Tiene como resultado la versión de un producto.

PSM: *Practical Software and System Measurements*.

Requisito: condición o capacidad que debe cumplir un sistema.

Requisitos funcionales: condición que tiene que cumplir un sistema.

Rol: función que cumple una persona en determinado contexto.

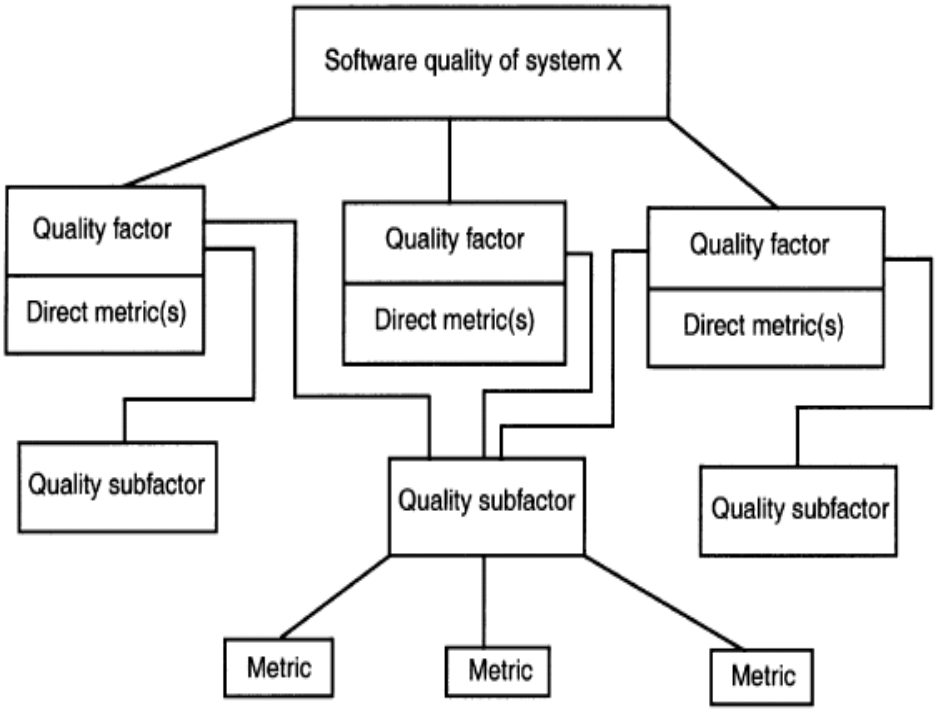
Sistema: conjunto de elementos que relacionados entre sí de forma ordenada contribuyen al logro de un objetivo predeterminado.

***Software*:** conjunto de programas e instrucciones para ejecutar determinadas tareas en una computadora.

Usuario: persona o sistema que interactúa con el sistema que se desarrolla.

Anexos

Anexo 1: Marco de trabajo de las métricas de calidad de *software*



Anexo 2: Métricas de *Software*

Item	Description
Name	Name given to the metric.
Costs	Costs of using the metric (see 4.2.2.1).
Benefits	Benefits of using the metric (see 4.2.2.2).
Impact	Indication of whether a metric can be used to alter or halt the project (ask, "Can the metric be used to indicate deficient software quality?").
Target value	Numerical value of the metric that is to be achieved in order to meet quality requirements. Include the critical value and the range of the metric.
Quality factors	Quality factors that are related to this metric.
Tools	Software or hardware tools that are used to gather and store data, compute the metric, and analyze the results.
Application	Description of how the metric is used and what its area of application is.
Data items	Input values that are necessary for computing the metric values.
Computation	Explanation of the steps involved in the metrics computation.
Interpretation	Interpretation of the results of the metrics computation (see 4.4.1).
Considerations	Considerations of the appropriateness of the metric (e.g., Can data be collected for this metric? Is the metric appropriate for this application?).
Training required	Training required to implement or use the metric.
Example	An example of applying the metric.
Validation history	Names of projects that have used the metric, and the validity criteria the metric has satisfied.
References	References, such as a list of projects and project details, giving further details on understanding or implementing the metric.

Anexo 3: Descripción de los datos de las métricas

Item	Description
Name	Name given to the data item.
Metrics	Metrics that are associated with the data item.
Definition	Straightforward description of the data item.
Source	Location of where the data item originates.
Collector	Entity responsible for collecting the data.
Timing	Time(s) in life cycle at which the data item is to be collected. (Some data items are collected more than once.)
Procedures	Methodology (e.g., automated or manual) used to collect the data.
Storage	Location of where the data are stored.
Representation	Manner in which the data are represented, e.g., precision and format (Boolean, dimensionless, etc.).
Sample	Method used to select the data to be collected and the percentage of the available data that is to be collected.
Verification	Manner in which the collected data are to be checked for errors.
Alternatives	Methods that may be used to collect the data other than the preferred method.
Integrity	Person(s) or organization(s) authorized to alter the data item and under what conditions.

Anexo 4: Encuestas a realizada a los expertos

Encuesta a especialistas para someter a sus criterios la propuesta de un Sistema de Métricas para Requisitos Funcionales.

Estimado(a) Profesor:

La presente encuesta forma parte de la aplicación del Método de Valoración de Especialistas. Con este fin solicitamos su valiosa colaboración, y de antemano le aseguramos, que sus opiniones se tendrán en cuenta para aplicación del Sistema de Métricas para Requisitos Funcionales a proyectos.
Muchas Gracias.

Nombre y Apellidos: _____
Fecha de graduación: _____ Puesto de trabajo actual: _____
Calificación profesional: Ingeniero___ Licenciado en Educación __ Máster ___ Doctor___
Años de experiencia como especialista de calidad: _____
Categoría Docente: Prof. Instructor___ Prof. Asistente____ Prof. Auxiliar____ Prof. Titular____ Prof. Adjunto_____

1. Seleccione de escala el valor que corresponda al grado de conocimientos que usted posee acerca del tema de investigación que desarrollamos, (1: no tiene ningún conocimiento, 5: pleno conocimiento de la problemática tratada.)

1	2	3	4	5	6	7	8	9	10

2. Valore el grado de influencia que cada una de las fuentes que le presentamos a continuación ha tenido en su conocimiento y criterios sobre el tema que se investiga.

FUENTES DE ARGUMENTACIÓN	Grado de Influencia de Cada Fuente		
	ALTO	MEDIO	BAJO
Conocimientos teóricos que posee acerca del tema.			
Su experiencia obtenida en la actividad práctica			
Certificaciones que ha obtenido en esta área			

3. ¿Considera Ud. importante la aplicación de un Sistema de Métricas para Requisitos Funcionales en los proyectos que se desarrollan en la Universidad?

Sí__ No__ No sé__

¿Por qué?

_____

_____

_____

4. De una valoración del 1 al 5 de los criterios expuestos a continuación según la propuesta a validar:

--- Necesidad del empleo de la propuesta.

--- Posibilidad de aplicación.

--- Aporte en el proceso de Gestión de Requerimientos desarrollado en un proyecto.

--- Impacto en la Calidad de los Requerimientos.

5. ¿Considera Ud que se puede aplicar exitosamente el Sistema de Métricas propuesto en los proyectos productivos de la Universidad, de manera que permita mejorar la calidad de la obtención de los requisitos?

Sí__ No__ No sé__

¿Por qué?

_____

_____

_____

6. ¿Que factores cree usted que podría atentar contra la correcta aplicación de este Sistema de Métricas en la UCI y cuales lo facilitarían?

8. Luego de conocer el sistema de métricas para requisitos funcionales, valore la eficiencia que este podría tener

__ No sería eficaz

__ Pudiera ser o no

__ Seria eficaz

9. ¿Considera Ud que las métricas seleccionadas son correctas para cumplir el objetivo de elevar la calidad de la obtención de los requisitos funcionales?

Sí__ No__ No sé__

¿Por qué?

_____

_____

_____

10. Haga un comentario o aporte sobre el manual que usted esta evaluando. (El comentario es libre y debe reflejar algún elemento de interés que aporte elementos a la mejora del mismo):

_____

_____

_____

11. Según su criterio, ¿Qué otras métricas debería tener el sistema propuesto?

_____

_____

_____

Anexo 5: Encuesta a integrantes de proyectos productivos de la UCI

Encuesta a integrantes de proyectos productivos.

Rol: _____ **Proyecto:** _____ **Facultad:** __ **Fecha:** _____

Marque con una X su respuesta:

1. ¿Conoce qué es una métrica?
   __ Sí __No
2. ¿Conoce si en su proyecto se emplean métricas?
   __ Sí __No
   a) ¿Cuáles?:
   _____
   _____
3. ¿Ha empleado métricas alguna vez en algún proyecto?
   __ Sí __No
   b) ¿Cuáles?:
   _____
   _____
4. ¿Considera importante el uso de las métricas en su proyecto?
   __ Sí __No
5. ¿Conoce si la universidad emplea algún sistema de métricas estándar en sus proyectos?
   __ Sí __No
   a) ¿Cuáles?:
   _____
   _____

NOTA: Si es Líder de Proyecto o Analista de Requisitos, responda además las preguntas al dorso.

___ **Líder de Proyecto**

1. ¿Existe algún analista de medición en su proyecto?
   __ Sí __No

2. ¿Conoce si en la universidad existe alguna política que estipule el uso de métricas en el levantamiento de requisitos?
   __ Sí __No

___ **Analista de Requisitos**

1. ¿Conoce si en la universidad existe alguna política que estipule el uso de métricas en el levantamiento de requisitos?
   __ Sí __No

2. ¿Ha empleado métricas orientadas a requisitos funcionales alguna vez en un proyecto?
   __ Sí __No

3. ¿Qué tiempo demora en la obtención de requisitos funcionales? _____ (semanas)

4. ¿Son muy extensos estos requisitos funcionales?
   __ Sí __No
   a) Cantidad de requisitos funcionales extensos: ___
   b) Cantidad de requisitos funcionales medianos: ___

NOTA: Un requisito extenso es cuando sobrepasa las 6 palabras. Un requisito funcional medio es cuando las palabras que lo contiene son menores ó iguales que 6.